aws

사용자 가이드

AWS Database Migration Service

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Database Migration Service: 사용자 가이드

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon의 상표 및 트레이드 드레스는 Amazon 외 제품 또는 서비스와 함께 사용하여 고객에게 혼란을 초래하거나 Amazon을 폄하 또는 브랜드 이미지에 악영향을 끼치는 목적으로 사용할 수 없습니다. Amazon이 소유하지 않은 기타 모든 상표는 Amazon과 제휴 관계이거나 관련이 있거나 후원 관계와 관계없이 해당 소유자의 자산입니다.

Table of Contents

AWS Database Migration Service란 무엇인가요? 1
AWS DMS에서 수행하는 마이그레이션 작업 2
AWS DMS 작동 방식 4
개괄적인 관점 AWS DMS 4
구성 요소 6
소스 12
데이터 마이그레이션에 사용할 수 있는 소스 12
DMS Fleet Advisor가 지원하는 소스 15
DMS Schema Conversion이 지원하는 소스 15
DMS 동종 데이터 마이그레이션이 지원하는 소스 16
대상 17
대상 마이그레이션에 사용할 수 있는 대상 17
DMS Fleet Advisor가 지원하는 대상 19
DMS Schema Conversion이 지원하는 대상 20
DMS 동종 데이터 마이그레이션이 지원하는 대상 20
Amazon 리소스 이름 21
다른 AWS 서비스와 함께 23
에 대한 지원 AWS CloudFormation 23
시작하기 25
설정 25
AWS 계정에 등록 25
관리 사용자 생성 26
사전 조건 27
VPC 생성 28
Amazon RDS 파라미터 그룹 생성 29
소스 Amazon RDS 데이터베이스 생성 30
대상 Amazon RDS 데이터베이스 생성 32
Amazon EC2 클라이언트 생성 33
소스 데이터베이스 채우기 34
스키마 마이그레이션 36
복제 38
1단계: 복제 인스턴스 생성 38
2단계: 소스 및 대상 엔드포인트 지정 40
3단계: 태스크 생성 및 데이터 마이그레이션 42

4단계: 복제 테스트 44
5단계: AWS DMS 리소스 정리 46
추가 리소스 47
마이그레이션을 위한 데이터베이스 검색 49
지원되는 AWS 리전 50
시작하기 52
설정 52
필요한 리소스 생성 53
데이터베이스 사용자 생성 62
데이터 수집기 68
권한 69
데이터 수집기 생성 70
데이터 수집기 설치 72
OS 및 데이터베이스 서버 검색 75
모니터링되는 객체 관리 78
SSL 사용 81
데이터 수집 82
문제 해결 87
인벤토리 89
데이터베이스 인벤토리 사용 90
스키마 인벤토리 사용 91
대상 권장 사항 93
대상 인스턴스 93
DMS Fleet Advisor는 대상 사양을 어떻게 결정합니까? 94
대상 권장 사항 생성 95
권장 사항 세부 정보 97
대상 권장 사항 내보내기 98
마이그레이션 제한 99
문제 해결 116
제한 사항 117
데이터베이스 스키마 변환 118
지원됨 AWS 리전 119
특성 120
제한 사항 121
시작하기 122
필수 조건 122

1단계: 인스턴스 프로파일 생성 127
2단계: 데이터 공급자 구성 128
3단계: 마이그레이션 프로젝트 생성 129
4단계: 평가 보고서 생성 129
5단계: 소스 코드 변환 130
6단계: 변환된 코드 적용 131
7단계: 정리 및 문제 해결 131
네트워크 설정 132
단일 VPC 구성 133
여러 VPC 구성 133
AWS Direct Connect 또는 VPN 사용 133
인터넷 연결 사용 134
인터넷 게이트웨이가 없는 환경 사용 134
소스 데이터 공급자 생성 135
SQL Server를 소스로 사용 135
Oracle을 소스로 사용 137
Oracle Data Warehouse를 소스로 사용 138
PostgreSQL을 소스로 사용 140
MySQL을 소스로 사용 141
대상 데이터 공급자 생성 141
MySQL을 대상으로 사용 142
PostgreSQL을 대상으로 사용 144
Amazon Redshift 대상으로 사용 144
마이그레이션 프로젝트 관리 145
마이그레이션 프로젝트 설정 지정 146
데이터베이스 마이그레이션 평가 보고서 147
평가 보고서 생성 147
평가 보고서 보기 148
평가 보고서 저장 149
스키마 변환 151
변환 규칙 설정 152
데이터베이스 스키마 변환 154
스키마 변환 설정 지정 157
데이터베이스 스키마 새로 고침 162
스키마 저장 및 적용 162
확장 팩 사용 164

동종 데이터베이스 마이그레이션 166
지원되는 AWS 리전 167
특성 168
제한 사항 168
개요 169
설정 170
IAM 리소스 생성 170
네트워크 설정 175
소스 데이터 공급자 생성 178
소스로 MySQL 또는 MariaDB 사용 179
PostgreSQL을 소스로 사용 182
MongoDB 또는 Amazon DocumentDB를 소스로 사용 185
대상 데이터 공급자 생성 188
대상으로 MySQL 또는 MariaDB 사용 189
PostgreSQL을 대상으로 사용 190
Amazon DocumentDB를 대상으로 사용 192
데이터 마이그레이션 192
데이터 마이그레이션 생성 193
선택 규칙 195
데이터 마이그레이션 관리 198
데이터 마이그레이션을 모니터링 199
마이그레이션 상태 201
MySQL에서 데이터 마이그레이션 202
PostgreSQL에서 데이터 마이그레이션 204
MongoDB에서 데이터 마이그레이션 206
문제 해결 207
데이터 마이그레이션 생성 207
데이터 마이그레이션 시작 208
연결 문제 208
PostgreSQL에서 뷰를 테이블로 마이그레이션합니다. 209
마이그레이션 프로젝트 작업 210
서브넷 그룹 생성 211
인스턴스 프로파일 생성 211
데이터 공급자 생성 213
마이그레이션 프로젝트 생성 215
마이그레이션 프로젝트 관리 216

모범 사례 218
AWS Database Migration Service 마이그레이션 계획 218
스키마 변환 220
AWS DMS 문서 검토 220
개념 증명 실행 220
성능 개선 221
자체 온프레미스 이름 서버 사용 225
AWS DMS에서 Amazon Route 53 Resolver 사용 226
대용량 이진 객체(LOB) 마이그레이션 227
제한적 LOB 모드 사용 228
개선된 LOB 성능 229
행 필터링을 사용하여 큰 테이블 마이그레이션 시 성능 향상 231
지속적 복제 232
소스 데이터베이스의 로드 감소 233
대상 데이터베이스에서 병목 현상 감소 233
데이터 유효성 검사 사용 233
지표 모니터링 234
이벤트 234
작업 로그 사용 235
Time Travel을 통한 복제 문제 해결 235
Oracle 대상에서 사용자 및 스키마 변경 236
Oracle 대상에 대한 테이블 및 인덱스 테이블스페이스 변경 236
복제 인스턴스 업그레이드 237
마이그레이션 비용 이해 238
서버리스로 작업하기 AWS DMS 239
DMS 서버리스 구성 요소 240
지원되는 엔진 버전 242
서버리스 애플리케이션 생성 243
서버리스 복제 수정 AWS DMS 245
컴퓨팅 구성 248
서버리스의 오토스케일링에 대한 이해 AWS DMS 249
서버리스 복제 모니터링 AWS DMS 250
서버리스의 제한 사항 254
복제 인스턴스 작업 256
복제 인스턴스 유형 선택 260
사용할 인스턴스 클래스 결정 265

무제한 모드 버스트 가능 인스턴스 266
복제 인스턴스 크기 조정 267
고려해야 할 요소 267
일반적인 문제 268
모범 사례 269
복제 엔진 버전 269
콘솔을 사용하여 엔진 버전 업그레이드 269
를 사용하여 엔진 버전 업그레이드 AWS CLI 270
퍼블릭 및 프라이빗 복제 인스턴스 271
IP 주소 지정 및 네트워크 유형 272
복제 인스턴스용으로 네트워크 설정 273
데이터베이스 마이그레이션을 위한 네트워크 구성 274
복제 서브넷 그룹 생성 282
DNS를 사용하여 도메인 엔드포인트 확인 283
암호화 키 설정 283
복제 인스턴스 생성 284
복제 인스턴스 수정 289
복제 인스턴스 재부팅 293
복제 인스턴스 삭제 296
DMS 유지 관리 기간 297
유지 관리가 기존 마이그레이션 작업에 미치는 영향 297
유지 관리 기간 설정 변경 298
엔드포인트 300
소스 및 대상 엔드포인트 생성 300
데이터 마이그레이션용 소스 305
Oracle을 소스로 사용 305
SQL Server를 소스로 사용 368
Azure SQL 데이터베이스를 소스로 사용 395
Microsoft Azure SQL 관리형 인스턴스를 소스로 사용 395
Azure Database for PostgreSQL을 소스로 사용 395
Azure Database for MySQL을 소스로 사용 397
OCI MySQL Heatwave를 소스로 사용 397
Google Cloud for MySQL을 소스로 사용 398
Google Cloud for PostgreSQL을 소스로 사용 399
PostgreSQL을 소스로 사용 400
MySQL을 소스로 사용 433

SAP ASE를 소스로 사용 ... 446
MongoDB를 소스로 사용 ... 454
Amazon DocumentDB를 소스로 사용 ... 471
Amazon S3를 소스로 사용 ... 488
IBM Db2 LUW를 소스로 사용 ... 499
IBM Db2 for z/OS를 소스로 사용 ... 505
마이그레이션에 적합한 대상 ... 546
Oracle을 대상으로 사용 ... 547
SQL Server를 대상으로 사용 ... 557
PostgreSQL을 대상으로 사용 ... 563
MySQL을 대상으로 사용 ... 575
Amazon Redshift 대상으로 사용 ... 582
SAP ASE를 대상으로 사용 ... 607
Amazon S3를 대상으로 사용 ... 610
대상으로 Amazon DynamoDB 사용 ... 656
Amazon Kinesis Data Streams를 대상으로 사용 ... 676
Apache Kafka를 대상으로 사용 ... 694
OpenSearch을 대상으로 사용 ... 720
Amazon DocumentDB를 대상으로 사용 ... 725
Amazon Neptune을 대상으로 사용 ... 738
Redis를 대상으로 사용 ... 754
Babelfish를 대상으로 사용 ... 761
Amazon Timestream을 대상으로 사용 ... 769
Db2를 대상으로 사용 ... 779
데이터 마이그레이션에 사용할 수 있는 VPC 엔드포인트 ... 781
AWS DMS 버전 3.4.7 이상으로 마이그레이션할 경우 영향을 받는 사용자 ... 781
AWS DMS 버전 3.4.7 이상으로 마이그레이션할 경우 영향을 받지 않는 사용자 ... 782
AWS DMS 버전 3.4.7 이상으로 마이그레이션 준비 ... 782
지원되는 DDL 문 ... 783
Tasks ... 785
작업 생성 ... 789
작업 설정 ... 796
LOB 지원 설정 ... 846
여러 작업 생성 ... 847
지속적 복제 작업 ... 848
CDC 시작 지점에서 복제 시작 ... 849

양방향 복제 수행 ... 854
작업 수정 ... 858
태스크 이동 ... 858
작업 중 테이블 다시 로드 ... 859
AWS Management Console ... 860
테이블 매핑 ... 861
콘솔에서 테이블 선택 및 변환 규칙 지정 ... 862
JSON을 사용하여 테이블 선택 및 변환 지정 ... 866
선택 규칙 및 작업 ... 868
테이블 매핑의 와일드카드 ... 874
변환 규칙 및 작업 ... 875
변환 규칙 표현식을 사용하여 열 내용 정의 ... 896
테이블 및 컬렉션 설정 규칙과 작업 ... 910
소스 필터 사용 ... 939
필터 적용 ... 940
시간과 날짜를 기준으로 필터링 ... 946
마이그레이션 전 평가 활성화 및 활용 ... 947
필수 조건 ... 948
평가 실행 지정, 시작, 보기 ... 950
개별 평가 ... 954
데이터 형식 평가 시작 및 보기 ... 974
평가 실행 문제 해결 ... 978
보충 데이터 지정 ... 979
작업 모니터링 ... 981
작업 상태 ... 983
작업 중 테이블 상태 ... 985
Amazon CloudWatch를 사용한 복제 태스크 모니터링 ... 986
AWS Database Migration Service 지표 ... 988
복제 인스턴스 지표 ... 991
복제 작업 지표 ... 993
AWS DMS 로그 보기 및 관리 ... 996
AWS CloudTrail을 사용하여 AWS DMS API 호출 로깅 ... 998
CloudTrail의 AWS DMS 정보 ... 998
AWS DMS 로그 파일 항목 이해 ... 999
컨텍스트 로깅 ... 1002
객체 유형 ... 1003

로깅 예제 1004
제한 사항 1005
EventBridge 이벤트 사용 1007
AWS DMS에서 Amazon EventBridge 이벤트 규칙 사용 1008
AWS DMS 이벤트 범주 및 이벤트 메시지 1009
복제 인스턴스 이벤트 메시지 1009
복제 태스크 이벤트 메시지 1013
복제 이벤트 메시지 1015
Amazon SNS 이벤트 작업 1017
이벤트 구독을 Amazon EventBridge로 이전 1017
Amazon SNS 이벤트 및 알림 작업 1018
AWS DMS 이벤트 카테고리 및 SNS 알림용 이벤트 메시지 1019
SNS를 사용한 AWS DMS 이벤트 알림 구독 1023
AWS Management Console 사용 1023
SNS 주제의 액세스 정책 검증 1026
데이터 유효성 검사 1028
복제 작업 통계 1029
Amazon CloudWatch를 사용한 복제 태스크 통계 1031
작업 중 테이블 다시 검증 1032
AWS Management Console 1032
JSON 편집기를 사용하여 검증 규칙 수정 1033
검증 전용 태스크 1034
전체 로드 검증 전용 1034
CDC 검증 전용 1035
검증 전용 사용 사례 1035
문제 해결 1036
Redshift 검증 성능 1037
제한 사항 1038
S3 검증 1039
사전 조건 1040
권한 1040
제한 사항 1042
검증 전용 태스크 1043
리소스에 태그 지정 1044
API 1045
보안 1048

데이터 보호 1050
데이터 암호화 1050
인터네트워크 트래픽 개인 정보 1051
DMS Fleet Advisor의 데이터 보호 1051
Identity and Access Management(IAM) 1053
고객 1053
보안 인증을 통한 인증 1054
정책을 사용한 액세스 관리 1057
IAM의 AWS Database Migration Service 작동 방식 1059
보안 인증 기반 정책 예 1066
리소스 기반 정책 예제 1073
보안 암호를 사용하여 리소스에 액세스하기 1078
서비스 연결 역할 사용 1087
문제 해결 1094
필요한 IAM 권한 1097
CLI와 API에 대한 IAM 역할 1102
교차 서비스 혼동된 대리인 방지 1108
AWS 관리형 정책 1111
규정 준수 확인 1118
복원성 1120
인프라 보안 1121
세분화된 액세스 제어 1124
리소스 이름을 사용하여 액세스 제어 1124
태그를 사용하여 액세스 제어 1127
암호화 키 설정 1134
네트워크 보안 1137
SSL 사용 1139
AWS DMS에서 SSL을 사용할 때 적용되는 제한 사항 1141
인증서 관리 1141
MySQL 호환, PostgreSQL 또는 SQL Server 엔드포인트에서 SSL 활성화 1142
데이터베이스 암호 변경 1144
Limits 1145
AWS Database Migration Service에 대한 리소스 할당량 1145
API 요청 제한에 대한 이해 1146
문제 해결 및 진단 지원 1148
마이그레이션 태스크가 느리게 실행됨 1149

태스크 상태 표시줄이 움직이지 않음 1150
태스크가 완료되었지만 아무것도 마이그레이션되지 않음 1150
외부 키와 보조 인덱스 누락 1150
AWS DMS CloudWatch 로그를 생성하지 않습니다. 1151
Amazon RDS에 연결할 때 문제가 발생함 1151
오류 메시지: Incorrect thread connection string: incorrect thread value 0 1152
네트워킹 문제 발생 1152
전체 로드 후 CDC 중단 1153
태스크 재시작 시 프라이머리 키 위반 오류 발생 1153
스키마 초기 로드 실패 1153
알 수 없는 오류로 인해 태스크 실패 1153
시작부터 작업 재시작 시 테이블 로드 1154
태스크당 테이블 수로 인해 문제 발생 1154
LOB 열에 프라이머리 키가 생성되면 태스크가 실패함 1154
프라이머리 키가 없는 대상 테이블에서 중복 레코드 발생 1154
소스 엔드포인트가 예약된 IP 범위에 속함 1154
Amazon Athena 쿼리에서 타임스탬프가 깨져 보임 1155
Oracle 관련 문제 해결 1155
보기에서 데이터 가져오기 1156
12c에서 LOB 마이그레이션 1156
LogMiner오라클과 바이너리 리더 간 전환 1156
오류: Oracle CDC stopped 122301 Oracle CDC maximum retry counter exceeded. 1157
보충 로깅을 Oracle 소스 엔드포인트에 자동으로 추가합니다. 1157
LOB 변경 사항이 캡처되지 않음 1158
오류: ORA-12899: value too large for column *column-name* 1158
NUMBER 데이터 형식 오해 1158
전체 로드 중 레코드 누락 1158
테이블 오류 1159
오류: Cannot retrieve Oracle archived Redo log destination ids 1159
Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능 평가 1159
MySQL 관련 문제 해결 1161
이진 로깅 비활성화로 인해 Amazon RDS DB 인스턴스 엔드포인트에서 CDC 작업 실패 1162
작업 중에 대상 MySQL 인스턴스 연결이 끊김 1162
MySQL 호환 엔드포인트에 자동 커밋 추가 1163
대상 MySQL 호환 엔드포인트에서 외래 키 비활성화 1163
물음표로 대체된 문자 1164

'Bad event' 로그 항목 1164
MySQL 5.5로 변경 데이터 캡처 1164
Amazon RDS DB 인스턴스를 위한 이진 로그 보존 기간 증가 1164
로그 메시지: 소스 데이터베이스의 일부 변경 사항은 대상 데이터베이스에 적용될 때 영향이 전혀 없었습니다. 1165
오류: Identifier too long 1165
오류: 지원되지 않는 문자 집합으로 인해 필드 데이터 변환 실패 1165
오류: Codepage 1252 to UTF8 [120112] A field data conversion failed 1166
인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음 1166
PostgreSQL 관련 문제 해결 1168
잘리는 JSON 데이터 형식 1169
사용자 정의 데이터 형식 열이 올바르게 마이그레이션되지 않음 1169
오류: 생성하도록 선택된 스키마가 없음 1169
테이블 삭제와 업데이트가 CDC를 사용하여 복제되지 않음 1170
자르기 문이 전파되지 않고 있음 1170
PostgreSQL에서 DDL이 캡처되지 않음 1170
DDL 캡처를 위해 데이터베이스 객체가 생성되는 스키마 선택 1170
PostgreSQL로 마이그레이션한 후 Oracle 테이블 누락 1170
ReplicationSlotDiskUsage ETL 워크로드와 같은 긴 트랜잭션 중에는 restart_lsn이 증가하고 restart_lsn이 앞으로 이동하지 않습니다. 1171
소스로 보기를 사용한 작업에서 복사된 행이 없음 1171
Microsoft SQL Server 관련 문제 해결 1171
SQL Server 데이터베이스에서 변경 사항을 캡처하는 동안 발생하는 오류 1172
자격 증명 열 누락 1172
오류: SQL Server doesn't support publications 1172
대상에 변경 사항이 표시되지 않음 1173
파티션 간에 매핑된 비균일 테이블 1173
Amazon Redshift 관련 문제 해결 1174
다른 AWS 리전에서 Amazon Redshift 클러스터에 로드 1174
오류: Relation "awsdms_apply_exceptions" already exists 1174
awsdms_changes"로 시작하는 테이블 이름이 있는 오류 1174
dms.awsdms_changes000000000XXXX와 같은 이름과 함께 클러스터에 테이블 표시 1175
Amazon Redshift 사용에 필요한 권한 1175
Amazon Aurora MySQL 관련 문제 해결 1175
오류: CHARACTER SET UTF8 fields terminated by ',' enclosed by '"' lines terminated by '\n' 1175

SAP ASE 관련 문제 해결 1176
오류: LOB columns have NULL values when source has a composite unique index with NULL values 1176
IBM Db2 관련 문제 해결 1176
오류: Resume from timestamp is not supported Task 1176
지연 시간 문제 해결 1177
CDC 지연 시간 유형 1177
CDC 지연 시간이 발생하는 일반적인 원인 1178
지연 시간 문제 해결 1181
진단 지원 스크립트 작업 1195
Oracle 지원 스크립트 1197
SQL Server 지원 스크립트 1200
MySQL과 호환되는 지원 스크립트 1225
PostgreSQL 지원 스크립트 1227
진단 지원 AMI 작업 1229
새 AWS DMS 진단 Amazon EC2 인스턴스 시작 1230
IAM 역할 생성 1230
진단 테스트 실행 1231
다음 단계 1235
리전별 AMI ID 1235
Reference 1238
AWS DMS 데이터 형식 1238
릴리스 정보 1241
AWS DMS 3.5.2 릴리스 노트 1241
AWS DMS 3.5.1 릴리스 노트 1242
AWS DMS 3.5.0 베타 릴리스 노트 1252
AWS DMS 3.4.7 릴리스 노트 1257
AWS DMS 3.4.6 릴리스 노트 1265
AWS DMS 3.4.5 릴리스 노트 1271
AWS DMS 3.4.4 릴리스 노트 1274
AWS DMS 3.4.3 릴리스 노트 1276
AWS DMS 3.4.2 릴리스 노트 1278
AWS DMS 3.4.1 릴리스 노트 1279
AWS DMS 3.4.0 릴리스 노트 1280
AWS DMS 3.3.4 릴리스 노트 1282
AWS DMS 3.3.3 릴리스 노트 1283

문서 기록 1285
AWS 용어집 1289
..... mccxc

AWS Database Migration Service란 무엇인가요?

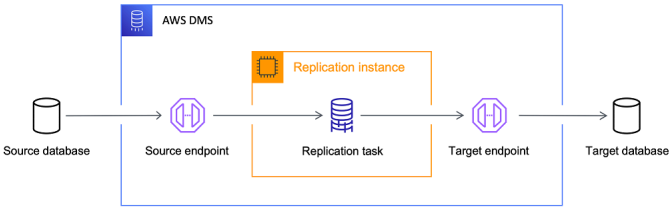
AWS Database Migration Service(AWS DMS)는 관계형 데이터베이스, 데이터 웨어하우스, NoSQL 데이터베이스 및 기타 유형의 데이터 스토어를 마이그레이션할 수 있는 클라우드 서비스입니다. AWS DMS를 사용하여 데이터를 AWS 클라우드로 마이그레이션하거나, 클라우드와 온프레미스 설정 조합 간에 마이그레이션할 수 있습니다.

AWS DMS를 사용하면 소스 데이터 스토어를 검색하고, 소스 스키마를 변환하고, 데이터를 마이그레이션할 수 있습니다.

- DMS Fleet Advisor를 사용하여 소스 데이터 인프라를 검색할 수 있습니다. 이 서비스는 온프레미스 데이터베이스 및 분석 서버에서 데이터를 수집하고 AWS 클라우드로 마이그레이션할 수 있는 서버, 데이터베이스, 스키마의 인벤토리를 구축합니다.
- 다른 데이터베이스 엔진으로 마이그레이션하려면 DMS Schema Conversion을 사용하면 됩니다. 이 서비스는 소스 스키마를 자동으로 평가한 후 새로운 대상 엔진으로 변환합니다. 또는 AWS Schema Conversion Tool(AWS SCT)을 로컬 PC에 다운로드하여 소스 스키마를 변환할 수도 있습니다.
- 소스 스키마를 변환하고 변환된 코드를 대상 데이터베이스에 적용한 후, AWS DMS를 사용하여 데이터를 마이그레이션할 수 있습니다. 마이그레이션을 한 번 수행하거나, 지속적인 변경 사항을 복제하여 소스와 대상을 동기화 상태로 유지할 수 있습니다. AWS DMS는 AWS 클라우드의 일부이기 때문에 AWS 서비스에서 제공하는 비용 효율성, 빠른 시장 출시 속도, 보안, 유연성을 누릴 수 있습니다.

기본 수준에서 보자면 AWS DMS는 복제 소프트웨어를 실행하는 AWS 클라우드의 서버입니다. 소스 및 대상 연결을 생성하여 데이터를 추출할 위치와 데이터를 로드할 위치를 AWS DMS에 지시합니다. 그 다음, 이 서버에서 실행되는 태스크를 예약하여 데이터를 이동합니다. AWS DMS는 테이블 및 관련 프라이머리 키가 대상에 없는 경우 이러한 항목을 생성합니다. 원하는 경우 대상 테이블을 직접 생성할 수 있습니다. 또는 AWS Schema Conversion Tool(AWS SCT)을 사용하여 대상 테이블, 인덱스, 보기, 트리거 등을 일부 또는 모두 생성할 수 있습니다.

아래 다이어그램에 AWS DMS 프로세스가 나와 있습니다.



참조

- AWS DMS를 지원하는 AWS 리전 – AWS DMS를 지원하는 AWS 리전 지원에 대한 자세한 내용은 AWS DMS 복제 인스턴스 사용 섹션을 참조하세요.
- 데이터베이스 마이그레이션 비용 – 데이터베이스 마이그레이션 비용에 대한 자세한 내용은 AWS Database Migration Service 요금 페이지 섹션을 참조하세요.
- AWS DMS 기능 및 이점 – AWS DMS 기능 및 이점에 대한 내용은 AWS Database Migration Service 기능 섹션을 참조하세요.
- 사용 가능한 데이터베이스 옵션 – Amazon Web Services Services에서 사용할 수 있는 다양한 데이터베이스 옵션에 대해 자세히 알아보려면 조직에 적합한 데이터베이스 선택을 참조하세요.

AWS DMS에서 수행하는 마이그레이션 작업

AWS DMS는 마이그레이션 프로젝트와 관련된 까다롭거나 지루한 수많은 태스크를 대신 처리합니다.

- 기존 솔루션에서는 용량 분석을 수행하고, 하드웨어와 소프트웨어를 구매하고, 시스템을 설치 및 관리하고, 설치를 테스트 및 디버깅해야 합니다. AWS DMS는 마이그레이션에 필요한 모든 하드웨어 및 소프트웨어의 배포, 관리, 모니터링을 자동으로 관리합니다. AWS DMS 구성 프로세스를 시작한 지 몇 분 이내에 마이그레이션을 실행할 수 있습니다.
- AWS DMS를 사용하면 실제 워크로드에 맞춰 필요에 따라 마이그레이션 리소스를 스케일 업(또는 스케일 다운)할 수 있습니다. 예를 들어, 추가 스토리지가 필요하다고 판단되면 할당된 스토리지를 쉽게 늘리고, 대게 몇 분 내로 마이그레이션을 다시 시작할 수 있습니다.
- AWS DMS는 사용한 만큼만 지불하는 모델을 사용합니다. 선결제 구매 비용과 지속적인 유지 관리 비용이 부과되는 기존 라이선스 모델과 달리, AWS DMS 리소스는 사용하는 동안에만 비용을 지불하면 됩니다.
- AWS DMS는 하드웨어 및 소프트웨어, 소프트웨어 패치 적용, 오류 보고 등을 비롯하여 마이그레이션 서버를 지원하는 모든 인프라를 자동으로 관리합니다.
- AWS DMS는 자동 장애 조치를 제공합니다. 어떤 이유로든 기본 복제 서버에 장애가 발생할 경우, 서비스 중단이 거의 발생하지 않은 채로 백업 복제 서버가 작업을 이어받을 수 있습니다.
- AWS DMS Fleet Advisor는 데이터 인프라의 인벤토리를 자동으로 생성합니다. 마이그레이션 대상을 식별하고, 마이그레이션을 계획하는 데 도움이 되는 보고서를 생성합니다.
- AWS DMS Schema Conversion은 소스 데이터 공급자의 마이그레이션 복잡성을 자동으로 평가합니다. 또한 데이터베이스 스키마와 코드 객체를 대상 데이터베이스와 호환되는 형식으로 변환한 다음, 변환된 코드를 적용합니다.

- AWS DMS는 현재 실행 중인 엔진보다 더 비용 효율적인 최신 데이터베이스 엔진으로 전환하는 데 도움이 될 수 있습니다. 예를 들어, AWS DMS는 Amazon Relational Database Service(RDS) 또는 Amazon Aurora에서 제공하는 관리형 데이터베이스 서비스를 활용하는 데 도움이 될 수 있습니다. 또는 Amazon Redshift에서 제공하는 관리형 데이터 웨어하우스 서비스, Amazon DynamoDB와 같은 NoSQL 플랫폼 또는 Amazon Simple Storage Service(S3)와 같은 저비용 스토리지 플랫폼으로 전환할 수 있습니다. 반대로 기존 인프라에서 벗어나 동일한 데이터베이스 엔진을 계속 사용하려는 경우에도, AWS DMS는 이러한 프로세스를 지원합니다.
- AWS DMS는 오늘날 가장 널리 사용되는 거의 모든 DBMS 엔진을 소스 엔드포인트로 지원합니다. 자세한 내용은 데이터 마이그레이션용 소스 섹션을 참조하세요.
- AWS DMS는 사용 가능한 대상 엔진을 광범위하게 제공합니다. 자세한 내용은 마이그레이션에 적합한 대상 섹션을 참조하세요.
- 지원되는 데이터 소스에서 지원되는 데이터 대상으로 마이그레이션할 수 있습니다. AWS DMS는 지원되는 엔진 간의 완전한 이기종 데이터 마이그레이션을 지원합니다.
- AWS DMS는 데이터 마이그레이션의 보안을 보장합니다. 저장 데이터는 AWS Key Management Service(AWS KMS) 암호화로 암호화됩니다. 마이그레이션하는 동안 Secure Socket Layer(SSL)를 사용하여 소스에서 대상으로 데이터를 이동할 때 처리 중인 데이터를 암호화할 수 있습니다.

AWS Database Migration Service의 작동 방식

AWS Database Migration Service (AWS DMS) 는 원본 데이터 저장소의 데이터를 대상 데이터 저장소로 마이그레이션하는 데 사용할 수 있는 웹 서비스입니다. 이러한 두 데이터 스토어는 엔드포인트라고 합니다. Oracle 데이터베이스에서 Oracle 데이터베이스로와 같이 동일한 데이터베이스 엔진을 사용하는 원본 및 대상 엔드포인트를 마이그레이션할 수 있습니다. Oracle 데이터베이스에서 PostgreSQL 데이터베이스로와 같이 다른 데이터베이스 엔진을 사용하는 원본 및 대상 엔드포인트를 마이그레이션할 수 있습니다. 사용할 AWS DMS 수 있는 유일한 요구 사항은 엔드포인트 중 하나가 AWS 서비스에 있어야 한다는 것입니다. 를 AWS DMS 사용하여 온프레미스 데이터베이스에서 다른 온프레미스 데이터베이스로 마이그레이션할 수 없습니다.

데이터베이스 마이그레이션 비용에 대한 자세한 내용은 AWS Database Migration Service 요금 페이지를 참조하세요.

더 잘 이해하려면 다음 항목을 참조하십시오. AWS DMS

주제

- 개괄적인 관점 AWS DMS
- 구성 요소: AWS DMS
- 출처: AWS DMS
- 대상: AWS DMS
- 에 대한 아마존 리소스 이름 (ARN) 생성 AWS DMS
- 다른 AWS DMSAWS 서비스와 함께 사용

개괄적인 관점 AWS DMS

데이터베이스 마이그레이션을 수행하려면 원본 데이터 저장소에 AWS DMS 연결하고 원본 데이터를 읽고 대상 데이터 저장소에서 사용할 수 있도록 데이터 형식을 지정합니다. 그런 다음 데이터를 대상 데이터 스토어에 로드합니다. 이러한 처리 대다수는 메모리에서 나타나지만, 큰 트랜잭션에는 디스크로의 일부 버퍼링이 필요할 수 있습니다. 캐시된 트랜잭션과 로그 파일도 디스크에 기록됩니다.

상위 수준에서 사용할 AWS DMS 때는 다음과 같이 하십시오.

- 네트워크 환경에서 마이그레이션에 적합한 데이터베이스를 검색합니다.
- 소스 데이터베이스 스키마와 대부분의 데이터베이스 코드 객체를 대상 데이터베이스와 호환되는 형식으로 자동 변환합니다.

- 복제 서버를 생성합니다.
- 데이터 스토어에 대한 연결 정보가 있는 소스 및 대상 엔드포인트를 생성합니다.
- 하나 이상의 마이그레이션 작업을 생성하여 원본과 대상 데이터 스토어 간 데이터를 마이그레이션 합니다.

작업은 세 가지 주요 단계로 구성되어 있습니다.

- 기존 데이터 마이그레이션(전체 로드)
- 캐시된 변경 사항 적용
- 지속적 복제(변경 데이터 캡처)

원본의 기존 데이터를 대상으로 이동하는 전체 부하 마이그레이션 중에는 원본 데이터 저장소의 테이블에서 대상 데이터 저장소의 테이블로 데이터를 AWS DMS 로드합니다. 전체 로드가 진행되는 동안 로드 중인 테이블에 적용된 변경 사항은 복제 서버에서 캐시되고, 이것은 캐시된 변경 사항입니다. 한 가지 중요한 점은 해당 테이블의 전체 로드가 시작될 때까지는 해당 테이블의 변경 내용을 캡처하지 AWS DMS 않는다는 점입니다. 즉, 변경 캡처가 시작되는 시점은 각 테이블별로 다릅니다.

지정된 테이블에 대한 전체 로드가 완료되면 해당 테이블에 대해 캐시된 변경 사항을 AWS DMS 즉시 적용하기 시작합니다. 테이블이 로드되고 캐시된 변경 사항이 적용되면 진행 중인 복제 단계를 위한 트랜잭션으로 변경 사항을 AWS DMS 수집하기 시작합니다. 트랜잭션에 테이블이 아직 완전히 로드되지 않은 경우 변경 사항은 복제 인스턴스에 로컬로 저장됩니다. 캐시된 변경 사항을 모두 AWS DMS 적용한 후에는 테이블이 트랜잭션 측면에서 일관성을 유지합니다. 이 시점에서 는 진행 중인 복제 단계로 AWS DMS 이동하여 변경 내용을 트랜잭션으로 적용합니다.

지속적 복제 단계가 시작되면, 트랜잭션의 백로그는 일반적으로 원본과 대상 데이터베이스 사이에서 일부 지연 시간을 유발합니다. 결국 마이그레이션은 이 트랜잭션 백로그를 통해 진행된 후 일정한 상태에 도달합니다. 이 시점에서 애플리케이션을 종료하고, 남은 트랙잭션을 대상에 적용하도록 허용하고, 애플리케이션을 불러오면 이제 대상 데이터베이스를 가리킵니다.

AWS DMS 데이터 마이그레이션을 수행하는 데 필요한 대상 스키마 객체를 생성합니다. 를 사용하여 AWS DMS 최소한의 접근 방식을 취하고 데이터를 효율적으로 마이그레이션하는 데 필요한 객체만 만들 수 있습니다. 이 방법을 사용하면 테이블, 기본 키 및 경우에 따라 고유 인덱스가 생성되지만 원본에서 데이터를 효율적으로 마이그레이션하는 데 필요하지 않은 다른 객체는 생성되지 않습니다. AWS DMS

또는 내에서 AWS DMS DMS 스키마 변환을 사용하여 원본 데이터베이스 스키마와 대부분의 데이터베이스 코드 개체를 대상 데이터베이스와 호환되는 형식으로 자동 변환할 수 있습니다. 이 변환에는 테

이블, 뷰, 저장 프로시저, 함수, 데이터 형식, 동의어 등이 포함됩니다. DMS Schema Conversion에서 자동으로 변환할 수 없는 모든 객체는 명확하게 표시됩니다. 마이그레이션을 완료하기 위해 이러한 객체를 수동으로 변환할 수 있습니다.

구성 요소: AWS DMS

이 섹션에서는 의 AWS DMS 내부 구성 요소와 이러한 구성 요소가 함께 작동하여 데이터 마이그레이션을 수행하는 방법에 대해 설명합니다. AWS DMS 의 기본 구성 요소를 이해하면 데이터를 더 효율적으로 마이그레이션할 수 있고 문제를 해결하거나 원인을 파악할 때 문제를 더 심도 있게 파악할 수 있습니다.

AWS DMS 마이그레이션은 마이그레이션할 데이터베이스 검색, 자동 스키마 변환, 복제 인스턴스, 원본 및 대상 엔드포인트, 복제 작업의 다섯 가지 구성 요소로 구성됩니다. 에서 필요한 복제 인스턴스, 엔드포인트 및 작업을 생성하여 AWS DMS 마이그레이션을 생성합니다. AWS 리전

데이터베이스 검색

DMS Fleet Advisor는 여러 데이터베이스 환경에서 데이터를 수집하여 데이터 인프라에 대한 인사이트를 제공합니다. DMS Fleet Advisor는 모든 컴퓨터에 설치할 필요 없이 하나 이상의 중앙 위치에서 온프레미스 데이터베이스 및 분석 서버의 데이터를 수집합니다. 현재 DMS Fleet Advisor는 Microsoft SQL Server, MySQL, Oracle 및 PostgreSQL 데이터베이스 서버를 지원합니다.

DMS Fleet Advisor는 네트워크에서 검색된 데이터를 기반으로 인벤토리를 작성합니다. 이 인벤토리를 검토하여 모니터링할 데이터베이스 서버 및 객체를 결정할 수 있습니다. 이러한 서버, 데이터베이스 및 스키마에 대한 세부 정보가 수집되면 의도한 데이터베이스 마이그레이션의 실행 가능성을 분석할 수 있습니다.

스키마 및 코드 마이그레이션

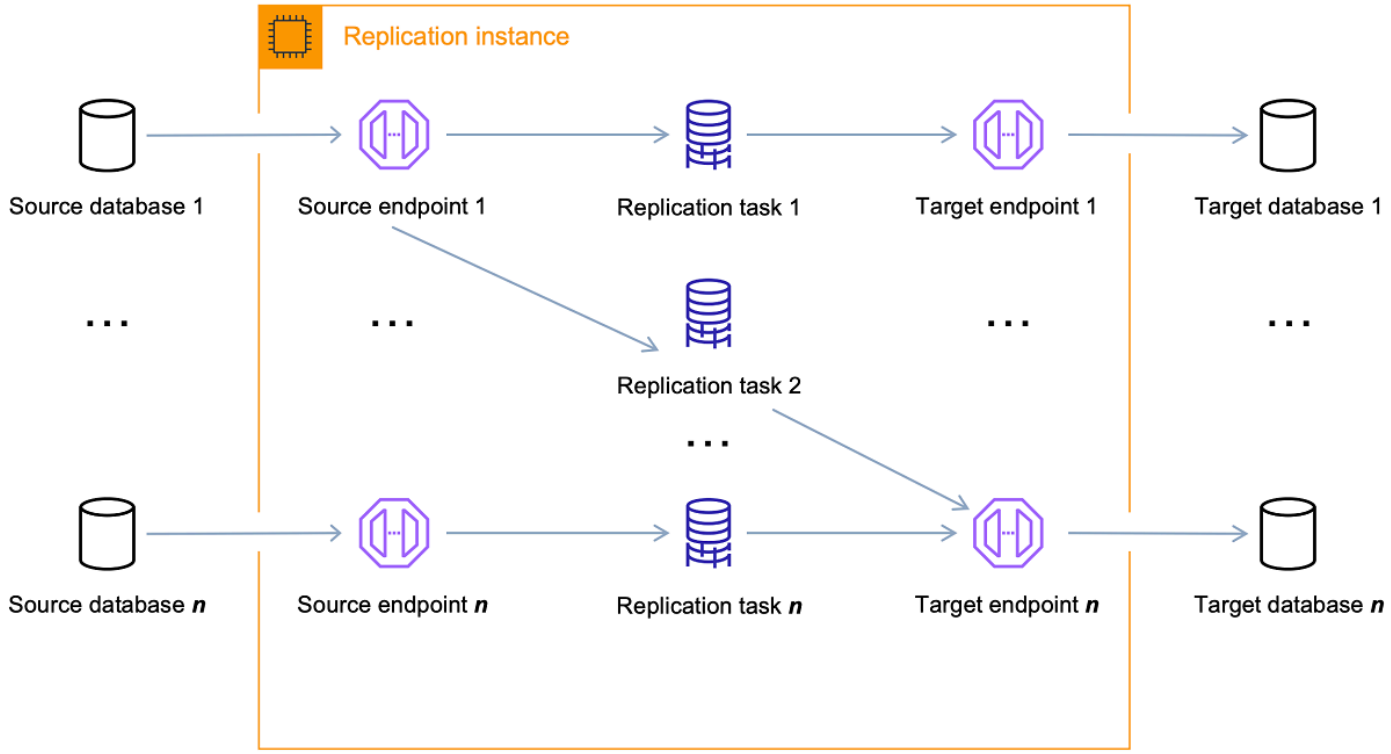
DMS 스키마 변환을 통해 서로 다른 유형의 데이터베이스 간 데이터베이스 마이그레이션을 보다 예측 AWS DMS 가능하게 할 수 있습니다. DMS Schema Conversion을 사용하여 소스 데이터 공급자의 마이그레이션 복잡성을 평가한 다음 이를 사용하여 데이터베이스 스키마 및 코드 객체를 변환할 수 있습니다. 그런 다음 변환된 코드를 대상 데이터베이스에 적용할 수 있습니다.

상위 수준에서 DMS Schema Conversion은 인스턴스 프로파일, 데이터 공급자, 마이그레이션 프로젝트 등 3가지 구성 요소를 사용하여 작동합니다. 인스턴스 프로파일은 네트워크 및 보안 설정을 지정합니다. 데이터 공급자는 데이터베이스 연결 보안 인증 정보를 저장합니다. 마이그레이션 프로젝트에는 데이터 공급자, 인스턴스 프로필 및 마이그레이션 규칙이 포함됩니다. AWS DMS 데이터 공급자와 인스턴스 프로필을 사용하여 데이터베이스 스키마와 코드 개체를 변환하는 프로세스를 설계합니다.

복제 인스턴스

상위 수준에서 AWS DMS 복제 인스턴스는 간단히 말해 하나 이상의 복제 작업을 호스팅하는 관리형 Amazon Elastic Compute Cloud (Amazon EC2) 인스턴스입니다.

다음 그림은 몇 가지 연결된 복제 작업을 실행하는 복제 인스턴스의 예시입니다.



마이그레이션의 특성과 복제 서버의 용량에 따라 단일 복제 인스턴스에서 하나 이상의 복제 작업을 호스팅할 수 있습니다. AWS DMS 다양한 복제 인스턴스를 제공하므로 사용 사례에 맞는 최적의 구성을 선택할 수 있습니다. 복제 인스턴스의 다양한 클래스에 대한 자세한 내용은 마이그레이션에 적합한 AWS DMS 복제 인스턴스 선택 섹션을 참조하세요.

AWS DMS Amazon EC2 인스턴스에 복제 인스턴스를 생성합니다. 소형 인스턴스 클래스 중에는 서비스를 테스트하거나 소규모 마이그레이션을 하기에 충분한 것이 있습니다. 마이그레이션에 대량의 테이블이 포함되어 있는 경우 또는 여러 동시 복제 작업을 실행해야 하는 경우, 대형 인스턴스 중 하나를 사용하는 것이 좋습니다. AWS DMS 는 상당량의 메모리와 CPU를 소모할 수 있으므로 이 접근 방식을 권장합니다.

선택하는 Amazon EC2 인스턴스 클래스에 따라 복제 인스턴스에는 50GB 또는 100GB의 데이터 스토리지가 포함되어 있습니다. 대개 이 정도 크기면 대부분 고객에게 충분합니다. 그러나 마이그레이션에 대규모 트랜잭션 또는 대량 데이터 변경 사항이 포함되어 있다면 기본 스토리지 할당을 늘려야 할 수 있습니다. 변경 데이터 캡처(CDC)로 인해 데이터가 디스크에 쓰일 수 있는데, 이는 대

상이 변경 사항을 쓰는 속도에 달려 있습니다. 로그 파일도 디스크에 기록되므로 로깅의 심각도가 높아지면 스토리지 사용량도 증가합니다.

AWS DMS 다중 AZ 배포를 사용하여 고가용성 및 장애 조치 지원을 제공할 수 있습니다. 다중 AZ 배포에서는 복제 인스턴스의 예비 복제본을 다른 가용 영역에 AWS DMS 자동으로 프로비저닝하고 유지 관리합니다. 기본 복제 인스턴스는 대기 복제본에 동기식으로 복제됩니다. 기본 복제 인스턴스에 장애가 발생하거나 무응답 상태가 되면 대기 복제본은 중단을 최소화하면서 실행 중이던 작업을 다시 시작합니다. 기본 복제본은 자신의 상태를 끊임없이 대기 복제본으로 복제하기 때문에 다중 AZ 배포는 약간의 성능 오버헤드를 발생시킵니다.

AWS DMS 복제 인스턴스에 대한 자세한 내용은 을 참조하십시오. AWS DMS 복제 인스턴스 사용

복제 인스턴스를 만들고 관리하는 대신 AWS DMS 서버리스를 사용하여 복제를 자동으로 AWS DMS 프로비저닝하도록 할 수 있습니다. 자세한 정보는 AWS DMS 서버리스로 작업하기을 참조하세요.

엔드포인트

AWS DMS 엔드포인트를 사용하여 원본 또는 대상 데이터 스토어에 액세스합니다. 구체적인 연결 정보는 데이터 스토어에 따라 다르지만, 일반적으로 사용자는 엔드포인트 생성 시 다음과 같은 정보를 제공합니다.

- 엔드포인트 유형 소스 또는 대상.
- 엔진 유형 - Oracle 또는 PostgreSQL 같은 데이터베이스 엔진의 유형.
- 서버 이름 — 연결할 AWS DMS 수 있는 서버 이름 또는 IP 주소.
- 포트 - 데이터베이스 서버 연결에 사용되는 포트 번호.
- 암호화 - 연결을 암호화하는 데 SSL(보안 소켓 계층)이 사용되는 경우, SSL 모드.
- 자격 증명 - 필수 액세스 권한이 있는 계정의 사용자 이름 및 암호.

콘솔을 사용하여 엔드포인트를 생성하는 경우 AWS DMS 콘솔에서 엔드포인트 연결을 테스트해야 합니다. 테스트에 성공해야 엔드포인트를 AWS DMS 작업에 사용할 수 있습니다. 연결 정보와 마찬가지로 구체적인 테스트 기준은 엔진 유형에 따라 다릅니다. 일반적으로 AWS DMS 에서는 주어진 서버 이름 및 포트에 데이터베이스가 있는지, 제공된 자격 증명을 마이그레이션을 수행하는 데 필요한 권한이 있는 데이터베이스에 연결하는 데 사용할 수 있는지 확인합니다. 연결 테스트에 성공하면 나중에 작업 구성 중에 사용할 스키마 정보를 AWS DMS 다운로드하여 저장합니다. 스키마 정보에는 테이블 정의, 기본 키 정의, 고유 키 정의 등이 포함될 수 있습니다.

복수의 복제 작업에서 단일 엔드포인트를 사용할 수 있습니다. 예를 들어, 동일한 소스 데이터베이스에서 호스팅되는 논리적으로 구분된 애플리케이션이 두 개 있는데, 사용자가 이를 서로 구분하여 마이그레이션하고자 하는 경우가 있을 수 있습니다. 이 경우 각 애플리케이션 테이블 집합에 대해

하나씩 두 개의 복제 작업을 생성합니다. 두 작업에서 동일한 AWS DMS 엔드포인트를 사용할 수 있습니다.

엔드포인트 설정을 사용하여 엔드포인트의 동작을 사용자 지정할 수 있습니다. 엔드포인트 설정은 로깅 세부 정보, 파일 크기, 기타 파라미터 등 다양한 동작을 제어할 수 있습니다. 데이터 스토어 엔진 유형마다 사용 가능한 엔드포인트 설정이 다릅니다. 데이터 스토어의 소스 혹은 대상 섹션에서 각 데이터 스토어의 특정 엔드포인트 설정을 찾을 수 있습니다. 지원되는 원본 및 대상 데이터 스토어 목록은 출처: AWS DMS 및 대상: AWS DMS 섹션을 참조하세요.

AWS DMS 엔드포인트에 대한 자세한 내용은 을 참조하십시오AWS DMS 엔드포인트 작업.

복제 작업

AWS DMS 복제 작업을 사용하여 원본 엔드포인트에서 대상 엔드포인트로 데이터 세트를 이동합니다. 복제 작업 생성은 마이그레이션을 시작하기 전에 수행해야 할 최종 단계입니다.

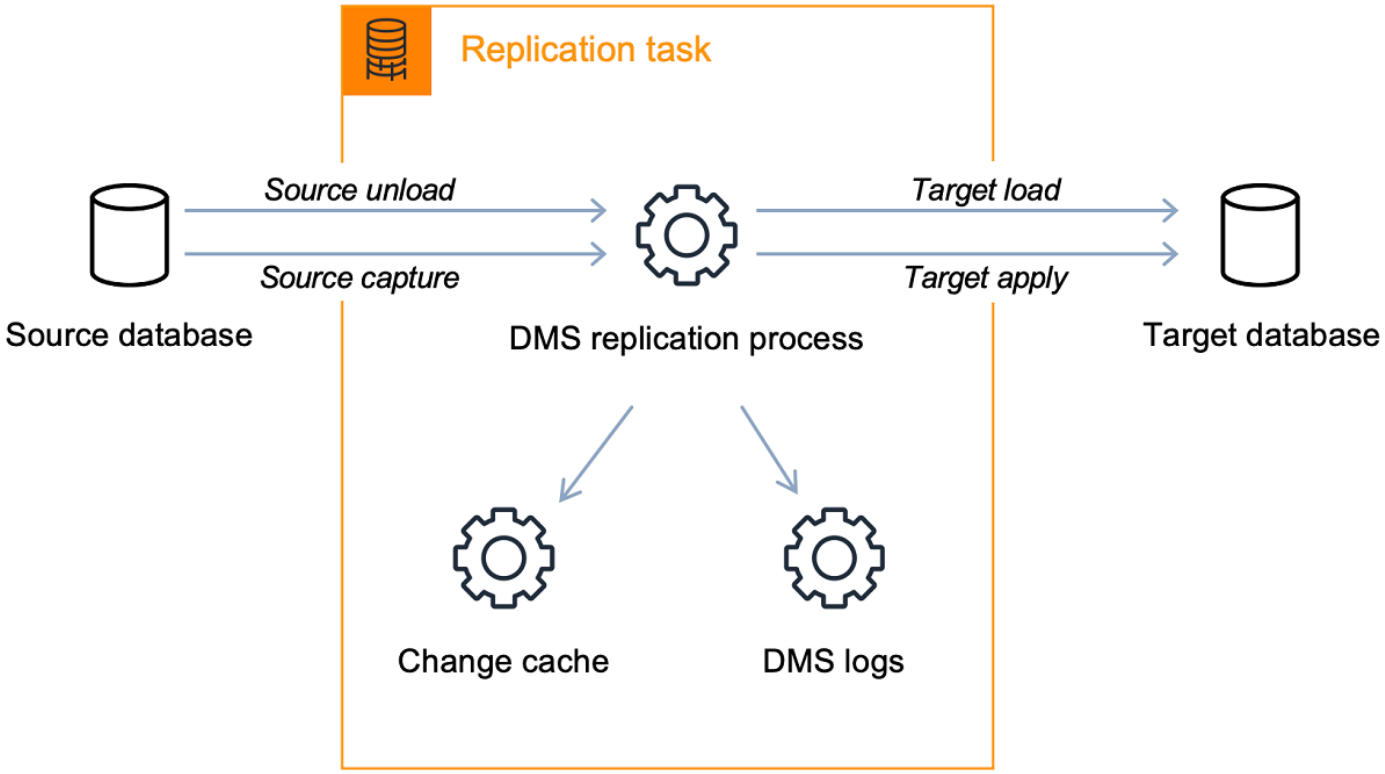
복제 작업을 생성할 때 다음 작업 설정을 지정합니다.

- 복제 인스턴스 - 태스크를 호스팅하고 실행할 인스턴스
- [소스 엔드포인트]
- [타겟 엔드포인트]
- 마이그레이션 유형 옵션은 다음과 같습니다. 마이그레이션 유형 옵션에 대한 자세한 설명은 작업 생성을 참조하세요.
  - 전체 로드(기존 데이터 마이그레이션) - 기존 데이터를 복사할 수 있을 정도로 긴 중단을 감당할 수 있는 경우 이 옵션이 적합합니다. 이 옵션은 간단히 원돈 데이터베이스에서 대상 데이터베이스로 데이터를 마이그레이션하여 필요할 경우 테이블을 생성합니다.
  - 전체 로드 및 CDC(기존 데이터를 마이그레이션하고 진행 중 변경 사항 복제) - 이 옵션을 선택하면 소스에서 변경 사항을 캡처하는 동안 전체 데이터 로드를 수행합니다. 전체 로드가 완료되고 나면, 캡처된 변경 사항을 대상에 적용합니다. 결과적으로 변경 사항 적용은 안정적인 상태에 도달합니다. 이 시점에서 애플리케이션을 종료할 수 있고, 남은 변경 사항이 대상으로 흐르도록 한 후 대상을 나타내는 애플리케이션을 다시 시작할 수 있습니다.
  - CDC 전용(데이터 변경 사항만 복제) - 경우에 따라서는 AWS DMS이외의 방법을 사용하여 기존 데이터를 복사하는 것이 더 효율적일 수 있습니다. 예를 들어, 같은 유형의 마이그레이션에서 기본 내보내기 및 가져오기 도구를 사용하면 대량 데이터를 로드하기에 더 효과적일 수 있습니다. 이 경우 대량 로드를 시작할 때부터 변경 내용을 AWS DMS 복제하여 원본 및 대상 데이터베이스를 동기화하고 유지하는 데 사용할 수 있습니다.
- 대상 테이블 준비 모드 옵션은 다음과 같습니다. 대상 테이블 모드에 관한 완전한 설명을 보려면 작업 생성 섹션을 참조하세요.

  - 아무 것도 하지 마세요. 대상 테이블이 타겟에 미리 AWS DMS 생성되었다고 가정합니다.
  - 타겟에 테이블 삭제 — 대상 테이블을 AWS DMS 삭제하고 다시 생성합니다.
  - 자르기 - 대상에 테이블을 생성했다면 AWS DMS는 마이그레이션을 시작하기 전에 이 테이블을 자릅니다. 테이블이 없는 상태에서 이 옵션을 선택하면 누락된 테이블이 모두 AWS DMS 생성됩니다.
- LOB 모드 옵션은 다음과 같습니다. LOB 모드에 관한 완전한 설명을 보려면 태스크의 소스 데이터베이스에 대한 LOB 지원 설정 AWS DMS 섹션을 참조하세요.
  - LOB 열 포함 안 함 - LOB 열이 마이그레이션에서 제외됩니다.
  - 전체 LOB 모드 - 크기에 관계없이 전체 LOB를 마이그레이션합니다. AWS DMS 최대 LOB 크기 매개 변수에 의해 제어되는 청크 단위로 LOB를 마이그레이션합니다. 이 모드는 제한적 LOB 모드를 사용하는 것보다 더 느립니다.
  - 제한적 LOB 모드 - LOB를 최대 LOB 크기 파라미터에 지정된 값으로 자릅니다. 이 모드는 전체 LOB 모드를 사용하는 것보다 더 빠릅니다.
- 테이블 매핑 - 마이그레이션할 테이블과 마이그레이션 방법을 나타냅니다. 자세한 정보는 작업 설정을 지정하기 위한 테이블 매핑 사용을 참조하세요.
- 데이터 변환은 다음과 같습니다. 데이터 변환에 대한 자세한 내용은 JSON을 사용하여 테이블 선택 및 변환 지정 섹션을 참조하세요.
  - 스키마, 테이블 및 열 이름 변경.
  - 테이블스페이스 이름 변경(Oracle 대상 엔드포인트의 경우).
  - 대상에서 기본 키와 고유한 인덱스 정의.
- 데이터 유효성 검사
- 아마존 CloudWatch 로깅

해당 작업을 사용하여 소스 엔드포인트에서 대상 엔드포인트로 데이터를 마이그레이션하고 나면 복제 인스턴스에서 작업 처리가 완료됩니다. 마이그레이션할 테이블 및 스키마와 모든 특수 처리(로깅 요구 사항, 제어 테이블 데이터 및 오류 처리 등)를 지정합니다.

개념적으로 AWS DMS 복제 작업은 다음 다이어그램과 같이 서로 다른 두 가지 기능을 수행합니다.



전체 로드 프로세스는 이해하기 쉽습니다. 데이터는 대량 추출 방식으로 소스에서 추출되어 대상으로 직접 로드됩니다. AWS DMS 콘솔의 고급 설정에서 병렬로 추출 및 로드할 테이블 수를 지정할 수 있습니다.

AWS DMS 작업에 대한 자세한 내용은 을 참조하십시오AWS DMS 작업 사용.

지속적인 복제 또는 변경 데이터 캡처(CDC)

데이터를 대상으로 마이그레이션하는 동안 AWS DMS 작업을 사용하여 원본 데이터 저장소의 진행 중인 변경 사항을 캡처할 수도 있습니다. 원본 엔드포인트에서 진행 중인 변경 내용을 복제할 때 AWS DMS 사용하는 변경 사항 캡처 프로세스는 데이터베이스 엔진의 네이티브 API를 사용하여 데이터베이스 로그의 변경 내용을 수집합니다.

CDC 프로세스에서 복제 작업은 인 메모리 버퍼를 사용해 전송 중인 데이터를 유지하는 방식으로 원본에서 대상으로 변경 사항을 스트림하도록 설계되어 있습니다. 어떤 이유로든 인 메모리 버퍼가 남지 않게 되면 복제 작업은 보류 중인 변경 사항을 디스크의 변경 사항 캐시로 유출합니다. 예를 들어 원본에서 변경 내용을 캡처하는 AWS DMS 속도가 대상에 적용할 수 있는 속도보다 빠른 경우 이러한 상황이 발생할 수 있습니다. 이 경우 사용자는 작업의 대상 지연 시간이 작업의 소스 지연 시간을 초과하는지 확인해야 합니다.

AWS DMS 콘솔에서 작업으로 이동한 다음 작업 모니터링 탭을 열면 이를 확인할 수 있습니다. CDC LatencyTarget 및 CDC LatencySource 그래프는 페이지 하단에 표시됩니다. 대상 지연 시간

을 표시하는 작업이 있다면 애플리케이션 속도를 높이는 데 필요한 대상 엔드포인트 튜닝이 있을 가능성이 있습니다.

복제 태스크에서도 앞서 설명한 태스크 로그용 스토리지를 사용합니다. 복제 인스턴스로 사전 구성되어 제공되는 디스크 공간은 대개 로깅 및 유출된 변경 사항에 충분한 크기입니다. 추가 디스크 공간이 필요하다면(예: 마이그레이션 문제를 조사하기 위해 세부 디버깅을 사용하는 경우) 복제 인스턴스를 수정하여 더 많은 공간을 할당할 수 있습니다.

출처: AWS DMS

AWS DMS 피처마다 다른 원본 데이터 저장소를 사용할 수 있습니다. 다음 섹션에는 각 AWS DMS 기능에 지원되는 소스 데이터 저장소 목록이 포함되어 있습니다.

주제

- 데이터 마이그레이션에 사용할 수 있는 소스 엔드포인트
- DMS Fleet Advisor가 지원하는 소스 데이터베이스
- DMS Schema Conversion이 지원하는 소스 데이터 공급자
- DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자

데이터 마이그레이션에 사용할 수 있는 소스 엔드포인트

AWS DMS를 사용하여 다음 데이터 스토어를 데이터 마이그레이션용 소스 엔드포인트로 사용할 수 있습니다.

온프레미스 및 EC2 인스턴스 데이터베이스

- Enterprise, Standard, Standard One 및 Standard Two 에디션용 Oracle 버전 10.2 이상(버전 10.x의 경우), 11g 및 12.2 이하, 18c 및 19c
- Microsoft SQL Server 버전 2005, 2008, 2008R2, 2012, 2014, 2016, 2017, 2019, 2022.
  - 엔터프라이즈, 스탠다드, 워크그룹, 개발자 및 웹 에디션은 전체 로드 복제를 지원합니다.
  - 엔터프라이즈, 스탠다드 (버전 2016 이상) 및 개발자 에디션은 전체 로드 외에도 CDC (지속적인) 복제를 지원합니다.
  - Express 에디션은 지원되지 않습니다.
- MySQL 버전 5.5, 5.6, 5.7 및 8.0.

> **Note**
>
> MySQL 8.0을 소스로 지원하는 것은 트랜잭션 페이로드가 압축된 경우를 제외하고 AWS DMS 버전 3.4.0 이상에서 사용할 수 있습니다. MySQL 8.0용 Google Cloud Support는 AWS DMS 버전 3.4.6 이상에서 사용할 수 있습니다.

- MariaDB(MySQL 호환 데이터 소스로 지원됨) 버전 10.0(버전 10.0.24 이상만 해당), 10.2, 10.3, 10.4, 10.5 및 10.6.

> **Note**
>
> MariaDB를 소스로 지원하는 것은 MySQL이 지원되는 모든 AWS DMS 버전에서 사용할 수 있습니다.

- PostgreSQL 버전 9.4 이상(버전 9.x의 경우), 10.x, 11.x, 12.x, 13.x 14.x 및 15.x.

> **Note**
>
> AWS DMS 버전 3.5.1 이상의 PostgreSQL 버전 15.x만 지원합니다.

- MongoDB 버전 3.x, 4.0, 4.2, 4.4, 5.0, 6.0

> **Note**
>
> AWS DMS 버전 3.5.0 이상은 3.6 이전의 MongoDB 버전을 지원하지 않습니다.

- SAP Adaptive Server Enterprise(ASE) 버전 12.5, 15, 15.5, 15.7, 16 이상
- Linux, UNIX 및 Windows용 IBM Db2(Db2 LUW) 버전:
  - 버전 9.7, 모든 수정 팩
  - 버전 10.1, 모든 수정 팩
  - 버전 10.5, 수정 팩 5를 제외한 모든 수정 팩
  - 버전 11.1, 모든 수정 팩
  - 버전 11.5, 수정 팩 제로만 포함된 Mod(0~8)
- IBM Db2 for z/OS 버전 12

서드 파티 관리형 데이터베이스 서비스:

- Microsoft Azure SQL Database
- Microsoft Azure PostgreSQL Flexible Server 버전 11.2, 12.15, 13.11, 14.8 및 15.3.
- Microsoft Azure MySQL Flexible Server 버전 5.7 및 8.
- Google Cloud for MySQL 버전 5.6, 5.7 및 8.0.
- Google Cloud for PostgreSQL 버전 9.6, 10, 11, 12, 13, 14 및 15.
- OCI MySQL Heatwave 버전 8.0.34.

Amazon RDS 인스턴스 데이터베이스 및 Amazon Simple Storage Service(Amazon S3)

- Enterprise, Standard, Standard One 및 Standard Two 에디션용 Oracle 버전 11g(버전 11.2.0.4 이상) 및 12.2 이하, 18c 및 19c
- Enterprise, Standard, Workgroup 및 Developer 에디션용 Microsoft SQL Server 버전2012, 2014, 2016, 2017, 2019, 2022

> **Note**
>
> AWS DMS SQL 서버 익스프레스를 지원하지 않습니다. Web 에디션은 전체 로드 전용 복제에만 지원됩니다.

- MySQL 버전 5.5, 5.6, 5.7 및 8.0.

> **Note**
>
> MySQL 8.0을 소스로 지원하는 것은 트랜잭션 페이로드가 압축된 경우를 제외하고 AWS DMS 버전 3.4.0 이상에서 사용할 수 있습니다.

- MariaDB(MySQL 호환 데이터 소스로 지원됨) 버전 10.0.24~10.0.28, 10.2, 10.3, 10.4, 10.5 및 10.6.

> **Note**
>
> MariaDB를 소스로 지원하는 것은 MySQL이 지원되는 모든 AWS DMS 버전에서 사용할 수 있습니다.

- PostgreSQL 버전 10.x, 11.x, 12.x, 13.x, 14.x 및 15.x.

> **Note**
>
> AWS DMS 버전 3.5.1 이상에서는 PostgreSQL 15.x만 지원합니다.

- MySQL 호환 Amazon Aurora(MySQL 호환 데이터 소스로 지원됨)
- PostgreSQL 호환 Amazon Aurora(PostgreSQL 호환 데이터 소스로 지원됨)
- Amazon S3
- 아마존 DocumentDB (MongoDB 호환 포함) 버전 3.6, 4.0, 5.0
- IBM Db2 LUW용 Amazon RDS.

특정 소스로 작업하는 방법에 대한 자세한 내용은 엔드포인트 작업을 참조하십시오. AWS DMS

지원되는 대상 엔드포인트에 대한 자세한 내용은 데이터 마이그레이션에 사용할 수 있는 대상 엔드포인트 섹션을 참조하세요.

DMS Fleet Advisor가 지원하는 소스 데이터베이스

DMS Fleet Advisor는 다음 소스 데이터베이스를 지원합니다.

- Microsoft SQL Server 버전 2012 및 2019 이하
- MySQL 버전 5.6 및 8 이하
- Oracle 버전 11g 릴리스 2 및 12c 이하, 19c 및 21c
- PostgreSQL 버전 9.6 및 13 이하

특정 소스를 사용한 작업에 대한 자세한 내용은 AWS DMS Fleet Advisor용 데이터베이스 사용자 생성 섹션을 참조하세요.

DMS Fleet Advisor가 대상 권장 사항을 생성하는 데 사용하는 데이터베이스의 목록은 DMS Fleet Advisor가 지원하는 대상 섹션을 참조하세요.

DMS Schema Conversion이 지원하는 소스 데이터 공급자

DMS Schema Conversion은 다음 데이터 공급자를 마이그레이션 프로젝트의 소스로 지원합니다.

- Microsoft SQL Server 버전 2008 R2, 2012, 2014, 2016, 2017 및 2019
- Oracle 버전 10.2 이상, 11g 및 12.2 이하, 18c 및 19c, Oracle Data Warehouse

- PostgreSQL 버전 9.2 이상
- MySQL 버전 5.5 이상

소스 데이터베이스 공급자는 온프레미스 또는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서 실행되는 자체 관리형 엔진일 수 있습니다.

특정 소스를 사용한 작업에 대한 자세한 내용은 DMS Schema Conversion에서 소스 데이터 공급자 생성 섹션을 참조하세요.

지원되는 대상 데이터베이스에 관한 자세한 내용은 DMS Schema Conversion이 지원하는 대상 데이터 공급자 섹션을 참조하세요.

AWS Schema Conversion Tool (AWS SCT) 는 DMS 스키마 변환보다 더 많은 소스 및 대상 데이터베이스를 지원합니다. AWS SCT 지원하는 데이터베이스에 대한 자세한 내용은 무엇입니까를 AWS Schema Conversion Tool 참조하십시오.

DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자

다음 데이터 공급자를 동종 데이터 마이그레이션의 소스로 사용할 수 있습니다.

- MySQL 버전 5.7 이상
- MariaDB 버전 10.2 이상
- PostgreSQL 버전 10.4~14.x.
- 몽고DB 버전 4.x, 5.x, 6.0
- 아마존 DocumentDB 버전 3.6, 4.0, 5.0

소스 데이터베이스 공급자는 온프레미스 또는 Amazon EC2 인스턴스에서 실행되는 자체 관리형 엔진일 수 있습니다. 또한 Amazon RDS DB 인스턴스를 소스 데이터 공급자로 사용할 수 있습니다.

특정 소스를 사용한 작업에 대한 자세한 내용은 에서 동종 데이터 마이그레이션을 위한 소스 데이터 공급자 생성 AWS DMS 섹션을 참조하세요.

지원되는 대상 데이터베이스에 관한 자세한 내용은 DMS 동종 데이터 마이그레이션이 지원하는 대상 데이터 공급자 섹션을 참조하세요.

대상: AWS DMS

AWS DMS 기능마다 다른 대상 데이터 저장소를 사용할 수 있습니다. 다음 섹션에는 각 AWS DMS 기능에 지원되는 대상 데이터 저장소 목록이 포함되어 있습니다.

주제

- 데이터 마이그레이션에 사용할 수 있는 대상 엔드포인트
- DMS Fleet Advisor가 지원하는 대상 데이터베이스
- DMS Schema Conversion이 지원하는 대상 데이터 공급자
- DMS 동종 데이터 마이그레이션이 지원하는 대상 데이터 공급자

데이터 마이그레이션에 사용할 수 있는 대상 엔드포인트

AWS DMS를 사용하여 다음 데이터 스토어를 데이터 마이그레이션용 대상 엔드포인트로 사용할 수 있습니다.

온프레미스 및 Amazon EC2 인스턴스 데이터베이스

- Enterprise, Standard, Standard One 및 Standard Two 에디션용 Oracle 버전 10g, 11g, 12c, 18c 및 19c
- Enterprise, Standard, Workgroup 및 Developer 에디션용 Microsoft SQL Server 버전 2005, 2008, 2008R2, 2012, 2014, 2016, 2017, 2019, 2022.

> **Note**
>
> AWS DMS SQL Server 웹 및 익스프레스 에디션은 지원하지 않습니다.

- MySQL 버전 5.5, 5.6, 5.7 및 8.0.
- MariaDB(MySQL 호환 데이터 소스로 지원됨) 버전 10.0.24~10.0.28, 10.2, 10.3, 10.4, 10.5 및 10.6.

> **Note**
>
> MariaDB를 대상으로 지원하는 것은 MySQL이 지원되는 모든 AWS DMS 버전에서 사용할 수 있습니다.

- PostgreSQL 버전 9.4 이상(버전 9.x의 경우), 10.x, 11.x, 12.x, 13.x, 14.x 및 15.x.

> **Note**
>
> AWS DMS 버전 3.5.1 이상에서는 PostgreSQL 15.x만 지원합니다.

- SAP Adaptive Server Enterprise(ASE) 버전 15, 15.5, 15.7, 16 이상
- Redis 버전 6.x

아마존 RDS 인스턴스 데이터베이스, 아마존 레드시프트, 아마존 레드시프트 서버리스, 아마존 다이나모DB, 아마존 S3, 아마존 서비스, 아마존 레디스용 아마존 OpenSearch , 아마존 키네시스 데이터 스트림, ElastiCache 아마존 DocumentDB, 아마존 넵튠, 아파치 카프카

- Enterprise, Standard, Standard One 및 Standard Two 에디션용 Oracle 버전 11g(버전 11.2.0.3.v1 이상), 12c, 18c 및 19c
- Enterprise, Standard, Workgroup 및 Developer 에디션용 Microsoft SQL Server 버전2012, 2014, 2016, 2017, 2019, 2022

> **Note**
>
> AWS DMS SQL Server 웹 및 익스프레스 에디션은 지원하지 않습니다.

- MySQL 버전 5.5, 5.6, 5.7 및 8.0.
- MariaDB(MySQL 호환 데이터 소스로 지원됨) 버전 10.0.24~10.0.28, 10.2, 10.3, 10.4, 10.5 및 10.6.

> **Note**
>
> MariaDB를 대상으로 지원하는 것은 MySQL이 지원되는 모든 AWS DMS 버전에서 사용할 수 있습니다.

- PostgreSQL 버전 10.x, 11.x, 12.x, 13.x, 14.x 및 15.x

> **Note**
>
> AWS DMS 버전 3.5.1 이상에서는 PostgreSQL 15.x만 지원합니다.

- IBM Db2 LUW 버전 11.1 및 11.5
- Amazon Aurora MySQL 호환 버전

- Amazon Aurora PostgreSQL 호환 에디션
- Amazon Aurora Serverless v2
- Amazon Redshift
- Amazon Redshift Serverless
- Amazon S3
- Amazon DynamoDB
- 아마존 OpenSearch 서비스
- 아마존 포 ElastiCache 레디스용
- Amazon Kinesis Data Streams
- Amazon DocumentDB(MongoDB 호환)
- Amazon Neptune
- Apache Kafka – Amazon Managed Streaming for Apache Kafka(Amazon MSK) 및 자체 관리형 Apache Kafka
- Aurora PostgreSQL(버전 15.3/14.8 이상)용 Babelfish(버전 3.2.0 이상)

특정 대상 작업에 대한 자세한 내용은 AWS DMS 엔드포인트 사용을 참조하십시오.

지원되는 소스 엔드포인트에 대한 자세한 내용은 데이터 마이그레이션에 사용할 수 있는 소스 엔드포인트 섹션을 참조하세요.

DMS Fleet Advisor가 지원하는 대상 데이터베이스

DMS Fleet Advisor는 다음 대상 데이터베이스의 최신 버전을 사용하여 대상 권장 사항을 생성합니다.

- Amazon Aurora MySQL
- Amazon Aurora PostgreSQL
- Amazon RDS for MySQL
- Amazon RDS for Oracle
- Amazon RDS for PostgreSQL
- Amazon RDS for SQL Server

DMS Fleet Advisor의 대상 권장 사항에 대한 자세한 내용은 AWS DMS Fleet Advisor 대상 권장 기능 사용 섹션을 참조하세요.

지원되는 소스 데이터베이스에 대한 자세한 내용은 DMS Fleet Advisor가 지원하는 소스 데이터베이스 단원을 참조하세요.

DMS Schema Conversion이 지원하는 대상 데이터 공급자

DMS Schema Conversion은 다음 데이터 공급자를 마이그레이션 프로젝트의 대상으로 지원합니다.

- Amazon Aurora MySQL 8.0.23
- Amazon Aurora PostgreSQL 14.5
- Amazon RDS for MySQL 8.0.23
- Amazon RDS for PostgreSQL 14.x
- Amazon Redshift

특정 대상을 사용한 작업에 대한 자세한 내용은 DMS Schema Conversion에서 대상 데이터 공급자 생성 섹션을 참조하세요.

지원되는 소스 데이터베이스에 대한 자세한 내용은 DMS Schema Conversion이 지원하는 소스 데이터 공급자 단원을 참조하세요.

DMS 동종 데이터 마이그레이션이 지원하는 대상 데이터 공급자

다음 데이터 공급자를 동종 데이터 마이그레이션의 대상으로 사용할 수 있습니다.

- Amazon Aurora MySQL 버전 5.7 이상
- Amazon Aurora PostgreSQL 버전 10.4~14.x
- Amazon Aurora Serverless v2
- Amazon RDS for MySQL 버전 5.7 이상
- Amazon RDS for MariaDB 버전 10.2 이상
- Amazon RDS for PostgreSQL 버전 10.4~14.x
- 아마존 DocumentDB 버전 4.0, 5.0 및 DocumentDB 엘라스틱 클러스터

특정 대상을 사용한 작업에 대한 자세한 내용은 AWS DMS에서 동종 데이터 마이그레이션을 위한 대상 데이터 공급자 생성 섹션을 참조하세요.

지원되는 소스 데이터베이스에 대한 자세한 내용은 DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자 단원을 참조하세요.

에 대한 아마존 리소스 이름 (ARN) 생성 AWS DMS

AWS CLI 또는 AWS DMS API를 사용하여 데이터베이스 마이그레이션을 자동화하는 경우 Amazon 리소스 이름 (ARN) 을 사용합니다. Amazon Web Services에서 생성되는 각 리소스는 고유한 식별자인 ARN으로 식별됩니다. AWS CLI 또는 AWS DMS API를 사용하여 데이터베이스 마이그레이션을 설정하는 경우 작업하려는 리소스의 ARN을 제공합니다.

AWS DMS 리소스의 ARN은 다음 구문을 사용합니다.

`arn:aws:dms:`*`region`*`:`*`account number`*`:`*`resourcetype`*`:`*`resourcename`*

이 구문에는 다음이 적용됩니다.

- *`region`* AWS DMS 리소스가 생성된 AWS 리전 위치의 ID입니다 (예`us-west-2`:).

  다음 표에는 AWS 리전 ARN을 구성할 때 사용해야 하는 이름과 값이 나와 있습니다.

리전	명칭
아시아 태평양(도쿄) 리전	ap-northeast-1
아시아 태평양(서울) 리전	ap-northeast-2
아시아 태평양(뭄바이) 리전	ap-south-1
아시아 태평양(싱가포르) 리전	ap-southeast-1
아시아 태평양(시드니) 리전	ap-southeast-2
캐나다(중부) 리전	ca-central-1
중국(북경) 지역	cn-north-1
중국(닝샤) 리전	cn-northwest-1
유럽(스톡홀름) 리전	eu-north-1
유럽(밀라노) 리전	eu-south-1
EU(프랑크푸르트) 리전	eu-central-1

리전	명칭
유럽(아일랜드) 리전	eu-west-1
EU(런던) 리전	eu-west-2
EU(파리) 리전	eu-west-3
South America (São Paulo) Region	sa-east-1
미국 동부(버지니아 북부) 리전	us-east-1
US East (Ohio) Region	us-east-2
US West (N. California) Region	us-west-1
미국 서부(오레곤) 리전	us-west-2

- *account number*는 대시가 생략된 계정 번호입니다. 계정 번호를 찾으려면 http://aws.amazon.com 에서 AWS 계정에 로그인하고 내 계정/콘솔을 선택한 다음 내 계정을 선택합니다.
- *resourcetype*리소스 AWS DMS 유형입니다.

  다음 표에는 특정 리소스에 대한 ARN을 구성할 때 사용할 리소스 유형이 나와 있습니다 AWS DMS .

AWS DMS 리소스 유형	ARN 형식
복제 인스턴스	`arn:aws:dms:` *region*`:` *account*`:rep:` *resourcename*
엔드포인트	`arn:aws:dms:` *region*`:`*account*`:endpoint:` *resourcename*
복제 작업	`arn:aws:dms:` *region*`:`*account*`:task:`*resourcename*
서브넷 그룹	`arn:aws:dms:` *region*`:`*account*`:subgrp:`*resourcename*

- *resourcename*리소스에 할당된 AWS DMS 리소스 이름입니다. 이것은 생성된 임의 문자입니다.

다음 표에는 AWS DMS 리소스용 ARN의 예가 나와 있습니다. 여기서는 123456789012라는 AWS 계정을 가정합니다. 이 계정은 미국 동부(버지니아 북부) 리전에서 생성되었고 리소스 이름이 있습니다.

리소스 유형	샘플 ARN
복제 인스턴스	arn:aws:dms:us-east-1:123456789012:rep:QLXQZ64MH7CXF4QCQMGRVYVXAI
엔드포인트	arn:aws:dms:us-east-1:123456789012:endpoint:D3HMZ2IGUCGFF3NTAXUXGF6S5A
복제 작업	arn:aws:dms:us-east-1:123456789012:task:2PVREMWNPGYJCVU2IBPTOYTIV4
서브넷 그룹	arn:aws:dms:us-east-1:123456789012:subgrp:test-tag-grp

다른 AWS DMSAWS 서비스와 함께 사용

다른 여러 AWS 서비스와 AWS DMS 함께 사용할 수 있습니다.

- Amazon EC2 인스턴스 또는 Amazon RDS DB 인스턴스를 데이터 마이그레이션 대상으로 사용할 수 있습니다.
- AWS Schema Conversion Tool (AWS SCT) 를 사용하여 소스 스키마와 SQL 코드를 동등한 대상 스키마와 SQL 코드로 변환할 수 있습니다.
- Amazon S3를 데이터 스토리지 사이트로 사용하거나, 많은 데이터를 마이그레이션할 때 중간 단계로 사용할 수 있습니다.
- 를 AWS CloudFormation 사용하여 인프라 관리 또는 배포를 위한 AWS 리소스를 설정할 수 있습니다. 예를 들어 복제 인스턴스, 작업, 인증서, 엔드포인트와 같은 AWS DMS 리소스를 프로비저닝할 수 있습니다. 원하는 모든 리소스를 설명하는 템플릿을 만들고 해당 AWS 리소스를 자동으로 AWS CloudFormation 프로비저닝 및 구성합니다.

AWS DMS 에 대한 지원 AWS CloudFormation

를 사용하여 AWS DMS 리소스를 프로비저닝할 수 AWS CloudFormation있습니다. AWS CloudFormation 인프라 관리 또는 배포를 위한 AWS 리소스를 모델링하고 설정하는 데 도움이 되는 서비스입니다. 예를 들어 복제 인스턴스, 작업, 인증서, 엔드포인트와 같은 AWS DMS 리소스를 프로비저닝할 수 있습니다. 원하는 모든 리소스를 설명하고 해당 AWS 리소스를 자동으로 AWS CloudFormation 프로비저닝 및 구성하는 템플릿을 생성합니다.

개발자나 시스템 관리자는 반복적인 마이그레이션 작업이나 조직에 리소스 배포 작업에 사용할 수 있는 이러한 리소스의 모음을 생성 및 관리할 수 있습니다. 에 대한 AWS CloudFormation자세한 내용은 AWS CloudFormation 사용 설명서의 AWS CloudFormation 개념을 참조하십시오.

AWS DMS 다음을 사용하여 AWS CloudFormation다음과 같은 AWS DMS 리소스를 만들 수 있습니다.

- AWS::DMS::Certificate
- AWS::DMS::Endpoint
- AWS::DMS::EventSubscription
- AWS::DMS::ReplicationInstance
- AWS::DMS::ReplicationSubnet그룹
- AWS::DMS::ReplicationTask

AWS Database Migration Service 시작하기

다음 자습서에서는 AWS Database Migration Service(AWS DMS)를 사용하여 데이터베이스 마이그레이션을 수행하는 방법을 확인할 수 있습니다.

데이터베이스 마이그레이션을 수행하려면 다음 단계를 따릅니다.

1. AWS Database Migration Service에 대한 설정의 단계에 따라 AWS 계정을 설정합니다.
2. 샘플 데이터베이스와 Amazon EC2 클라이언트를 생성하여 소스 데이터베이스를 채우고 복제를 테스트합니다. 또한 Amazon Virtual Private Cloud(Amazon VPC) 서비스를 기반으로 Virtual Private Cloud(VPC)를 생성하여 자습서 리소스를 포함시킵니다. 이러한 리소스를 생성하려면 AWS Database Migration Service의 사전 조건의 단계를 따릅니다.
3. 샘플 데이터베이스 생성 스크립트를 사용하여 소스 데이터베이스를 채웁니다.
4. DMS Schema Conversion 또는 AWS Schema Conversion Tool(AWS SCT)를 사용하여 소스 데이터베이스의 스키마를 대상 데이터베이스로 변환합니다. DMS Schema Conversion을 사용하려면 DMS Schema Conversion 시작하기의 단계를 따릅니다. AWS SCT를 사용하여 스키마를 변환하려면 스키마 마이그레이션의 단계를 따릅니다.
5. 마이그레이션을 위한 모든 프로세스를 수행하는 복제 인스턴스를 생성합니다. 이 태스크 및 후속 태스크를 수행하려면 복제의 단계를 수행합니다.
6. 소스 및 대상 엔드포인트를 지정합니다. 엔드포인트를 생성하는 방법에 대한 자세한 내용은 소스 및 대상 엔드포인트 생성 섹션을 참조하세요.
7. 사용할 테이블 및 복제 프로세스를 정의하는 태스크를 생성하고 복제를 시작합니다. 데이터베이스 마이그레이션 태스크를 생성하는 방법에 대한 자세한 내용은 작업 생성 섹션을 참조하세요.
8. 대상 데이터베이스에서 쿼리를 실행하여 복제가 제대로 작동하는지 확인합니다.

AWS Database Migration Service에 대한 설정

AWS 계정에 등록

AWS 계정 항목이 없으면 다음 절차에 따라 생성하세요.

AWS 계정에 가입하려면

1. https://portal.aws.amazon.com/billing/signup을 엽니다.
2. 온라인 지시 사항을 따르세요.

가입 절차 중 전화를 받고 전화 키패드로 확인 코드를 입력하는 과정이 있습니다.

AWS 계정에 가입하면 AWS 계정 루트 사용자 항목이 생성됩니다. 루트 사용자에게 계정의 모든 AWS 서비스 및 리소스에 대한 액세스 권한이 있습니다. 보안 모범 사례는 관리 사용자에게 관리자 액세스 권한을 할당하고, 루트 사용자만 루트 사용자 액세스 권한이 필요한 태스크를 수행하는 것입니다.

AWS 항목은 가입 절차 완료된 후 사용자에게 확인 이메일을 전송합니다. 언제든지 https://aws.amazon.com/으로 이동하고 내 계정을 선택하여 현재 계정 활동을 보고 계정을 관리할 수 있습니다.

관리 사용자 생성

AWS 계정에 가입하고 AWS 계정 루트 사용자에 보안 조치를 한 다음, AWS IAM Identity Center을 활성화하고 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 생성합니다.

귀하의 AWS 계정 루트 사용자 보호

1. 루트 사용자를 선택하고 AWS 계정 이메일 주소를 입력하여 AWS Management Console에 계정 소유자로 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

   루트 사용자를 사용하여 로그인하는 데 도움이 필요하면 AWS 로그인 사용 설명서의 Sign in as the root user을 참조하세요.
2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

   지침은 IAM 사용 설명서의 AWS 계정 루트 사용자용 가상 MFA 디바이스 활성화(콘솔) 섹션을 참조하세요.

관리 사용자 생성

1. IAM Identity Center를 활성화합니다.

   지침은 AWS IAM Identity Center 사용 설명서에서 Enabling AWS IAM Identity Center를 참조하세요.
2. IAM Identity Center에서 관리 사용자에게 관리 액세스 권한을 부여합니다.

IAM Identity Center 디렉터리를 ID 소스로 사용하는 방법에 대한 자습서는 AWS IAM Identity Center 사용 설명서의 Configure user access with the default IAM Identity Center 디렉터리를 참조하세요.

관리자 사용자로 로그인

- IAM 자격 증명 센터 사용자로 로그인하려면 IAM 자격 증명 센터 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

  IAM Identity Center 사용자로 로그인하는 데 도움이 필요한 경우 AWS 로그인 사용 설명서의 AWS 액세스 포털에 로그인을 참조하세요.

AWS Database Migration Service의 사전 조건

이 단원에서는 소스 및 대상 데이터베이스 설정과 같은 AWS DMS 관련 사전 조건 작업에 대해 알아볼 수 있습니다. 이러한 작업의 일부로 Amazon VPC 서비스 기반의 Virtual Private Cloud(VPC)를 설정하여 리소스를 포함시킵니다. 또한 소스 데이터베이스를 채우고 대상 데이터베이스에서 복제를 확인하는 데 사용할 Amazon EC2 인스턴스를 설정합니다.

> **Note**
>
> 소스 데이터베이스를 채우는 데 최대 45분이 걸립니다.

이 자습서에서는 MariaDB 데이터베이스를 소스로 생성하고 PostgreSQL 데이터베이스를 대상으로 생성합니다. 이 시나리오에서는 일반적으로 사용되는 저렴한 데이터베이스 엔진을 사용하여 복제를 시연합니다. 다양한 데이터베이스 엔진을 사용하면 이기종 플랫폼 간에 데이터를 마이그레이션하는 AWS DMS 기능을 시연할 수 있습니다.

이 자습서의 리소스에서는 미국 서부(오레곤) 리전을 사용합니다. 다른 AWS 리전을 사용하려면 미국 서부(오레곤)가 나타나는 위치에 대신 선택한 리전을 지정합니다.

> **Note**
>
> 단순성을 위해 이 자습서에서 생성하는 데이터베이스는 암호화 또는 기타 고급 보안 기능을 사용하지 않습니다. 프로덕션 데이터베이스를 안전하게 보호하려면 보안 기능을 사용해야 합니다. 자세한 내용은 Amazon EC2의 보안을 참조하세요.

사전 조건 단계는 다음 항목을 참조하세요.

주제

- VPC 생성
- Amazon RDS 파라미터 그룹 생성
- 소스 Amazon RDS 데이터베이스 생성
- 대상 Amazon RDS 데이터베이스 생성
- Amazon EC2 클라이언트 생성
- 소스 데이터베이스 채우기

VPC 생성

이 단원에서는 AWS 리소스를 포함할 VPC를 생성합니다. AWS 리소스를 사용할 때는 VPC를 사용하는 것이 모범 사례입니다. 그러면 데이터베이스, Amazon EC2 인스턴스, 보안 그룹 등이 논리적으로 구성되고 안전하게 보호됩니다.

VPC를 자습서 리소스로 사용하면 자습서를 완료했을 때 사용한 모든 리소스도 삭제됩니다. VPC를 삭제하려면 먼저 VPC에 포함된 모든 리소스를 삭제해야 합니다.

AWS DMS에 사용할 VPC를 생성하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/vpc/에서 Amazon VPC 콘솔을 엽니다.
2. 탐색 창에서 VPC 대시보드, 대시보드 생성을 차례로 선택합니다.
3. VPC 생성 페이지에서 다음 옵션을 선택합니다.
   - 생성할 리소스: VPC 등
   - 이름 태그 자동 생성: 자동 생성을 선택하고 **DMSVPC**을 입력합니다.
   - IPv4 블록: **10.0.1.0/24**
   - IPv6 CIDR 블록: IPv6 CIDR 블록 없음
   - 테넌시: 기본값
   - 가용 영역 수: 2
   - 퍼블릭 서브넷 수: 2
   - 프라이빗 서브넷 수: 2

- NAT 게이트웨이($): 없음
- VPC 엔드포인트: 없음

VPC 생성을 선택합니다.

4. 탐색 창에서 VPC를 선택합니다. DMSVPC의 VPC ID를 적어둡니다.
5. 탐색 창에서 보안 그룹을 선택합니다.
6. DMSVPC에 대해 적어둔 ID와 일치하는 VPC ID를 가진 default라는 그룹을 선택합니다.
7. 인바운드 규칙 탭을 선택하고 인바운드 규칙 편집을 선택합니다.
8. 규칙 추가를 선택합니다. MySQL/Aurora 유형의 규칙을 추가하고 소스로 Anywhere-IPv4를 선택합니다.
9. 규칙 추가를 다시 선택합니다. PostgreSQL 유형의 규칙을 추가하고 소스로 Anywhere-IPv4를 선택합니다.
10. 규칙 저장을 선택합니다.

Amazon RDS 파라미터 그룹 생성

AWS DMS용 소스 및 대상 데이터베이스의 설정을 지정하려면 Amazon RDS 파라미터 그룹을 사용합니다. 데이터베이스 간 초기 복제와 지속적 복제를 허용하려면 다음을 구성해야 합니다.

- AWS DMS가 복제해야 할 증분 업데이트를 결정할 수 있도록 소스 데이터베이스의 이진 로그.
- AWS DMS가 초기 데이터 전송 시 외래 키 제약 조건을 무시하도록 대상 데이터베이스의 복제 역할. 이 설정을 사용하면 AWS DMS가 데이터를 순서에 맞지 않게 마이그레이션할 수 있습니다.

AWS DMS에서 사용할 파라미터 그룹을 생성하려면

1. https://console.aws.amazon.com/rds/에서 Amazon RDS 콘솔을 엽니다.
2. 탐색 창에서 파라미터 그룹을 선택합니다.
3. 파라미터 그룹 페이지에서 파라미터 그룹 생성을 선택합니다.
4. 파라미터 그룹 생성페이지에서 다음 설정을 입력합니다.

   - 파라미터 그룹 패밀리: mariadb10.6
   - 그룹 이름: **`dms-mariadb-parameters`**
   - 설명: **`Group for specifying binary log settings for replication`**

생성을 선택하세요.

5. 매개변수 그룹 페이지에서 dms-mariadb-parameters을 선택하고 dms-mariadb-parameters페이지에서 편집을 선택합니다.
6. 다음 파라미터를 다음 값 중 하나로 설정합니다.

   - binlog_checksum: 없음
   - binlog_format: 행

   변경 사항 저장을 선택합니다.
7. 파라미터 그룹 페이지에서 파라미터 그룹 생성을 다시 선택합니다.
8. 파라미터 그룹 생성페이지에서 다음 설정을 입력합니다.

   - 파라미터 그룹 패밀리: postgres13
   - 그룹 이름: **dms-postgresql-parameters**
   - 설명: **Group for specifying role setting for replication**

   생성을 선택하세요.
9. 파라미터 그룹 페이지에서 원하는 항목을 선택합니다 dms-postgresql-parameters.
10. dms-postgresql-parameters페이지에서 편집을 선택하고 session_replication_role 매개변수를 복제본으로 설정합니다. 참고로, session_replication_role 파라미터는 파라미터의 첫 페이지에 없습니다. 페이지 매김 컨트롤 또는 검색 필드를 사용하여 이 파라미터를 찾으세요.
11. 변경 사항 저장을 선택합니다.

소스 Amazon RDS 데이터베이스 생성

다음 절차에 따라 소스 Amazon RDS 데이터베이스를 생성합니다.

소스 Amazon RDS for MariaDB 데이터베이스를 생성하려면

1. https://console.aws.amazon.com/rds/에서 Amazon RDS 콘솔을 엽니다.
2. 대시보드 페이지의 데이터베이스 섹션에서 데이터베이스 생성을 선택합니다. 페이지 상단의 새로운 MySQL 및 PostgreSQL용 Amazon RDS 다중 AZ 배포 옵션 사용해 보기 섹션에서 데이터베이스 생성을 선택하지 마세요.

3. 데이터베이스 생성 페이지에서 다음 옵션을 선택합니다.

   - 데이터베이스 생성 방법 선택: 표준 생성을 선택합니다.
   - 엔진 옵션: 엔진 유형에서 MariaDB를 선택합니다. 버전의에서 MariaDB 10.6.14를 선택한 상태로 둡니다.
   - 템플릿: 개발 및 테스트를 선택합니다.
   - 설정:
     - DB 인스턴스 식별자: **dms-mariadb**를 입력합니다.
     - 보안 인증 정보 설정 섹션에서 다음을 입력합니다.
       - 마스터 사용자 이름: **admin**을 그대로 둡니다.
       - AWS Secrets Manager에서 마스터 보안 인증 정보 관리를 선택하지 않은 상태로 둡니다.
       - 암호 자동 생성: 선택하지 않은 상태로 둡니다.
       - 마스터 암호: **changeit**을 입력합니다.
       - 암호 확인: **changeit**을 다시 입력합니다.
   - 인스턴스 구성:
     - DB 인스턴스 클래스: 표준 클래스가 선택된 상태로 둡니다.
     - DB 인스턴스 클래스에서 db.m5.large를 선택합니다.
   - 스토리지:
     - 스토리지 자동 조정 활성화 확인란의 선택을 취소합니다.
     - 나머지 설정은 그대로 둡니다.
   - 가용성 및 내구성: 대기 인스턴스를 생성하지 않음을 선택한 상태로 둡니다.
   - 연결:
     - 컴퓨팅 리소스 EC2 컴퓨팅 리소스에 연결하지 않음을 그대로 둡니다.
     - 네트워크 유형: IPv4를 선택한 상태로 둡니다.
     - Virtual Private Cloud: DMSVPC-vpc
     - 퍼블릭 액세스: 예 AWS Schema Conversion Tool를 사용하려면 퍼블릭 액세스를 활성화해야 합니다.
     - 가용 영역: us-west-2a
     - 나머지 설정은 그대로 둡니다.
   - 데이터베이스 인증: 암호 인증을 선택한 상태로 둡니다.
   - 모니터링에서 Performance Insights 켜기 확인란의 선택을 취소합니다. 추가 구성 섹션을 확장하고 향상된 모니터링 활성화 확인란의 선택을 취소합니다.

- 추가 구성을 확장합니다.
  - 데이터베이스 옵션에서 초기 데이터베이스 이름에 **dms_sample**을 입력합니다.
  - DB 파라미터 그룹에서 선택합니다. dms-mariadb-parameters
  - 옵션 그룹에서 default:mariadb-10-6을 선택한 상태로 둡니다.
  - 백업에서 다음을 수행합니다.
    - 자동 백업 활성화를 선택한 상태로 둡니다. 지속적 복제를 지원하려면 소스 데이터베이스에 자동 백업이 활성화되어 있어야 합니다.
    - 백업 보존 기간으로 1일을 선택합니다.
    - 백업 기간에서 기본 설정 없음을 선택된 상태로 둡니다.
    - 스냅샷에 태그 복사 확인란의 선택을 취소합니다.
    - 다른 AWS 리전에서 복제 활성화를 선택하지 않은 상태로 둡니다.
  - 암호화에서 암호화 활성화 확인란의 선택을 취소합니다.
  - 로그 내보내기 섹션은 그대로 둡니다.
  - 유지 관리에서 자동 마이너 버전 업그레이드 활성화 확인란의 선택을 취소하고 유지 관리 기간 설정을 기본 설정 없음으로 유지합니다.
  - 삭제 방지 활성화를 선택하지 않은 상태로 둡니다.
4. 데이터베이스 생성를 선택합니다.

대상 Amazon RDS 데이터베이스 생성

이전 절차를 반복하되 다음과 같이 변경하여 대상 Amazon RDS 데이터베이스를 생성합니다.

대상 RDS for PostgreSQL 데이터베이스를 생성하려면

1. 이전 절차의 1 및 2단계를 반복합니다.
2. 데이터베이스 생성 페이지에서 다음을 제외하고 동일한 옵션을 설정합니다.
   a. 엔진 옵션에서 PostgreSQL을 선택합니다.
   b. 버전에서 PostgreSQL 13.7-R1을 선택합니다.
   c. DB 인스턴스 식별자에 **dms-postgresql**을 입력합니다.
   d. 마스터 사용자 이름은 **postgres**을 선택된 상태로 둡니다.
   e. DB 파라미터 그룹의 경우 선택합니다 dms-postgresql-parameters.
   f. 자동 백업 활성화 확인란의 선택을 취소합니다.

3. 데이터베이스 생성을 선택합니다.

Amazon EC2 클라이언트 생성

이 단원에서는 Amazon EC2 클라이언트를 생성합니다. 이 클라이언트를 사용하여 복제할 데이터로 소스 데이터베이스를 채웁니다. 또한 이 클라이언트를 사용하여 대상 데이터베이스에서 쿼리를 실행하여 복제를 확인할 수 있습니다.

Amazon EC2 클라이언트를 사용하여 데이터베이스에 액세스하면 인터넷을 통해 데이터베이스에 액세스하는 것에 비해 다음과 같은 이점이 있습니다.

- 데이터베이스에 대한 액세스를 동일한 VPC에 있는 클라이언트로만 제한할 수 있습니다.
- 이 자습서에서 사용하는 도구는 Amazon Linux 2023에서 작동하고 설치하기도 쉽다는 것을 확인했으므로 이 자습서에서는 이 도구를 권장합니다.
- VPC의 구성 요소 간 데이터 작업은 일반적으로 인터넷을 통한 작업보다 성능이 좋습니다.

Amazon EC2 클라이언트를 생성 및 구성하여 소스 데이터베이스를 채우려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 대시보드에서 인스턴스 시작을 선택합니다.
3. 인스턴스 시작 페이지에서 다음 값을 입력합니다.

    a. 이름 및 태그 섹션에서 이름에 **`DMSClient`**를 입력합니다.

    b. 애플리케이션 및 OS 이미지(Amazon Machine Image) 섹션에서 설정을 그대로 유지합니다.

    c. 인스턴스 유형 섹션에서 t2.xlarge를 선택합니다.

    d. 키 페어(로그인) 섹션에서 새 키 페어 생성을 선택합니다.

    e. 키 페어 생성 페이지에서 다음을 입력합니다.

    - Key pair name: **`DMSKeyPair`**
    - 키 페어 유형: RSA를 그대로 둡니다.
    - 프라이빗 키 파일 형식: macOS 또는 Linux의 OpenSSH에는 pem을 선택하고, Windows의 PuTTY에는 ppk를 선택합니다.

    메시지가 표시되면 키 파일을 저장합니다.

> **Note**
>
> 새 키 페어를 생성하는 대신 기존 Amazon EC2 키 페어를 사용할 수도 있습니다.

f. 네트워크 설정 섹션에서 편집을 선택합니다. 다음 설정을 선택합니다.

- VPC - 필수: DMSVPC-vpc VPC에 대해 적어둔 ID가 있는 VPC를 선택합니다.
- 서브넷에서 첫 번째 퍼블릭 서브넷을 선택합니다.
- 퍼블릭 IP 자동 할당: 활성화를 선택합니다.

나머지 설정은 그대로 두고 인스턴스 시작을 선택합니다.

소스 데이터베이스 채우기

이 단원에서는 나중에 사용할 소스 및 대상 데이터베이스의 엔드포인트를 찾고 다음 도구를 사용하여 소스 데이터베이스를 채웁니다.

- Git - 소스 데이터베이스를 채우는 스크립트를 다운로드합니다.
- MariaDB 클라이언트 - 이 스크립트를 실행합니다.

엔드포인트 가져오기

나중에 사용할 수 있도록 MariaDB용 RDS와 PostgreSQL DB 인스턴스용 RDS의 엔드포인트를 찾아 기록해 둡니다.

DB 인스턴스 엔드포인트를 찾으려면

1. https://console.aws.amazon.com/rds/에서 AWS Management Console에 로그인한 후 Amazon RDS 콘솔을 엽니다.
2. 탐색 창에서 데이터베이스를 선택합니다.
3. dms-mariadb 데이터베이스를 선택하고 데이터베이스의 엔드포인트 값을 적어둡니다.
4. dms-postgresql 데이터베이스에 대해 이전 단계를 반복합니다.

소스 데이터베이스 채우기

다음으로 클라이언트 인스턴스에 연결하고, 필요한 소프트웨어를 설치하고, Git에서 AWS 샘플 데이터베이스 스크립트를 다운로드하고, 스크립트를 실행하여 소스 데이터베이스를 채웁니다.

소스 데이터베이스를 채우려면

1. 이전 단계에서 저장한 호스트 이름 및 퍼블릭 키를 사용하여 클라이언트 인스턴스에 연결합니다.

   Amazon EC2 인스턴스에 연결하는 방법에 대한 자세한 내용은 Linux 인스턴스용 Amazon EC2 사용 설명서의 인스턴스 액세스를 참조하세요.

   > **Note**
   >
   > PuTTY를 사용하는 경우 비활성 상태로 인해 연결이 시간 초과되지 않도록 연결 설정 페이지에서 TCP keepalive를 활성화합니다.

2. Git, MariaDB 및 PostgreSQL을 설치합니다. 필요에 따라 설치를 확인합니다.

   ```
   $ sudo yum install git
   $ sudo dnf install mariadb105
   $ sudo dnf install postgresql15
   ```

3. 다음 명령을 실행하여 에서 데이터베이스 생성 스크립트를 다운로드합니다. GitHub

   ```
   git clone https://github.com/aws-samples/aws-database-migration-samples.git
   ```

4. 디렉터리를 `aws-database-migration-samples/mysql/sampledb/v1/`로 변경합니다.
5. 다음 명령을 실행합니다. 이전에 적어둔 소스 RDS 인스턴스의 엔드포인트를 제공합니다(예: `dms-mariadb.cdv5fbeyiy4e.us-east-1.rds.amazonaws.com`).

   ```
   mysql -h dms-mariadb.abcdefghij01.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
    dms_sample < ~/aws-database-migration-samples/mysql/sampledb/v1/install-rds.sql
   ```

6. 데이터베이스 생성 스크립트를 실행합니다. 이 스크립트가 스키마를 생성하고 데이터를 채우는 데 최대 45분이 걸립니다. 스크립트에 표시되는 오류 및 경고는 무시해도 됩니다.

AWS SCT를 사용하여 소스 스키마를 대상 데이터베이스로 마이그레이션합니다.

이 단원에서는 AWS Schema Conversion Tool를 사용하여 소스 스키마를 대상 데이터베이스로 마이그레이션합니다. 또는 DMS Schema Conversion을 사용하여 소스 데이터베이스 스키마를 변환할 수 있습니다. 자세한 내용은 DMS Schema Conversion 시작하기 섹션을 참조하세요.

AWS SCT를 사용하여 소스 스키마를 대상 데이터베이스로 마이그레이션하려면

1. AWS Schema Conversion Tool를 설치합니다. 자세한 내용은 AWS 사용 설명서의 AWS SCT 설치, 확인 또는 업데이트를 참조하세요.

   MySQL 및 PostgreSQL용 JDBC 드라이버를 다운로드할 때 도구에서 드라이버 위치를 묻는 메시지가 표시될 경우를 대비하여 드라이버를 저장한 위치를 적어두세요.
2. AWS Schema Conversion Tool을 엽니다. 파일을 선택하고 새 프로젝트를 선택합니다.
3. 새 프로젝트 창에서 다음 값을 설정합니다.

   - 프로젝트 이름을 **DMSProject**로 설정합니다.
   - AWS SCT 프로젝트를 기본 폴더에 저장하도록 위치를 그대로 유지합니다.

   확인을 선택합니다.
4. 소스 추가를 선택하여 소스 MySQL 데이터베이스를 프로젝트에 추가한 후 MySQL을 선택하고 다음을 선택합니다.
5. 소스 추가 페이지에서 다음 값을 설정합니다.

   - 연결 이름: **source**
   - 서버 이름: 이전에 적어둔 MySQL 데이터베이스의 엔드포인트를 입력합니다.
   - 서버 포트: **3306**
   - 사용자 이름: **admin**
   - 암호: **changeit**
6. 대상 추가를 선택하여 대상 Amazon RDS for PostgreSQL 데이터베이스를 프로젝트에 추가한 다음 Amazon RDS for PostgreSQL을 선택합니다. 다음을 선택합니다.
7. 대상 추가 페이지에서 다음 값을 설정합니다.

   - 연결 이름: **target**

- 서버 이름: 이전에 적어둔 PostgreSQL 데이터베이스의 엔드포인트를 입력합니다.
- 서버 포트: **5432**
- 데이터베이스: PostgreSQL 데이터베이스의 이름을 입력합니다.
- 사용자 이름: **postgres**
- 암호: **changeit**

8. 왼쪽 창의 스키마에서 dms_sample을 선택합니다. 오른쪽 창에서 대상 Amazon RDS for PostgreSQL 데이터베이스를 선택합니다. 매핑 생성을 선택합니다. 단일 AWS SCT 프로젝트에 여러 매핑 규칙을 추가할 수 있습니다. 매핑 규칙에 대한 자세한 내용은 매핑 규칙 생성을 참조하세요.
9. 기본 보기를 선택합니다.
10. 왼쪽 창의 스키마에서 dms_sample을 선택합니다. 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) 스키마 변환을 선택합니다. 작업을 확인합니다.

    도구가 스키마를 변환하면 오른쪽 창에 dms_sample 스키마가 나타납니다.
11. 오른쪽 창의 스키마에서 dms_sample의 컨텍스트 메뉴를 열고(마우스 오른쪽 버튼 클릭) 데이터베이스에 적용을 선택합니다. 작업을 확인합니다.

스키마 마이그레이션이 완료되었는지 확인합니다. 다음 단계를 수행합니다.

스키마 마이그레이션을 확인하려면

1. Amazon EC2 클라이언트에 연결합니다.
2. 다음 명령을 사용하여 PSQL 클라이언트를 시작합니다. PostgreSQL 데이터베이스 엔드포인트를 지정하고 메시지가 표시되면 데이터베이스 암호를 입력합니다.

```
psql \
   --host=dms-postgresql.abcdefg12345.us-west-2.rds.amazonaws.com \
   --port=5432 \
   --username=postgres \
   --password \
   --dbname=dms_sample
```

3. (빈) 테이블 중 하나를 쿼리하여 AWS SCT가 스키마를 올바르게 적용했는지 확인합니다.

```
dms_sample=> SELECT * from dms_sample.player;
 id | sport_team_id | last_name | first_name | full_name
----+---------------+-----------+------------+-----------
```

```
(0 rows)
```

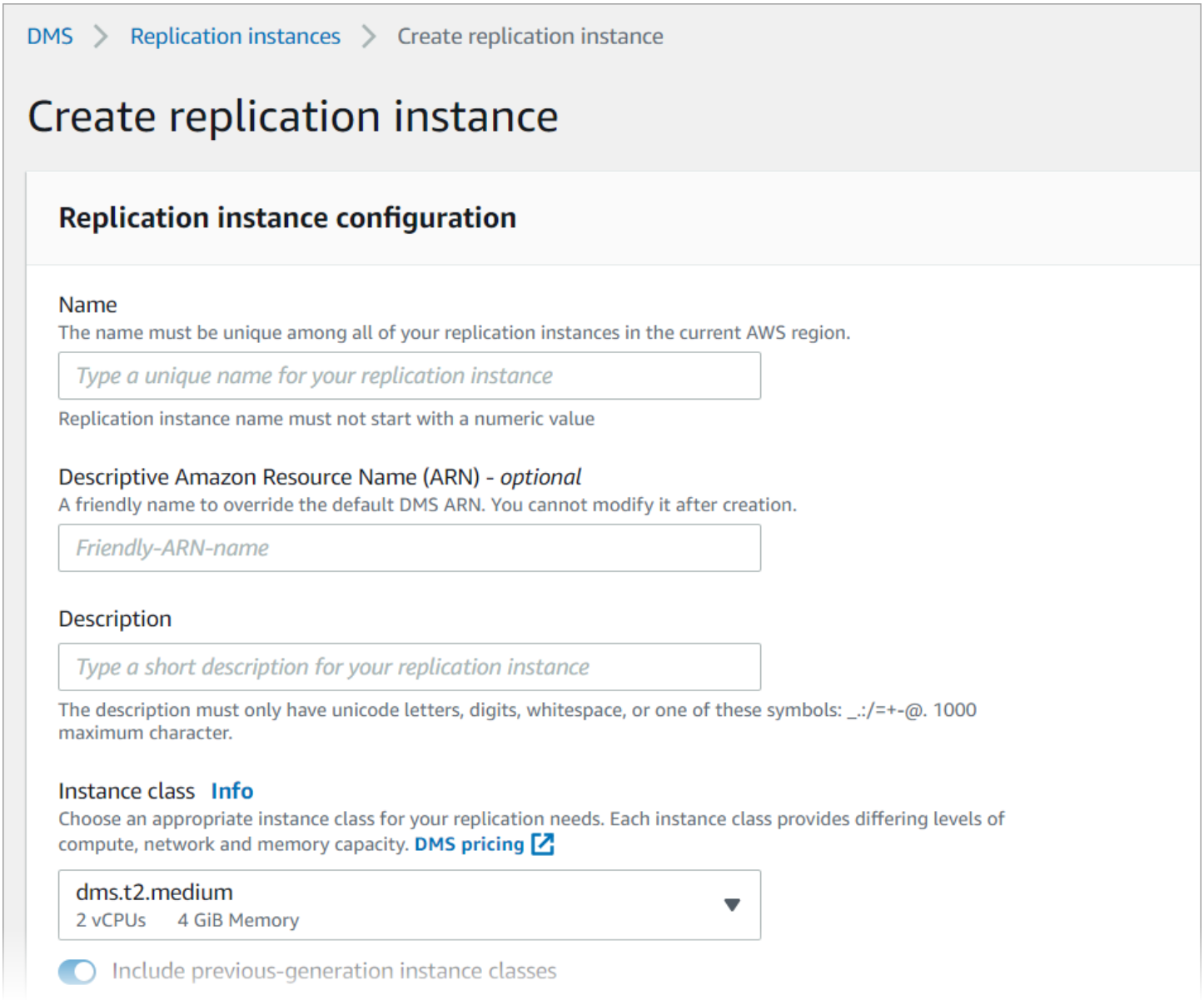
AWS Database Migration Service에서 복제 설정

이 항목에서는 소스 데이터베이스와 대상 데이터베이스 간의 복제를 설정합니다.

1단계: AWS DMS 콘솔을 사용하여 복제 인스턴스 생성

AWS DMS를 시작하기 위해 복제 인스턴스를 생성합니다.

복제 인스턴스는 소스 엔드포인트와 대상 엔드포인트 간에 실제 데이터 마이그레이션을 수행합니다. 소스 데이터베이스에서 대상 데이터베이스로 데이터를 마이그레이션하는 작업을 수행하려면 인스턴스에 충분한 스토리지 및 처리 능력이 필요합니다. 이 복제 인스턴스의 크기는 마이그레이션할 데이터의 양과 인스턴스가 수행해야 하는 태스크에 따라 달라집니다. 복제 인스턴스에 대한 자세한 내용은 AWS DMS 복제 인스턴스 사용 섹션을 참조하세요.



콘솔을 사용하여 복제 인스턴스를 생성하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/dms/v2/에서 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 복제 인스턴스를 선택하고, 복제 인스턴스 생성을 선택합니다.
3. 복제 인스턴스 생성 페이지에서 복제 인스턴스 구성을 지정합니다.
   a. 이름에 **`DMS-instance`**를 입력합니다.
   b. 설명에 복제 인스턴스에 대한 간단한 설명을 입력합니다(선택 사항).
   c. 인스턴스 클래스에서 dms.t3.medium을 선택한 상태로 둡니다.

인스턴스에는 마이그레이션을 위한 충분한 스토리지, 네트워킹 및 처리 능력이 필요합니다. 인스턴스 클래스를 선택하는 방법에 대한 자세한 내용은 마이그레이션에 적합한 AWS DMS 복제 인스턴스 선택 섹션을 참조하세요.

d. 엔진 버전에서 기본값을 그대로 사용합니다.

e. 다중 AZ에서 개발 또는 테스트 워크로드(단일 AZ)를 선택합니다.

f. 할당된 스토리지(GiB)에서 기본값인 50GiB를 그대로 사용합니다.

AWS DMS에서는 스토리지가 로그 파일 및 캐시된 트랜잭션에 주로 사용됩니다. 캐시 트랜잭션에서 스토리지는 캐시된 트랜잭션을 디스크에 기록해야 할 때에만 사용됩니다. 따라서 AWS DMS는 스토리지를 많이 사용하지 않습니다.

g. 네트워크 유형에서 IPv4를 선택합니다.

h. VPC에서 DMSVPC를 선택합니다.

i. 복제 서브넷 그룹에서 현재 선택한 복제 서브넷 그룹을 그대로 둡니다.

j. 퍼블릭 액세스 가능 확인란의 선택을 취소합니다.

4. 필요한 경우 고급 보안 및 네트워크 구성 탭을 선택하여 네트워크 및 암호화 설정에 대한 값을 설정합니다.

a. 가용 영역에서 us-west-2a를 선택합니다.

b. VPC 보안 그룹에서 기본 보안 그룹을 선택합니다(이미 선택되지 않은 경우).

c. AWS KMS key에서 (기본값) aws/dms를 선택한 상태로 둡니다.

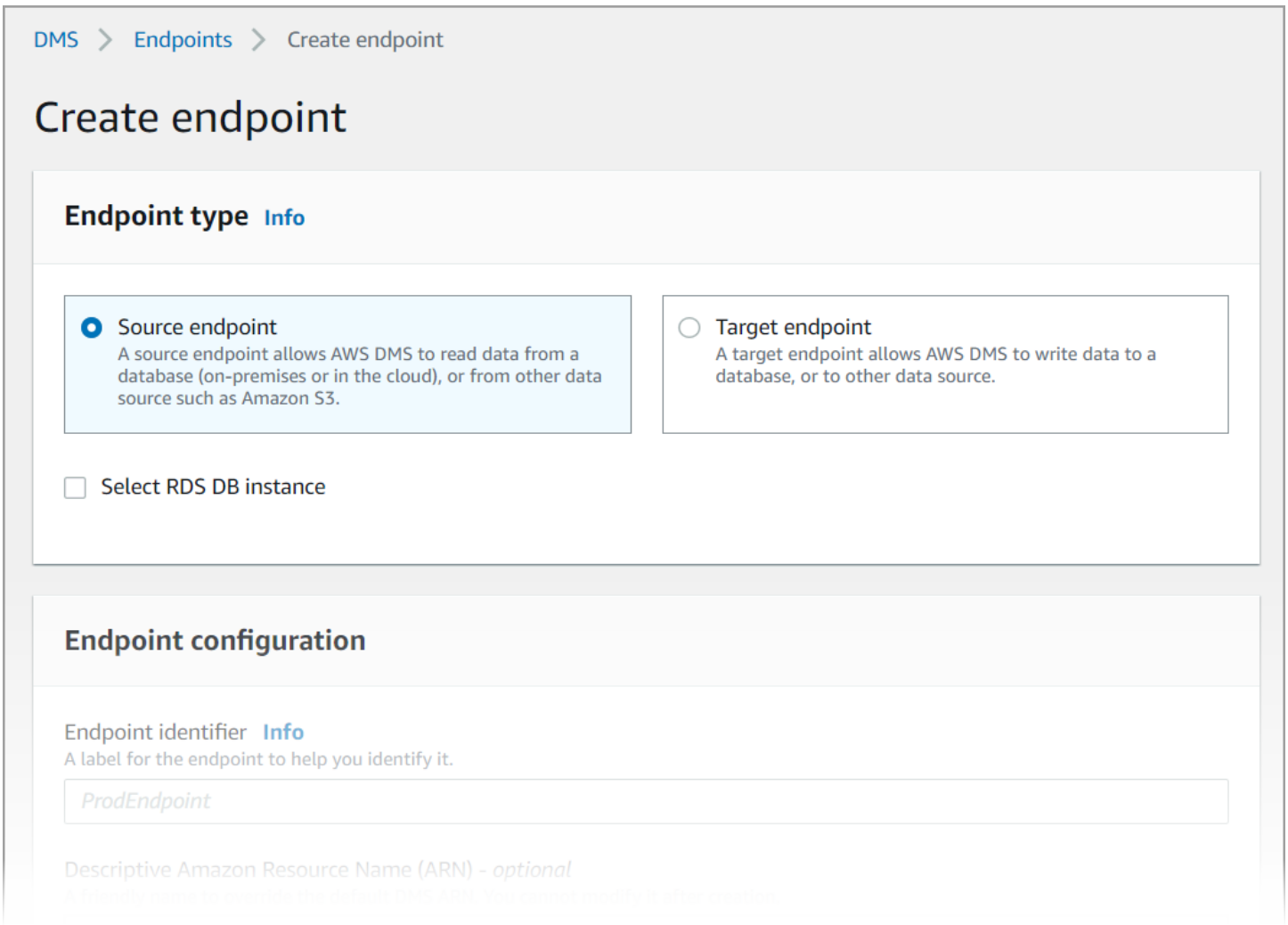
5. 유지 관리 탭의 설정을 그대로 둡니다. 기본값은 AWS 리전별로 8시간의 시간 블록 중 임의로 선택한 30분의 기간이며, 발생하는 요일은 무작위입니다.

6. 생성을 선택합니다.

AWS DMS가 마이그레이션을 수행할 복제 인스턴스를 생성합니다.

2단계: 소스 및 대상 엔드포인트 지정

복제 인스턴스가 생성되는 동안 앞서 생성한 Amazon RDS 데이터베이스를 위한 소스 및 대상 데이터 스토어를 지정할 수 있습니다. 각 엔드포인트를 개별적으로 생성합니다.



AWS DMS 콘솔을 사용하여 소스 엔드포인트와 데이터베이스 엔드포인트를 지정하려면

1. 콘솔의 탐색 창에서 엔드포인트를 선택하고 엔드포인트 생성을 선택합니다.
2. 엔드포인트 생성 페이지에서 소스 엔드포인트 유형을 선택합니다. RDS DB 인스턴스 선택 확인란을 선택하고 dms-mariadb 인스턴스를 선택합니다.
3. 엔드포인트 구성 섹션에서 엔드포인트 식별자로 **`dms-mysql-source`**를 입력합니다.
4. 소스 엔진에서 MySQL을 선택한 상태로 둡니다.
5. 엔드포인트 데이터베이스 액세스에서 수동으로 액세스 정보 제공을 선택합니다. 포트, 보안 소켓 계층(SSL) 모드, 사용자 이름 및 암호가 올바른지 확인합니다.
6. 엔드포인트 연결 테스트(선택 사항) 탭을 선택합니다. VPC에서 DMSVPC를 선택합니다.
7. 복제 인스턴스에서 dms-instance를 선택한 상태로 둡니다.
8. 테스트 실행을 선택합니다.

테스트 실행을 선택하면 AWS DMS가 제공된 세부 정보로 엔드포인트를 생성하고 엔드포인트에 연결합니다. 연결이 실패할 경우 엔드포인트 정의를 편집하고 연결을 다시 테스트하세요. 엔드포인트를 수동으로 삭제할 수도 있습니다.

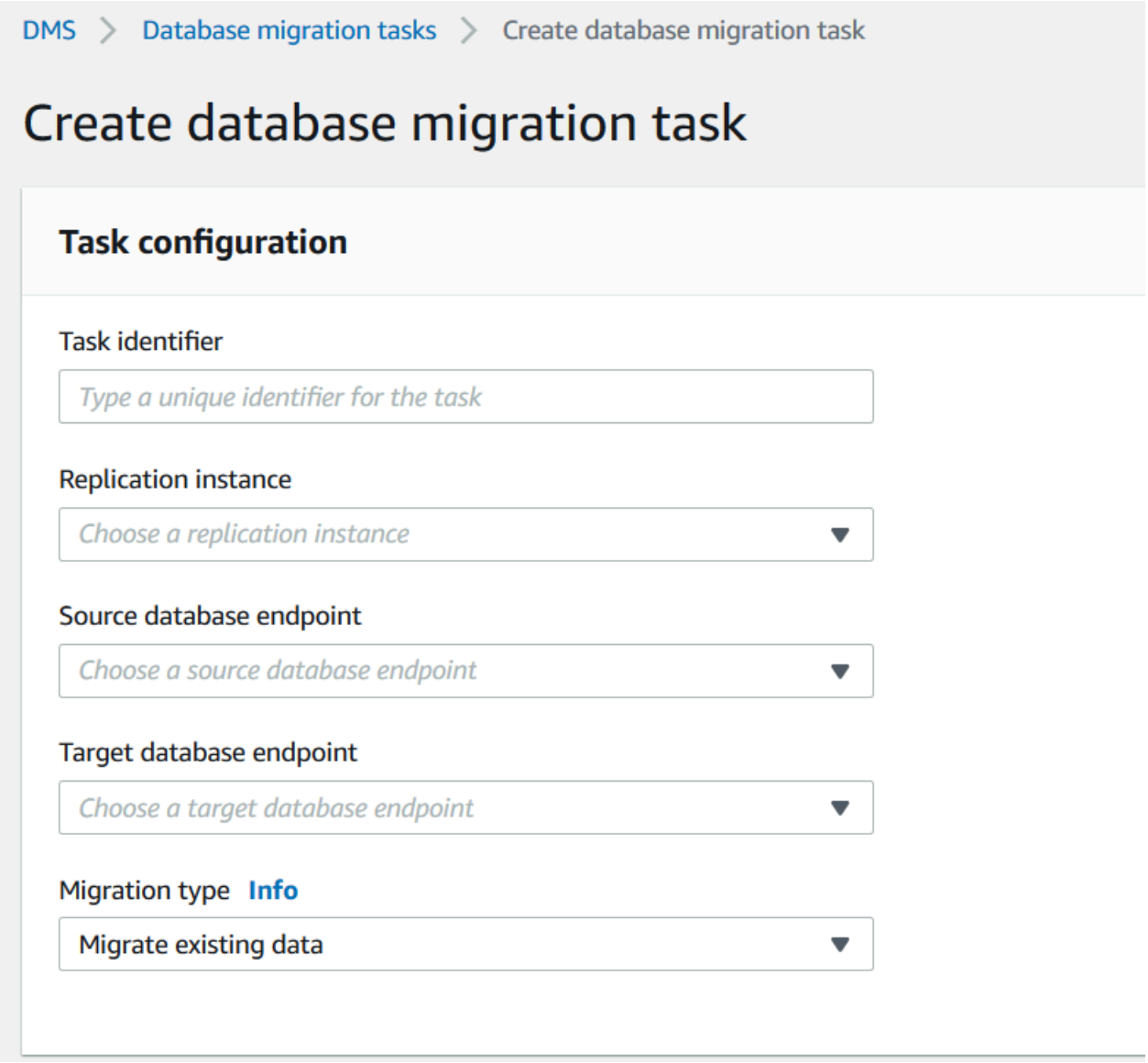
9. 테스트에 성공하면 엔드포인트 생성을 선택합니다.
10. AWS DMS 콘솔을 사용하여 대상 데이터베이스 엔드포인트를 지정합니다. 이렇게 하려면 다음 설정을 사용하여 이전 단계를 반복합니다.

    - 엔드포인트 유형: 대상 엔드포인트
    - RDS 인스턴스: dms-postgresql
    - 엔드포인트 식별자: **dms-postgresql-target**
    - 대상 엔진: **PostgreSQL**을 선택한 상태로 둡니다.

엔드포인트에 대한 모든 정보를 모두 제공했다면 AWS DMS는 데이터베이스 마이그레이션 중에 사용할 소스 및 대상 엔드포인트를 생성합니다.

3단계: 태스크 생성 및 데이터 마이그레이션

이 단계에서는 생성한 데이터베이스 간에 데이터를 마이그레이션하는 태스크를 생성합니다.



마이그레이션 태스크를 생성하고 데이터베이스 마이그레이션을 시작하려면

1. 콘솔 탐색 창에서 데이터베이스 마이그레이션 태스크를 선택한 다음 태스크 생성을 선택합니다. 데이터베이스 마이그레이션 태스크 생성 페이지가 열립니다.
2. 태스크 구성 섹션에서 다음 태스크 옵션을 지정합니다.

   - 태스크 식별자: **`dms-task`**를 입력합니다.
   - 복제 인스턴스: 복제 인스턴스를 선택합니다(dms-instance-vpc-*<vpc id>*).
   - 소스 데이터베이스 엔드포인트: dms-mysql-source를 선택합니다.
   - 대상 데이터베이스 엔드포인트: dms-postgresql-target을 선택합니다.
   - 마이그레이션 유형: 기존 데이터 마이그레이션 및 진행 중 변경 사항 복제를 선택합니다.

3. 태스크 설정 탭을 선택합니다. 다음 설정을 지정합니다.

   - 대상 테이블 준비 모드: 아무 작업 안 함
   - 전체 로드 완료 후 태스크 중지: 중지 안 함

4. 테이블 매핑 탭을 선택하고 선택 규칙을 확장합니다. 선택 규칙 추가를 선택합니다. 다음 설정을 지정합니다.

   - 스키마: 스키마 입력
   - 스키마 이름: **dms_sample**

5. 마이그레이션 태스크 시작 구성 탭을 선택한 다음 생성 시 자동으로를 선택합니다.
6. [태스크 생성]을 선택합니다.

그러면 AWS DMS가 마이그레이션 태스크를 생성하고 시작합니다. 초기 데이터베이스 복제는 약 10분이 걸립니다. AWS DMS가 데이터 마이그레이션을 완료하기 전에 자습서의 다음 단계를 수행해야 합니다.

4단계: 복제 테스트

이 단원에서는 초기 복제 도중 및 이후에 소스 데이터베이스에 데이터를 삽입하고 대상 데이터베이스에서 삽입된 데이터를 쿼리합니다.

복제를 테스트하려면

1. 데이터베이스 마이그레이션 태스크의 상태가 실행 중으로 표시되지만 이전 단계에서 시작한 초기 데이터베이스 복제가 완료되지 않았는지 확인합니다.
2. Amazon EC2 클라이언트에 연결하고 다음 명령을 사용하여 MySQL 클라이언트를 시작합니다. MySQL 데이터베이스 엔드포인트를 제공합니다.

   ```
   mysql -h dms-mysql.abcdefg12345.us-west-2.rds.amazonaws.com -P 3306 -u admin -
   pchangeit dms_sample
   ```

3. 다음 명령을 실행하여 소스 데이터베이스에 레코드를 삽입합니다.

   ```
   MySQL [dms_sample]> insert person (full_name, last_name, first_name) VALUES ('Test
    User1', 'User1', 'Test');
   Query OK, 1 row affected (0.00 sec)
   ```

4. MySQL 클라이언트를 종료합니다.

```
MySQL [dms_sample]> exit
Bye
```

5. 복제가 완료되기 전에 대상 데이터베이스에서 새 레코드를 쿼리합니다.

   다음 명령으로 Amazon EC2 인스턴스에서 대상 데이터베이스 엔드포인트를 제공하여 대상 데이터베이스에 연결합니다.

```
psql \
   --host=dms-postgresql.abcdefg12345.us-west-2.rds.amazonaws.com \
   --port=5432 \
   --username=postgres \
   --password \
   --dbname=dms_sample
```

   메시지가 표시되면 암호(**changeit**)를 제공합니다.

6. 복제가 완료되기 전에 대상 데이터베이스에서 새 레코드를 쿼리합니다.

```
dms_sample=> select * from dms_sample.person where first_name = 'Test';
 id | full_name | last_name | first_name
----+-----------+-----------+------------
(0 rows)
```

7. 마이그레이션 태스크가 실행되는 동안 데이터베이스 마이그레이션의 진행 상황을 모니터링할 수 있습니다.

   - DMS 콘솔 탐색 창에서 데이터베이스 마이그레이션 태스크를 선택합니다.
   - dms-task를 선택합니다.
   - 테이블 통계를 선택합니다.

   모니터링에 대한 자세한 내용은 AWS DMS 태스크 모니터링 섹션을 참조하세요.

8. 복제가 완료되면 대상 데이터베이스에서 새 레코드를 다시 쿼리합니다. AWS DMS는 초기 복제가 완료된 후 새 레코드를 마이그레이션합니다.

```
dms_sample=> select * from dms_sample.person where first_name = 'Test';
   id    | full_name  | last_name | first_name
---------+------------+-----------+------------
 7077784 | Test User1 | User1     | Test
```

```
(1 row)
```

9. psql 클라이언트를 종료합니다.

```
dms_sample=> quit
```

10. 1단계를 반복하여 소스 데이터베이스에 다시 연결합니다.
11. person 테이블에 다른 레코드를 삽입합니다.

```
MySQL [dms_sample]> insert person (full_name, last_name, first_name) VALUES ('Test
 User2', 'User2', 'Test');
Query OK, 1 row affected (0.00 sec)
```

12. 3단계 및 4단계를 반복하여 소스 데이터베이스의 연결을 끊고 대상 데이터베이스에 연결합니다.
13. 대상 데이터베이스에서 복제된 데이터를 다시 쿼리합니다.

```
dms_sample=> select * from dms_sample.person where first_name = 'Test';
   id    | full_name  | last_name | first_name
---------+------------+-----------+------------
 7077784 | Test User1 | User1     | Test
 7077785 | Test User2 | User2     | Test
(2 rows)
```

5단계: AWS DMS 리소스 정리

시작하기 자습서를 완료하면 생성한 리소스를 삭제할 수 있습니다. AWS 명령을 사용하여 제거할 수도 있습니다. 복제 인스턴스 및 엔드포인트를 삭제하기 전에 마이그레이션 태스크를 삭제해야 합니다.

콘솔을 사용하여 마이그레이션 태스크를 삭제하려면

1. AWS DMS 콘솔 탐색 창에서 데이터베이스 마이그레이션 태스크를 선택합니다.
2. dms-task를 선택합니다.
3. 작업, 삭제를 선택합니다.

콘솔을 사용하여 복제 인스턴스를 삭제하려면

1. AWS DMS 콘솔 탐색 창에서 복제 인스턴스를 선택합니다.
2. DMS-instance를 선택합니다.

3. 작업, 삭제를 선택합니다.

AWS DMS가 복제 인스턴스를 삭제하고 복제 인스턴스 페이지에서 제거합니다.

콘솔을 사용하여 엔드포인트를 제거하려면

1. AWS DMS 콘솔 탐색 창에서 엔드포인트를 선택합니다.
2. dms-mysql-source를 선택합니다.
3. 작업, 삭제를 선택합니다.

AWS DMS 리소스를 삭제한 후에는 다음 리소스도 삭제해야 합니다. 다른 서비스의 리소스를 삭제하는 데 도움이 필요하면 각 서비스의 설명서를 참조하세요.

- RDS 데이터베이스.
- RDS 데이터베이스 파라미터 그룹.
- RDS 서브넷 그룹.
- 데이터베이스 및 복제 인스턴스와 함께 생성된 모든 Amazon CloudWatch 로그
- Amazon VPC 및 Amazon EC2 클라이언트용으로 생성된 보안 그룹. launch-wizard-1 보안 그룹에 대한 기본값에서 인바운드 규칙을 제거해야 합니다. 이는 보안 그룹을 삭제하는 데 필요합니다.
- Amazon EC2 클라이언트.
- Amazon VPC.
- Amazon EC2 클라이언트용 Amazon EC2 키 페어

AWS Database Migration Service에서 사용할 추가 리소스

이 설명서의 뒷부분에서는 AWS DMS를 사용하여 가장 널리 사용되는 상용 및 오픈 소스 데이터베이스 간에 데이터를 마이그레이션하는 방법을 배울 수 있습니다.

또한 데이터베이스 마이그레이션 프로젝트를 준비하고 수행할 때 다음 리소스를 확인하는 것이 좋습니다.

- AWS DMS 단계별 마이그레이션 가이드 — 이 가이드는 데이터를 AWS로 마이그레이션하는 프로세스를 단계별로 안내합니다.
- AWS DMS API 참조 - 이 참조에서는 모든 AWS Database Migration Service용 API 작업을 자세히 설명합니다.

- AWS DMS용 AWS CLI - 이 참조에서는 AWS Command Line Interface(AWS CLI)를 AWS DMS와 함께 사용하는 방법에 대한 정보를 제공합니다.

AWS DMS Fleet Advisor를 사용하여 마이그레이션할 데이터베이스 검색 및 평가

DMS Fleet Advisor를 사용하여 여러 데이터베이스 환경에서 메타데이터 및 성능 지표를 수집할 수 있습니다. 이렇게 수집된 지표는 데이터 인프라에 대한 인사이트를 제공합니다. DMS Fleet Advisor는 모든 컴퓨터에 설치할 필요 없이 하나 이상의 중앙 위치에서 온프레미스 데이터베이스 및 분석 서버의 메타데이터와 지표를 수집합니다. 현재 DMS 플릿 어드바이저는 Microsoft SQL Server, MySQL, 오라클 및 PostgreSQL 데이터베이스 서버의 검색 및 메트릭 수집을 지원합니다.

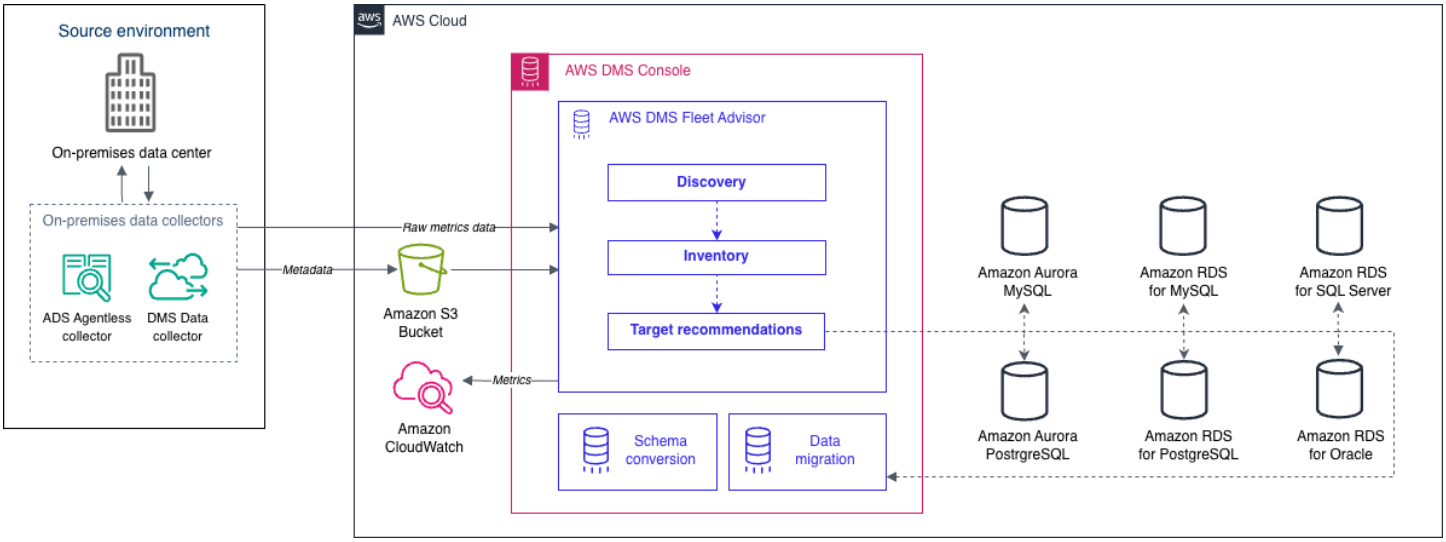
네트워크에서 검색된 데이터를 기반으로 인벤토리를 작성하여 추가 데이터 수집을 위한 데이터베이스 서버 목록을 정의할 수 있습니다. AWS DMS가 서버, 데이터베이스 및 스키마에 대한 정보를 수집한 후에는 의도한 데이터베이스 마이그레이션의 실행 가능성을 분석할 수 있습니다.

AWS 클라우드로 마이그레이션하려는 인벤토리 내의 데이터베이스에 대해 DMS Fleet Advisor가 적절한 규모의 대상 권장 사항을 생성합니다. 대상 권장 사항을 생성하기 위해 DMS Fleet Advisor는 데이터 수집기의 지표와 기본 설정을 고려합니다. DMS Fleet Advisor가 권장 사항을 생성한 후에는 각 대상 데이터베이스 구성에 대한 세부 정보를 볼 수 있습니다. 조직의 데이터베이스 엔지니어 및 관리자는 DMS Fleet Advisor 대상 권장 사항을 사용하여 온프레미스 데이터베이스를 AWS로 마이그레이션할 계획을 수립할 수 있습니다. 사용 가능한 다양한 마이그레이션 옵션을 탐색하고 이러한 권장 사항을로 내보내 비용을 더욱 최적화할 수 있습니다AWS Pricing Calculator.

지원되는 소스 데이터베이스의 목록은 DMS Fleet Advisor가 지원하는 소스 섹션을 참조하세요.

DMS Fleet Advisor가 대상 권장 사항을 생성하는 데 사용하는 데이터베이스의 목록은 DMS Fleet Advisor가 지원하는 대상 섹션을 참조하세요. DMS 플릿 어드바이저는 예를 들어 소스 Oracle에서 대상 Oracle 데이터베이스까지 비슷한 수준의 권장 사항을 생성합니다. 또한 DMS 플릿 어드바이저는 소스 Oracle 또는 Microsoft SQL Server에서 PostgreSQL 또는 Aurora용 타겟 RDS로 마이그레이션하는 것과 같은 이기종 권장 사항을 생성합니다.

다음 다이어그램은 AWS DMS Fleet Advisor 대상 권장 사항 프로세스를 보여 줍니다.



AWS DMS Fleet Advisor를 사용하는 방법을 더 잘 이해하려면 다음 항목을 참조하세요.

주제

- 지원되는 AWS 리전
- DMS Fleet Advisor 시작
- AWS DMS Fleet Advisor 설정
- 데이터 수집기를 사용하여 마이그레이션할 데이터베이스 검색
- AWS DMS Fleet Advisor에서 인벤토리를 분석에 사용하기
- AWS DMS Fleet Advisor 대상 권장 기능 사용
- DMS 플릿 어드바이저 제한 사항

지원되는 AWS 리전

DMS Fleet Advisor는 다음 AWS 리전에서 사용할 수 있습니다.

리전 이름	리전
미국 동부(버지니아 북부)	us-east-1
미국 동부(오하이오)	us-east-2
미국 서부(캘리포니아 북부)	us-west-1

리전 이름	리전
미국 서부(오레곤)	us-west-2
아시아 태평양(홍콩)	ap-east-1
아시아 태평양(도쿄)	ap-northeast-1
아시아 태평양(서울)	ap-northeast-2
아시아 태평양(오사카)	ap-northeast-3
아시아 태평양(뭄바이)	ap-south-1
아시아 태평양(싱가포르)	ap-southeast-1
아시아 태평양(시드니)	ap-southeast-2
아시아 태평양(자카르타)	ap-southeast-3
캐나다(중부)	ca-central-1
유럽(프랑크푸르트)	eu-central-1
유럽(스톡홀름)	eu-north-1
유럽(아일랜드)	eu-west-1
유럽(런던)	eu-west-2
유럽(파리)	eu-west-3
유럽(밀라노)	eu-south-3
캐나다(중부)	ca-central-1
남아메리카(상파울루)	sa-east-1
중동(바레인)	me-south-1
아프리카(케이프타운)	af-south-1

DMS Fleet Advisor 시작

DMS Fleet Advisor를 사용하여 AWS 클라우드로 마이그레이션할 소스 온프레미스 데이터베이스를 검색할 수 있습니다. 그러면 각 온프레미스 데이터베이스에 적합한 마이그레이션 대상을 AWS 클라우드에서 결정할 수 있습니다. 다음 워크플로를 사용하여 소스 데이터베이스의 인벤토리를 만들고 대상 권장 사항을 생성하십시오.

1. Amazon S3 버킷, IAM 정책, 역할 및 사용자를 생성합니다. 자세한 내용은 필요한 리소스 생성 섹션을 참조하세요.
2. DMS 데이터 수집기에 필요한 최소 권한을 가진 데이터베이스 사용자를 생성하십시오. 자세한 내용은 데이터베이스 사용자 생성 섹션을 참조하세요.
3. 데이터 수집기를 생성하고 다운로드합니다. 자세한 내용은 데이터 수집기 생성 섹션을 참조하세요.
4. 로컬 환경에 데이터 수집기를 설치하십시오. 그런 다음, 수집된 데이터를 DMS Fleet Advisor로 전송할 수 있도록 데이터 수집기를 구성하십시오. 자세한 내용은 데이터 수집기 설치 섹션을 참조하세요.
5. 데이터 환경의 OS 및 데이터베이스 서버를 확인해 보십시오. 자세한 내용은 OS 및 데이터베이스 서버 검색 섹션을 참조하세요.
6. 데이터베이스 메타데이터와 리소스 사용률 지표를 수집하세요. 자세한 내용은 데이터 수집 섹션을 참조하세요.
7. 소스 데이터베이스와 스키마를 분석하세요. DMS Fleet Advisor는 데이터베이스의 대규모 평가를 실행하여 유사한 스키마를 식별합니다. 자세한 내용은 AWS DMS Fleet Advisor에서 인벤토리를 분석에 사용하기 섹션을 참조하세요.
8. 소스 데이터베이스에 대한 대상 권장 사항의 로컬 사본을 생성, 확인 및 저장합니다. 자세한 내용은 대상 권장 사항 섹션을 참조하세요.

각 소스 데이터베이스의 마이그레이션 대상을 결정한 후에는 DMS Schema Conversion을 사용하여 데이터베이스 스키마를 새 플랫폼으로 변환할 수 있습니다. 그런 다음, 데이터를 마이그레이션하는 데 AWS DMS를 사용할 수 있습니다. 자세한 내용은 DMS Schema Conversion을 사용하여 데이터베이스 스키마 변환 및 AWS Database Migration Service란 무엇인가요? 섹션을 참조하세요.

이 동영상은 DMS Schema Conversion 사용자 인터페이스를 소개하고 이 서비스의 핵심 구성 요소를 숙지하는 데 도움이 됩니다f.

AWS DMS Fleet Advisor 설정

AWS DMS Fleet Advisor를 설정하려면 다음 사전 조건 작업을 완료하십시오.

주제

- AWS DMS Fleet Advisor에 필요한 AWS 리소스 생성
- AWS DMS Fleet Advisor용 데이터베이스 사용자 생성

AWS DMS Fleet Advisor에 필요한 AWS 리소스 생성

DMS Fleet Advisor가 인벤토리 정보를 전달하고 가져오며 DMS 데이터 수집기의 상태를 업데이트하려면 계정에 일련의 AWS 리소스가 필요합니다.

처음으로 데이터를 수집하고 데이터베이스 및 스키마의 인벤토리를 생성하려면 먼저 다음과 같은 사전 조건을 완료해야 합니다.

Amazon S3 버킷과 IAM 리소스를 구성하려면 다음 중 하나를 수행합니다.

- AWS CloudFormation을 사용하여 Amazon S3 및 IAM 리소스 구성(권장).
- AWS Management Console에서 Amazon S3 및 IAM 리소스 구성

AWS CloudFormation을 사용하여 Amazon S3 및 IAM 리소스 구성

CloudFormation 스택이란 하나의 단위로 관리할 수 있는 AWS 리소스의 모음입니다. DMS 플릿 어드바이저에 필요한 리소스를 간편하게 생성하기 위해 AWS CloudFormation 템플릿 파일을 사용하여 CloudFormation 스택을 생성할 수 있습니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 AWS CloudFormation 콘솔에서 스택 생성을 참조하세요.

> **Note**
>
> 이 섹션은 독립형 DMS Fleet Advisor 수집기 사용에만 적용됩니다. 단일 온프레미스 수집기를 사용하여 데이터베이스와 서버 모두에 대한 정보를 수집하는 방법에 대한 자세한 내용은 AWS Application Discovery Service 사용 설명서의 Application Discovery Service Agentless Collector를 참조하세요.

에서 생성한 Amazon S3 및 IAM 리소스 CloudFormation

CloudFormation 템플릿을 사용하면 다음과 같은 리소스가 포함된 스택이 생성됩니다. AWS 계정

- `dms-fleetadvisor-data-`*`accountId`*`-`*`region`*이라는 Amazon S3 버킷

- `FleetAdvisorCollectorUser-region`이라는 IAM 사용자 이름
- `FleetAdvisorS3Role-region`이라는 IAM 서비스 역할
- `FleetAdvisorS3Role-region-Policy`라는 액세스 정책
- `FleetAdvisorCollectorUser-region-Policy`라는 액세스 정책
- `AWSServiceRoleForDMSFleetAdvisor`라는 IAM 서비스 연결 역할(SLR)

아래 나열된 단계에 따라 리소스를 구성하십시오. CloudFormation

- 1단계: CloudFormation 템플릿 파일 다운로드
- 2단계: 다음을 사용하여 Amazon S3 및 IAM을 구성합니다. CloudFormation

1단계: CloudFormation 템플릿 파일 다운로드

CloudFormation 템플릿은 스택을 구성하는 AWS 리소스의 선언입니다. 템플릿은 JSON 파일로 저장됩니다.

CloudFormation 템플릿 파일을 다운로드하려면

1. 다음 링크 중 하나에 대한 컨텍스트(마우스 오른쪽 클릭) 메뉴를 열고 링크 저장을 선택합니다.
   - DMS 플릿 어드바이저를 사용하려면 dms-fleetadvisor-iam-slr-s3.zip 를 선택하십시오. DMS 플릿 어드바이저용 SLR을 이미 만든 경우 3.zip 를 선택하십시오. dms-fleetadvisor-iam-s
   - AWS애플리케이션 디스커버리 서비스 (ADS) 에이전트리스 컬렉터를 사용할 계획이고 이에 대한 SLR을 만들지 않았다면 -slr-s3.zip 를 선택하십시오. dms-fleetadvisor-ads-iam 이전에 ADS를 사용하여 DMS 플릿 어드바이저용 SLR을 만든 적이 있다면 -s3.zip 를 선택하십시오. dms-fleetadvisor-ads-iam
2. 파일을 컴퓨터에 저장합니다.

2단계: 다음을 사용하여 Amazon S3 및 IAM을 구성합니다. CloudFormation

IAM용 CloudFormation 템플릿을 사용하면 이전에 나열된 Amazon S3 및 IAM 리소스가 생성됩니다.

를 사용하여 Amazon S3 및 IAM을 구성하려면 CloudFormation

1. https://console.aws.amazon.com/cloudformation 에서 CloudFormation 콘솔을 엽니다.
2. 드롭다운 목록에서 스택 생성 및 새 리소스 사용을 선택하여 스택 생성 마법사를 시작합니다.
3. Create stack(스택 생성) 페이지에서 다음을 수행합니다.

a. Prepare template(템플릿 준비)에서 Template is ready(템플릿 준비가 완료되었습니다)를 선택합니다.

b. Template source(템플릿 소스)로 Upload a template file(템플릿 파일 업로드)을 선택합니다.

c. 파일 선택에서 다음으로 이동한 다음 -s3.json, dms-fleetadvisor-iam-slr-s3.json을 선택합니다. dms-fleetadvisor-iam dms-fleetadvisor-ads-iamdms-fleetadvisor-ads-iam, -slr-s3.zip 또는 -s3.zip.

d. Next(다음)를 선택합니다.

4. 스택 세부 정보 지정 페이지에서 다음 작업을 수행합니다.

a. 스택 이름에 **dms-fleetadvisor-iam-slr-s3**, **dms-fleetadvisor-iam-s3**, **dms-fleetadvisor-ads-iam-slr-s3** 또는 **dms-fleetadvisor-ads-iam-s3**를 입력합니다.

b. 다음을 선택합니다.

5. Configure stack options(스택 옵션 구성) 페이지에서 Next(다음)를 선택합니다.

6. 리뷰 dms-fleetadvisor-iam-slr -s3, 리뷰 dms-fleetadvisor-iam-s 3, 리뷰 dms-fleetadvisor-ads-iam -slr-s3 또는 리뷰 dms-fleetadvisor-ads-iam -s3 페이지에서 다음을 수행하십시오.

a. 사용자 지정 이름을 사용하여 AWS CloudFormation이 IAM 리소스를 생성할 수 있음에 동의합니다 확인란을 선택합니다.

b. Submit(제출)을 선택합니다.

CloudFormation DMS 플릿 어드바이저에 필요한 S3 버킷 및 IAM 역할 및 사용자를 생성합니다. 왼쪽 패널에서 -s3, dms-fleetadvisor-iam-s3, dms-fleetadvisor-iam-slrdms-fleetadvisor-ads-iam-slr-s3 또는 dms-fleetadvisor-ads-iam -s3에 CREATE_COMPLETE가 표시되면 다음 단계로 진행하십시오.

7. 왼쪽 패널에서 -s3, 3, -slr-s3 또는 -s3를 선택합니다dms-fleetadvisor-iam-slr. dms-fleetadvisor-iam-s dms-fleetadvisor-ads-iam dms-fleetadvisor-ads-iam 오른쪽 패널에서 다음을 수행합니다.

a. 스택 정보를 선택합니다. ***### ID# arn:aws:cloudformation: ##: account-no:stack/ -s3/ ###, arn:aws:cloudformation: ##: ## ##:stack/ 3/ ###, arn:aws:cloudformation: ##: ## ##:stack/ -slr-s3/ dms-fleetadvisor-iam-slr ### ## arn:AWS:cloudformation: region: account-formation: account-account: account-formation: account: account: account-formation: account: account ##: ##/ -s3/ ### dms-fleetadvisor-iam-s dms-fleetadvisor-ads-iam dms-fleetadvisor-ads-iam*** .

b. 리소스를 선택합니다. 다음과 같은 모양이어야 합니다.

- `dms-fleetadvisor-data-accountId-region`이라는 Amazon S3 버킷
- `FleetAdvisorS3Role-region`이라는 서비스 역할
- `FleetAdvisorCollectorUser-region`이라는 IAM 사용자 이름
- `AWSServiceRoleForDMSFleetAdvisor`라는 IAM SLR(`dms-fleet-advisor-iam-slr-s3.zip` 또는 `dms-fleet-advisor-ads-iam-slr-s3.zip`을 다운로드한 경우)
- `FleetAdvisorS3Role-region-Policy`라는 액세스 정책
- `FleetAdvisorCollectorUser-region-Policy`라는 액세스 정책

AWS Management Console에서 Amazon S3 및 IAM 리소스 구성

Amazon S3 버킷 생성

인벤토리 메타데이터를 저장할 수 있는 Amazon S3 버킷을 생성합니다. DMS Fleet Advisor를 사용하기 전에 이 S3 버킷을 미리 구성하는 것이 좋습니다. AWS DMS는 DMS Fleet Advisor 인벤토리 메타데이터를 이 S3 버킷에 저장합니다.

S3 버킷 만들기에 관한 자세한 내용은 Amazon S3 사용 설명서의 첫 번째 S3 버킷 생성을 참조하세요.

> **Note**
>
> DMS 플릿 어드바이저는 SSE-S3 암호화 버킷만 지원합니다.

Amazon S3 버킷을 생성하여 로컬 데이터 환경 정보를 저장하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 버킷 만들기를 선택합니다.
3. 버킷 만들기 페이지에서 버킷의 로그인 이름이 포함된 세계적으로 고유한 이름(예: fa-bucket-***yoursignin***)을 입력합니다.
4. DMS Fleet Advisor를 사용할 AWS 리전을 선택합니다.
5. 나머지 설정을 유지하고 버킷 만들기를 선택합니다.

IAM 리소스 생성

이 섹션에서는 데이터 수집기, IAM 사용자, DMS Fleet Advisor를 위한 IAM 리소스를 생성합니다.

주제

- 데이터 수집기를 위한 IAM 리소스 생성
- DMS Fleet Advisor 서비스 연결 역할 생성

데이터 수집기를 위한 IAM 리소스 생성

데이터 수집기가 올바르게 작동하는지 확인하고 수집된 메타데이터를 Amazon S3 버킷으로 업로드하려면 다음 정책을 생성하십시오. 그런 다음, 다음과 같은 최소한의 권한을 가진 IAM 사용자를 생성합니다. DMS 수집기에 관한 자세한 내용은 데이터 수집기를 사용하여 마이그레이션할 데이터베이스 검색 섹션을 참조하세요.

DMS Fleet Advisor 및 데이터 수집기가 Amazon S3에 액세스할 수 있도록 IAM 정책을 생성하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 생성 페이지에서 JSON 탭을 선택합니다.
5. 다음 JSON을 편집기에 붙여 넣으면서 예제 코드를 바꿉니다. *fa_bucket*을 이전 단원에서 생성한 Amazon S3 버킷의 이름으로 바꿉니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:GetBucket*",
                "s3:List*",
                "s3:DeleteObject*",
                "s3:PutObject*"
            ],
            "Resource": [
                "arn:aws:s3:::fa_bucket",
```

```
                "arn:aws:s3:::fa_bucket/*"
            ]
        }
    ]
}
```

6. 다음: 태그와 다음: 검토를 선택합니다.
7. 이름*의 경우 **FleetAdvisorS3Policy**를 입력한 후 정책 생성을 선택합니다.

DMS 데이터 수집기가 DMS Fleet Advisor에 액세스할 수 있도록 IAM 정책을 생성하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 생성 페이지에서 JSON 탭을 선택합니다.
5. 다음 JSON 코드를 편집기에 붙여 넣으면서 예제 코드를 바꿉니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dms:DescribeFleetAdvisorCollectors",
                "dms:ModifyFleetAdvisorCollectorStatuses",
                "dms:UploadFileMetadataList"
            ],
            "Resource": "*"
        }
    ]
}
```

6. 다음: 태그와 다음: 검토를 선택합니다.
7. 이름*의 경우 **DMSCollectorPolicy**를 입력한 후 정책 생성을 선택합니다.

DMS 데이터 수집기를 사용할 수 있는 최소한의 권한을 가진 IAM 사용자를 생성하려면

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 사용자를 선택합니다.
3. 사용자 추가를 선택합니다.
4. 사용자 추가 페이지의 사용자 이름*에 **`FleetAdvisorCollectorUser`**를 입력합니다. AWS 액세스 유형 선택에서 액세스 키 - 프로그래밍 방식 액세스를 선택합니다. 다음: 권한을 선택합니다.
5. 권한 설정 섹션에서 기존 정책 직접 연결을 선택합니다.
6. 검색 컨트롤을 사용하여 이전에 만든 DMS CollectorPolicy 및 FleetAdvisorS3 정책 정책을 찾아 선택할 수 있습니다. 다음: 태그를 선택합니다.
7. 태그 페이지에서 다음: 검토를 선택합니다.
8. 검토 페이지에서 사용자 생성을 선택합니다. 다음 페이지에서 .csv 다운로드를 선택하여 새 사용자 보안 인증을 저장합니다. 필요한 최소 액세스 권한을 얻으려면 DMS Fleet Advisor와 함께 이 보안 인증을 사용하십시오.

DMS Fleet Advisor 및 데이터 수집기가 Amazon S3에 액세스할 수 있도록 IAM 역할을 생성하려면

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 선택 페이지의 신뢰할 수 있는 엔터티 유형 아래에서 AWS 서비스를 선택합니다. 기타 AWS 서비스 사용 사례에서 DMS를 선택합니다.
5. DMS 확인란을 선택하고 다음을 선택합니다.
6. 권한 추가 페이지에서 S3Policy를 선택합니다. FleetAdvisor 다음을 선택합니다.
7. 이름 지정, 검토 및 생성 페이지에서 역할 이름에 **`FleetAdvisorS3Role`**을 입력하고 역할 생성을 선택합니다.
8. 역할 페이지에서 역할 이름에 **`FleetAdvisorS3Role`**을 입력합니다. S3Role을 선택합니다 FleetAdvisor.
9. FleetAdvisorS3Role 페이지에서 신뢰 관계 탭을 선택합니다. 신뢰 정책 편집을 선택합니다.
10. 신뢰 정책 편집 페이지에서 기존 텍스트를 대체하여 다음 JSON을 편집기에 붙여넣습니다.

```
{
 "Version": "2012-10-17",
 "Statement": [
    {
     "Sid": "",
     "Effect": "Allow",
     "Principal": {
       "Service": [
         "dms.amazonaws.com",
         "dms-fleet-advisor.amazonaws.com"
        ]
      },
     "Action": "sts:AssumeRole"
    }
  ]
}
```

위 정책은 AWS DMS가 Amazon S3 버킷에서 수집된 데이터를 가져오는 데 사용하는 서비스에 `sts:AssumeRole` 권한을 부여합니다.

11. 정책 업데이트를 선택합니다.

DMS Fleet Advisor 서비스 연결 역할 생성

DMS 플릿 어드바이저는 서비스 연결 역할을 사용하여 사용자의 Amazon CloudWatch 지표를 관리합니다. AWS 계정 DMS 플릿 어드바이저는 이 서비스 연결 역할을 사용하여 수집된 데이터베이스 성능 지표를 사용자를 대신하여 게시합니다. CloudWatch

DMS Fleet Advisor에 대한 서비스 연결 역할을 생성하려면

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다. 그런 다음 역할 생성을 선택합니다.
3. 신뢰할 수 있는 엔터티 유형에서 AWS 서비스를 선택합니다.
4. 기타 AWS 서비스 사용 사례에서 DMS – Fleet Advisor를 선택합니다.
5. DMS – Fleet Advisor 확인란을 선택하고 다음을 선택합니다.
6. 권한 추가 페이지에서 다음을 선택합니다.
7. 이름 지정, 검토 및 생성 페이지에서 역할 생성을 선택합니다.

또는 AWS API 또는 AWS CLI에서 이 서비스 연결 역할을 생성할 수 있습니다. 자세한 설명은 AWS DMS Fleet Advisor에 대한 서비스 연결 역할 생성 섹션을 참조하세요.

DMS Fleet Advisor의 서비스 연결 역할을 생성한 후에는 대상 권장 사항에서 소스 데이터베이스의 성능 지표를 확인할 수 있습니다. 또한 계정에서 이러한 지표를 확인할 수 있습니다. CloudWatch 자세한 설명은 대상 권장 사항 섹션을 참조하세요.

DMS Fleet Advisor 서비스 연결 역할에 필요한 IAM 정책을 생성하려면

서비스 연결 역할을 생성하는 데 필요한 최소 권한은 `DMSFleetAdvisorCreateServiceLinkedRolePolicy` 정책에 지정되어 있습니다. 서비스 연결 역할을 생성할 수 없는 경우 계정에 이 IAM 정책을 생성합니다.

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 생성 페이지에서 JSON 탭을 선택합니다.
5. 다음 JSON 코드를 편집기에 붙여 넣으면서 예제 코드를 바꿉니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/dms-fleet-
advisor.amazonaws.com/AWSServiceRoleForDMSFleetAdvisor*",
            "Condition": {"StringLike": {"iam:AWSServiceName": "dms-fleet-
advisor.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/dms-fleet-
advisor.amazonaws.com/AWSServiceRoleForDMSFleetAdvisor*"
        }
    ]
```

```
}
```

6. 다음: 태그와 다음: 검토를 선택합니다.
7. 이름*의 경우 **DMSFleetAdvisorCreateServiceLinkedRolePolicy**를 입력한 후 정책 생성을 선택합니다.

이제 이 정책을 사용하여 DMS Fleet Advisor에 대한 서비스 연결 역할을 생성할 수 있습니다.

AWS DMS Fleet Advisor용 데이터베이스 사용자 생성

이 섹션에서는 DMS 데이터 수집기에 필요한 최소 권한을 가진 소스 데이터베이스 사용자를 생성하는 방법을 설명합니다.

이 섹션은 다음 주제를 포함합니다:

- AWS DMS Fleet Advisor와 함께 데이터베이스 사용자 사용
- MySQL을 사용하여 데이터베이스 사용자 생성
- Oracle을 사용하여 데이터베이스 사용자 생성
- PostgreSQL을 사용하여 데이터베이스 사용자 생성
- Microsoft SQL Server를 사용하여 데이터베이스 사용자 생성
- 데이터베이스 사용자 삭제

AWS DMS Fleet Advisor와 함께 데이터베이스 사용자 사용

DMS 데이터 수집기에서는 `root` 이외의 데이터베이스 사용자를 사용할 수 있습니다. 인벤토리에 데이터베이스를 추가한 후 데이터 수집기를 실행하기 전에 사용자 이름과 암호를 지정하십시오. 인벤토리에 데이터베이스를 추가하는 방법에 관한 자세한 내용은 모니터링되는 객체 관리 단원을 참조하십시오.

DMS 데이터 수집기 사용을 마친 후에는 생성한 데이터베이스 사용자를 삭제할 수 있습니다. 자세한 설명은 데이터베이스 사용자 삭제 섹션을 참조하세요.

> **Important**
>
> 다음 예제에서는 *{your_user_name}*을 데이터베이스용으로 만든 데이터베이스 사용자 이름으로 바꿉니다. 그런 다음, *{your_password}*를 안전한 비밀번호로 바꾸십시오.

MySQL을 사용하여 데이터베이스 사용자 생성

MySQL 소스 데이터베이스에서 데이터베이스 사용자를 생성하려면 다음 스크립트를 사용합니다. MySQL 데이터베이스 버전에 따라 한 가지 버전의 GRANT 문을 유지해야 합니다.

```
CREATE USER {your_user_name} identified BY '{your_password}';

GRANT PROCESS ON *.* TO {your_user_name};
GRANT REFERENCES ON *.* TO {your_user_name};
GRANT TRIGGER ON *.* TO {your_user_name};
GRANT EXECUTE ON *.* TO {your_user_name};

# For MySQL versions lower than 8.0, use the following statement.
GRANT SELECT, CREATE TEMPORARY TABLES ON `temp`.* TO {your_user_name};

# For MySQL versions 8.0 and higher, use the following statement.
GRANT SELECT, CREATE TEMPORARY TABLES ON `mysql`.* TO {your_user_name};

GRANT SELECT ON performance_schema.* TO {your_user_name};

SELECT
    IF(round(Value1 + Value2 / 100 + Value3 / 10000, 4) > 5.0129, 'GRANT EVENT ON *.*
 TO {your_user_name};', 'SELECT ''Events are not applicable'';') sql_statement
INTO @stringStatement
FROM (
       SELECT
         substring_index(ver, '.', 1)                              value1,
         substring_index(substring_index(ver, '.', 2), '.', - 1) value2,
         substring_index(ver, '.', - 1)                            value3
       FROM  (
         SELECT
           IF((@@version regexp '[^0-9\.]+') != 0, @@innodb_version, @@version) AS ver
         FROM dual
       ) vercase
     ) v;

PREPARE sqlStatement FROM @stringStatement;
SET @stringStatement := NULL;
EXECUTE sqlStatement;
DEALLOCATE PREPARE sqlStatement;
```

Oracle을 사용하여 데이터베이스 사용자 생성

Oracle 소스 데이터베이스에서 데이터베이스 사용자를 생성하려면 다음 스크립트를 사용합니다.

이 SQL 스크립트를 실행하려면 SYSDBA 권한을 사용하여 Oracle 데이터베이스에 연결하십시오. 이 SQL 스크립트를 실행한 후 이 스크립트로 생성한 사용자의 보안 인증을 사용하여 데이터베이스에 연결합니다. 또한 이 사용자의 보안 인증을 사용하여 DMS 데이터 수집기를 실행할 수 있습니다.

다음 스크립트는 Oracle 멀티테넌트 컨테이너 데이터베이스(CDB)의 사용자 이름에 C## 접두사를 추가합니다.

```
CREATE USER {your_user_name} IDENTIFIED BY "{your_password}";
GRANT CREATE SESSION TO {your_user_name};
GRANT SELECT ANY DICTIONARY TO {your_user_name};
GRANT SELECT ON DBA_WM_SYS_PRIVS TO {your_user_name};
BEGIN
    DBMS_NETWORK_ACL_ADMIN.CREATE_ACL(
        acl => UPPER('{your_user_name}') || '_Connect_Access.xml',
        description => 'Connect Network',
        principal => UPPER('{your_user_name}'),
        is_grant => TRUE,
        privilege => 'resolve',
        start_date => NULL,
        end_date => NULL);

    DBMS_NETWORK_ACL_ADMIN.ASSIGN_ACL(
        acl => UPPER('{your_user_name}') || '_Connect_Access.xml',
        host => '*',
        lower_port => NULL,
        upper_port => NULL);
END;
```

PostgreSQL을 사용하여 데이터베이스 사용자 생성

PostgreSQL 소스 데이터베이스에서 데이터베이스 사용자를 생성하려면 다음 스크립트를 사용합니다.

```
CREATE USER "{your_user_name}" WITH LOGIN PASSWORD '{your_password}';
GRANT pg_read_all_settings TO "{your_user_name}";

-- For PostgreSQL versions 10 and higher, add the following statement.
GRANT EXECUTE ON FUNCTION pg_ls_waldir() TO "{your_user_name}";
```

Microsoft SQL Server를 사용하여 데이터베이스 사용자 생성

Microsoft SQL Server 소스 데이터베이스에서 데이터베이스 사용자를 생성하려면 다음 스크립트를 사용합니다.

```
USE master
GO

IF NOT EXISTS (SELECT * FROM sys.sql_logins WHERE name = N'{your_user_name}')

 CREATE LOGIN [{your_user_name}] WITH PASSWORD=N'{your_password}',
 DEFAULT_DATABASE=[master], DEFAULT_LANGUAGE=[us_english], CHECK_EXPIRATION=OFF,
 CHECK_POLICY=OFF

GO

GRANT VIEW SERVER STATE TO [{your_user_name}]

GRANT VIEW ANY DEFINITION TO [{your_user_name}]

GRANT VIEW ANY DATABASE TO [{your_user_name}]

IF LEFT(CONVERT(SYSNAME,SERVERPROPERTY('ProductVersion')), CHARINDEX('.',
 CONVERT(SYSNAME,SERVERPROPERTY('ProductVersion')), 0)-1) >= 12
 EXECUTE('GRANT CONNECT ANY DATABASE TO [{your_user_name}]')

DECLARE @dbname VARCHAR(100)
DECLARE @statement NVARCHAR(max)

DECLARE db_cursor CURSOR
LOCAL FAST_FORWARD
FOR
  SELECT
    name
  FROM MASTER.sys.databases
  WHERE state = 0
    AND is_read_only = 0
        OPEN db_cursor
FETCH NEXT FROM db_cursor INTO @dbname
    WHILE @@FETCH_STATUS = 0
BEGIN

  SELECT @statement = 'USE '+ quotename(@dbname) +';'+ '
```

```
    IF NOT EXISTS (SELECT * FROM sys.syslogins WHERE name = ''{your_user_name}'') OR
 NOT EXISTS (SELECT * FROM sys.sysusers WHERE name = ''{your_user_name}'')
   CREATE USER [{your_user_name}] FOR LOGIN [{your_user_name}];

  EXECUTE sp_addrolemember N''db_datareader'', [{your_user_name}]'

  BEGIN TRY
    EXECUTE sp_executesql @statement
  END TRY
  BEGIN CATCH
    DECLARE @err NVARCHAR(255)

    SET @err = error_message()

    PRINT @dbname
    PRINT @err
  END CATCH

  FETCH NEXT FROM db_cursor INTO @dbname
END
CLOSE db_cursor
  DEALLOCATE db_cursor

USE msdb
GO

GRANT EXECUTE ON dbo.agent_datetime TO [{your_user_name}]
```

데이터베이스 사용자 삭제

모든 데이터 수집 작업을 완료한 후에는 DMS 데이터 수집기에 대해 생성한 데이터베이스 사용자를 삭제할 수 있습니다. 다음 스크립트를 사용하여 데이터베이스에서 최소 권한을 가진 사용자를 삭제할 수 있습니다.

MySQL 데이터베이스에서 사용자를 삭제하려면 다음 스크립트를 실행합니다.

```
DROP USER IF EXISTS "{your_user_name}";
```

Oracle 데이터베이스에서 사용자를 삭제하려면 다음 스크립트를 실행합니다.

```
DECLARE
  -- Input parameters, please set correct value
  cnst$user_name CONSTANT VARCHAR2(255) DEFAULT '{your_user_name}';
```

```
  -- System variables, please, don't change
  var$is_exists INTEGER DEFAULT 0;
BEGIN
  SELECT COUNT(hal.acl) INTO var$is_exists
  FROM dba_host_acls hal
  WHERE hal.acl LIKE '%' || UPPER(cnst$user_name) || '_Connect_Access.xml';
  IF var$is_exists > 0 THEN
    DBMS_NETWORK_ACL_ADMIN.DROP_ACL(
      acl => UPPER(cnst$user_name) || '_Connect_Access.xml');
  END IF;
  SELECT COUNT(usr.username) INTO var$is_exists
  FROM all_users usr
  WHERE usr.username = UPPER(cnst$user_name);
  IF var$is_exists > 0 THEN
    EXECUTE IMMEDIATE 'DROP USER ' || cnst$user_name || ' CASCADE';
  END IF;
END;
```

PostgreSQL 데이터베이스에서 사용자를 삭제하려면 다음 스크립트를 실행합니다.

```
DROP USER IF EXISTS "{your_user_name}";
```

SQL Server 데이터베이스에서 사용자를 삭제하려면 다음 스크립트를 실행합니다.

```
USE msdb
GO

REVOKE EXECUTE ON dbo.agent_datetime TO [{your_user_name}]

USE master
GO

DECLARE @dbname VARCHAR(100)
DECLARE @statement NVARCHAR(max)

DECLARE db_cursor CURSOR
LOCAL FAST_FORWARD
FOR
SELECT
  name
FROM MASTER.sys.databases
WHERE state = 0
```

```
  AND is_read_only = 0
    OPEN db_cursor
FETCH NEXT FROM db_cursor INTO @dbname
  WHILE @@FETCH_STATUS = 0
BEGIN

SELECT @statement = 'USE '+ quotename(@dbname) +';'+ '
  EXECUTE sp_droprolemember N''db_datareader'', [{your_user_name}]

  IF EXISTS (SELECT * FROM sys.syslogins WHERE name = ''{your_user_name}'')
    OR EXISTS (SELECT * FROM sys.sysusers WHERE name = ''{your_user_name}'')
    DROP USER [{your_user_name}];'

BEGIN TRY
EXECUTE sp_executesql @statement
END TRY
BEGIN CATCH
  DECLARE @err NVARCHAR(255)

  SET @err = error_message()

  PRINT @dbname
  PRINT @err
END CATCH

FETCH NEXT FROM db_cursor INTO @dbname
END
CLOSE db_cursor
  DEALLOCATE db_cursor

GO

IF EXISTS (SELECT * FROM sys.sql_logins WHERE name = N'{your_user_name}')
  DROP LOGIN [{your_user_name}]  -- Use for SQL login

GO
```

데이터 수집기를 사용하여 마이그레이션할 데이터베이스 검색

소스 데이터 인프라를 검색하려는 경우 AWS Application Discovery Service Agentless Collector 또는 AWS DMS 데이터 수집기를 사용할 수 있습니다. ADS Agentless Collector는 에이전트가 필요 없는 방법을 통해 서버 프로파일 정보(예: OS, CPU 수, RAM 용량), 데이터베이스 메타데이터, 사용률 지표 등

온프레미스 환경에 대한 정보를 수집하는 온프레미스 애플리케이션입니다. OVA(Open Virtualization Archive) 파일을 사용하여 Agentless Collector를 VMware vCenter Server 환경의 VM(가상 머신)으로 설치합니다. AWS DMS데이터 수집기는 로컬 환경에 설치하는 Windows 애플리케이션입니다. 이 애플리케이션은 데이터 환경에 연결하고 온프레미스 데이터베이스 및 분석 서버에서 메타데이터 및 성능 지표를 수집합니다. ADS Agentless Collector 또는 DMS 데이터 수집기를 통해 데이터베이스 메타데이터 및 성능 지표가 수집되면 DMS Fleet Advisor가 AWS 클라우드로 마이그레이션할 수 있는 서버, 데이터베이스 및 스키마의 인벤토리를 작성합니다.

DMS 데이터 수집기는.NET 라이브러리, 커넥터 및 데이터 공급자를 사용하여 데이터베이스 검색 및 데이터 수집을 위해 원본 데이터베이스에 연결하는 Windows 응용 프로그램입니다.

DMS 데이터 수집기는 Windows에서 실행됩니다. 하지만 DMS 데이터 수집기는 지원되는 데이터베이스 공급업체가 실행하는 OS 서버에 관계없이 지원되는 모든 데이터베이스 공급업체로부터 데이터를 수집할 수 있습니다.

DMS 데이터 수집기는 TLS 암호화를 적용한 보호된 RTPS 프로토콜을 사용하여 DMS Fleet Advisor와의 보안 연결을 설정합니다. 따라서 기본적으로 전송 중에 데이터가 암호화됩니다.

AWS DMS는 AWS 계정에 대해 생성할 수 있는 최대 수의 데이터 수집기가 있습니다. AWS DMS 서비스 할당량 AWS Database Migration Service에 대한 할당량에 관한 자세한 내용은 다음 섹션을 참조하세요.

주제

- DMS 데이터 수집기에 대한 권한
- AWS DMS Fleet Advisor용 데이터 수집기 만들기
- 데이터 수집기 설치 및 구성
- 모니터링할 OS 및 데이터베이스 서버 검색
- 모니터링되는 객체 관리
- AWS DMS Fleet Advisor에서 SSL 사용
- AWS DMS Fleet Advisor를 위한 데이터 수집
- DMS 데이터 수집기 문제 해결

DMS 데이터 수집기에 대한 권한

DMS 데이터 수집기를 위해 생성한 데이터베이스 사용자에게는 읽기 권한이 있어야 합니다. 그러나 경우에 따라서는 데이터베이스 사용자에게 EXECUTE 권한이 필요합니다. 자세한 설명은 AWS DMS Fleet Advisor용 데이터베이스 사용자 생성 섹션을 참조하세요.

검색 스크립트를 실행하려면 DMS 데이터 수집기에 추가 권한이 필요합니다.

- OS 검색의 경우, DMS 데이터 수집기에는 도메인 서버가 LDAP 프로토콜을 사용하여 요청을 실행하기 위한 보안 인증이 필요합니다.
- Linux에서 데이터베이스를 검색하려면 DMS 데이터 수집기에 `sudo` SSH 권한 부여가 포함된 보안 인증이 필요합니다. 또한 원격 SSH 스크립트를 실행할 수 있도록 Linux 서버를 구성해야 합니다.
- Windows에서 데이터베이스를 검색하려면 DMS 데이터 수집기가 WMI(Windows Management Instrumentation) 및 WQL(WMI Query Language) 쿼리를 실행하고 레지스트리를 읽을 수 있는 권한이 있는 보안 인증이 필요합니다. 또한 원격 WMI, WQL 및 스크립트를 실행할 수 있도록 Windows 서버를 구성해야 합니다. PowerShell

AWS DMS Fleet Advisor용 데이터 수집기 만들기

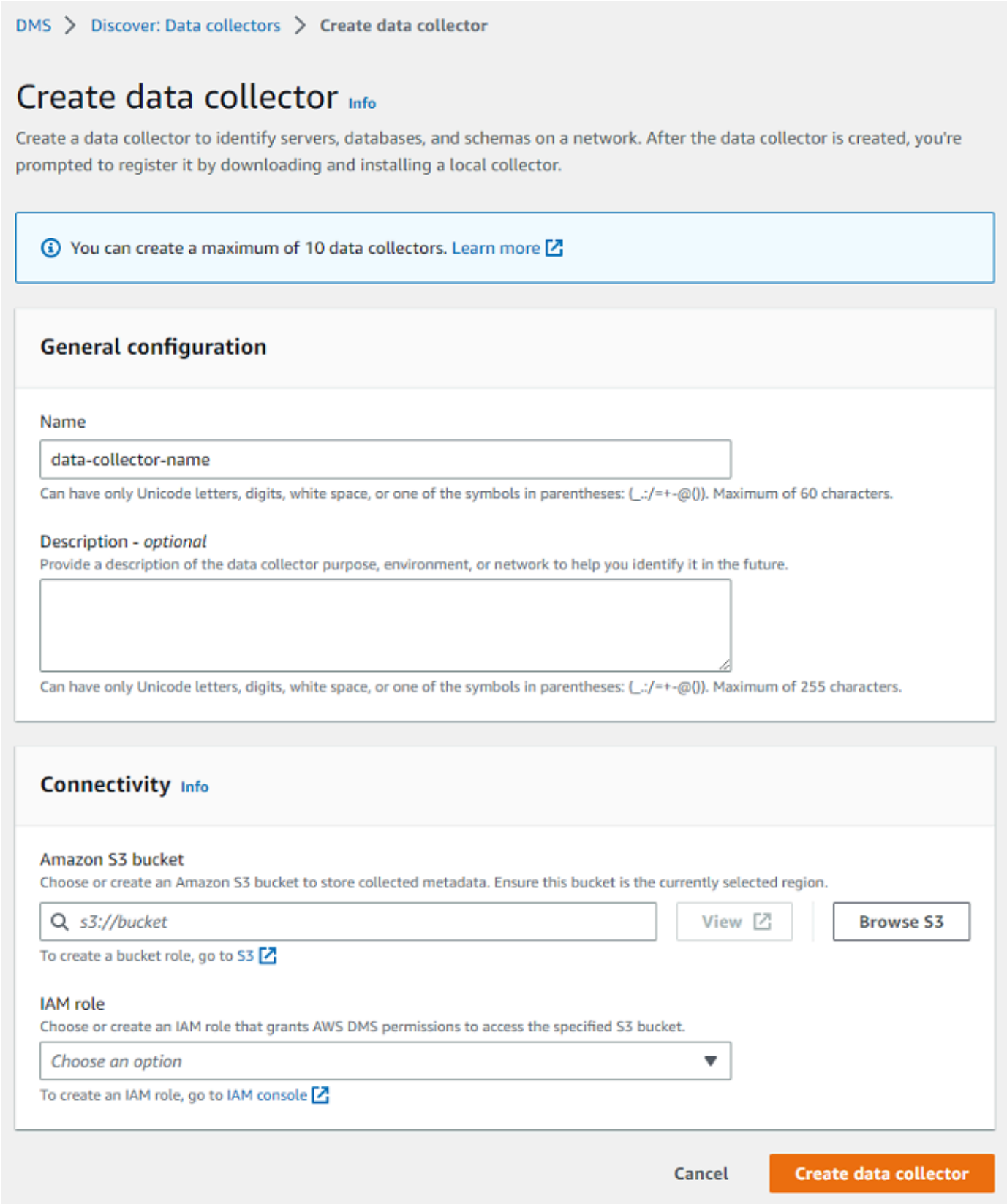
DMS 데이터 수집기를 생성하고 다운로드하는 방법을 알아봅니다.

데이터 수집기를 생성하기 전에 IAM 콘솔을 사용해 DMS Fleet Advisor에 대한 서비스 연결 역할을 생성해야 합니다. 이 역할을 통해 주체는 지표 데이터 포인트를 Amazon에 게시할 수 있습니다. CloudWatch DMS Fleet Advisor는 이 역할을 사용하여 데이터베이스 지표와 함께 차트를 표시합니다. 자세한 설명은 AWS DMS Fleet Advisor에 대한 서비스 연결 역할 생성 섹션을 참조하세요.

데이터 수집기를 생성하고 다운로드하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.

   DMS Fleet Advisor를 사용할 리전을 선택합니다.
2. 탐색 창에서 검색 아래에 있는 데이터 수집기를 선택합니다. 데이터 수집기 페이지가 열립니다.
3. 데이터 수집기 생성을 선택합니다. 데이터 수집기 생성 페이지가 열립니다.



4. 일반 구성 섹션의 이름에 데이터 수집기의 이름을 입력합니다.

5. 연결 섹션에서 Browse S3을 선택합니다. 표시되는 목록에서 사전 구성한 Amazon S3 버킷을 선택합니다.

   AWS DMS는 DMS Fleet Advisor 인벤토리 메타데이터를 이 S3 버킷에 저장합니다. AWS DMS Fleet Advisor가 현재 실행 중인 AWS 리전과 동일한 위치에 Amazon S3 버킷에 있는지 확인하십시오.

   > **Note**
   >
   > DMS 플릿 어드바이저는 SSE-S3 암호화 버킷만 지원합니다.

6. IAM 역할 목록에서 화면에 표시되는 해당 목록 중 사전 구성한 IAM 역할을 선택합니다. 이 역할은 지정된 Amazon S3 버킷에 대한 액세스 권한을 AWS DMS에 부여합니다.
7. 데이터 수집기 생성을 선택합니다. 데이터 수집기 페이지가 열리고 생성된 데이터 수집기가 목록에 나타납니다.

   첫 번째 데이터 수집기를 생성할 때 AWS DMS는 DMS Fleet Advisor에서 사용할 수 있도록 데이터 형식을 지정하고 속성을 저장하는 Amazon S3 버킷 환경을 구성합니다.
8. 정보 배너에서 로컬 수집기 다운로드를 선택하여 새로 생성된 데이터 수집기를 다운로드합니다. 다운로드가 진행 중임을 알리는 메시지가 나타납니다. 다운로드가 완료된 후 `AWS_DMS_Collector_Installer_`*`version_number`*`.msi` 파일에 액세스할 수 있습니다.

이제 클라이언트에 DMS 데이터 수집기를 설치할 수 있습니다. 자세한 설명은 데이터 수집기 설치 및 구성 섹션을 참조하세요.

데이터 수집기 설치 및 구성

DMS 데이터 수집기를 설치하는 방법, 데이터 전달 보안 인증을 지정하는 방법, 프로젝트에 LDAP 서버를 추가하는 방법을 알아봅니다.

다음 표에는 DMS 데이터 수집기를 설치하기 위한 하드웨어 및 소프트웨어 요구 사항이 설명되어 있습니다.

최소	권장
프로세서: CPU 벤치마크 점수가 8,000점 이상인 코어 2개	프로세서: CPU 벤치마크 점수가 11,000점 이상인 코어 4개

최소	권장
RAM: 8GB	RAM: 16GB
하드 드라이브 크기: 256GB	하드 드라이브 크기: 512GB
운영 체제: Microsoft Windows Server 2012 또는 이후 버전	운영 체제: Windows Server 2016 또는 이후 버전

네트워크의 클라이언트에 데이터 수집기를 설치하려면

1. .MSI 설치 관리자를 실행합니다. AWS DMS Fleet Advisor 수집기 설치 마법사 페이지가 나타납니다.
2. 다음을 선택합니다. 최종 사용자 라이선스 계약이 표시됩니다.
3. 최종 사용자 라이선스 계약을 읽고 동의하세요.
4. 다음을 선택합니다. 대상 폴더 페이지가 나타납니다.
5. 다음을 선택하여 기본 디렉터리에 데이터 수집기를 설치합니다.

   또는 변경을 선택하여 다른 설치 디렉터리를 입력합니다. 다음을 선택합니다.
6. 바탕 화면 바로 가기 페이지에서 확인란을 선택하여 바탕 화면에 아이콘을 설치합니다.
7. 설치를 선택합니다. 데이터 수집기는 선택한 디렉터리에 설치됩니다.
8. 완료된 DMS 수집기 설치 마법사 페이지에서 AWS DMS 수집기 시작을 선택한 후 완료를 선택합니다.

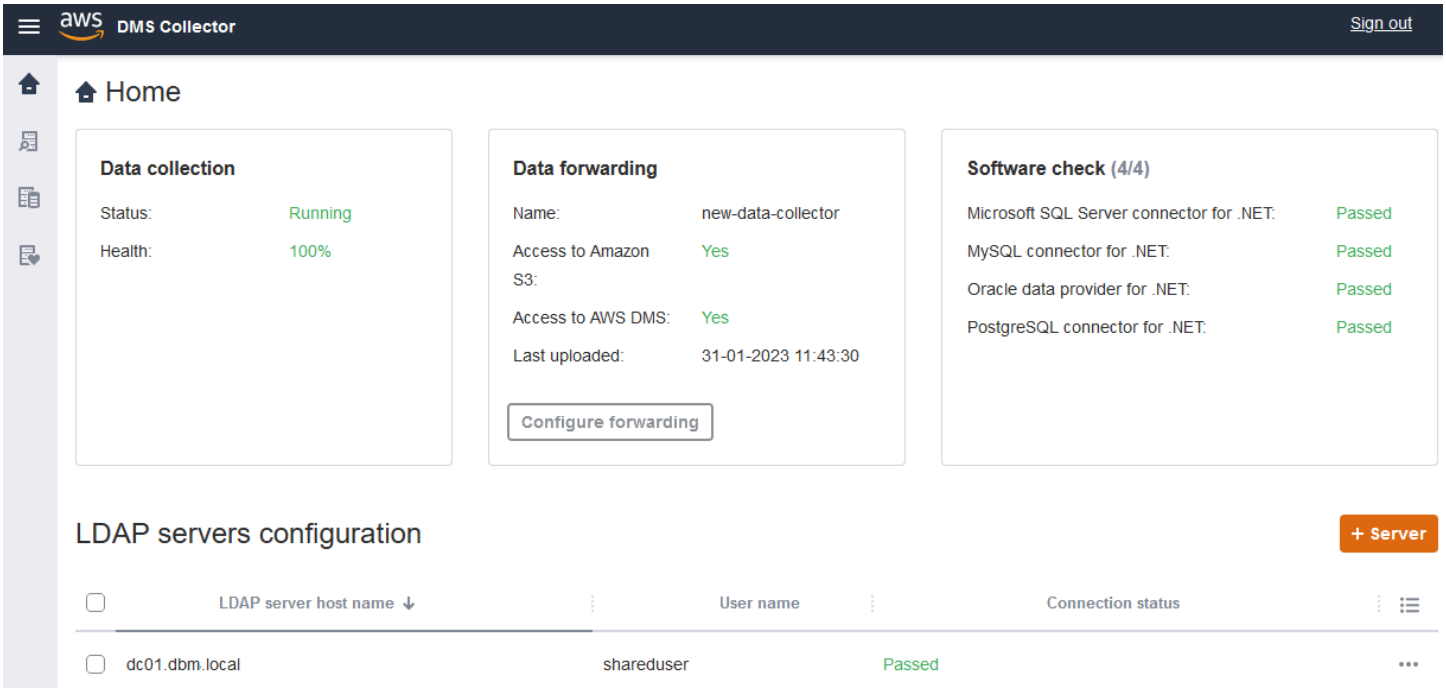
DMS 데이터 수집기는 .NET 라이브러리, 커넥터 및 데이터 공급자를 사용하여 소스 데이터베이스에 연결합니다. DMS 데이터 수집기 설치 프로그램은 지원되는 모든 데이터베이스에 필요한 이 소프트웨어를 서버에 자동으로 설치합니다.

데이터 수집기를 설치한 후 **`http://localhost:11000/`**을 주소로 입력하여 브라우저에서 실행할 수 있습니다. Microsoft Windows 시작 메뉴의 프로그램 목록에서 AWS DMS 수집기를 선택할 수도 있습니다. DMS 데이터 수집기를 처음 실행하면 보안 인증을 구성하라는 메시지가 표시됩니다. 데이터 수집기에 로그인하기 위한 사용자 이름과 암호를 생성합니다.

DMS 데이터 수집기 홈페이지에서 다음 상태 조건을 포함하여 메타데이터 수집을 준비하고 실행하기 위한 정보를 찾을 수 있습니다.

- 데이터 수집의 상태.

- Amazon S3 버킷 및 AWS DMS에 대한 접근성을 통해 데이터 수집기가 데이터를 AWS DMS로 전달할 수 있음.
- 설치된 데이터베이스 드라이버에 대한 연결.
- 초기 검색을 수행하기 위한 LDAP 서버의 보안 인증.



DMS 데이터 수집기는 LDAP 디렉터리를 사용하여 네트워크의 시스템 및 데이터베이스 서버에 관한 정보를 수집합니다. LDAP(Lightweight Directory Access Protocol)는 개방형 표준 애플리케이션 프로토콜입니다. LDAP는 IP 네트워크를 통해 분산 디렉터리 정보 서비스에 액세스하고 이 서비스를 유지 관리하는 데 사용됩니다. 시스템 인프라에 관한 정보를 검색하는 데 사용할 수 있는 데이터 수집기에 대한 기존 LDAP 서버를 프로젝트에 추가할 수 있습니다. 이렇게 하려면 +Server 옵션을 선택한 다음, 정규화된 도메인 이름(FQDN)과 도메인 컨트롤러의 보안 인증을 지정하십시오. 서버를 추가한 후 연결 확인을 검증합니다. 검색 프로세스를 시작하려면 모니터링할 OS 및 데이터베이스 서버 검색을 참조하십시오.

데이터 전달을 위한 보안 인증 구성

데이터 수집기를 설치한 후에는 이 애플리케이션이 수집된 데이터를 AWS DMS Fleet Advisor로 전송할 수 있는지 확인하십시오.

AWS DMS Fleet Advisor에서 데이터 전달을 위한 보안 인증을 구성하려면

1. DMS 데이터 수집기 홈페이지의 데이터 전달 섹션에서 전달 구성을 선택합니다. 데이터 전달을 위한 보안 인증 구성 대화 상자가 열립니다.
2. DMS Fleet Advisor를 사용할 AWS 리전 위치를 선택합니다.
3. 이전에 IAM 리소스를 생성할 때 얻은 AWS 액세스 키 ID와 AWS 비밀 액세스 키를 입력합니다. 자세한 설명은 IAM 리소스 생성 섹션을 참조하세요.
4. 데이터 수집기 찾아보기를 선택합니다.

   지정된 리전에서 데이터 수집기를 아직 생성하지 않았다면 계속 진행하기 전에 데이터 수집기를 생성하세요. 자세한 설명은 데이터 수집기 생성 섹션을 참조하세요.
5. 데이터 수집기 선택 창의 목록에서 데이터 수집기를 선택하고 선택을 선택합니다.
6. 데이터 전달을 위한 보안 인증 구성 대화 상자에서 저장을 선택합니다.

DMS 수집기 홈페이지의 데이터 전달 카드에서 Amazon S3에 대한 액세스 및 AWS DMS에 대한 액세스의 상태가 예로 설정되어 있는지 확인합니다.

Amazon S3에 대한 액세스 또는 AWS DMS에 대한 액세스의 상태가 아니요로 설정된 경우, Amazon S3 및 DMS Fleet Advisor에 액세스하기 위한 IAM 리소스를 생성했는지 확인하십시오. 필요한 모든 권한을 사용하여 이러한 IAM 리소스를 생성한 후 데이터 전달을 다시 구성하십시오. 자세한 설명은 IAM 리소스 생성 섹션을 참조하세요.

모니터링할 OS 및 데이터베이스 서버 검색

DMS 데이터 수집기를 사용하여 네트워크에서 사용 가능한 모든 서버를 찾아 나열할 수 있습니다. 네트워크에서 사용 가능한 모든 데이터베이스 서버를 검색하는 것이 좋지만 필수 사항은 아닙니다. 선택적으로 추가 데이터 수집을 위해 서버 목록을 수동으로 추가하거나 업로드할 수 있습니다. 서버 목록을 수동으로 추가하는 방법에 관한 자세한 내용은 모니터링되는 객체 관리 섹션을 참조하세요.

운영 체제(OS) 서버에서 데이터베이스를 검색하기 전에 먼저 모든 OS 서버를 검색하는 것이 좋습니다. OS 서버를 검색하려면 원격, 보안 셸 (SSH) PowerShell, Windows 관리 기기 (WMI) 스크립트와 명령을 실행할 수 있는 권한과 Windows 레지스트리에 대한 액세스 권한이 있어야 합니다. 네트워크에서 데이터베이스 서버를 검색하고 해당 서버에서 메타데이터를 수집하려면 원격 데이터베이스 연결을 위한 읽기 전용 관리자 권한이 필요합니다. 검색을 진행하기 전에 LDAP 서버를 추가했는지 확인하십시오. 자세한 설명은 데이터 전달을 위한 보안 인증 구성 섹션을 참조하세요.

DMS 데이터 수집기를 시작하려면 다음 작업을 완료하십시오.

- 네트워크에 있는 모든 OS 서버를 검색하십시오.
- 특정 OS 서버를 모니터링할 객체로 추가합니다.
- 모니터링되는 OS 서버의 연결을 확인합니다.
- OS 서버에서 실행되는 Microsoft SQL Server, MySQL, Oracle 및 PostgreSQL 데이터베이스를 검색합니다.
- 데이터 수집을 위해 데이터베이스 서버를 추가합니다.
- 모니터링되는 데이터베이스에 대한 연결을 확인합니다.

네트워크에서 모니터링할 수 있는 OS 서버를 검색하려면

1. DMS 데이터 수집기 탐색 창에서 검색을 선택합니다. 탐색 창을 표시하려면 DMS 데이터 수집기 홈페이지의 왼쪽 위 모서리에 있는 메뉴 아이콘을 선택합니다.

   검색 페이지가 열립니다.
2. OS 서버 탭이 선택되어 있는지 확인한 다음, 검색 실행을 선택합니다. 검색 파라미터 대화 상자가 표시됩니다.
3. 네트워크를 스캔하는 데 사용할 LDAP 서버를 입력합니다.
4. 검색 실행을 선택합니다. 이 페이지에는 데이터베이스를 실행하는지 여부에 관계없이 네트워크 내에서 검색된 모든 OS 서버의 목록이 표시됩니다.

   모든 OS 서버에서 데이터베이스 검색을 실행하기 전에 모든 OS 서버에 대해 검색을 실행하는 것이 좋습니다. 보안 인증을 사용하면 먼저 호스트 서버를 검색한 다음, 호스트 서버에 있는 데이터베이스를 검색할 수 있습니다. OS 서버에서 데이터베이스 검색을 실행하기 전에 필요하다면 먼저 OS 서버를 검색합니다. LDAP 서버가 네트워크의 OS 서버를 찾는 데 사용하는 보안 인증은 특정 OS 서버의 데이터베이스를 검색하는 데 필요한 보안 인증과 다를 수 있다는 점에 유의하십시오. 따라서 모니터링되는 객체에 OS 서버를 추가하고 보안 인증을 확인하여 필요하다면 수정한 다음, 계속 진행하기 전에 연결 상태를 확인하는 것이 좋습니다.

이제 네트워크에서 검색된 OS 서버의 목록에서 모니터링되는 객체에 추가할 서버를 선택할 수 있습니다.

OS 서버를 모니터링할 객체로 선택하려면

1. 검색 페이지에서 OS 서버 탭을 선택합니다.
2. 검색된 OS 서버 목록이 화면에 표시되면 이 목록에서 모니터링할 각 서버의 옆에 있는 확인란을 선택합니다.

3. 모니터링되는 객체에 추가를 선택합니다.

객체 모니터링 페이지에서 연결을 모니터링하고 확인할 OS 서버의 목록을 볼 수 있습니다.

모니터링할 서버로서 선택한 OS 서버의 연결을 확인하려면

1. DMS 데이터 수집기 탐색 창에서 모니터링되는 객체를 선택합니다.
2. 모니터링되는 객체 페이지에서 OS 서버 탭을 선택합니다. 모니터링할 서버로서 검색된 OS 서버의 목록이 나타납니다.
3. 열 상단의 확인란을 선택하여 목록에 나열된 OS 서버를 모두 선택합니다.
4. 작업을 선택한 다음, 연결 확인을 선택합니다. 각 서버 객체의 경우, 연결 상태 열에서 결과를 확인합니다.
5. 연결 상태가 성공 이외의 상태에 해당하는 서버를 선택합니다. 작업을 선택한 후, 편집을 선택합니다. 서버 편집 대화 상자가 열립니다.
6. 정보가 정확한지 확인하고 필요하다면 편집하십시오. 확인 또는 편집이 완료되면 저장을 선택합니다. 보안 인증 재정의 대화 상자가 열립니다.
7. 덮어쓰기를 선택합니다. DMS 데이터 수집기는 각 연결의 상태를 성공으로 확인하고 업데이트합니다.

이제 모니터링하도록 선택한 서버에 있는 데이터베이스를 검색할 수 있습니다.

서버에서 실행되는 데이터베이스를 검색

1. DMS 데이터 수집기 탐색 창에서 검색을 선택합니다.
2. 데이터베이스 서버 탭을 선택하고 검색 실행을 선택합니다. 검색 파라미터 대화 상자가 열립니다.
3. 검색 파라미터 대화 상자의 검색 기준에서 모니터링되는 객체 선택합니다. 서버의 경우 데이터베이스 검색을 실행할 OS 서버를 선택합니다.
4. 검색 실행을 선택합니다. 이 페이지에는 모니터링하기로 선택한 OS 서버에 있는 모든 데이터베이스 목록이 표시됩니다.

데이터베이스 주소, 서버 이름, 데이터베이스 엔진 등의 정보를 보면 모니터링할 데이터베이스를 선택하는 데 도움이 됩니다.

모니터링할 데이터베이스를 선택하려면

1. 검색 페이지에서 데이터베이스 서버 탭을 선택합니다.

2. 검색된 데이터베이스 목록이 화면에 표시되면 이 목록에서 모니터링할 모든 데이터베이스의 옆에 있는 확인란을 선택합니다.
3. 모니터링되는 객체에 추가를 선택합니다.

이제 모니터링하기로 선택한 데이터베이스와의 연결을 확인할 수 있습니다.

모니터링되는 데이터베이스에 대한 연결을 확인하려면

1. DMS 데이터 수집기 탐색 창에서 모니터링되는 객체를 선택합니다.
2. 모니터링되는 객체 페이지에서 데이터베이스 서버 탭을 선택합니다. 모니터링할 서버로서 선택하는 검색된 모든 데이터베이스 서버의 목록이 나타납니다.
3. 열 상단의 확인란을 선택하여 목록에 나열된 데이터베이스 서버를 모두 선택합니다.
4. 작업을 선택한 다음, 연결 확인을 선택합니다. 각 데이터베이스의 경우, 연결 상태 열에서 결과를 확인합니다.
5. 상태가 정의되지 않았거나(비어 있음) 상태이거나 실패 상태가 있는 연결을 선택합니다. 작업을 선택한 후, 편집을 선택합니다. 모니터링되는 객체 편집 대화 상자가 열립니다.
6. 로그인 및 암호 보안 인증을 입력한 다음, 저장을 선택합니다. 보안 인증 변경 대화 상자가 열립니다.
7. 덮어쓰기를 선택합니다. DMS 데이터 수집기는 각 연결의 상태를 성공으로 확인하고 업데이트합니다.

모니터링할 OS 서버 및 데이터베이스를 검색한 후, 모니터링되는 객체를 관리하는 작업을 수행할 수도 있습니다.

모니터링되는 객체 관리

OS 및 데이터베이스 서버 검색에 설명된 대로 서버 검색 프로세스를 실행할 때 모니터링할 객체를 선택할 수 있습니다. 또한 운영 체제(OS) 서버 및 데이터베이스 서버와 같은 객체를 수동으로 관리할 수 있습니다. 다음 작업을 수행하여 모니터링되는 객체를 관리할 수 있습니다.

- 모니터링할 새 객체 추가
- 기존 객체 제거
- 기존 객체 편집
- 모니터링할 객체 목록 내보내기 및 가져오기
- 객체와의 연결 확인

- 데이터 수집 시작

예를 들어, 모니터링할 객체를 수동으로 추가할 수 있습니다.

모니터링할 객체를 수동으로 추가하려면

1. 모니터링되는 객체 페이지에서 +서버를 선택합니다. 모니터링되는 객체 추가 대화 상자가 열립니다.
2. 서버에 관한 정보를 추가한 다음 저장을 선택합니다.

.csv 파일을 사용하여 모니터링할 객체의 큰 목록을 가져올 수도 있습니다. 다음 .csv 파일 형식을 사용하여 객체 목록을 DMS 데이터 수집기로 가져올 수 있습니다.

```
Hostname - Hostname or IP address of Monitored Object
Port - TCP port of Monitored Object
Engine: (one of the following)
        • Microsoft SQL Server
        • Microsoft Windows
        • Oracle Database
        • Linux
        • MySQL Server
        • PostgreSQL
Connection type: (one of the following)
        • Login/Password Authentication
        • Windows Authentication
        • Key-Based Authentication
Domain name:(Windows authentication)
        • Use domain name for the account
User name
Password
```

모니터링할 객체 목록이 있는 .csv 파일을 가져오려면

1. 가져오기를 선택합니다. 모니터링되는 객체 가져오기 페이지가 열립니다.
2. 가져오려는 .csv 파일을 찾은 후 다음을 선택합니다.

모든 객체를 볼 수 있으며 필요하다면 메타데이터 수집을 시작할 객체를 선택할 수 있습니다.

수동으로 추가한 데이터베이스에 OS 서버를 연결

DMS Fleet Advisor는 MySQL 및 PostgreSQL 데이터베이스에서 성능 지표를 직접 수집할 수 없습니다. 대상 권장 사항에 필요한 지표를 수집하기 위해 DMS Fleet Advisor는 데이터베이스가 실행되는 OS 지표를 사용합니다.

MySQL 및 PostgreSQL 데이터베이스를 모니터링되는 객체 목록에 수동으로 추가하는 경우, DMS 데이터 수집기는 이러한 데이터베이스가 실행되는 OS 서버를 식별할 수 없습니다. 이러한 문제가 있기 때문에 MySQL 및 PostgreSQL 데이터베이스를 OS 서버에 연결해야 합니다.

DMS Fleet Advisor가 자동으로 검색한 데이터베이스와 OS 서버를 수동으로 연결할 필요는 없습니다.

OS 서버를 데이터베이스에 연결하려면

1. DMS 데이터 수집기 탐색 창에서 모니터링되는 객체를 선택합니다.
2. 모니터링되는 객체 페이지에서 데이터베이스 서버 탭을 선택합니다. 데이터베이스 서버 목록이 나타납니다.
3. 수동으로 추가한 MySQL 또는 PostgreSQL 데이터베이스 서버 옆에 있는 확인란을 선택합니다.
4. 작업을 선택한 후, 편집을 선택합니다. 데이터베이스 편집 대화 상자가 열립니다.
5. 이 데이터베이스가 실행되는 OS 서버를 DMS 데이터 수집기가 이미 발견했다면 자동 감지를 선택합니다. DMS 데이터 수집기는 SQL 스크립트를 실행하여 데이터베이스가 실행되는 OS 서버를 자동으로 식별합니다. 그러면 DMS 데이터 수집기가 이 OS 서버를 데이터베이스에 연결합니다. 다음 단계를 건너뛴 후, 편집한 데이터베이스 구성을 저장합니다.

   DMS 데이터 수집기가 데이터베이스의 OS 서버를 자동으로 식별할 수 없는 경우, 올바른 보안 인증을 사용하고 데이터베이스 액세스 권한을 제공해야 합니다. 필요에 따라 OS 서버를 수동으로 추가할 수 있습니다.
6. OS 서버를 수동으로 추가하려면 +OS 서버 추가를 선택합니다. 호스트 OS 서버 추가 대화 상자가 열립니다.

   OS 서버에 관한 정보를 추가한 다음, 저장을 선택합니다.
7. 데이터베이스 편집 대화 상자에서 연결 확인을 선택하여 DMS 데이터 수집기가 OS 서버에 연결할 수 있는지 확인합니다.
8. 연결을 확인한 후 저장을 선택합니다.

소스 데이터베이스에 대한 관련 OS 서버를 변경하는 경우, DMS Fleet Advisor는 업데이트된 지표를 사용하여 권장 사항을 생성합니다. 하지만 Amazon CloudWatch 차트에는 데이터베이스 서버의 이전 데이터가 표시됩니다. CloudWatch 차트에 대한 자세한 내용은 을 참조하십시오권장 사항 세부 정보.

AWS DMS Fleet Advisor에서 SSL 사용

데이터를 보호하기 위해 AWS DMS Fleet Advisor는 SSL을 사용하여 데이터베이스에 액세스할 수 있습니다.

지원되는 데이터베이스

AWS DMS Fleet Advisor는 SSL을 사용하여 다음 데이터베이스에 액세스할 수 있도록 지원합니다.

- Microsoft SQL Server
- MySQL
- PostgreSQL

SSL 설정

SSL을 사용하여 데이터베이스에 액세스하려면 SSL을 지원하도록 데이터베이스 서버를 구성하십시오. 자세한 내용은 다음의 데이터베이스 설명서를 참조하세요.

- SQL Server: 데이터베이스 엔진에 암호화된 연결을 활성화
- MySQL: 암호화된 연결을 사용하도록 MySQL 구성
- PostgreSQL: SSL을 이용한 TCP/IP 연결의 보안

SSL을 사용하여 데이터베이스에 연결하려면 서버를 수동으로 추가할 때 서버 인증서 신뢰 및 SSL 사용을 선택합니다. MySQL 데이터베이스의 경우, 사용자 지정 인증서를 사용할 수 있습니다. 사용자 지정 인증서를 사용하려면 CA 확인 확인란을 선택합니다. 서버 추가에 관한 자세한 내용은 모니터링되는 객체 관리 섹션을 참조하세요.

SQL Server용 서버 인증 기관(CA) 인증서 확인

SQL Server용 서버 인증 기관(CA) 인증서의 유효성을 검사하려면 서버를 추가할 때 서버 인증서 신뢰를 지우십시오. 서버가 잘 알려진 CA를 사용하고 있으며 해당 CA가 OS에 기본적으로 설치되어 있다면 확인은 정상적으로 작동할 것입니다. DMS Fleet Advisor가 데이터베이스 서버에 연결할 수 없는 경우, 데이터베이스 서버에서 사용하는 CA 인증서를 설치하십시오. 자세한 내용은 클라이언트 구성 섹션을 참조하세요.

AWS DMS Fleet Advisor를 위한 데이터 수집

데이터 수집을 시작하려면 모니터링되는 객체 페이지에서 해당 객체를 선택하고 데이터 수집 실행을 선택합니다. DMS 데이터 수집기는 한 번에 최대 100개의 데이터베이스를 수집할 수 있습니다. 또한 DMS 데이터 수집기는 최대 8개의 병렬 스레드를 사용하여 사용자 환경의 데이터베이스에 연결할 수 있습니다. DMS 데이터 수집기는 이 8개의 스레드에서 최대 5개의 병렬 스레드를 사용하여 단일 데이터베이스 인스턴스에 연결할 수 있습니다.

> **Important**
>
> 데이터 수집을 시작하기 전에 DMS 데이터 수집기 홈페이지의 소프트웨어 검사 섹션을 확인하십시오. 모니터링하려는 모든 데이터베이스 엔진이 통과 상태인지 확인하십시오. 일부 데이터베이스 엔진이 실패 상태이고 모니터링되는 객체 목록에 해당 엔진이 포함된 데이터베이스 서버가 있는 경우, 계속 진행하기 전에 문제를 해결하십시오. 소프트웨어 검사 섹션에 나열된 실패 상태 옆에 위치한 팁을 확인할 수 있습니다.

DMS 데이터 수집기는 두 가지 모드 즉, 단일 실행 모드 또는 지속적 모니터링 모드로 작동할 수 있습니다. 데이터 수집을 시작하면 데이터 수집 실행 대화 상자가 열립니다. 이어서 다음 옵션 중 하나를 선택합니다.

메타데이터 및 데이터베이스 용량

DMS 데이터 수집기는 데이터베이스 또는 OS 서버에서 정보를 수집합니다. 여기에는 스키마, 버전, 에디션, CPU, 메모리 및 디스크 용량이 포함됩니다. 또한 DMS 데이터 수집기는 IOPS, I/O 처리량 및 활성 데이터베이스 서버 연결과 같은 메트릭을 수집하고 제공합니다. 이 정보를 기반으로 DMS Fleet Advisor에서 대상 권장 사항을 고려할 수 있습니다. 소스 데이터베이스가 과다 프로비저닝되거나 과소 프로비저닝된 경우, 대상 권장 사항도 과다 프로비저닝되거나 과소 프로비저닝됩니다.

이것은 기본 옵션에 해당됩니다.

메타데이터, 데이터베이스 용량 및 리소스 사용률

DMS 데이터 수집기는 메타데이터 및 데이터베이스 용량 정보 외에도 데이터베이스 또는 OS 서버의 CPU, 메모리 및 디스크 용량에 대한 실제 사용률 지표를 수집합니다. 또한 DMS 데이터 수집기는 IOPS, I/O 처리량 및 활성 데이터베이스 서버 연결과 같은 메트릭을 수집하여 제공합니다. 제공된 대상 권장 사항은 실제 데이터베이스 워크로드를 기반으로 하기 때문에 더 정확할 것입니다.

이 옵션을 선택하는 경우, 데이터 수집 기간을 설정합니다. 다음 7일 동안 데이터를 수집하거나 사용자 지정 범위를 1~60일로 설정할 수 있습니다.

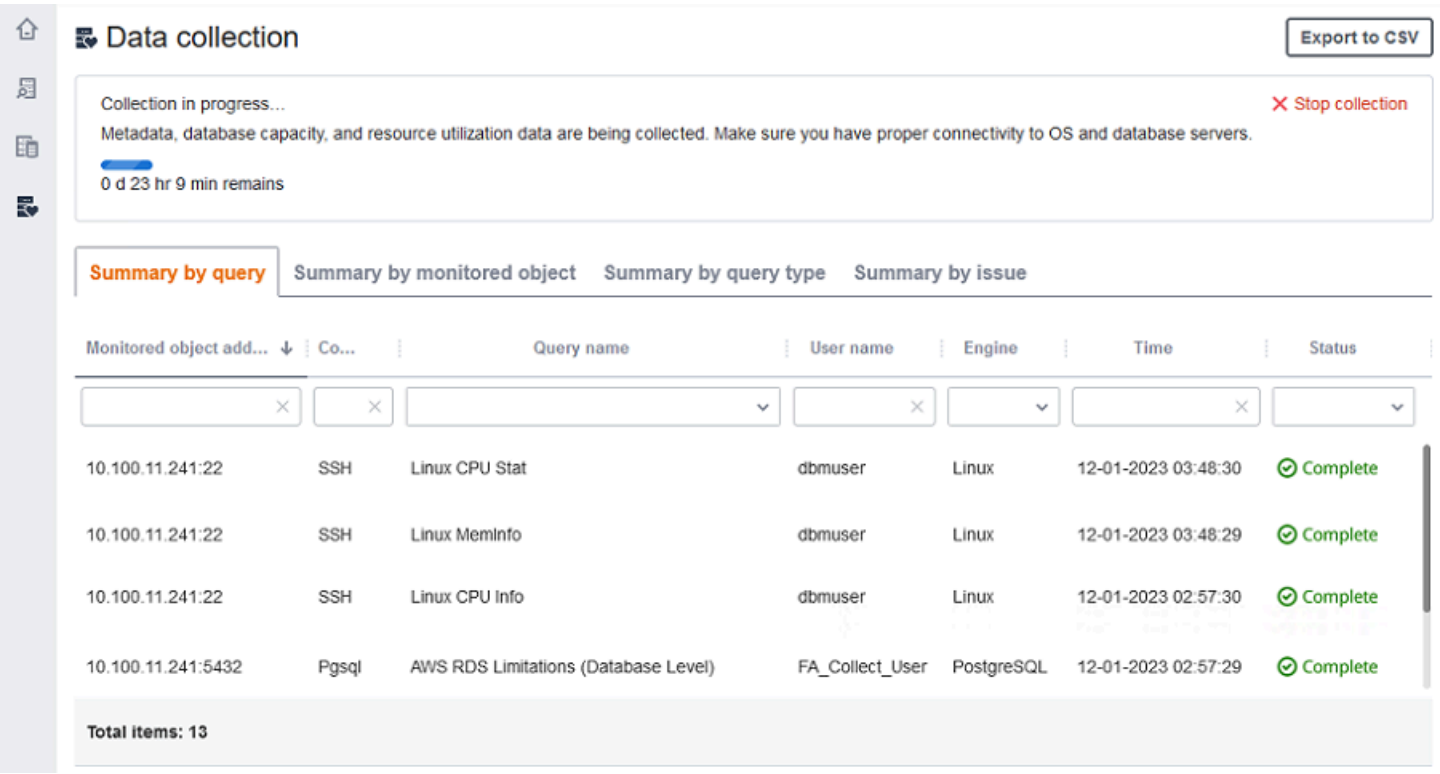
데이터 수집이 시작되면 데이터 수집 페이지로 리디렉션됩니다. 이 페이지에서 수집 쿼리의 실행 방식을 확인하고 실시간 진행 상황을 모니터링할 수 있습니다. 여기에서 전체 수집 상태를 보거나 DMS 데이터 수집기 홈페이지에서 확인할 수 있습니다. 전체 데이터 수집 상태가 100% 미만인 경우, 수집과 관련된 문제를 해결해야 할 수 있습니다.

메타데이터 및 데이터베이스 용량 모드에서 DMS 데이터 수집기를 실행하면 데이터 수집 페이지에서 완료된 쿼리 수를 확인할 수 있습니다.

메타데이터, 데이터베이스 용량 및 리소스 사용률 모드에서 DMS 데이터 수집기를 실행하면 DMS 데이터 수집기가 모니터링을 완료하기까지 남은 시간을 확인할 수 있습니다.

데이터 수집 페이지에서 각 객체의 수집 상태를 볼 수 있습니다. 무언가가 제대로 작동하지 않는 경우, 발생한 문제의 수를 보여주는 메시지가 나타납니다. 세부 정보를 확인하면 문제의 해결 방법을 결정하는 데 도움이 됩니다. 다음 탭에는 잠재적 문제들이 나열되어 있습니다.

- 쿼리별 요약 – Ping 테스트와 같은 테스트의 상태를 표시합니다. 상태 열에서 결과를 필터링할 수 있습니다. 상태 열은 데이터 수집 중에 발생한 오류 수를 나타내는 메시지를 제공합니다.
- 모니터링되는 객체별 요약 - 객체별 전체 상태를 보여줍니다.
- 쿼리 유형별 요약 - SQL, Secure Shell(SSH) 또는 WMI(Windows Management Instrumentation) 호출 등 수집기 쿼리 유형의 상태를 표시합니다.
- 문제별 요약 - 발생한 모든 고유한 문제를 문제 이름 및 각 문제의 발생 횟수와 함께 표시합니다.



수집 결과를 내보내려면 CSV로 내보내기를 선택합니다.

문제를 식별하고 해결한 후 수집 시작을 선택하고 데이터 수집 프로세스를 다시 실행합니다. 데이터 수집을 수행한 후 데이터 수집기는 보안 연결을 사용하여 수집된 데이터를 DMS Fleet Advisor 인벤토리에 업로드합니다. DMS Fleet Advisor는 Amazon S3 버킷에 정보를 저장합니다. 데이터 전달을 위한 보안 인증 구성에 관한 자세한 내용은 데이터 전달을 위한 보안 인증 구성를 참조하십시오.

AWS DMS Fleet Advisor를 통한 용량 및 리소스 사용률 지표 수집

메타데이터와 성능 지표를 두 가지 모드 즉, 단일 실행 모드 또는 지속적 모니터링 모드로 수집할 수 있습니다. 선택한 옵션에 따라 DMS 데이터 수집기는 데이터 환경에서 다양한 지표를 추적합니다. 단일 실행 모드 중 DMS 데이터 수집기는 데이터베이스 및 OS 서버의 메타데이터 지표만 추적합니다. 지속적 모니터링 모드 중 DMS 데이터 수집기는 리소스의 실제 사용률을 추적합니다.

AWS DMSDMS 데이터 수집기를 한 번 실행하는 동안 다음 메타데이터와 지표를 수집합니다.

- OS 서버의 가용 메모리
- OS 서버의 가용 스토리지
- 데이터베이스 버전 및 에디션
- OS 서버의 CPU 수
- 스키마 수

- 저장된 프로시저의 수
- 테이블 수
- 트리거 수
- 조회수
- 스키마 구조

DMS Fleet Advisor는 이러한 지표를 사용하여 데이터베이스 및 OS 서버의 인벤토리를 구축합니다. 또한 DMS Fleet Advisor는 이러한 메타데이터와 지표를 사용하여 소스 데이터베이스 스키마를 분석합니다.

DMS 플릿 어드바이저는 데이터 수집기를 한 번 실행하는 동안 수집된 지표를 사용하여 대상 권장 사항을 생성할 수 있습니다. 하지만 이 경우 오버프로비저닝된 원본 데이터베이스의 경우 대상 권장 사항도 오버프로비저닝됩니다. 따라서 의 리소스를 유지 관리하는 데 추가 비용이 발생합니다. AWS 클라우드 과소 프로비저닝된 소스 데이터베이스의 경우, 대상 권장 사항도 과소 프로비저닝되어 성능 문제가 발생할 수 있습니다. DMS 데이터 수집기의 메타데이터, 데이터베이스 용량 및 리소스 사용 모드를 선택하여 지속적인 모니터링을 통해 데이터를 수집하는 것이 좋습니다.

AWS DMS는 지속적 모니터링 중에 다음 지표를 수집합니다. 1~60일 동안 DMS 데이터 수집기를 실행할 수 있습니다.

- 데이터베이스 서버의 I/O 처리량
- 데이터베이스 서버의 초당 입출력 작업 처리량(IOPS)
- OS 서버에서 사용하는 CPU의 수
- OS 서버의 메모리 사용량
- 활성 데이터베이스 및 OS 서버 연결 수

DMS Fleet Advisor는 이러한 지표를 사용하여 정확한 대상 권장 사항을 생성하므로 대상 데이터베이스가 성능 요구 사항을 충족할 수 있습니다. 이렇게 하면 에서 리소스를 유지 관리하는 데 발생하는 추가 비용을 방지할 수 있습니다. AWS 클라우드

AWS DMS Fleet Advisor는 용량 및 리소스 사용률 지표를 어떻게 수집하나요?

DMS Fleet Advisor는 1분마다 성능 지표를 수집합니다.

Oracle 및 SQL Server의 경우, DMS Fleet Advisor는 SQL 쿼리를 실행하여 각 데이터베이스 지표의 값을 캡처합니다.

MySQL 및 PostgreSQL의 경우, DMS Fleet Advisor는 데이터베이스가 실행되는 OS 서버에서 성능 지표를 수집합니다. Windows에서 DMS Fleet Advisor는 WQL(WMI Query Language) 스크립트를 실행하고 WMI 데이터를 수신합니다. Linux에서 DMS Fleet Advisor는 OS 서버 지표를 캡처하는 명령을 실행합니다.

> **Important**
>
> 원격 SQL 스크립트를 실행하면 프로덕션 데이터베이스의 성능에 영향을 미칠 수 있습니다. 하지만 데이터 수집 쿼리에는 계산 로직이 포함되어 있지 않습니다. 따라서 데이터 수집 프로세스에서는 데이터베이스 리소스의 1% 이상을 사용할 가능성이 없습니다.

데이터 수집기가 지표를 수집하기 위해 실행하는 모든 쿼리를 볼 수 있습니다. 이렇게 하려면 `DMSCollector.Collections.json` 파일을 여십시오. 이 파일은 데이터 수집기를 설치한 동일한 폴더에 있는 `etc` 폴더에서 찾을 수 있습니다. 기본 경로는 `C:\ProgramData\Amazon\AWS DMS Collector\etc\DMSCollector.Collections.json`입니다.

DMS 데이터 수집기는 수집된 모든 데이터의 임시 저장소로 로컬 파일 시스템을 사용합니다. DMS 데이터 수집기는 수집된 데이터를 JSON 형식으로 저장합니다. 오프라인 모드에서 로컬 수집기를 사용하고 데이터 전달을 구성하기 전에 수집된 파일을 수동으로 검사하거나 확인할 수 있습니다. 수집된 모든 파일은 DMS 데이터 수집기를 설치한 동일한 폴더에 있는 `out` 폴더에서 확인할 수 있습니다. 기본 경로는 `C:\ProgramData\Amazon\AWS DMS Collector\out`입니다.

> **Important**
>
> DMS 데이터 수집기를 오프라인 모드에서 실행하고 수집된 데이터를 서버에 14일 이상 저장하는 경우 CloudWatch Amazon을 사용하여 이러한 지표를 표시할 수 없습니다. 하지만 DMS Fleet Advisor는 여전히 이 데이터를 사용하여 권장 사항을 생성합니다. CloudWatch 차트에 대한 자세한 내용은 을 참조하십시오권장 사항 세부 정보.

온라인 모드에서 수집된 데이터 파일을 검사하거나 확인할 수도 있습니다. DMS 데이터 수집기는 모든 데이터를 DMS 데이터 수집기 설정에서 지정한 Amazon S3 버킷으로 전달합니다.

DMS 데이터 수집기를 사용하여 온프레미스 데이터베이스에서 데이터를 수집할 수 있습니다. 또한 Amazon RDS 및 Aurora 데이터베이스에서 데이터를 수집할 수 있습니다. 하지만 Amazon RDS 또는 Aurora와 온프레미스 DB 인스턴스 간의 차이로 인해 클라우드에서 모든 DMS 데이터 수집기 쿼리를 성공적으로 실행할 수는 없습니다. DMS 데이터 수집기는 호스트 OS에서 MySQL 및 PostgreSQL 데

이터베이스의 사용률 지표를 수집하기 때문에 Amazon RDS 및 Aurora에서는 이 방법을 사용할 수 없습니다.

DMS 데이터 수집기 문제 해결

다음 목록에서는 데이터 수집기로 데이터를 수집하는 동안 특정 문제가 발생할 때 취할 수 있는 조치를 찾을 수 있습니다.

주제

- 네트워크 및 서버 연결과 관련된 데이터 수집 문제
- WMI(Windows Management Instrumentation)와 관련된 데이터 수집 문제
- Windows 웹 페이지 작성기와 관련된 데이터 수집 문제
- SSL과 관련된 데이터 수집 문제

네트워크 및 서버 연결과 관련된 데이터 수집 문제

NET: ping 요청 중에 예외가 발생했습니다.

컴퓨터의 이름을 확인하여 IP 주소로 확인할 수 없는 상태인지 확인합니다.

예를 들어 컴퓨터가 꺼져 있는지, 네트워크 연결이 끊겼는지, 사용 중지되었는지 확인합니다.

NET: 시간 초과됨

인바운드 방화벽 규칙 '파일 및 프린터 공유(에코 요청 - ICMPv4-In)'를 활성화합니다. 예:

```
* Inbound ICMPv4
```

NET: DestinationHostUnreachable

컴퓨터의 IP 주소를 확인합니다. 특히 DMS 데이터 수집기를 실행하는 컴퓨터와 동일한 서브넷에 있는지, 주소 확인 프로토콜(ARP) 요청에 응답하는지 확인하십시오.

컴퓨터가 다른 서브넷에 있는 경우, 게이트웨이의 IP 주소를 미디어 액세스 제어(MAC) 주소로 확인할 수 없습니다.

또한 컴퓨터가 꺼져 있는지, 네트워크 연결이 끊겼는지, 사용 중지되었는지 확인합니다.

WMI(Windows Management Instrumentation)와 관련된 데이터 수집 문제

WMI: RPC 서버를 사용할 수 없습니다. (HRESULT 예외: 0x800706BA)

인바운드 방화벽 규칙 'Windows Management Instrumentation(DCOM–in)'을 활성화합니다. 예:

`* Inbound TCP/IP at local port 135.`

또한 인바운드 방화벽 규칙 'Windows Management Instrumentation(WMI–In)'도 활성화합니다. 예:

`* Inbound TCP/IP at local port 49152 – 65535`: Windows Server 2008 및 이후 버전용.

`* Inbound TCP/IP at local port 1025 – 5000`: Windows Server 2003 및 이전 버전용.

WMI: 액세스가 거부되었습니다. (HRESULT 예외: 0x80070005(E_ACCESSDENIED))

다음을 시도해 보십시오.

- DMS 데이터 수집기 사용자를 Windows 그룹, 분산 COM 사용자 또는 관리자에 추가합니다.
- WMI(Windows Management Instrumentation) 서비스를 시작하고 시작 유형을 자동으로 설정합니다.
- DMS 데이터 수집기 사용자 이름이 \ 형식인지 확인하세요.

WMI: 액세스가 거부됨

루트 WMI 네임스페이스의 DMS 데이터 수집기 사용자에게 원격 활성화 권한을 추가합니다.

고급 설정을 사용하고 권한이 '이 네임스페이스 및 하위 네임스페이스'에 적용되는지 확인하십시오.

WMI: 메시지 필터에 의해 호출이 취소되었습니다. (HRESULT 예외: 0x80010002...)

WMI(Windows Management Instrumentation) 서비스를 다시 시작합니다.

Windows 웹 페이지 작성기와 관련된 데이터 수집 문제

WPC: 네트워크 경로를 찾을 수 없습니다

인바운드 방화벽 규칙 '파일 및 프린터 공유(SMB–in)'를 활성화합니다. 예:

`* Inbound TCP/IP at local port 445.`

또한 원격 레지스트리 서비스를 시작하고 시작 유형을 자동으로 설정합니다.

WPC: 액세스 거부됨

DMS 데이터 수집기 사용자를 성능 모니터 사용자 또는 관리자 그룹에 추가합니다.

WPC: 카테고리가 존재하지 않음

`loader /r`을 실행하여 성능 카운터 캐시를 재구축한 다음, 컴퓨터를 다시 시작합니다.

> **Note**
>
> AWS Database Migration Service(AWS DMS)을 사용하여 데이터를 마이그레이션할 때의 문제 해결에 관한 자세한 내용은 문제 해결 및 진단 지원을 참조하십시오.

SSL과 관련된 데이터 수집 문제

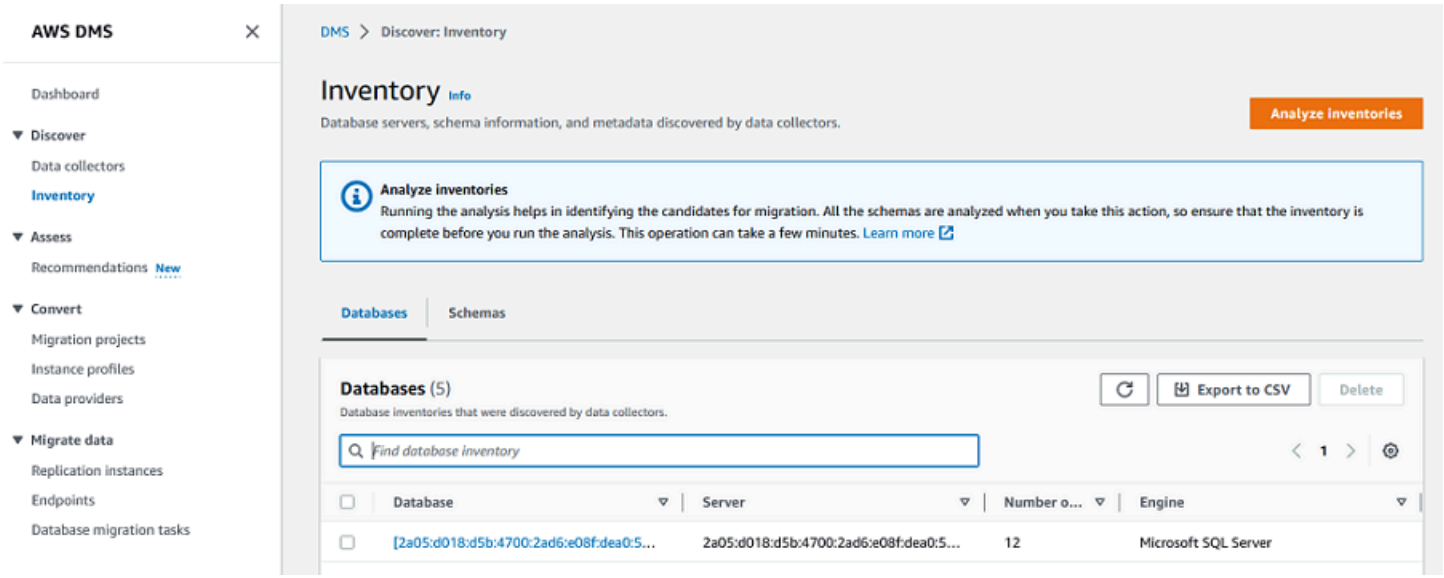
SSL 오류

데이터베이스에 보안 SSL 연결이 필요한데 해당 연결에 대해 CA 확인 및 SSL 사용 옵션을 활성화하지 않았습니다. 이 두 옵션을 활성화한 후 데이터베이스에서 사용하는 인증 기관(CA)이 로컬 OS에 설치되어 있는지 확인하십시오. 자세한 내용은 SSL 설정을(를) 참조하세요.

AWS DMS Fleet Advisor에서 인벤토리를 분석에 사용하기

검색된 데이터베이스 및 스키마의 인벤토리를 활용하면 잠재적 데이터베이스 마이그레이션의 타당성을 확인할 수 있습니다. 이러한 인벤토리의 정보를 사용하여 마이그레이션에 적합한 데이터베이스 및 스키마를 파악할 수 있습니다.

콘솔에서 데이터베이스 및 스키마 인벤토리에 액세스할 수 있습니다. 이렇게 하려면 콘솔에서 인벤토리를 선택합니다.



DMS Fleet Advisor는 데이터베이스 스키마를 분석하여 여러 스키마의 유사성을 확인합니다. 이 분석에서는 객체의 실제 코드를 비교하지 않습니다. DMS Fleet Advisor는 함수 및 프로시저와 같은 스키마 객체의 이름만 비교하여 서로 다른 데이터베이스 스키마에서 유사한 객체를 식별합니다.

주제

- 분석을 위한 데이터베이스 인벤토리 사용
- 분석을 위한 스키마 인벤토리 사용

분석을 위한 데이터베이스 인벤토리 사용

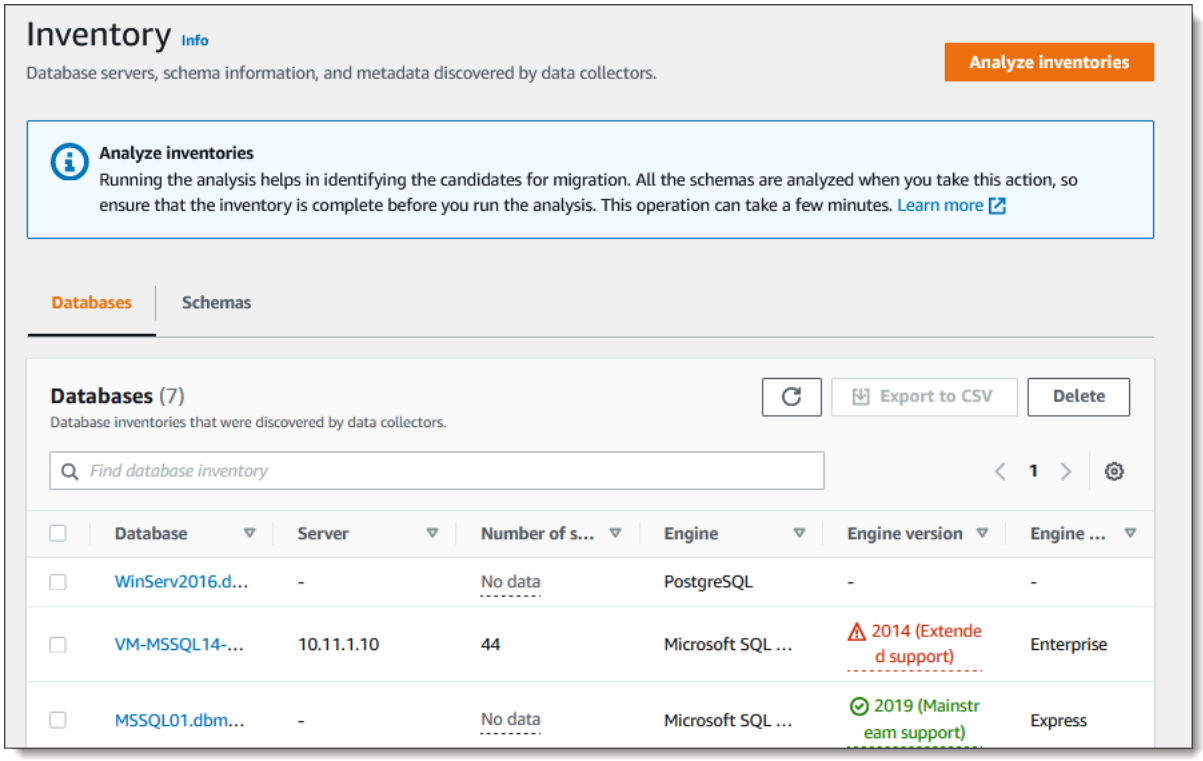
데이터가 수집된 네트워크 내 검색된 모든 서버의 모든 데이터베이스 목록을 보려면 다음 절차를 적용하십시오.

데이터를 수집한 출처에 해당하는 네트워크 서버의 데이터베이스 목록을 보려면

1. 콘솔에서 인벤토리를 선택합니다.

   인벤토리 페이지가 열립니다.
2. 데이터베이스 탭을 선택합니다.

   검색된 데이터베이스의 목록이 나타납니다.



3. 인벤토리 분석을 선택하여 유사성 및 복잡성과 같은 스키마 속성을 확인합니다. 프로세스에 걸리는 시간은 분석할 객체의 수에 따라 차이가 있으며 다만 1시간 이상 걸리지는 않습니다. 분석 결과는 인벤토리 페이지에 있는 스키마 탭에서 확인할 수 있습니다.

   DMS Fleet Advisor는 검색된 모든 데이터베이스의 스키마를 분석하여 객체의 교차점을 정의합니다. 분석 결과는 백분율로 표시됩니다. DMS Fleet Advisor는 객체 교차율이 50% 이상인 스키마를 중복으로 간주합니다. 원본 스키마는 중복이 발견된 스키마로 식별됩니다. 이렇게 하면 먼저 변환하거나 마이그레이션할 원본 스키마를 식별하는 데 도움이 됩니다.

   전체 인벤토리를 함께 분석하여 중복된 스키마를 식별합니다.

분석을 위한 스키마 인벤토리 사용

데이터가 수집된 네트워크 내 서버에서 검색된 데이터베이스 스키마의 목록을 볼 수 있습니다. 다음 절차를 수행하십시오.

데이터를 수집한 출처에 해당하는 네트워크 서버의 스키마 목록을 보려면

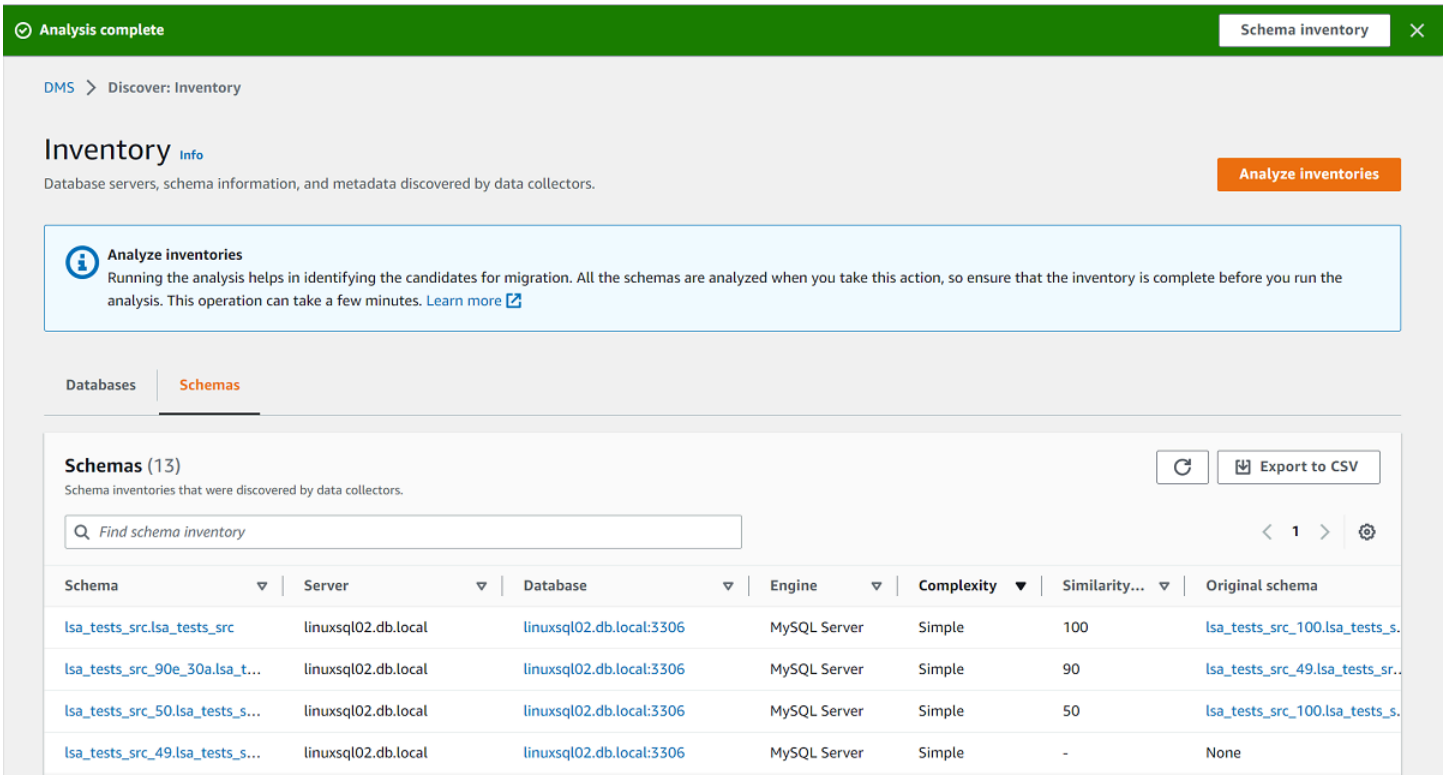
1. 콘솔에서 인벤토리를 선택합니다. 인벤토리 페이지가 열립니다.
2. 스키마 탭을 선택합니다. 스키마 목록이 나타납니다.

3. 목록에서 스키마를 선택하면 서버, 데이터베이스, 크기, 복잡성 등 스키마에 관한 정보를 볼 수 있습니다.

   각 스키마에 대해 객체 유형, 객체 수, 객체 크기 및 코드 줄에 관한 정보를 제공하는 객체 요약을 볼 수 있습니다.

4. (선택 사항) 인벤토리 분석을 선택하여 중복된 스키마를 식별합니다. DMS Fleet Advisor는 데이터베이스 스키마를 분석하여 해당 객체의 교차율을 정의합니다.

5. 추가 검토를 위해 인벤토리 정보를 `.csv` 파일로 내보낼 수 있습니다.



마이그레이션할 스키마를 식별하고 마이그레이션 대상을 결정하려면 AWS Schema Conversion Tool (AWS SCT) 또는 DMS 스키마 변환을 사용할 수 있습니다. 자세한 내용은 AWS SCT의 새 프로젝트 마법사 사용을 참조하십시오.

마이그레이션할 스키마를 식별한 후에는 AWS SCT 또는 DMS Schema Conversion을 사용하여 스키마를 변환할 수 있습니다. DMS Schema Conversion에 관한 자세한 내용은 DMS Schema Conversion을 사용하여 데이터베이스 스키마 변환 섹션을 참조하세요.

AWS DMS Fleet Advisor 대상 권장 기능 사용

최적의 마이그레이션 대상을 탐색하고 선택하기 위해 DMS Fleet Advisor에서 소스 온프레미스 데이터베이스에 대한 대상 권장 사항을 생성할 수 있습니다. 권장 사항에는 소스 온프레미스 데이터베이스의 마이그레이션을 위해 선택할 수 있는 하나 이상의 AWS 대상 엔진이 포함됩니다. 이러한 가능한 대상 엔진 중에서 DMS Fleet Advisor는 단일 대상 엔진을 적절한 크기의 마이그레이션 대상으로 제안하고 이 대상을 DMS 권장으로 표시합니다. 적정 규모의 마이그레이션 대상을 결정하기 위해 DMS Fleet Advisor는 데이터 수집기에서 수집한 인벤토리 메타데이터와 지표를 사용합니다.

마이그레이션을 시작하기 전에 권장 사항을 사용하여 마이그레이션 옵션을 검색하고 비용을 절감하며 위험을 줄일 수 있습니다. 권장 사항을 CSV(쉼표로 구분된 값) 파일로 내보내고 이를 주요 이해 관계자와 공유하여 의사 결정을 촉진할 수 있습니다. 권장 사항을 로 내보내 유지 관리 비용을 더욱 최적화할 수 있습니다. AWS Pricing Calculator 자세한 내용은 https://calculator.aws/#/를 참조하세요.

DMS Fleet Advisor에서는 대상 권장 사항을 수정할 수 없습니다. 따라서 가정 분석에는 DMS Fleet Advisor를 사용할 수 없습니다. 가정 분석은 대상 파라미터를 변경하여 이러한 변경이 권장 사항의 예상 요금에 어떤 영향을 미치는지 확인하는 프로세스입니다. AWS Pricing Calculator에서 권장 대상 파라미터를 출발점으로 사용하여 AWS Pricing Calculator에서 가정 분석을 실행할 수 있습니다. 자세한 내용은 https://calculator.aws/#/를 참조하세요.

DMS Fleet Advisor 권장 사항을 마이그레이션 계획의 출발점으로 삼는 것이 좋습니다. 그런 다음, 권장 인스턴스 파라미터를 변경하기로 결정하면 데이터베이스 워크로드의 비용 또는 성능을 최적화할 수 있습니다.

주제

- 권장 대상 인스턴스
- DMS Fleet Advisor는 권장 사항의 대상 인스턴스 사양을 어떻게 결정합니까?
- AWS DMS Fleet Advisor를 사용하여 대상 권장 사항 생성
- AWS DMS Fleet Advisor와 함께 대상 권장 사항의 세부 정보 살펴보기
- AWS DMS Fleet Advisor를 사용하여 대상 권장 사항 내보내기
- 플릿 어드바이저를 통한 마이그레이션 제한 발견 및 분석 AWS DMS
- 대상 권장 사항에 대한 문제 해결

권장 대상 인스턴스

대상 권장 사항의 경우, DMS Fleet Advisor는 메모리에 최적화되고 버스트 가능한 다음과 같은 범용 Amazon RDS DB 인스턴스를 고려합니다.

- db.m5
- db.m6i
- db.r5
- db.r6i
- db.t3
- db.x1
- db.x1e
- db.z1d

Amazon RDS DB 인스턴스 클래스에 관한 자세한 내용은 Amazon RDS 사용 설명서의 DB 인스턴스 클래스를 참조하십시오.

DMS Fleet Advisor는 권장 사항의 대상 인스턴스 사양을 어떻게 결정합니까?

DMS Fleet Advisor는 데이터베이스 용량 또는 사용률을 기반으로 권장 사항을 생성할 수 있습니다.

- 데이터베이스 용량을 기반으로 권장 사항을 생성하도록 선택하면 DMS Fleet Advisor는 기존 데이터베이스 용량을 가장 가까운 인스턴스 클래스의 사양에 매핑합니다.
- 리소스 사용률을 기반으로 권장 사항을 생성하도록 선택하면 DMS Fleet Advisor가 CPU, 메모리, IO 처리량 및 IOPS와 같은 지표의 95번째 백분위수 값을 결정합니다. 95번째 백분위수는 수집된 데이터의 95%가 이 값보다 낮다는 것을 의미합니다. 그러면 DMS Fleet Advisor는 이 값을 가장 가까운 인스턴스 클래스의 사양에 매핑합니다.

대상 데이터베이스의 크기를 결정하기 위해 DMS Fleet Advisor는 소스 데이터베이스의 크기에 관한 정보를 수집합니다. 그러면 DMS Fleet Advisor는 대상 스토리지에 동일한 크기를 사용할 것을 권장합니다. 소스 데이터베이스 스토리지가 과다 프로비저닝되면 대상 스토리지의 권장 크기도 과다 프로비저닝됩니다.

AWS DMS를 사용하여 데이터를 마이그레이션하려면 대상 DB 인스턴스의 IOPS 프로비저닝을 늘려야 할 수 있습니다. DMS Fleet Advisor가 대상 권장 사항을 생성할 때 서비스는 소스 데이터베이스 지표만 고려합니다. DMS Fleet Advisor는 데이터 마이그레이션 작업을 실행하는 데 필요할 수 있는 추가 IOPS를 고려하지 않습니다. 자세한 설명은 마이그레이션 태스크가 느리게 실행됨 섹션을 참조하세요.

IOPS 비용을 추정하기 위해 DMS 플릿 어드바이저는 소스 IOPS 사용량 one-to-one 매핑을 기준으로 사용합니다. DMS Fleet Advisor는 피크 로드를 기준선 값으로 간주하고 IOPS 요금 계산 시 100% 사용률을 고려합니다.

PostgreSQL 및 MySQL 소스 데이터베이스의 경우, DMS Fleet Advisor는 대상 권장 사항에 Aurora 및 Amazon RDS DB 인스턴스를 포함할 수 있습니다. Aurora 구성이 소스 요구 사항에 매핑되는 경우, DMS Fleet Advisor는 이 옵션을 권장 옵션으로 표시합니다.

AWS DMS Fleet Advisor를 사용하여 대상 권장 사항 생성

데이터베이스 및 분석 플릿의 데이터 수집 및 인벤토리를 완료한 후 DMS Fleet Advisor에서 대상 권장 사항을 생성할 수 있습니다. 이렇게 하려면 소스 데이터베이스를 선택하고 DMS Fleet Advisor 대상 권장 기능이 대상 인스턴스의 크기를 결정하는 데 사용하는 설정을 구성하십시오. 또한 DMS Fleet Advisor 대상 권장 기능은 소스 데이터베이스에서 수집한 용량 및 사용률 지표를 사용합니다.

대상 권장 사항을 생성하려면

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 콘솔을 엽니다. AWS DMS

   DMS Fleet Advisor를 사용할 AWS 리전 위치를 선택해야 합니다.
2. 탐색 창의 평가에서 권장 사항을 선택한 다음, 권장 사항 생성을 선택합니다.
3. 소스 데이터베이스 선택 패널에서 AWS 클라우드로 마이그레이션할 데이터베이스의 이름 옆에 있는 확인란을 선택합니다.

   소스 데이터베이스 검색에 데이터베이스의 이름을 입력하여 인벤토리를 필터링합니다.

   DMS Fleet Advisor는 한 번에 최대 100개의 데이터베이스에 대한 권장 사항을 생성할 수 있습니다.
4. 가용성 및 내구성을 위해 선호하는 배포 옵션을 선택하십시오.

   프로덕션 데이터베이스의 대상 권장 사항을 계산하려면 프로덕션(다중 AZ)을 선택합니다. DMS Fleet Advisor는 대상 권장 사항에 서로 다른 가용 영역(AZ)에 있는 두 개의 DB 인스턴스를 포함합니다. 이 다중 AZ 배포 옵션은 고가용성, 데이터 중복 및 장애 조치 지원을 제공합니다.

   Aurora가 권장 대상 엔진이고 가용성 및 내구성이 다중 AZ 배포인 경우 대상 권장 사항에는 리더 및 라이터 DB 인스턴스가 포함됩니다.

개발 또는 테스트에 사용하는 데이터베이스의 대상 권장 사항을 계산하려면 개발 및 테스트(단일 AZ)를 선택합니다. DMS Fleet Advisor는 대상 권장 사항에 단일 DB 인스턴스를 포함합니다. 이 단일 AZ 배포 옵션은 유지 관리 비용을 줄여줍니다.

5. 대상 인스턴스 크기 조정에서는 DMS Fleet Advisor가 대상 권장 사항을 계산하는 데 사용하는 선호 옵션을 선택하십시오.

   소스 데이터베이스 또는 OS 서버 구성을 기반으로 대상 권장 사항을 계산하려면 총 용량을 선택합니다. DMS Fleet Advisor는 소스 데이터베이스 또는 OS 서버의 총 CPU, 메모리, 디스크 용량과 같은 지표를 사용하여 대상 권장 사항을 생성합니다. 그런 다음, DMS Fleet Advisor는 데이터베이스 용량 지표를 가장 가까운 Amazon RDS DB 인스턴스 클래스의 사양에 매핑합니다.

   소스 데이터베이스 또는 OS 서버의 실제 사용률을 기반으로 대상 권장 사항을 계산하려면 리소스 사용률을 선택합니다. DMS Fleet Advisor는 소스 데이터베이스 또는 OS 서버의 CPU, 메모리 및 디스크 용량에 대한 사용률 지표를 사용하여 대상 권장 사항을 생성합니다. DMS Fleet Advisor는 사용률 지표에서 각 지표의 95번째 백분위수를 계산합니다. 95번째 백분위수는 기간 내 데이터의 95%가 이 값보다 낮음을 의미합니다. 그러면 DMS Fleet Advisor는 이 값을 가장 가까운 Amazon RDS DB 인스턴스 클래스에 매핑합니다.

   더 정확한 권장 사항을 확인하려면 리소스 사용률 옵션을 사용하는 것이 좋습니다. 이를 위해 총 용량 및 리소스 사용률 지표를 수집했는지 확인하십시오.

6. 생성을 선택합니다.

DMS Fleet Advisor는 선택한 데이터베이스에 대한 대상 권장 사항을 생성합니다. 권장 사항을 성공적으로 생성한 경우, DMS Fleet Advisor는 상태를 계산됨으로 설정합니다. 또한 DMS Fleet Advisor는 AWS Pricing Calculator를 사용하여 권장 대상 DB 인스턴스의 월별 예상 비용을 결정합니다. 이제 생성된 권장 사항의 자세한 내용을 살펴볼 수 있습니다. 자세한 설명은 권장 사항 세부 정보 섹션을 참조하세요.

데이터 인벤토리의 월별 총 비용을 추정하려면 클라우드로 이전하려는 데이터베이스의 확인란을 선택합니다. DMS Fleet Advisor는 월별 총 예상 비용과 AWS 클라우드 내 대상 데이터베이스의 요약 정보를 표시합니다. DMS Fleet Advisor는 사용자가 알아야 하는 요금 내역만 제공하기 위해 AWS 가격표 Query API를 사용합니다. 실제 요금은 AWS 서비스의 실제 사용량을 포함한 다양한 요인에 따라 달라집니다. AWS 서비스 요금에 관한 자세한 내용은 클라우드 서비스 요금을 참조하세요.

AWS DMS Fleet Advisor와 함께 대상 권장 사항의 세부 정보 살펴보기

DMS Fleet Advisor가 대상 권장 사항을 생성한 후 권장 사항 테이블에서 권장 마이그레이션 대상의 주요 파라미터를 볼 수 있습니다. 이러한 주요 파라미터에는 대상 엔진, 인스턴스 클래스, 가상 CPU 수, 메모리, 스토리지 및 스토리지 유형이 포함됩니다. DMS Fleet Advisor는 이러한 파라미터 외에도 이 권장 마이그레이션 대상의 월별 예상 비용을 표시합니다.

각 권장 사항에는 가능한 AWS 대상 엔진이 하나 이상 포함될 수 있습니다. 권장 사항에 대상 엔진이 여러 개 포함되어 있는 경우, AWS DMS는 그 중 하나를 권장 엔진으로 표시합니다. 또한 AWS DMS는 권장 사항 테이블에 이 권장 옵션에 대한 파라미터와 월별 예상 비용을 표시합니다.

대상 권장 사항을 소스 데이터베이스의 사용률 및 용량과 비교하려면 권장 사항의 세부 내용을 살펴보십시오. 또한 선택한 권장 사항에 대한 마이그레이션 제한 사항을 볼 수 있습니다. 이러한 제한 사항에는 지원되지 않는 데이터베이스 기능, 작업 항목 및 기타 마이그레이션 고려 사항이 포함됩니다.

권장 사항의 자세한 내용을 살펴보려면

1. DMS Fleet Advisor를 사용하여 대상 권장 사항을 생성하십시오. 자세한 설명은 대상 권장 사항 생성 섹션을 참조하세요.
2. 권장 사항 테이블의 권장 사항 이름을 선택합니다. 권장 사항 페이지가 열립니다.
3. 권장 사항에 대상 옵션이 두 개 이상 포함된 경우, 대상 권장 사항의 경우 대상 옵션을 선택하십시오.
4. 소스 사용률 및 용량 섹션을 확장하십시오. DMS Fleet Advisor는 다음 지표에 대한 리소스 사용률 차트를 표시합니다.

   - CPU 수
   - 메모리
   - I/O 처리량
   - 초당 입출력 작업 처리량(IOPS)
   - 스토리지
   - 활성 데이터베이스 서버 연결 수

   이 차트를 사용하여 DMS 데이터 수집기의 소스 데이터베이스 지표를 선택한 대상 엔진의 지표와 비교할 수 있습니다.

소스 사용률 및 용량 섹션을 확장한 후 차트가 표시되지 않으면 IAM 사용자에게 Amazon CloudWatch 대시보드를 볼 수 있는 권한을 부여했는지 확인하십시오. 자세한 내용은 Amazon 사용 설명서의 Amazon CloudWatch 대시보드 사용을 참조하십시오. CloudWatch

5. 선택한 대상 엔진 이름이 있는 링크를 선택합니다. 대상 세부 정보 페이지가 열립니다.
6. 대상 추천을 CSV로 내보내려면 작업 드롭다운에서 CSV로 내보내기 옵션을 선택합니다.
7. 대상 추천을 로 내보내려면 작업 AWS Pricing Calculator 드롭다운에서 AWS Pricing Calculator옵션으로 비용 최적화를 선택합니다.
8. 구성 섹션에서 소스 데이터베이스 파라미터의 값을 대상 엔진의 파라미터와 비교합니다. 대상 엔진의 경우, DMS Fleet Advisor는 클라우드 리소스의 월별 예상 비용을 표시합니다. DMS Fleet Advisor는 사용자가 알아야 하는 요금 내역만 제공하기 위해 AWS 가격표 Query API를 사용합니다. 실제 요금은 AWS 서비스의 실제 사용량을 포함한 다양한 요인에 따라 달라집니다. AWS 서비스 요금에 관한 자세한 내용은 https://aws.amazon.com/pricing/ 클라우드 서비스 요금을 참조하세요.
9. 마이그레이션 제한 사항 섹션에서 마이그레이션 제한 사항을 확인하세요. 소스 데이터베이스를 AWS 클라우드로 마이그레이션할 때는 이러한 제한 사항을 고려하는 것이 좋습니다.

AWS DMS Fleet Advisor를 사용하여 대상 권장 사항 내보내기

대상 권장 사항을 생성한 후 권장 사항 목록의 사본을 CSV(쉼표로 구분된 값) 파일로 저장할 수 있습니다.

대상 권장 사항을 생성하려면

1. AWS Management Console로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.

   DMS Fleet Advisor를 사용할 AWS 리전 위치를 선택해야 합니다.
2. 탐색 창의 평가 에서 권장 사항을 선택한 다음, CSV 파일에 포함할 권장 사항을 선택합니다.
3. CSV로 내보내기를 선택하고 파일 이름을 입력한 다음, PC에서 이 파일을 저장할 폴더를 선택합니다.
4. CSV 파일을 엽니다.

권장 사항이 수록된 CSV 파일에는 다음 정보가 포함됩니다.

- CreatedDate— DMS 플릿 어드바이저가 타겟 엔진 권장 사항을 생성한 날짜.

- DatabaseId— DMS 플릿 어드바이저가 이 권장 사항을 생성한 소스 데이터베이스의 식별자입니다.
- DeploymentOption— 권장 Amazon RDS DB 인스턴스를 위한 배포 옵션입니다.
- EngineEdition— 권장 대상 Amazon RDS 엔진 에디션입니다.
- EngineName— 대상 엔진의 이름.
- InstanceMemory— 권장 Amazon RDS DB 인스턴스의 메모리 양.
- InstanceSizingType— 대상 인스턴스의 크기.
- InstanceType— 권장 대상 Amazon RDS 인스턴스 유형입니다.
- InstanceVcpu— 권장 Amazon RDS DB 인스턴스의 가상 CPU 수입니다.
- 선호됨 – 이 대상 옵션이 권장됨을 나타내는 부울 플래그.
- 상태 - 대상 엔진 권장 사항의 상태.
- StorageIops— 권장 Amazon RDS DB 인스턴스에서 초당 완료한 I/O 작업 수 (IOPS) 입니다.
- StorageSize— 권장 Amazon RDS DB 인스턴스의 스토리지 크기입니다.
- StorageType— 권장 Amazon RDS DB 인스턴스의 스토리지 유형입니다.
- WorkloadType— 대상 엔진의 배포 옵션 (예: 다중 AZ 또는 단일 AZ 배포)

플릿 어드바이저를 통한 마이그레이션 제한 발견 및 분석 AWS DMS

DMS 데이터 수집기를 사용하여 대상 엔진이 지원하지 않는 데이터베이스 기능을 검색할 수 있습니다. 올바른 마이그레이션 대상을 선택하려면 이러한 제한 사항을 고려해야 합니다.

DMS 데이터 수집기는 특정 소스 데이터베이스 기능을 검색합니다. 그런 다음 DMS Fleet Advisor는 지정된 대상으로의 마이그레이션 관점에서 소스 기능을 분석하고 제한에 대한 추가 정보를 제공하며 이러한 제한을 해결하거나 방지하기 위한 권장 조치를 포함합니다. 또한 DMS Fleet Advisor는 이러한 제한 사항에 따른 영향을 고려합니다.

제한 목록은 타겟 엔진 세부 정보 페이지에서 확인할 수 있습니다. 왼쪽 탐색 메뉴의 권장 사항 페이지에서 이 페이지로 이동합니다. 대상 목록에서 검사할 대상 엔진을 선택합니다. 제한 목록은 페이지 하단에 있습니다.

다음 표에는 Amazon RDS for MySQL이 지원하지 않는 MySQL 데이터베이스 기능이 나와 있습니다.

제한 사항	설명	영향
인증 플러그인	Amazon RDS에서는 MySQL 인증 플러그인을 지원하지 않습니다.	낮음(Low)
시스템 로그에 오류 로깅	Amazon RDS는 시스템 로그에 오류 로그를 쓰는 것을 지원하지 않습니다.	낮음(Low)
글로벌 트랜잭션 식별자(GTID)	모든 RDS for MySQL 버전 5.7과 RDS for MySQL 버전 8.0.26 및 그 이후의 RDS for MySQL 8.0 버전에서 글로벌 트랜잭션 식별자를 사용할 수 있습니다.	낮음(Low)
그룹 복제	Amazon RDS는 MySQL 그룹 복제 플러그인을 지원하지 않습니다.	낮음(Low)
InnoDB 테이블스페이스 암호화	Amazon RDS에서는 InnoDB 테이블스페이스 암호화를 지원하지 않습니다.	낮음(Low)
InnoDB 예약어	InnoDB는 Amazon RDS for MySQL의 예약어입니다. MySQL 데이터베이스에는 이 이름을 사용할 수 없습니다.	낮음(Low)
키링 플러그인	Amazon RDS는 MySQL 키링 플러그인을 지원하지 않습니다.	낮음(Low)
암호 확인 플러그인	Amazon RDS는 MySQL `validate_password` 플러그인을 지원하지 않습니다.	낮음(Low)

제한 사항	설명	영향
지속 시스템 변수	Amazon RDS에서는 MySQL 지속 시스템 변수를 지원하지 않습니다.	낮음(Low)
제한된 액세스	Amazon RDS는 고급 권한이 필요한 특정 시스템 절차와 테이블에 대한 액세스를 제한합니다. 또한 Amazon RDS는 Telnet, Secure Shell(SSH) 또는 Windows 원격 데스크톱 연결을 사용하여 DB 인스턴스에 대한 직접적인 호스트 액세스를 허용하지 않습니다.	낮음(Low)
리라이터 쿼리 다시 쓰기 플러그인	Amazon RDS는 MySQL 리라이터 쿼리 다시 쓰기 플러그인을 지원하지 않습니다.	낮음(Low)
반동기식 복제	Amazon RDS는 MySQL 반동기식 복제를 지원하지 않습니다.	낮음(Low)
전송 가능 테이블스페이스	Amazon RDS는 MySQL 전송 가능 테이블스페이스 기능을 지원하지 않습니다.	낮음(Low)
X 플러그인	Amazon RDS는 MySQL X 플러그인을 지원하지 않습니다.	낮음(Low)

다음 표에는 Amazon RDS for Oracle이 지원하지 않는 Oracle 데이터베이스 기능이 나와 있습니다.

제한 사항	설명	영향
Active Data Guard	Active Data Guard는 Oracle 멀티테넌트 컨테이너 데이터베이	중간(Medium)

제한 사항	설명	영향
	스(CDB)와 함께 사용할 수 없습니다.	
Automatic Storage Management(ASM)	Amazon RDS는 Oracle Automatic Storage Management(Oracle ASM)를 지원하지 않습니다.	중간(Medium)
데이터베이스 활동 스트림	Amazon RDS는 단일 테넌트 아키텍처에 대해 Oracle 데이터베이스 활동 스트림을 지원하지 않습니다.	높음(High)
파일 크기 제한	RDS for Oracle DB 인스턴스에서 단일 파일의 최대 크기는 16TiB입니다.	중간(Medium)
FTP 및 SFTP	Amazon RDS에서는 FTP와 SFTP를 지원하지 않습니다.	중간(Medium)
파티셔닝된 하이브리드 테이블	Amazon RDS에서는 파티셔닝된 Oracle 하이브리드 테이블을 지원하지 않습니다.	높음(High)
Oracle Data Guard	Amazon RDS는 단일 테넌트 아키텍처에 대한 Oracle Data Guard를 지원하지 않습니다.	높음(High)
Oracle Database Vault	Amazon RDS에서는 Oracle Database Vault를 지원하지 않습니다.	높음(High)
Oracle DBA 권한 저장소	Amazon RDS는 Oracle DBA 권한에 대한 제한 사항이 있습니다. 자세한 내용은 Oracle DBA 권한에 대한 제한 사항을 참조하십시오.	높음(High)

제한 사항	설명	영향
Oracle Enterprise Manager	Amazon RDS는 단일 테넌트 아키텍처에 대한 Oracle Enterprise Manager를 지원하지 않습니다.	높음(High)
Oracle Enterprise Manager Agent	Amazon RDS는 단일 테넌트 아키텍처에 대한 Oracle Enterprise Manager Agent를 지원하지 않습니다.	중간(Medium)
Oracle Enterprise Manager Cloud Control Management Repository	Amazon RDS for Oracle DB 인스턴스는 Oracle Enterprise Manager Cloud Control Management Repository에 사용할 수 없습니다.	높음(High)
Oracle Flashback Database	Amazon RDS는 Oracle Flashback Database 기능을 지원하지 않습니다.	높음(High)
Oracle 레이블 보안	Amazon RDS는 단일 테넌트 아키텍처에 대한 Oracle Label Security를 지원하지 않습니다. Oracle Label Security는 멀티테넌트 컨테이너 데이터베이스(Oracle CDB)에만 사용할 수 있습니다.	높음(High)
Oracle Messaging Gateway	Amazon RDS에서는 Oracle Messaging Gateway를 지원하지 않습니다.	높음(High)
Oracle Real Application Clusters	Amazon RDS는 Oracle Real Application Clusters(Oracle RAC)를 지원하지 않습니다.	높음(High)

제한 사항	설명	영향
Oracle Real Application Testing	Amazon RDS는 Oracle Real Application Testing을 지원하지 않습니다.	높음(High)
Oracle Snapshot Standby 데이터베이스	Amazon RDS는 Oracle Snapshot Standby 데이터베이스를 지원하지 않습니다.	높음(High)
공개 동의어	Amazon RDS는 Oracle에서 제공한 스키마에 대한 공개 동의어를 지원하지 않습니다.	중간(Medium)
지원되지 않는 기능에 대한 스키마	Amazon RDS는 시스템 권한이 필요한 Oracle 기능 및 구성 요소에 대한 스키마를 지원하지 않습니다.	높음(High)
순수 통합 감사	Amazon RDS에서는 순수 통합 감사를 지원하지 않습니다. 통합 감사는 혼합 모드에서 사용할 수 있습니다.	중간(Medium)
Workspace Manager	Amazon RDS는 Oracle Database Workspace Manager `WMSYS` 스키마를 지원하지 않습니다.	높음(High)

다음 표에는 Amazon RDS for PostgreSQL이 지원하지 않는 PostgreSQL 데이터베이스 기능이 나와 있습니다.

제한 사항	설명	영향
동시 연결	RDS for PostgreSQL 인스턴스에 동시에 연결할 수 있는 최대	낮음(Low)

제한 사항	설명	영향
	수는 `max_connections` 파라미터에 의해 제한됩니다.	
최신 버전	Amazon RDS는 자동으로 메이저 버전 업그레이드를 적용하지 않습니다. 메이저 버전 업그레이드를 수행하려면 DB 인스턴스를 수동으로 수정합니다. 자세한 내용은 PostgreSQL에 대한 메이저 버전 업그레이드 선택을 참조하세요.	낮음(Low)
예약된 연결	Amazon RDS는 시스템 유지 관리를 위해 최대 3개의 연결을 예약합니다. 사용자 연결 파라미터의 값을 지정할 경우, 사용할 것으로 예상하는 연결 개수에 3을 더합니다.	낮음(Low)
지원되는 확장	RDS for PostgreSQL은 PostgreSQL 데이터베이스 엔진을 위한 한정된 수의 확장을 지원합니다. 해당 PostgreSQL 버전의 기본 DB 파라미터 그룹 내에서 지원되는 확장 기능의 목록을 확인할 수 있습니다. 또한 `rds.extensions` 파라미터를 표시하면 `psql`을 사용하여 현재 확장 기능 목록을 확인할 수도 있습니다.	낮음(Low)
테이블스페이스 분할 또는 격리	I/O 분할 또는 격리를 위해 테이블스페이스를 사용할 수 없습니다. RDS for PostgreSQL에서 모든 스토리지는 단일 논리 볼륨에 있습니다.	낮음(Low)

다음 표에는 Amazon RDS for SQL Server가 지원하지 않는 SQL Server 데이터베이스 기능이 나와 있습니다.

제한 사항	설명	영향
Microsoft Azure Blob Storage로 백업	RDS for SQL Server는 Microsoft Azure Blob Storage에 백업하는 것을 지원하지 않습니다.	중간(Medium)
버퍼 풀 확장	RDS for SQL Server는 버퍼 풀 확장을 지원하지 않습니다.	높음(High)
사용자 지정 암호 정책	RDS for SQL Server는 사용자 지정 암호 정책을 지원하지 않습니다.	중간(Medium)
데이터 품질 서비스	RDS for SQL Server는 SQL Server DQS(Data Quality Services)를 지원하지 않습니다.	높음(High)
데이터베이스 로그 전달	RDS for SQL Server는 데이터베이스 로그 전달을 지원하지 않습니다.	높음(High)
데이터베이스 이름	데이터베이스 이름에는 다음과 같은 제한이 있습니다. rdsadmin으로 시작할 수 없고, 공백이나 탭으로 시작하거나 끝날 수 없으며, 새 줄을 생성하는 문자를 포함할 수 없으며, 작은따옴표(')를 포함할 수 없습니다.	중간(Medium)
데이터베이스 스냅샷	RDS for SQL Server는 데이터베이스 스냅샷을 지원하지 않습니다. Amazon RDS에서는	중간(Medium)

제한 사항	설명	영향
	DB 인스턴스 스냅샷만 사용할 수 있습니다.	
확장된 저장 프로시저	RDS for SQL Server는 `xp_cmdshell` 을 포함하여 확장된 저장 프로시저를 지원하지 않습니다.	높음(High)
파일 테이블	RDS for SQL Server는 파일 테이블을 지원하지 않습니다.	중간(Medium)
FILESTREAM 지원	RDS for SQL Server는 FILESTREAM을 지원하지 않습니다.	중간(Medium)
연결된 서버	RDS for SQL Server는 연결된 서버를 제한적으로 지원합니다.	높음(High)
기계 학습 및 R 서비스	RDS for SQL Server는 기계 학습 및 R 서비스를 지원하지 않습니다. 이러한 서비스를 설치하려면 OS 액세스가 필요하기 때문입니다.	높음(High)
유지 관리 계획	RDS for SQL Server는 유지 관리 계획을 지원하지 않습니다.	높음(High)
성능 데이터 수집기	RDS for SQL Server는 성능 데이터 수집기를 지원하지 않습니다.	높음(High)
정책 기반 관리	RDS for SQL Server는 정책 기반 관리를 지원하지 않습니다.	중간(Medium)
PolyBase	SQL Server용 RDS는 지원하지 PolyBase 않습니다.	높음(High)

제한 사항	설명	영향
복제	RDS for SQL Server는 복제를 지원하지 않습니다.	중간(Medium)
리소스 거버너	RDS for SQL Server는 리소스 거버너를 지원하지 않습니다.	높음(High)
서버 수준 트리거	RDS for SQL Server는 서버 수준 트리거를 지원하지 않습니다.	중간(Medium)
서비스 브로커 엔드포인트	RDS for SQL Server는 서비스 브로커 엔드포인트를 지원하지 않습니다.	높음(High)
SSAS	RDS for SQL Server에서 SQL Server Analysis Services(SSAS)를 실행하는 데 적용되는 제한 사항을 고려하십시오. 자세한 내용은 제한 사항을 참조하세요.	낮음(Low)
SSIS	RDS for SQL Server에서 SQL Server Integration Services(SSIS)를 실행하는 데 적용되는 제한 사항을 고려하십시오. 자세한 내용은 제한 사항을 참조하세요.	낮음(Low)
SSRS	RDS for SQL Server에서 SQL Server Reporting Services(SSRS)를 실행하는 데 적용되는 제한 사항을 고려하십시오. 자세한 내용은 제한 사항을 참조하세요.	낮음(Low)

제한 사항	설명	영향
SQL Server DB 인스턴스의 스토리지 크기	SQL Server 범용(SSD) 스토리지 및 프로비저닝된 IOPS 스토리지 인스턴스의 최대 스토리지 크기는 16TiB입니다. SQL Server 마그네틱 스토리지 인스턴스를 위한 최대 스토리지 크기는 1TiB입니다.	높음(High)
Stretch 데이터베이스	RDS for SQL Server는 SQL Server Stretch 데이터베이스 기능을 지원하지 않습니다.	중간(Medium)
T-SQL 엔드포인트	RDS for SQL Server는 `CREATE ENDPOINT`를 사용하는 모든 작업을 지원하지는 않습니다.	높음(High)
TRUSTWORTHY 데이터베이스 속성	RDS for SQL Server는 `sysadmin` 역할이 필요하지 않으므로 `TRUSTWORTHY` 데이터베이스 속성을 지원하지 않습니다.	중간(Medium)

다음 표에는 권장 사항 문제 목록이 포함되어 있습니다. DMS Fleet Advisor는 소스 및 대상 데이터베이스 기능을 분석하고 이러한 마이그레이션 제한 사항을 제공합니다. 차단기 영향의 한계로 인해 DMS 플릿 어드바이저는 원본 데이터베이스에 대한 대상 권장 사항을 생성할 수 없습니다.

제한 사항	설명	영향
적절한 인스턴스를 찾을 수 없습니다.	AWS DMS소스 데이터베이스 지표 조합에 적합한 크기의 마이그레이션 대상으로 사용할 수 있는 대상 인스턴스를 찾을 수 없습니다.	블로커

제한 사항	설명	영향
IOPS에서 적절한 인스턴스를 찾을 수 없습니다.	원본 데이터베이스는 가능한 대상 DB 인스턴스의 최대 IOPS 수를 초과하는 IOPS 수를 사용합니다.	블로커
RAM에서 적절한 인스턴스를 찾을 수 없습니다.	원본 데이터베이스는 몇 GB의 RAM을 사용하는데, 이는 가능한 대상 DB 인스턴스의 최대 RAM 크기를 초과합니다.	블로커
스토리지 크기로 적절한 인스턴스를 찾을 수 없습니다.	원본 데이터베이스는 TB의 스토리지를 사용하는데, 이는 가능한 대상 DB 인스턴스의 최대 스토리지 크기를 초과합니다.	블로커
에디션별로 적절한 인스턴스를 찾을 수 없습니다.	원본 데이터베이스에는 Amazon RDS에서 지원하지 않는 에디션이 있습니다.	블로커
CPU 코어에서 적절한 인스턴스를 찾을 수 없습니다.	원본 데이터베이스의 CPU 코어 수가 가능한 대상 DB 인스턴스의 최대 CPU 코어 수를 초과합니다.	블로커
버전별로 적절한 인스턴스를 찾을 수 없습니다.	원본 데이터베이스의 버전이 AWS DMS 인식되지 않습니다.	블로커
CPU 매개변수가 정의되지 않았습니다.	DMS 데이터 수집기는 원본 데이터베이스에서 사용하는 CPU에 대한 정보를 수집하지 않았습니다. 데이터 수집기에서 데이터 전달에 필요한 측정치를 수집하고 자격 증명을 구성했는지 확인하세요. 데이터 전달을 위한 보안 인증 구성 섹션을 참조하세요.	차단기

제한 사항	설명	영향
메모리 매개변수가 정의되지 않았습니다.	DMS 데이터 수집기는 원본 데이터베이스에서 사용하는 메모리에 대한 정보를 수집하지 않았습니다. 데이터 수집기에서 데이터 전달에 필요한 측정치를 수집하고 자격 증명을 구성했는지 확인하세요. 데이터 전달을 위한 보안 인증 구성 섹션을 참조하세요.	차단기
스토리지 크기 매개변수가 정의되지 않았습니다.	DMS 데이터 수집기는 원본 데이터베이스가 사용하는 스토리지 크기에 대한 정보를 수집하지 않았습니다. 데이터 수집기에서 데이터 전달에 필요한 지표와 구성된 자격 증명을 수집했는지 확인하세요. 데이터 전달을 위한 보안 인증 구성 섹션을 참조하세요.	차단기
스토리지 IOPS 매개변수가 정의되지 않았습니다.	DMS 데이터 수집기가 소스 데이터베이스 사용에 대한 스토리지 IOPS 지표를 수집하지 않았습니다. 데이터 수집기에서 데이터 전달에 필요한 지표와 구성된 자격 증명을 수집했는지 확인하십시오.	차단기

제한 사항	설명	영향
데이터가 충분하지 않음	DMS 데이터 수집기가 목표 추천을 생성하기에 충분한 데이터를 수집하지 않았습니다. 데이터 수집기에서 데이터 전달을 위한 자격 증명을 구성했는지 확인하세요. 데이터 전달을 위한 보안 인증 구성 섹션을 참조하세요.	블로커
데이터베이스 에디션은 정의되지 않았습니다.	DMS 데이터 수집기는 소스 데이터베이스 에디션에 대한 정보를 수집하지 않았습니다. 데이터 수집기에서 데이터를 전달하는 데 필요한 측정치를 수집하고 자격 증명을 구성했는지 확인하세요. 데이터 전달을 위한 보안 인증 구성 섹션을 참조하세요.	차단기
알 수 없는 오류	DMS 플릿 어드바이저는 소스 데이터베이스에 대한 대상 권장 사항을 생성할 수 없습니다.	블로커

제한 사항	설명	영향
데이터베이스 버전이 정의되지 않았습니다.	DMS 플릿 어드바이저는 소스 데이터베이스 버전에 대한 정보를 수집하지 않았습니다. DMS 플릿 어드바이저는 원본 데이터베이스에 최신 데이터베이스 버전을 사용할 것을 권장합니다. 이 권장 사항을 선택하는 경우 데이터베이스 버전을 업그레이드해야 합니다. 원본 데이터베이스에 대해 생성된 대상 권장 사항을 검토하고 이러한 권장 사항이 요구 사항을 충족하는지 확인하십시오.	높음(High)
RDS 설정에서 데이터베이스 연결 수를 늘리십시오.	소스 데이터베이스에는 특정 개수의 연결이 필요합니다. 기본적으로 Amazon RDS 데이터베이스 인스턴스에 사용할 수 있는 연결 수는 다릅니다. RDS 데이터베이스 인스턴스를 생성할 때 이 기본값을 변경해야 합니다. 이렇게 하려면 파라미터 그룹에서 `max_connections` 파라미터 값을 업데이트하십시오.	중간(Medium)
타겟 에디션은 호환됩니다.	소스 데이터베이스의 대상 권장 사항은 다른 데이터베이스 에디션을 사용합니다. 소스 데이터베이스 에디션은 권장 대상 에디션과 동일한 기능을 지원합니다. 하지만 이 새 데이터베이스 에디션을 선택하면 비용이 증가할 수 있습니다.	중간(Medium)

제한 사항	설명	영향
스토리지 처리량 매개변수는 정의되지 않았습니다.	DMS 데이터 수집기는 소스 데이터베이스 사용에 대한 스토리지 처리량 메트릭을 수집하지 않았습니다. 원본 데이터베이스에 대해 생성된 대상 권장 사항을 검토하고 이러한 권장 사항이 요구 사항을 충족하는지 확인하십시오.	중간(Medium)
데이터베이스 연결 번호 매개변수는 정의되지 않았습니다.	DMS 데이터 수집기는 원본 데이터베이스에서 사용하는 연결 수에 대한 정보를 수집하지 않았습니다. 원본 데이터베이스에 대해 생성된 대상 권장 사항을 검토하고 이러한 권장 사항이 요구 사항을 충족하는지 확인하십시오. 또는 할당량 증가를 요청하세요.	중간(Medium)
데이터베이스 다운그레이드 버전	원본 데이터베이스는 Amazon RDS 데이터베이스보다 상위 버전에서 실행됩니다. 데이터베이스 버전을 다운그레이드하려면 하위 버전에서 구현되지 않은 기능을 사용하지 마십시오. Amazon EC2를 마이그레이션 타겟으로 사용할 수도 있습니다.	중간(Medium)

제한 사항	설명	영향
대상 에디션은 다릅니다.	소스 데이터베이스의 대상 권장 사항은 다른 데이터베이스 에디션을 사용합니다. 소스 데이터베이스 에디션은 권장 대상 에디션과 호환됩니다. 하지만 권장 대상 데이터베이스 에디션은 원본 데이터베이스 에디션의 일부 기능을 지원하지 않습니다. 이 새 데이터베이스 에디션을 선택하면 비용이 증가할 수 있습니다.	중간(Medium)
지원되지 않는 버전에서 업그레이드	소스 데이터베이스가 지원 종료 단계에 도달했습니다. 최신 DB 엔진 버전을 대상으로 사용하려면 마이그레이션 전에 데이터베이스를 업그레이드해야 합니다. Amazon EC2를 마이그레이션 타겟으로 사용할 수도 있습니다. 데이터베이스 엔진에 따라 다음 링크 중 하나를 사용하여 자세히 알아보십시오. MySQL 업그레이드 SQL 서버 업그레이드 오라클 DB 업그레이드 PostgreSQL 업그레이드	중간(Medium)

대상 권장 사항에 대한 문제 해결

다음 목록에서 DMS Fleet Advisor 대상 권장 기능에 문제가 발생할 때 취해야 할 조치를 찾을 수 있습니다.

주제

- 대상 권장 사항의 예상 가격을 볼 수 없습니다.
- 리소스 사용률 차트가 보이지 않습니다.
- 지표 수집 상태를 볼 수 없습니다.

대상 권장 사항의 예상 가격을 볼 수 없습니다.

성공 상태인 권장 사항의 월별 예상 비용에서 데이터 없음이 표시되는 경우, AWS 가격표 서비스 API에 액세스할 수 있는 권한을 IAM 사용자에게 부여했는지 확인하십시오. 이렇게 하려면 `pricing:GetProducts` 권한이 포함된 정책을 생성하여 IAM 리소스 생성에 설명된 대로 IAM 사용자에게 추가해야 합니다.

DMS Fleet Advisor는 실패 상태인 권장 사항에 대한 월별 예상 비용을 계산하지 않습니다.

리소스 사용률 차트가 보이지 않습니다.

소스 사용률 및 용량 섹션을 확장한 후 지표 로드 실패 메시지가 표시되면 IAM 사용자에게 Amazon CloudWatch 대시보드를 볼 수 있는 권한을 부여했는지 확인하십시오. 이렇게 하려면 IAM 리소스 생성에 설명된 대로 IAM 사용자에게 필수 정책을 추가해야 합니다.

또는 `cloudwatch:GetDashboard`, `cloudwatch:ListDashboards`, `cloudwatch:PutDashboard` 및 `cloudwatch:DeleteDashboards` 권한이 포함된 사용자 지정 정책을 생성할 수도 있습니다. 자세한 내용은 Amazon 사용 설명서의 Amazon CloudWatch 대시보드 사용을 참조하십시오. CloudWatch

지표 수집 상태를 볼 수 없습니다.

권장 사항 생성을 선택할 때 지표 수집에 사용 가능한 데이터 없음이라는 메시지가 표시되면 데이터를 수집했는지 확인하십시오. 자세한 설명은 AWS DMS Fleet Advisor를 위한 데이터 수집 섹션을 참조하세요.

데이터를 수집한 후 이 문제가 발생하는 경우 IAM 사용자에게 CloudWatch Amazon에 액세스할 수 있는 `cloudwatch:Get*` 권한을 부여했는지 확인하십시오. DMS Fleet Advisor는 서비스 연결 역할을

사용하여 수집된 데이터베이스 성능 지표를 사용자를 대신하여 CloudWatch 게시합니다. DMS 플릿 어드바이저와 함께 사용할 서비스 연결 역할을 생성해야 합니다. 자세한 내용은 IAM 리소스 생성을(를) 참조하세요.

DMS 플릿 어드바이저 제한 사항

DMS 플릿 어드바이저를 사용할 때의 제한 사항은 다음과 같습니다.

- DMS 플릿 어드바이저는 권장 사항을 생성합니다 one-to-one . 각 소스 데이터베이스에 대해 DMS Fleet Advisor는 단일 대상 엔진을 결정합니다. DMS Fleet Advisor는 멀티테넌트 서버를 처리하지 않으며 단일 대상 DB 인스턴스에서 여러 데이터베이스를 실행하기 위한 권장 사항도 제공하지 않습니다.
- DMS Fleet Advisor는 사용 가능한 데이터베이스 버전 업그레이드에 관한 권장 사항을 제공하지 않습니다.
- DMS Fleet Advisor는 한 번에 최대 100개의 데이터베이스에 대한 권장 사항을 생성합니다.
- Windows 애플리케이션인 DMS 데이터 수집기를 설치하는 경우, .NET 프레임워크 4.8 및 PowerShell 6.0 이상도 설치해야 합니다. 하드웨어 요구 사항은 데이터 수집기 설치을 참조하십시오.
- DMS 데이터 수집기에는 도메인 서버에서 LDAP 프로토콜을 사용하여 요청을 실행할 수 있는 권한이 필요합니다.
- DMS 데이터 수집기를 사용하려면 Linux에서 실행되는 sudo SSH 스크립트가 필요합니다.
- DMS 데이터 수집기를 사용하려면 Windows에서 원격 PowerShell, Windows 관리 기기 (WMI), WMI 쿼리 언어 (WQL) 및 레지스트리 스크립트를 실행할 수 있는 권한이 필요합니다.
- MySQL 및 PostgreSQL의 경우, DMS Fleet Advisor는 데이터베이스에서 성능 지표를 수집할 수 없습니다. 그 대신 DMS Fleet Advisor는 OS 서버 지표를 수집합니다. 따라서 Amazon RDS 및 Aurora에서 실행되는 MySQL 및 PostgreSQL 데이터베이스의 사용률 지표를 기반으로 권장 사항을 생성할 수 없습니다.

DMS Schema Conversion을 사용하여 데이터베이스 스키마 변환

AWS Database Migration Service (AWS DMS) 에서의 DMS 스키마 변환은 서로 다른 유형의 데이터베이스 간 데이터베이스 마이그레이션을 보다 예측 가능하게 만듭니다. DMS Schema Conversion을 사용하면 소스 데이터 공급자의 마이그레이션 복잡성을 평가하고 데이터베이스 스키마 및 코드 객체를 변환할 수 있습니다. 그런 다음 변환된 코드를 대상 데이터베이스에 적용할 수 있습니다.

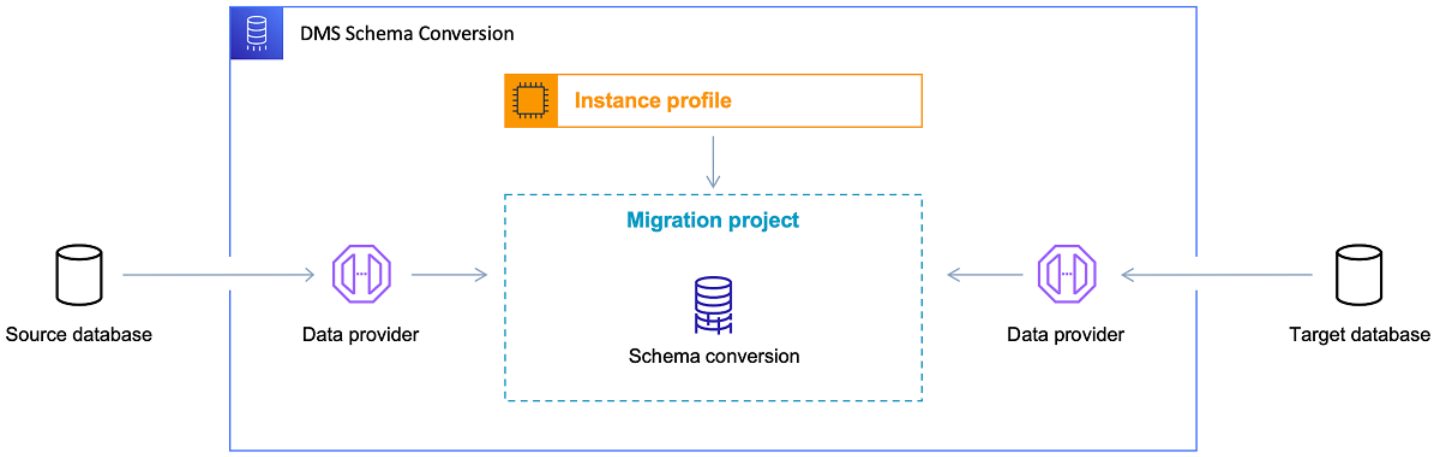
DMS Schema Conversion은 소스 데이터베이스 스키마와 대부분의 데이터베이스 코드 객체를 대상 데이터베이스와 호환되는 형식으로 자동 변환합니다. 이 변환에는 테이블, 뷰, 저장 프로시저, 함수, 데이터 형식, 동의어 등이 포함됩니다. DMS Schema Conversion에서 자동으로 변환할 수 없는 모든 객체는 명확하게 표시됩니다. 마이그레이션을 완료하기 위해 이러한 객체를 수동으로 변환할 수 있습니다.

상위 수준에서 DMS Schema Conversion은 인스턴스 프로파일, 데이터 공급자, 마이그레이션 프로젝트 등 3가지 구성 요소를 사용하여 작동합니다. 인스턴스 프로파일은 네트워크 및 보안 설정을 지정합니다. 데이터 공급자는 데이터베이스 연결 보안 인증 정보를 저장합니다. 마이그레이션 프로젝트에는 데이터 공급자, 인스턴스 프로필, 마이그레이션 규칙이 포함됩니다. AWS DMS 데이터 공급자와 인스턴스 프로필을 사용하여 데이터베이스 스키마와 코드 개체를 변환하는 프로세스를 설계합니다.

지원되는 소스 데이터베이스의 목록은 DMS Schema Conversion이 지원하는 소스 섹션을 참조하세요.

지원되는 대상 데이터베이스의 목록은 DMS Schema Conversion이 지원하는 대상 섹션을 참조하세요.

다음 다이어그램은 DMS Schema Conversion 프로세스를 보여 줍니다.



DMS Schema Conversion을 사용하는 방법을 더 잘 이해하려면 다음 항목을 참조하세요.

주제

- 지원됨 AWS 리전
- 스키마 변환 기능
- 스키마 변환 제한 사항
- DMS Schema Conversion 시작하기
- DMS Schema Conversion을 위한 네트워크 설정
- DMS Schema Conversion에서 소스 데이터 공급자 생성
- DMS Schema Conversion에서 대상 데이터 공급자 생성
- DMS Schema Conversion의 마이그레이션 프로젝트 관리
- DMS Schema Conversion을 사용하여 데이터베이스 마이그레이션 평가 보고서 생성
- DMS Schema Conversion 사용
- DMS Schema Conversion에서 확장 팩 사용

지원됨 AWS 리전

다음에서 DMS 스키마 변환 마이그레이션 프로젝트를 만들 수 있습니다. AWS 리전 다른 리전에서는 AWS Schema Conversion Tool를 사용할 수 있습니다. 에 대한 AWS SCT 자세한 내용은 AWS 스키마 변환 도구 사용 설명서를 참조하십시오.

리전 이름	리전
미국 동부(버지니아 북부)	us-east-1
미국 동부(오하이오)	us-east-2
미국 서부(오레곤)	us-west-2
아시아 태평양(도쿄)	ap-northeast-1
아시아 태평양(싱가포르)	ap-southeast-1
아시아 태평양(시드니)	ap-southeast-2
유럽(프랑크푸르트)	eu-central-1
유럽(스톡홀름)	eu-north-1

리전 이름	리전
유럽(아일랜드)	eu-west-1

스키마 변환 기능

DMS Schema Conversion은 다음과 같은 기능을 제공합니다.

- DMS 스키마 변환은 데이터베이스 마이그레이션 프로젝트에 필요한 AWS 클라우드 리소스를 자동으로 관리합니다. 이러한 리소스에는 인스턴스 프로필, 데이터 제공자 및 AWS Secrets Manager 비밀이 포함됩니다. 여기에는 AWS Identity and Access Management (IAM) 역할, Amazon S3 버킷 및 마이그레이션 프로젝트도 포함됩니다.
- DMS Schema Conversion을 사용하여 소스 데이터베이스에 연결하고, 메타데이터를 읽고, 데이터베이스 마이그레이션 평가 보고서를 생성할 수 있습니다. 그런 다음 Amazon S3 버킷에 보고서를 저장할 수 있습니다. 이 보고서를 통해 스키마 변환 작업의 요약 및 DMS Schema Conversion이 대상 데이터베이스로 자동 변환할 수 없는 항목에 대한 세부 정보를 얻을 수 있습니다. 데이터베이스 마이그레이션 평가 보고서는 DMS Schema Conversion이 마이그레이션 프로젝트를 얼마나 자동화할 수 있는지 평가하는 데 도움이 됩니다. 또한 이러한 보고서는 변환을 완료하는 데 필요한 수작업량을 추정하는 데 도움이 됩니다. 자세한 설명은 DMS Schema Conversion을 사용하여 데이터베이스 마이그레이션 평가 보고서 생성 섹션을 참조하세요.
- 소스 및 대상 데이터 공급자에 연결하면 DMS Schema Conversion이 기존 소스 데이터베이스 스키마를 대상 데이터베이스 엔진으로 변환할 수 있습니다. 소스 데이터베이스에서 변환할 스키마 항목을 선택할 수 있습니다. DMS Schema Conversion에서 데이터베이스 코드를 변환한 후 소스 코드와 변환된 코드를 검토할 수 있습니다. 또한 변환된 SQL 코드를 Amazon S3 버킷에 저장할 수 있습니다.
- 소스 데이터베이스 스키마를 변환하기 전에 변환 규칙을 설정할 수 있습니다. 변환 규칙을 사용하여 열의 데이터 형식을 변경하고, 한 스키마에서 다른 스키마로 객체를 이동하고, 객체 이름을 변경할 수 있습니다. 변환 규칙을 데이터베이스, 스키마, 테이블 및 열에 적용할 수 있습니다. 자세한 설명은 변환 규칙 설정 섹션을 참조하세요.
- 변환 설정을 변경하여 변환된 코드의 성능을 향상시킬 수 있습니다. 이러한 설정은 각 변환 쌍에 따라 다르고 코드에서 사용하는 소스 데이터베이스의 기능에 따라 달라집니다. 자세한 설명은 스키마 변환 설정 지정 섹션을 참조하세요.
- 경우에 따라 DMS Schema Conversion이 소스 데이터베이스 기능을 동등한 Amazon RDS 기능으로 변환할 수 없습니다. 이러한 경우 DMS Schema Conversion은 대상 데이터베이스에 확장 팩을 생성하여 변환되지 않은 기능을 에뮬레이션합니다. 자세한 설명은 확장 팩 사용 섹션을 참조하세요.

- 변환된 코드와 확장 팩 스키마를 대상 데이터베이스에 적용할 수 있습니다. 자세한 설명은 변환된 코드 적용 섹션을 참조하세요.
- DMS 스키마 변환은 최신 릴리스의 모든 기능을 지원합니다. AWS SCT 자세한 내용은 AWS SCT의 최신 릴리스 노트를 참조하십시오.
- DMS가 변환된 SQL 코드를 대상 데이터베이스로 마이그레이션하기 전에 해당 코드를 편집할 수 있습니다. 자세한 설명은 변환된 SQL 코드 편집 및 저장 섹션을 참조하세요.

스키마 변환 제한 사항

DMS 스키마 변환은 () 의 웹 버전입니다. AWS Schema Conversion Tool AWS SCT DMS Schema Conversion은 AWS SCT 데스크톱 애플리케이션에 비해 지원하는 데이터베이스 플랫폼 수가 적고 기능이 제한적입니다. 데이터 웨어하우스 스키마, 빅 데이터 프레임워크, 애플리케이션 SQL 코드 및 ETL 프로세스를 변환하려면 AWS SCT를 사용하세요. 에 대한 AWS SCT 자세한 내용은 AWS 스키마 변환 도구 사용 설명서를 참조하십시오.

데이터베이스 스키마 변환에 DMS Schema Conversion을 사용할 때는 다음과 같은 제한 사항이 적용됩니다.

- 마이그레이션 프로젝트를 저장한 후 오프라인 모드에서 사용할 수 없습니다.
- DMS 스키마 변환을 위한 마이그레이션 프로젝트에서 소스의 SQL 코드를 편집할 수 없습니다. 소스 데이터베이스의 SQL 코드를 편집하려면 일반 SQL 편집기를 사용하세요. 데이터베이스에서 새로 고침을 선택하여 마이그레이션 프로젝트에 업데이트된 코드를 추가합니다.
- DMS Schema Conversion의 마이그레이션 규칙은 열 데이터 정렬 변경을 지원하지 않습니다. 또한 마이그레이션 규칙을 사용하여 객체를 새 스키마로 이동할 수 없습니다.
- 소스 및 대상 데이터베이스 트리에 필터를 적용하여 필터 절을 충족하는 데이터베이스 객체만 표시할 수는 없습니다.
- DMS 스키마 변환 확장 팩에는 이메일 전송, 작업 예약 및 변환된 코드의 기타 AWS Lambda 기능을 에뮬레이션하는 기능이 포함되어 있지 않습니다.
- DMS 스키마 변환에서는 고객 리소스에 액세스하는 데 고객 관리형 KMS 키를 사용하지 않습니다. AWS 예를 들어, DMS Schema Conversion은 고객 관리형 KMS 키를 사용하여 Amazon S3의 고객 데이터에 액세스하는 것을 지원하지 않습니다.

DMS Schema Conversion 시작하기

DMS Schema Conversion을 시작하려면 다음 자습서를 사용하세요. 자습서에서는 DMS Schema Conversion을 설정하고 마이그레이션 프로젝트를 생성하며 데이터 공급자와 연결하는 방법을 배울 수 있습니다. 그런 다음, 마이그레이션의 복잡성을 평가하고 소스 데이터베이스를 대상 데이터베이스와 호환되는 형식으로 변환하는 방법을 배울 수 있습니다. 또한 변환된 코드를 대상 데이터베이스에 적용하는 방법도 배울 수 있습니다.

다음 자습서에서는 필수 작업에 관한 내용을 다루며 Amazon RDS for SQL Server 데이터베이스를 Amazon RDS for MySQL로 변환하는 방법을 보여줍니다. 지원되는 소스 및 대상 데이터 공급자를 모두 사용할 수 있습니다. 자세한 설명은 DMS Schema Conversion이 지원하는 소스 데이터 공급자 섹션을 참조하세요.

DMS 스키마 변환에 대한 자세한 내용은 Oracle에서 PostgreSQL로, SQL Server에서 MySQL로의 step-by-step 마이그레이션에 대한 마이그레이션 안내를 참조하십시오.

이 동영상은 DMS Schema Conversion 사용자 인터페이스를 소개하고 이 서비스의 핵심 구성 요소를 숙지하는 데 도움이 됩니다.

주제

- DMS Schema Conversion 작업을 위한 사전 조건
- 1단계: 인스턴스 프로파일 생성
- 2단계: 데이터 공급자 구성
- 3단계: 마이그레이션 프로젝트 생성
- 4단계: 평가 보고서 생성
- 5단계: 소스 코드 변환
- 6단계: 변환된 코드 적용
- 7단계: 정리 및 문제 해결

DMS Schema Conversion 작업을 위한 사전 조건

DMS Schema Conversion을 설정하려면 다음 작업을 완료하세요. 그런 다음, 인스턴스 프로파일을 설정하고 데이터 공급자를 추가하며 마이그레이션 프로젝트를 생성할 수 있습니다.

주제

- Amazon VPC를 기반으로 VPC 생성
- Amazon S3 버킷 생성
- 데이터베이스 자격 증명을 다음 위치에 저장합니다. AWS Secrets Manager
- IAM 역할 생성

Amazon VPC를 기반으로 VPC 생성

이 단계에서는 가상 사설 클라우드 (VPC) 를 생성합니다. AWS 계정 이 VPC는 아마존 가상 사설 클라우드 (Amazon VPC) 서비스를 기반으로 하며 사용자의 리소스를 포함합니다. AWS

DMS Schema Conversion을 위한 VPC를 생성하려면

1. AWS Management Console 로그인하고 https://console.aws.amazon.com/vpc/ 에서 Amazon VPC 콘솔을 엽니다.
2. VPC 생성을 선택합니다.
3. VPC 생성 페이지에서 다음 설정을 입력합니다.
   - 생성할 리소스 – VPC 등
   - 이름 태그 자동 생성 – 자동 생성을 선택하고 글로벌 고유 이름을 입력합니다. 예를 들면 **sc-vpc**를 입력합니다.
   - IPv4 CIDR block: - **10.0.1.0/24**
   - NAT 게이트웨이 – In 1 AZ
   - VPC 엔드포인트 – 없음
4. 해당 설정의 나머지 부분은 그대로 유지하고 VPC 만들기를 선택합니다.
5. 서브넷을 선택하고 퍼블릭 및 프라이빗 서브넷 ID를 기록해 둡니다.

   Amazon RDS 데이터베이스에 연결하려면 퍼블릭 서브넷이 포함된 서브넷 그룹을 생성하십시오.

   온프레미스 데이터베이스에 연결하려면 프라이빗 서브넷이 포함된 서브넷 그룹을 생성하십시오. 자세한 설명은 1단계: 인스턴스 프로파일 생성 섹션을 참조하세요.
6. NAT 게이트웨이를 선택합니다. NAT 게이트웨이를 선택하고 탄력적 IP 주소를 기록해 둡니다.

   이 NAT 게이트웨이의 퍼블릭 IP 주소에서 소스 온프레미스 데이터베이스에 액세스할 AWS DMS 수 있도록 네트워크를 구성하십시오. 자세한 설명은 VPC에 인터넷 연결 사용 섹션을 참조하세요.

Amazon RDS에서 인스턴스 프로파일과 대상 데이터베이스를 생성할 때 이 VPC를 사용하십시오.

Amazon S3 버킷 생성

마이그레이션 프로젝트의 정보를 저장하려면 Amazon S3 버킷을 생성하십시오. DMS Schema Conversion은 이 Amazon S3 버킷을 사용하여 평가 보고서, 변환된 SQL 코드, 데이터베이스 스키마 객체에 관한 정보 등의 항목을 저장합니다.

DMS Schema Conversion에 대한 Amazon S3 버킷을 생성하려면

1. AWS Management Console 로그인하고 https://console.aws.amazon.com/s3/ 에서 Amazon S3 콘솔을 엽니다.
2. 버킷 생성을 선택합니다.
3. 버킷 만들기 페이지에서 S3 버킷의 글로벌 고유 이름을 선택합니다. 예를 들면 **`sc-s3-bucket`**을 입력합니다.
4. AWS 리전의 경우, 리전을 선택합니다.
5. 버킷 버전 관리의 경우, 활성화를 선택합니다.
6. 해당 설정의 나머지 부분은 그대로 유지하고 버킷 만들기를 선택합니다.

데이터베이스 자격 증명을 다음 위치에 저장합니다. AWS Secrets Manager

소스 및 대상 데이터베이스 자격 증명을 에 저장합니다 AWS Secrets Manager. 이러한 암호를 자신의 AWS 리전 비밀에 복제해야 합니다. DMS Schema Conversion은 이러한 보안 암호를 사용하여 마이그레이션 프로젝트의 데이터베이스에 연결합니다.

데이터베이스 자격 증명을 저장하려면 AWS Secrets Manager

1. AWS Management Console 로그인하고 https://console.aws.amazon.com/secretsmanager/ 에서 AWS Secrets Manager 콘솔을 엽니다.
2. 새 보안 암호 저장을 선택합니다.
3. 보안 암호 유형 선택 페이지가 열립니다. 보안 암호 유형에서 저장할 데이터베이스 보안 인증 정보의 유형을 선택합니다.

   - Amazon RDS 데이터베이스의 보안 인증 정보 – Amazon RDS 데이터베이스의 자격 증명을 저장하려면 이 옵션을 선택합니다. 보안 인증 정보에서 데이터베이스에 대한 보안 인증 정보를 입력합니다. 데이터베이스에서 데이터베이스를 선택합니다.
   - 기타 데이터베이스의 보안 인증 정보 – Oracle 또는 SQL Server 소스 데이터베이스의 보안 인증 정보를 저장하려면 이 옵션을 선택합니다. 보안 인증 정보에서 데이터베이스에 대한 보안 인증 정보를 입력합니다.

- 기타 유형의 보안 암호 - 데이터베이스에 연결하는 데 필요한 사용자 이름과 암호만 저장하려면 이 옵션을 선택합니다. 행 추가를 선택하여 두 개의 키-값 쌍을 추가합니다. 키 이름에는 반드시 **username**과 **password**를 사용해야 합니다. 이러한 키와 관련된 값에는 데이터베이스의 보안 인증 정보를 입력합니다.

4. 암호화 키의 경우 Secrets Manager가 보안 값을 암호화하는 데 사용하는 AWS KMS 키를 선택합니다. 다음을 선택합니다.
5. 보안 암호 구성 페이지에서 설명이 포함된 보안 암호 이름을 입력합니다. 예를 들면, **sc-source-secret** 또는 **sc-target-secret**을 입력합니다.
6. 보안 암호 복제를 선택한 다음, AWS 리전에서 리전을 선택합니다. 다음을 선택합니다.
7. 로테이션 구성 페이지에서 다음을 선택합니다.
8. 검토 페이지에서 보안 암호 세부 정보를 검토한 후 저장을 선택합니다.

소스 및 대상 데이터베이스의 보안 인증 정보를 저장하려면 이 단계를 반복하세요.

IAM 역할 생성

마이그레이션 프로젝트에 사용할 AWS Identity and Access Management (IAM) 역할을 생성합니다. DMS Schema Conversion은 이러한 IAM 역할을 사용하여 AWS Secrets Manager에 저장된 Amazon S3 버킷 및 데이터베이스 보안 인증 정보에 액세스합니다.

Amazon S3 버킷에 대한 액세스 권한을 부여하는 IAM 정책을 생성하려면

1. AWS Management Console 로그인하고 https://console.aws.amazon.com/iam/ 에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 선택 페이지에서 AWS 서비스를 선택합니다. DMS를 선택합니다.
5. 다음을 선택합니다. 권한 추가 페이지가 열립니다.
6. 필터 정책의 경우 **S3**을 입력합니다. FullAccessAmazonS3를 선택합니다.
7. 다음을 선택합니다. 이름, 검토 및 생성 페이지가 열립니다.
8. 역할 이름에 설명이 포함된 이름을 입력합니다. 예를 들면 **sc-s3-role**를 입력합니다. 역할 생성을 선택합니다.
9. 역할 페이지에서 역할 이름에 **sc-s3-role**을 입력합니다. sc-s3-role을 선택합니다.
10. sc-s3-role 페이지에서 신뢰 관계 탭을 선택합니다. 신뢰 정책 편집을 선택합니다.

11. 신뢰 정책 편집 페이지에서 `schema-conversion.dms.amazonaws.com` 서비스 보안 주체를 신뢰할 수 있는 엔터티로 사용하도록 역할의 신뢰 관계를 편집합니다.
12. 신뢰 정책 업데이트를 선택합니다.

액세스 권한을 제공하는 IAM 역할을 만들려면 AWS Secrets Manager

1. https://console.aws.amazon.com/iam/ 에서 AWS Management Console 로그인하고 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 선택 페이지에서 AWS 서비스를 선택합니다. DMS를 선택합니다.
5. 다음을 선택합니다. 권한 추가 페이지가 열립니다.
6. 필터 정책의 경우 **`Secret`**을 입력합니다. 선택하세요 SecretsManagerReadWrite.
7. 다음을 선택합니다. 이름, 검토 및 생성 페이지가 열립니다.
8. 역할 이름에 설명이 포함된 이름을 입력합니다. 예를 들면 **`sc-secrets-manager-role`**를 입력합니다. 역할 생성을 선택합니다.
9. 역할 페이지에서 역할 이름에 **`sc-secrets-manager-role`**을 입력합니다. 선택하세요 sc-secrets-manager-role.
10. sc-secrets-manager-role페이지에서 신뢰 관계 탭을 선택합니다. 신뢰 정책 편집을 선택합니다.
11. 신뢰 정책 편집 페이지에서 사용할 역할의 신뢰 `schema-conversion.dms.amazonaws.com` 관계와 신뢰할 수 있는 주체로서의 AWS DMS 지역 서비스 주체를 편집합니다. 이 AWS DMS 지역 서비스 주체의 형식은 다음과 같습니다.

    ```
    dms.region-name.amazonaws.com
    ```

    리전의 이름인 *`region-name`*을 바꾸십시오(예: `us-east-1`).

    다음 코드 예제는 `us-east-1` 리전의 보안 주체를 보여줍니다.

    ```
    dms.us-east-1.amazonaws.com
    ```

    다음 코드 예제는 AWS DMS 스키마 변환에 액세스하기 위한 신뢰 정책을 보여줍니다.

    ```
    {
    ```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "dms.us-east-1.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "schema-conversion.dms.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

12. 신뢰 정책 업데이트를 선택합니다.

1단계: 인스턴스 프로파일 생성

인스턴스 프로파일을 생성하기 전에 인스턴스 프로파일의 서브넷 그룹을 구성하십시오. AWS DMS 마이그레이션 프로젝트의 서브넷 그룹을 만드는 방법에 대한 자세한 내용은 을 참조하십시오서브넷 그룹 생성.

다음 절차의 설명에 따라 인스턴스 프로파일을 생성할 수 있습니다. 이 인스턴스 프로파일에서는 DMS Schema Conversion 프로젝트의 네트워크 및 보안 설정을 지정합니다.

인스턴스 프로파일 생성

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 인스턴스 프로파일을 선택한 다음, 인스턴스 프로파일 생성을 선택합니다.
3. 이름에 인스턴스 프로파일의 고유한 이름을 입력합니다. 예를 들면 **sc-instance**를 입력합니다.
4. 네트워크 유형에서 IPv4를 선택하여 IPv4 주소 지정만 지원하는 인스턴스 프로파일을 생성합니다. IPv4 및 IPv6 주소 지정을 지원하는 인스턴스 프로파일을 생성하려면 듀얼 스택 모드를 선택합니다.
5. Virtual Private Cloud(VPC)의 경우, 사전 조건 단계에서 생성한 VPC를 선택합니다.

6. 서브넷 그룹의 경우, 인스턴스 프로파일에 대한 서브넷 그룹을 선택합니다. Amazon RDS 데이터베이스에 연결하려면 퍼블릭 서브넷이 포함된 서브넷 그룹을 생성하십시오. 온프레미스 데이터베이스에 연결하려면 프라이빗 서브넷이 포함된 서브넷 그룹을 생성하십시오.
7. 인스턴스 프로파일 생성을 선택합니다.

마이그레이션 프로젝트를 생성하려면 이 인스턴스 프로파일을 사용하십시오.

2단계: 데이터 공급자 구성

다음으로 소스 및 대상 데이터베이스를 설명하는 데이터 공급자를 생성합니다. 각 데이터 공급자에 대해 데이터 저장소 유형과 위치 정보를 지정합니다. 데이터베이스 보안 인증 정보는 데이터 공급자에 저장하지 않습니다.

소스 온프레미스 데이터베이스의 데이터 공급자를 생성하려면

1. 에 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 데이터 공급자를 선택한 다음, 데이터 공급자 생성을 선택합니다.
3. 이름에 소스 데이터 공급자의 고유한 이름을 입력합니다. 예를 들면 **sc-source**를 입력합니다.
4. 엔진 유형에서 데이터 공급자의 데이터베이스 엔진 유형을 선택합니다.
5. 소스 데이터베이스 연결 정보를 제공합니다. 연결 파라미터는 소스 데이터베이스 엔진에 따라 달라집니다. 자세한 설명은 데이터 공급자 생성 섹션을 참조하세요.
6. 보안 소켓 계층(SSL) 모드의 경우, SSL 적용 유형을 선택합니다.
7. 데이터 공급자 생성을 선택합니다.

대상 Amazon RDS 데이터베이스를 위한 데이터 공급자를 생성하려면

1. 에 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 데이터 공급자를 선택한 다음, 데이터 공급자 생성을 선택합니다.
3. 구성에서 RDS 데이터베이스 인스턴스를 선택합니다.
4. RDS의 데이터베이스에서 찾아보기를 선택하고 데이터베이스를 선택합니다. DMS Schema Conversion은 엔진 유형, 서버 이름 및 포트에 관한 정보를 자동으로 검색합니다.
5. 이름에 대상 데이터 공급자의 고유한 이름을 입력합니다. 예를 들면 **sc-target**을 입력합니다.
6. 데이터베이스 이름에 데이터베이스 이름을 입력합니다.
7. 보안 소켓 계층(SSL) 모드의 경우, SSL 적용 유형을 선택합니다.
8. 데이터 공급자 생성을 선택합니다.

3단계: 마이그레이션 프로젝트 생성

이제 마이그레이션 프로젝트를 생성할 수 있습니다. 마이그레이션 프로젝트에서 소스 및 대상 데이터 공급자와 인스턴스 프로파일을 지정합니다.

마이그레이션 프로젝트를 생성하려면

1. 마이그레이션 프로젝트를 선택한 다음, 마이그레이션 프로젝트 생성을 선택합니다.
2. 이름에 마이그레이션 프로젝트의 고유한 이름을 입력합니다. 예를 들면 **sc-project**를 입력합니다.
3. 인스턴스 프로파일에서 **sc-instance**를 선택합니다.
4. 소스에서 찾아보기를 선택한 다음, **sc-source**를 선택합니다.
5. 보안 암호 ID에서 **sc-source-secret**을 선택합니다.
6. IAM 역할에서 **sc-secrets-manager-role**을 선택합니다.
7. 대상에서 찾아보기를 선택한 다음, **sc-target**을 선택합니다.
8. 보안 암호 ID에서 **sc-target-secret**을 선택합니다.
9. IAM 역할에서 **schema-conversion-role**을 선택합니다.
10. 마이그레이션 프로젝트 생성을 선택합니다.

4단계: 평가 보고서 생성

마이그레이션의 복잡성을 평가하려면 데이터베이스 마이그레이션 평가 보고서를 생성하십시오. 이 보고서에는 DMS Schema Conversion이 자동으로 변환할 수 없는 모든 데이터베이스 객체의 목록이 포함됩니다.

평가 보고서를 생성하려면

1. 마이그레이션 프로젝트를 선택한 다음, **sc-project**를 선택하십시오.
2. 스키마 변환을 선택한 다음, 스키마 변환 시작을 선택합니다.
3. 소스 데이터베이스 창에서 평가할 데이터베이스 스키마를 선택합니다. 또한 이 스키마의 이름에 대한 확인란을 선택합니다.
4. 소스 데이터베이스 창의 작업 메뉴에서 평가를 선택합니다. 평가 대화 상자가 나타납니다.
5. 대화 상자에서 평가를 선택하여 선택을 확인합니다.

요약 탭에는 DMS Schema Conversion이 데이터베이스 저장소 객체 및 데이터베이스 코드 객체에 대해 자동으로 변환할 수 있는 항목의 수가 표시됩니다.

6. 작업 항목을 선택하면 DMS Schema Conversion이 자동으로 변환할 수 없는 모든 데이터베이스 객체의 목록을 볼 수 있습니다. 각 항목에 대한 권장 조치를 검토하세요.
7. 평가 보고서 사본을 저장하려면 결과 내보내기를 선택합니다. 그런 다음, CSV 또는 PDF 형식 중 하나를 선택합니다. 내보내기 대화 상자가 나타납니다.
8. 내보내기를 선택하여 선택을 확인합니다.
9. S3 버킷을 선택합니다. Amazon S3 콘솔이 열립니다.
10. 다운로드를 선택하여 평가 보고서를 저장합니다.

5단계: 소스 코드 변환

다음 절차에 따라 소스 데이터베이스 스키마를 변환할 수 있습니다. 그런 다음, 변환된 코드를 텍스트 파일에 SQL 스크립트로 저장할 수 있습니다.

데이터베이스 스키마를 변환하려면

1. 소스 데이터베이스 창에서 변환할 데이터베이스 스키마를 선택합니다. 또한 이 스키마의 이름에 대한 확인란을 선택합니다.
2. 소스 데이터베이스 창의 작업 메뉴에서 변환을 선택합니다. 변환 대화 상자가 나타납니다.
3. 대화 상자에서 변환을 선택하여 선택을 확인합니다.
4. 소스 데이터베이스 창에서 데이터베이스 객체를 선택합니다. DMS Schema Conversion은 이 객체의 소스 코드와 변환된 코드를 표시합니다.
5. 대상 데이터베이스 창에서 변환된 데이터베이스 스키마를 선택합니다. 또한 이 스키마의 이름에 대한 확인란을 선택합니다.
6. 작업에서 SQL로 저장을 선택합니다. 저장 대화 상자가 나타납니다.
7. SQL로 저장을 선택해 선택을 확인합니다.
8. S3 버킷을 선택합니다. Amazon S3 콘솔이 열립니다.
9. 다운로드를 선택하여 SQL 스크립트를 저장합니다.

6단계: 변환된 코드 적용

DMS Schema Conversion은 변환된 코드를 대상 데이터베이스에 즉시 적용하지 않습니다. 이전 단계에서 생성한 SQL 스크립트를 사용하면 대상 데이터베이스를 업데이트할 수 있습니다. 또는 다음 절차를 사용하여 DMS Schema Conversion에서 변환된 코드를 적용할 수 있습니다.

변환된 코드를 적용하려면

1. 대상 데이터베이스 창에서 변환된 데이터베이스 스키마를 선택합니다. 또한 이 스키마의 이름에 대한 확인란을 선택합니다.
2. 작업에서 변경 사항 적용을 선택합니다. 변경 사항 적용 대화 상자가 나타납니다.
3. 적용을 선택하여 선택을 확인합니다.

7단계: 정리 및 문제 해결

CloudWatch Amazon을 사용하여 DMS 스키마 변환 로그를 검토하거나 공유할 수 있습니다.

DMS Schema Conversion 로그를 검토하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 AWS Management Console 로그인하고 CloudWatch 콘솔을 엽니다.
2. 로그, 로그 그룹을 선택합니다.

   DMS Schema Conversion 로그 그룹의 이름은 `dms-tasks-sct`로 시작합니다. 생성 시간별로 로그 그룹을 정렬하여 DMS Schema Conversion 로그 그룹을 찾을 수 있습니다.

   또한 로그 그룹의 이름에는 마이그레이션 프로젝트의 Amazon 리소스 이름(ARN)이 포함됩니다. 프로젝트의 ARN은 DMS Schema Conversion의 마이그레이션 프로젝트 페이지에서 확인할 수 있습니다. 기본 설정에서 ARN을 선택했는지 확인하십시오.
3. 로그 그룹의 이름을 선택한 다음, 로그 스트림의 이름을 선택합니다.
4. 작업에서 결과 내보내기를 선택하여 DMS Schema Conversion 로그를 저장합니다.

DMS Schema Conversion에서 스키마 변환을 완료한 후에는 리소스를 정리합니다.

DMS Schema Conversion 리소스를 정리하려면

1. 에 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 마이그레이션 프로젝트를 선택합니다.

a. **sc-project**를 선택합니다.

b. 스키마 변환을 선택한 다음, 스키마 변환 종료를 선택합니다.

c. 삭제를 선택한 후 선택을 확인합니다.

3. 탐색 창에서 인스턴스 프로파일을 선택합니다.

a. **sc-instance**를 선택합니다.

b. 삭제를 선택한 후 선택을 확인합니다.

4. 탐색 창에서 데이터 공급자를 선택합니다.

a. **sc-source**와 **sc-target**을 선택합니다.

b. 삭제를 선택한 후 선택을 확인합니다.

또한 Amazon S3 버킷, 데이터베이스 암호, IAM 역할, 가상 사설 클라우드 (VPC) 등 생성한 다른 AWS 리소스를 정리해야 합니다. AWS Secrets Manager

DMS Schema Conversion을 위한 네트워크 설정

DMS Schema Conversion은 Amazon VPC 서비스를 기반으로 Virtual Private Cloud(VPC)에 스키마 변환 인스턴스를 생성합니다. 인스턴스 프로파일을 생성할 때 사용할 VPC를 지정합니다. 계정과 AWS 리전에 기본 VPC를 사용하거나 새 VPC를 생성할 수도 있습니다.

다양한 네트워크 구성을 사용하여 DMS Schema Conversion을 통해 소스 및 대상 데이터베이스의 상호 작용을 설정할 수 있습니다. 이러한 구성은 소스 데이터 공급자의 위치와 네트워크 설정에 따라 달라집니다. 다음 주제에서는 일반적인 네트워크 구성에 대해 설명합니다.

주제

- 소스 및 대상 데이터 공급자를 위한 단일 VPC 사용
- 소스 및 대상 데이터 공급자에 대해 여러 VPC를 사용
- AWS Direct Connect 또는 VPN을 사용하여 네트워크와 VPC 간 연결 구성
- VPC에 인터넷 연결 사용
- 인터넷 게이트웨이가 없는 환경 사용

소스 및 대상 데이터 공급자를 위한 단일 VPC 사용

DMS Schema Conversion을 위한 가장 간단한 네트워크 구성은 단일 VPC 구성입니다. 여기서 소스 데이터 공급자, 인스턴스 프로파일, 대상 데이터 공급자는 모두 동일한 VPC에 있습니다. 이 구성을 사용하여 Amazon EC2 인스턴스의 소스 데이터베이스를 변환할 수 있습니다.

이 구성을 사용하려면 인스턴스 프로파일에서 사용하는 VPC 보안 그룹에 데이터 공급자에 대한 액세스 권한이 있어야 합니다. 예를 들어, VPC Classless Inter-Domain Routing(CIDR) 범위 또는 Network Address Translation(NAT) 게이트웨이의 탄력적 IP 주소를 허용할 수 있습니다.

소스 및 대상 데이터 공급자에 대해 여러 VPC를 사용

소스 및 대상 데이터 공급자가 서로 다른 VPC에 있는 경우, VPC 중 하나에 인스턴스 프로파일을 생성할 수 있습니다. 그런 다음, VPC 피어링을 사용하여 이 두 VPC를 연결할 수 있습니다. 이 구성을 사용하여 Amazon EC2 인스턴스의 소스 데이터베이스를 변환할 수 있습니다.

VPC 피어링 연결은 두 VPC가 동일한 네트워크에 있는 것처럼 각 VPC의 프라이빗 IP 주소를 사용하여 라우팅을 활성화하는 두 VPC 간 네트워크 연결입니다. 자체 VPC 간, 다른 AWS 계정에서 VPC를 사용하여 또는 다른 AWS 리전에서 VPC를 사용하여 VPC 피어링 연결을 생성할 수 있습니다. VPC 피어링에 관한 자세한 내용은 Amazon VPC 사용 설명서에서 VPC 피어링을 참조하세요.

VPC 피어링을 구현하려면 Amazon VPC 사용 설명서의 VPC 피어링 연결 작업에 나와 있는 지침을 따르십시오. 한 VPC의 라우팅 테이블에 다른 VPC의 CIDR 블록이 포함되어 있는지 확인하십시오. 예를 들어, VPC A가 대상 10.0.0.0/16을 사용하고 VPC B가 대상 172.31.0.0을 사용한다고 가정해 보겠습니다. 이 경우, VPC A의 라우팅 테이블에는 172.31.0.0이 포함되어야 하고 VPC B의 라우팅 테이블에는 10.0.0.0/16이 포함되어야 합니다. 자세한 내용은 Amazon VPC 피어링 가이드의 VPC 피어링 연결에 대한 라우팅 테이블 업데이트를 참조하십시오.

AWS Direct Connect 또는 VPN을 사용하여 네트워크와 VPC 간 연결 구성

AWS Direct Connect 또는 소프트웨어나 하드웨어 VPN 연결과 같은 여러 옵션을 사용하여 원격 네트워크를 VPC에 연결할 수 있습니다. 이러한 옵션을 사용하여 내부 네트워크를 AWS 클라우드로 확장하여 기존 현장 서비스를 통합할 수 있습니다. 모니터링, 인증, 보안, 데이터 또는 기타 시스템과 같은 현장 서비스를 통합할 수 있습니다. 이러한 유형의 네트워크 확장을 활용하여 AWS에서 호스팅하는 리소스(예: VPC)에 현장 서비스를 원활하게 연결할 수 있습니다. 이 구성을 사용하여 소스 온프레미스 데이터베이스를 변환할 수 있습니다.

이 구성에서 VPC 보안 그룹에는 VPC CIDR 범위 또는 특정 IP 주소로 지정된 트래픽을 호스트로 전송하는 라우팅 규칙이 포함되어야 합니다. 이 호스트는 트래픽을 VPC에서 온프레미스 VPN으로 연결할

수 있어야 합니다. 이 경우, NAT 호스트에는 자체 보안 그룹 설정이 포함됩니다. 이러한 설정은 VPC CIDR 범위 또는 보안 그룹에서 NAT 인스턴스로 들어오는 트래픽을 허용해야 합니다. 자세한 내용은 AWS Site-to-Site VPN 사용 설명서의 Site-to-Site VPN 연결 생성을 참조하세요.

VPC에 인터넷 연결 사용

AWS 리소스에 연결하기 위해 VPN 또는 AWS Direct Connect를 사용하지 않는 경우, 인터넷 연결을 사용할 수 있습니다. 이 구성에는 인터넷 게이트웨이가 있는 VPC의 프라이빗 서브넷이 포함됩니다. 게이트웨이에는 대상 데이터 공급자와 인스턴스 프로파일이 포함됩니다. 이 구성을 사용하여 소스 온프레미스 데이터베이스를 변환할 수 있습니다.

VPC에 인터넷 게이트웨이를 추가하려면 Amazon VPC 사용 설명서의 인터넷 게이트웨이 연결을 참조하십시오.

VPC 라우팅 테이블에는 기본적으로 VPC로 향하지 않는 트래픽을 인터넷 게이트웨이로 보내는 라우팅 규칙이 포함되어야 합니다. 이 구성에서는 데이터 공급자에 대한 연결이 NAT 게이트웨이의 퍼블릭 IP 주소에서 시작되는 것으로 보입니다. 자세한 내용은 Amazon VPC 사용 설명서의 VPC 라우팅 테이블을 참조하세요.

인터넷 게이트웨이가 없는 환경 사용

인터넷 게이트웨이를 사용하지 않고 스키마 변환을 위한 환경을 생성하려면 다음과 같이 하십시오.

1. 시작하기 자습서의 1~3단계에 따라 다음과 같이 변경하십시오.
   - 퍼블릭 서브넷 대신 프라이빗 서브넷을 선택하십시오.
   - 인스턴스 생성 중에 퍼블릭 IP 할당에서 아니요를 선택합니다.
2. Amazon VPC 콘솔을 엽니다.
3. 엔드포인트를 선택한 다음, 엔드포인트 생성을 선택합니다.
4. 엔드포인트 생성 페이지에서 다음을 수행합니다.
   - 서비스 범주에서 AWS 서비스를 선택합니다.
   - 서비스 목록에서 com.amazonaws.***{region}***.secretsmanager를 선택합니다.
   - VPC 섹션에서 생성한 VPC를 선택합니다.
   - VPC에 대한 서브넷을 선택합니다.
   - VPC의 보안 그룹을 선택합니다.
   - 정책의 경우, 전체 액세스를 선택한 상태로 둡니다.

5. 시작하기 자습서의 나머지 부분을 완료하세요.

DMS Schema Conversion에서 소스 데이터 공급자 생성

Microsoft SQL Server, Oracle 또는 PostgreSQL 데이터베이스를 DMS 스키마 변환을 위한 마이그레이션 프로젝트의 소스 데이터 공급자로 사용할 수 있습니다. 소스 데이터베이스 공급자는 온프레미스 또는 Amazon EC2 인스턴스에서 실행되는 자체 관리형 엔진일 수 있습니다.

소스 데이터 공급자와 DMS Schema Conversion 간의 상호 작용을 허용하도록 네트워크를 구성해야 합니다. 자세한 정보는 DMS Schema Conversion을 위한 네트워크 설정을 참조하세요.

주제

- Microsoft SQL Server 데이터베이스를 DMS Schema Conversion의 소스로 사용
- DMS Schema Conversion에서 Oracle 데이터베이스를 소스로 사용
- DMS Schema Conversion에서 Oracle Data Warehouse 데이터베이스를 소스로 사용
- DMS 스키마 변환에서 PostgreSQL 데이터베이스를 원본으로 사용
- DMS 스키마 변환에서 MySQL 데이터베이스를 원본으로 사용

Microsoft SQL Server 데이터베이스를 DMS Schema Conversion의 소스로 사용

SQL Server 데이터베이스를 DMS Schema Conversion의 마이그레이션 소스로 사용할 수 있습니다.

DMS Schema Conversion을 사용하여 SQL Server의 데이터베이스 코드 객체를 다음 대상으로 변환할 수 있습니다.

- Aurora MySQL
- Aurora PostgreSQL
- RDS for MySQL
- RDS for PostgreSQL

지원되는 SQL Server 데이터베이스 버전에 관한 자세한 내용은 DMS Schema Conversion이 지원하는 소스 데이터 공급자을 참조하십시오.

원본 SQL Server 데이터베이스에서 DMS 스키마 변환을 사용하는 방법에 대한 자세한 내용은 SQL Server에서 step-by-step MySQL로의 마이그레이션 안내를 참조하십시오.

Microsoft SQL Server를 소스로 사용하기 위한 권한

Microsoft SQL Server를 소스로 사용하는 데 필요한 다음 권한 목록을 확인하십시오.

- 정의 보기
- 데이터베이스 상태 보기

퍼블릭 액세스 권한이 있는 사용자는 이 `VIEW DEFINITION` 권한을 통해 객체 정의를 볼 수 있습니다. DMS Schema Conversion은 `VIEW DATABASE STATE` 권한을 사용하여 SQL Server Enterprise 에디션의 기능을 확인합니다.

변환하려는 스키마의 각 데이터베이스에 대해 권한 부여를 반복합니다.

또한 `master` 데이터베이스에 다음 권한을 부여하십시오.

- 서버 상태 보기
- 정의 보기

DMS Schema Conversion은 `VIEW SERVER STATE` 권한을 사용하여 서버 설정 및 구성을 수집합니다. 데이터 공급자를 볼 수 있는 `VIEW ANY DEFINITION` 권한을 부여했는지 확인하십시오.

Microsoft Analysis Services에 관한 정보를 읽으려면 `master` 데이터베이스에서 다음 명령을 실행합니다.

```
EXEC master..sp_addsrvrolemember @loginame = N'<user_name>', @rolename = N'sysadmin'
```

앞의 예제에서 *<user_name>* 자리 표시자를 이전에 필요한 권한을 부여한 사용자의 이름으로 바꿉니다.

SQL Server 에이전트에 대한 정보를 읽으려면 사용자를 SQL 역할에 추가하십시오. AgentUser `msdb` 데이터베이스에서 다음 명령을 실행합니다.

```
EXEC sp_addrolemember <SQLAgentRole>, <user_name>;
```

앞의 예제에서 *<SQLAgentRole>* 자리 표시자를 SQL Server 에이전트 역할의 이름으로 바꿉니다. 그런 다음, *<user_name>* 자리 표시자를 이전에 필요한 권한을 부여한 사용자의 이름으로 바꿉니다. 자세한 내용은 Amazon RDS 사용 설명서의 SQL AgentUser 역할에 사용자 추가를 참조하십시오.

로그 전달을 감지하려면 msdb 데이터베이스에 대한 SELECT on dbo.log_shipping_primary_databases 권한을 부여하십시오.

데이터 정의 언어(DDL) 복제의 알림 접근 방식을 사용하려면 소스 데이터베이스에 대해 RECEIVE ON *<schema_name>*.*<queue_name>* 권한을 부여하십시오. 이 예제에서는 *<schema_name>* 자리 표시자를 데이터베이스의 스키마 이름으로 바꿉니다. 그런 다음, *<queue_name>* 자리 표시자를 대기열 테이블 이름으로 바꾸십시오.

DMS Schema Conversion에서 Oracle 데이터베이스를 소스로 사용

Oracle 데이터베이스를 DMS Schema Conversion의 마이그레이션 소스로 사용할 수 있습니다.

Oracle 데이터베이스에 연결하려면 Oracle 시스템 ID(SID)를 사용합니다. Oracle SID를 확인하려면 Oracle 데이터베이스에 다음 쿼리를 제출합니다.

```
SELECT sys_context('userenv','instance_name') AS SID FROM dual;
```

DMS Schema Conversion을 사용하여 Oracle 데이터베이스의 데이터베이스 코드 객체를 다음 대상으로 변환할 수 있습니다.

- Aurora MySQL
- Aurora PostgreSQL
- RDS for MySQL
- RDS for PostgreSQL

지원되는 Oracle 데이터베이스 버전에 관한 자세한 내용은 DMS Schema Conversion이 지원하는 소스 데이터 공급자을 참조하십시오.

원본 Oracle 데이터베이스에서 DMS 스키마 변환을 사용하는 방법에 대한 자세한 내용은 Oracle에서 PostgreSQL로의 마이그레이션 안내를 참조하십시오. step-by-step

Oracle을 소스로 사용하기 위한 권한

Oracle이 소스일 경우 필요한 권한은 다음과 같습니다.

- CONNECT
- SELECT_CATALOG_ROLE
- SELECT ANY DICTIONARY
- SELECT ON SYS.ARGUMENT$

DMS Schema Conversion에서 Oracle Data Warehouse 데이터베이스를 소스로 사용

Oracle Data Warehouse 데이터베이스를 DMS Schema Conversion의 마이그레이션 소스로 사용하여 데이터베이스 코드 객체와 애플리케이션 코드를 Amazon Redshift로 변환할 수 있습니다.

지원되는 Oracle 데이터베이스 버전에 대한 자세한 내용은 DMS Schema Conversion이 지원하는 소스 데이터 공급자 섹션을 참조하세요. 원본 Oracle 데이터베이스에서 DMS 스키마 변환을 사용하는 방법에 대한 자세한 내용은 Oracle에서 PostgreSQL로의 마이그레이션 안내를 참조하십시오. step-by-step

Oracle Data Warehouse 데이터베이스를 소스로 사용하기 위한 권한

Oracle Data Warehouse를 소스로 사용하기 위해 필요한 권한은 다음과 같습니다.

- CONNECT
- SELECT_CATALOG_ROLE
- SELECT ANY DICTIONARY

Oracle Data Warehouse에서 Amazon Redshift로의 변환 설정

DMS Schema Conversion 설정에 대한 자세한 내용은 마이그레이션 프로젝트에 대한 스키마 변환 설정 지정 섹션을 참조하세요.

Oracle Data Warehouse를 Amazon Redshift로 변환하는 설정에는 다음과 같은 옵션이 포함됩니다.

- 선택한 심각도 이상의 작업 항목에 대해 변환된 코드에 주석 추가: 이 설정은 변환된 코드에서 작업 항목에 대한 주석 수를 제한합니다. DMS는 선택한 수준 또는 그보다 높은 수준의 심각도에 해당하는 작업 항목에 대해 변환된 코드에 주석을 추가합니다.

  예를 들어, 변환된 코드의 설명 수를 최소화하려면 오류만을 선택합니다. 변환된 코드의 모든 작업 항목에 대한 설명을 포함하려면 모든 메시지를 선택합니다.
- 대상 Amazon Redshift 클러스터의 최대 테이블 수: 이 설정은 DMS가 대상 Amazon Redshift 클러스터에 적용할 수 있는 최대 테이블 수를 설정합니다. Amazon Redshift에는 여러 클러스터 노드 유형에 사용하는 테이블을 제한하는 할당량이 있습니다. 이 설정은 다음과 같은 값을 지원합니다.
  - 자동: DMS가 노드 유형에 따라 대상 Amazon Redshift 클러스터에 적용되는 테이블 수를 결정합니다.
  - 값 설정: 테이블 수를 수동으로 설정합니다.

테이블 수가 Amazon Redshift 클러스터가 저장할 수 있는 양보다 많더라도 DMS는 모든 소스 테이블을 변환합니다. DMS는 변환된 코드를 프로젝트에 저장하며 대상 데이터베이스에는 적용하지 않습니다. 변환된 코드를 적용할 때 테이블의 Amazon Redshift 클러스터 할당량에 도달하면 DMS에서 경고 메시지가 표시됩니다. 또한 DMS는 테이블 수가 한도에 도달할 때까지 대상 Amazon Redshift 클러스터에 테이블을 적용합니다.

Amazon Redshift 테이블 할당량에 대한 자세한 내용은 Amazon Redshift의 할당량 및 제한을 참조하세요.

- UNION ALL 보기 사용: 이 설정을 사용하면 DMS가 단일 소스 테이블에 대해 생성할 수 있는 대상 테이블의 최대 수를 설정할 수 있습니다.

  Amazon Redshift는 테이블 파티셔닝을 지원하지 않습니다. 테이블 파티셔닝을 에뮬레이션하고 쿼리를 더 빠르게 실행하기 위해 DMS는 소스 테이블의 각 파티션을 Amazon Redshift의 개별 테이블로 마이그레이션할 수 있습니다. 그런 다음 DMS는 생성하는 모든 대상 테이블의 데이터를 포함하는 보기를 생성합니다.

  DMS는 소스 테이블의 파티션 수를 자동으로 결정합니다. 소스 테이블 파티셔닝 유형에 따라서는 이 숫자가 Amazon Redshift 클러스터에 적용할 수 있는 테이블의 할당량을 초과할 수 있습니다. 이 할당량에 도달하지 않도록 하려면 DMS가 단일 소스 테이블의 파티션에 대해 생성할 수 있는 최대 대상 테이블 수를 입력합니다. 기본 옵션은 368개 테이블이며, 이는 1년 중 366일 동안의 파티션과 NO RANGE 및 UNKNOWN 파티션에 대한 테이블 두 개를 나타냅니다.

- Oracle 코드에서 사용하는 날짜 유형 형식 요소가 Amazon Redshift의 날짜/시간 형식 문자열과 유사: 이 설정을 사용하면 Amazon Redshift에서 지원하지 않는 날짜/시간 형식 요소를 포함하여 `TO_CHAR`, `TO_DATE`, `TO_NUMBER` 같은 데이터 유형 형식 지정 함수를 변환할 수 있습니다. 기본적으로 DMS는 확장 팩 함수를 사용하여 변환된 코드에서 지원되지 않는 이러한 형식 요소를 에뮬레이션합니다.

  Oracle의 날짜/시간 형식 모델에는 Amazon Redshift의 날짜/시간 형식 문자열보다 더 많은 요소가 포함되어 있습니다. 소스 코드에 Amazon Redshift가 지원하는 날짜/시간 형식 요소만 포함된 경우 변환된 코드에 확장 팩 함수가 없도록 하려면 이 값을 설정합니다. 확장 함수를 사용하지 않으면 변환된 코드가 더 빠르게 실행됩니다.

- Oracle 코드에서 사용하는 숫자 형식 요소가 Amazon Redshift의 숫자 형식 문자열과 유사: 이 설정을 사용하면 Amazon Redshift에서 지원하지 않는 숫자 데이터 유형 형식 지정 함수를 변환할 수 있습니다. 기본적으로 DMS는 확장 팩 함수를 사용하여 변환된 코드에서 지원되지 않는 이러한 형식 요소를 에뮬레이션합니다.

Oracle의 숫자 형식 모델에는 Amazon Redshift의 숫자 형식 문자열에 비해 더 많은 요소가 포함되어 있습니다. 소스 코드에 Amazon Redshift가 지원하는 숫자 형식 요소만 포함된 경우 변환된 코드에 확장 팩 함수가 없도록 하려면 이 값을 설정합니다. 확장 함수를 사용하지 않으면 변환된 코드가 더 빠르게 실행됩니다.

- NVL 함수를 사용하여 Oracle LEAD 및 LAG 함수의 동작 에뮬레이션: 소스 코드에서 LEAD 및 LAG 함수의 오프셋 기본값을 사용하지 않는 경우 DMS는 NVL 함수를 사용하여 이러한 함수를 에뮬레이션할 수 있습니다. 기본적으로 DMS는 각 LEAD 및 LAG 함수 사용에 대해 작업 항목을 발생시킵니다. NVL을 사용하여 이러한 함수를 에뮬레이션하면 변환된 코드가 더 빠르게 실행됩니다.
- 프라이머리 키와 고유 키의 동작 에뮬레이션: 이 설정을 지정하면 DMS가 대상 Amazon Redshift 클러스터에서 프라이머리 키와 고유 키 제약 조건 동작을 에뮬레이션합니다. Amazon Redshift는 프라이머리 키와 고유 키를 적용하지 않으며 정보 제공 목적으로만 사용합니다. 소스 코드에서 프라이머리 키 또는 고유 키 제약 조건을 사용하는 경우 DMS가 해당 동작을 에뮬레이션하도록 하려면 이 설정을 지정합니다.
- 압축 인코딩 사용: Amazon Redshift 테이블 열에 압축 인코딩을 적용하려면 이 설정을 지정합니다. DMS는 기본 Redshift 알고리즘을 사용하여 압축 인코딩을 자동으로 할당합니다. 압축 인코딩에 대한 자세한 내용은 Amazon Redshift 데이터베이스 개발자 안내서의 압축 인코딩을 참조하세요.

  기본적으로 Amazon Redshift는 정렬 및 배포 키로 정의된 열에 압축을 적용하지 않습니다. 이러한 열에 압축을 적용하려면 KEY 열에 압축 인코딩 사용을 설정합니다. 압축 인코딩 사용을 설정한 경우에만 이 옵션을 선택할 수 있습니다.

DMS 스키마 변환에서 PostgreSQL 데이터베이스를 원본으로 사용

PostgreSQL 데이터베이스를 DMS 스키마 변환에서 마이그레이션 소스로 사용할 수 있습니다.

DMS 스키마 변환을 사용하여 PostgreSQL 데이터베이스의 데이터베이스 코드 객체를 다음 대상으로 변환할 수 있습니다.

- MySQL
- Aurora MySQL

PostgreSQL을 소스로 사용하기 위해 필요한 권한은 다음과 같습니다.

- CONNECT ON DATABASE <database_name>
- USAGE ON SCHEMA <database_name>
- SELECT ON ALL TABLES IN SCHEMA <database_name>

- SELECT ON ALL SEQUENCES IN SCHEMA <database_name>

DMS 스키마 변환에서 MySQL 데이터베이스를 원본으로 사용

MySQL 데이터베이스를 DMS 스키마 변환의 마이그레이션 소스로 사용할 수 있습니다.

DMS 스키마 변환을 사용하여 MySQL 데이터베이스의 데이터베이스 코드 객체를 다음 대상으로 변환할 수 있습니다.

- PostgreSQL
- Aurora PostgreSQL

MySQL이 소스일 경우 필요한 권한은 다음과 같습니다.

- `SELECT ON *.*`
- `SHOW VIEW ON *.*`

MySQL에서 PostgreSQL로의 변환 설정

DMS Schema Conversion 설정에 대한 자세한 내용은 마이그레이션 프로젝트에 대한 스키마 변환 설정 지정 섹션을 참조하세요.

MySQL에서 PostgreSQL로의 변환 설정에는 다음이 포함됩니다.

- 변환된 SQL 코드의 주석: 선택한 심각도 이상의 작업 항목에 대해 변환된 코드에 주석을 추가하려면 이 설정을 설정하십시오.

  유효한 값:
  - 오류만
  - Errors and warnings(오류 및 경고)
  - 모든 메시지

DMS Schema Conversion에서 대상 데이터 공급자 생성

MySQL 및 PostgreSQL 데이터베이스를 DMS Schema Conversion을 위한 마이그레이션 프로젝트에서 대상 데이터 공급자로 사용할 수 있습니다. 대상 데이터 공급자는 Amazon EC2, Amazon RDS 또는 Amazon Aurora 인스턴스일 수 있습니다.

주제

- DMS Schema Conversion에서 MySQL 데이터베이스를 대상으로 사용
- DMS Schema Conversion에서 PostgreSQL 데이터베이스를 대상으로 사용하기
- Amazon Redshift 클러스터를 DMS Schema Conversion의 대상으로 사용

DMS Schema Conversion에서 MySQL 데이터베이스를 대상으로 사용

MySQL 데이터베이스를 DMS Schema Conversion의 마이그레이션 대상으로 사용할 수 있습니다.

지원되는 대상 데이터베이스에 관한 자세한 내용은 DMS Schema Conversion이 지원하는 대상 데이터 공급자 단원을 참조하십시오.

대상으로서 MySQL에 대한 권한

대상으로서 MySQL에 필요한 권한은 다음과 같습니다.

- `CREATE ON *.*`
- `ALTER ON *.*`
- `DROP ON *.*`
- `INDEX ON *.*`
- `REFERENCES ON *.*`
- `SELECT ON *.*`
- `CREATE VIEW ON *.*`
- `SHOW VIEW ON *.*`
- `TRIGGER ON *.*`
- `CREATE ROUTINE ON *.*`
- `ALTER ROUTINE ON *.*`
- `EXECUTE ON *.*`
- `CREATE TEMPORARY TABLES ON *.*`
- `AWS_LAMBDA_ACCESS`
- `INSERT, UPDATE ON AWS_ORACLE_EXT.*`
- `INSERT, UPDATE, DELETE ON AWS_ORACLE_EXT_DATA.*`

- INSERT, UPDATE ON AWS_SQLSERVER_EXT.*
- INSERT, UPDATE, DELETE ON AWS_SQLSERVER_EXT_DATA.*
- CREATE TEMPORARY TABLES ON AWS_SQLSERVER_EXT_DATA.*

다음 코드 예제를 사용하여 데이터베이스 사용자를 생성하고 권한을 부여할 수 있습니다.

```
CREATE USER 'user_name' IDENTIFIED BY 'your_password';
GRANT CREATE ON *.* TO 'user_name';
GRANT ALTER ON *.* TO 'user_name';
GRANT DROP ON *.* TO 'user_name';
GRANT INDEX ON *.* TO 'user_name';
GRANT REFERENCES ON *.* TO 'user_name';
GRANT SELECT ON *.* TO 'user_name';
GRANT CREATE VIEW ON *.* TO 'user_name';
GRANT SHOW VIEW ON *.* TO 'user_name';
GRANT TRIGGER ON *.* TO 'user_name';
GRANT CREATE ROUTINE ON *.* TO 'user_name';
GRANT ALTER ROUTINE ON *.* TO 'user_name';
GRANT EXECUTE ON *.* TO 'user_name';
GRANT CREATE TEMPORARY TABLES ON *.* TO 'user_name';
GRANT AWS_LAMBDA_ACCESS TO 'user_name';
GRANT INSERT, UPDATE ON AWS_ORACLE_EXT.* TO 'user_name';
GRANT INSERT, UPDATE, DELETE ON AWS_ORACLE_EXT_DATA.* TO 'user_name';
GRANT INSERT, UPDATE ON AWS_SQLSERVER_EXT.* TO 'user_name';
GRANT INSERT, UPDATE, DELETE ON AWS_SQLSERVER_EXT_DATA.* TO 'user_name';
GRANT CREATE TEMPORARY TABLES ON AWS_SQLSERVER_EXT_DATA.* TO 'user_name';
```

이전 예제에서 *user_name*을 사용자 이름으로 바꿉니다. 그런 다음 *your_password*를 안전한 암호로 바꿉니다.

Amazon RDS for MySQL 또는 Aurora MySQL을 대상으로 사용하려면 lower_case_table_names 파라미터를 1로 설정합니다. 이 값은 MySQL 서버가 테이블, 인덱스, 트리거 및 데이터베이스와 같은 객체 이름의 식별자를 대소문자 구분 없이 처리한다는 것을 의미합니다. 대상 인스턴스에서 이진 로깅을 활성화했다면 log_bin_trust_function_creators 파라미터를 1로 설정합니다. 이 경우 저장된 함수를 생성하기 위해 DETERMINISTIC, READS SQL DATA 또는 NO SQL 특성을 사용할 필요가 없습니다. 이들 파라미터를 구성하려면 새 DB 파라미터 그룹을 생성하거나 기존 DB 파라미터 그룹을 수정해야 합니다.

DMS Schema Conversion에서 PostgreSQL 데이터베이스를 대상으로 사용하기

PostgreSQL 데이터베이스를 DMS Schema Conversion에서 마이그레이션 대상으로 사용할 수 있습니다.

지원되는 대상 데이터베이스에 관한 자세한 내용은 DMS Schema Conversion이 지원하는 대상 데이터 공급자 단원을 참조하십시오.

대상으로서 PostgreSQL에 대한 권한

PostgreSQL을 대상으로 사용하려면 DMS Schema Conversion에 `CREATE ON DATABASE` 권한이 필요합니다. DMS Schema Conversion을 위한 마이그레이션 프로젝트에서 사용할 각 데이터베이스에 대해 사용자를 생성한 후 이 권한을 사용자에게 부여하십시오.

Amazon RDS for PostgreSQL을 대상으로 사용하려면 DMS Schema Conversion에 `rds_superuser` 역할이 필요합니다.

변환된 공개 동의어를 사용하려면 다음 명령을 사용하여 데이터베이스 기본 검색 경로를 변경합니다.

```
ALTER DATABASE <db_name> SET SEARCH_PATH = "$user", public_synonyms, public;
```

이 예제에서는 *<db_name>* 자리 표시자를 데이터베이스의 이름으로 바꿉니다.

PostgreSQL에서는 스키마 소유자 또는 `superuser`만 스키마를 삭제할 수 있습니다. 스키마 소유자가 일부 객체를 소유하지 않은 경우에도 스키마와 이 스키마에 포함된 모든 객체를 삭제할 수 있습니다.

여러 사용자를 사용하여 대상 데이터베이스를 변환하고 다른 스키마를 적용할 때 DMS Schema Conversion에서 스키마를 삭제할 수 없는 경우 오류 메시지가 표시될 수 있습니다. 이 오류 메시지가 표시되지 않도록 하려면 `superuser` 역할을 사용하십시오.

Amazon Redshift 클러스터를 DMS Schema Conversion의 대상으로 사용

Amazon Redshift 데이터베이스를 DMS Schema Conversion의 마이그레이션 대상으로 사용할 수 있습니다. 지원되는 대상 데이터베이스에 관한 자세한 내용은 DMS Schema Conversion이 지원하는 대상 데이터 공급자 섹션을 참조하세요.

Amazon Redshift를 대상으로 사용하기 위한 권한

Amazon Redshift를 DMS Schema Conversion의 대상으로 사용하려면 다음과 같은 권한이 필요합니다.

- CREATE ON DATABASE: DMS가 데이터베이스에 새 스키마를 생성할 수 있습니다.
- CREATE ON SCHEMA: DMS가 데이터베이스 스키마에 객체를 생성할 수 있습니다.
- GRANT USAGE ON LANGUAGE: DMS가 데이터베이스에 새 함수와 프로시저를 생성할 수 있습니다.
- GRANT SELECT ON ALL TABLES IN SCHEMA pg_catalog: Amazon Redshift 클러스터에 대한 시스템 정보를 사용자에게 제공합니다.
- GRANT SELECT ON pg_class_info: 사용자에게 테이블 배포 스타일에 대한 정보를 제공합니다.

다음 코드 예제를 사용하여 데이터베이스 사용자를 생성하고 권한을 부여할 수 있습니다. 예제 값을 사용자의 값으로 바꿉니다.

```
CREATE USER user_name PASSWORD your_password;
GRANT CREATE ON DATABASE db_name TO user_name;
GRANT CREATE ON SCHEMA schema_name TO user_name;
GRANT USAGE ON LANGUAGE plpythonu TO user_name;
GRANT USAGE ON LANGUAGE plpgsql TO user_name;
GRANT SELECT ON ALL TABLES IN SCHEMA pg_catalog TO user_name;
GRANT SELECT ON pg_class_info TO user_name;
GRANT SELECT ON sys_serverless_usage TO user_name;
GRANT SELECT ON pg_database_info TO user_name;
GRANT SELECT ON pg_statistic TO user_name;
```

변환된 코드를 적용하거나 데이터를 마이그레이션할 각 대상 스키마에 대해 `GRANT CREATE ON SCHEMA` 작업을 반복합니다.

대상 Amazon Redshift 데이터베이스에 확장 팩을 적용할 수 있습니다. 확장 팩은 객체를 Amazon Redshift로 변환할 때 필요한 소스 데이터베이스 함수를 에뮬레이션하는 추가 기능 모듈입니다. 자세한 내용은 DMS Schema Conversion에서 확장 팩 사용 섹션을 참조하세요.

DMS Schema Conversion의 마이그레이션 프로젝트 관리

스키마 변환을 위한 인스턴스 프로파일과 호환 가능한 데이터 공급자를 생성한 후 마이그레이션 프로젝트를 생성하십시오. 자세한 내용은 마이그레이션 프로젝트 생성 섹션을 참조하세요.

DMS Schema Conversion에서 이 새 프로젝트를 사용하려면 마이그레이션 프로젝트 페이지의 목록에서 프로젝트를 선택합니다. 그런 다음, 스키마 변환 탭에서 스키마 변환 시작을 선택합니다.

DMS Schema Conversion을 처음 실행하려면 몇 가지 설정이 필요합니다. AWS Database Migration Service(AWS DMS)을 실행하려면 스키마 변환 인스턴스가 필요한데, 이 작업에는 최대 15분이 소요됩

니다. 또한 이 프로세스는 소스 및 대상 데이터베이스에서 메타데이터를 읽습니다. 첫 번째 실행에 성공하면 DMS Schema Conversion에 더 빠르게 액세스할 수 있습니다.

Amazon은 마이그레이션 프로젝트에서 사용하는 스키마 변환 인스턴스를 프로젝트 완료 후 3일 이내에 종료합니다. DMS Schema Conversion에 사용하는 Amazon S3 버킷에서 변환된 스키마와 평가 보고서를 검색할 수 있습니다.

DMS Schema Conversion을 위한 마이그레이션 프로젝트 설정 지정

마이그레이션 프로젝트를 생성하고 스키마 변환을 시작한 후 마이그레이션 프로젝트 설정을 지정할 수 있습니다. 변환 설정을 변경하여 변환된 코드의 성능을 향상시킬 수 있습니다. 또한 스키마 변환 보기를 사용자 지정할 수 있습니다.

변환 설정은 소스 및 대상 데이터베이스 플랫폼에 따라 달라집니다. 자세한 정보는 소스 데이터 공급자 생성 및 대상 데이터 공급자 생성 섹션을 참조하십시오.

소스 및 대상 데이터베이스 창에 표시할 스키마와 데이터베이스를 지정하려면 트리 보기 설정을 사용합니다. 빈 스키마, 빈 데이터베이스, 시스템 데이터베이스, 사용자 정의 데이터베이스 또는 스키마를 숨길 수 있습니다.

트리 보기에서 데이터베이스와 스키마를 숨기려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/dms/v2/에서 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택하고 스키마 변환 탭에서 스키마 변환 시작을 선택합니다.
4. 설정을 선택합니다. 설정 페이지가 열립니다.
5. 트리 보기 섹션에서 다음을 수행합니다.

   - 빈 스키마를 숨기려면 빈 스키마 숨기기를 선택합니다.
   - 빈 데이터베이스를 숨기려면 빈 데이터베이스 숨기기를 선택합니다.
   - 시스템 데이터베이스 또는 스키마의 경우, 시스템 데이터베이스 및 스키마를 이름별로 선택하여 숨깁니다.
   - 사용자 정의 데이터베이스 또는 스키마의 경우, 숨기려는 사용자 정의 데이터베이스 및 스키마의 이름을 입력합니다. 추가를 선택합니다. 이름은 대/소문자를 구분하지 않습니다.

여러 데이터베이스 또는 스키마를 추가하려면 쉼표를 사용하여 이름을 구분합니다. 이름이 비슷한 객체를 여러 개 추가하려면 백분율(%)을 와일드카드로 사용합니다. 이 와일드카드는 데이터베이스 또는 스키마 이름에 있는 기호의 수를 대체합니다.

소스 및 대상 섹션에 대해 이 단계를 반복합니다.

6. 적용을 선택한 다음, 스키마 변환을 선택합니다.

DMS Schema Conversion을 사용하여 데이터베이스 마이그레이션 평가 보고서 생성

DMS Schema Conversion의 중요한 부분은 스키마를 변환하는 데 도움이 되도록 생성하는 보고서입니다. 이 데이터베이스 마이그레이션 평가 보고서는 모든 스키마 변환 태스크를 요약합니다. 또한 대상 DB 인스턴스의 DB 엔진으로 변환할 수 없는 스키마에 대한 작업 항목도 자세히 설명합니다. 이 보고서를 AWS DMS 콘솔에서 보거나 이 보고서의 사본을 PDF 또는 CSV(쉼표로 구분된 값) 파일로 저장할 수 있습니다.

마이그레이션 평가 보고서에는 다음 내용이 포함됩니다.

- 핵심 요약
- 서버 객체 변환, 백업 제안, 연결된 서버 변경을 포함한 권장 사항

DMS Schema Conversion이 자동으로 변환할 수 없는 항목이 있는 경우 보고서는 대상 DB 인스턴스에 대해 동등한 코드를 작성하는 데 필요한 예상 노력을 제시합니다.

주제

- 데이터베이스 마이그레이션 평가 보고서 생성
- 데이터베이스 마이그레이션 평가 보고서 보기
- 데이터베이스 마이그레이션 평가 보고서 저장

데이터베이스 마이그레이션 평가 보고서 생성

마이그레이션 프로젝트를 생성한 후 다음 절차를 사용하여 데이터베이스 마이그레이션 평가 보고서를 작성합니다.

데이터베이스 마이그레이션 평가 보고서를 작성하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/dms/v2/에서 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택한 다음, 스키마 변환을 선택합니다.
4. 스키마 변환 시작을 선택합니다. 스키마 변환 페이지가 열립니다.
5. 소스 데이터베이스 창에서 평가할 데이터베이스 스키마 또는 스키마 항목을 선택합니다. 보고서에 여러 객체를 포함하려면 모든 항목을 선택해야 합니다.
6. 평가하려는 모든 스키마 객체의 확인란을 선택한 후에는 선택한 객체의 상위 노드를 선택해야 합니다. 이제 소스 데이터베이스 창의 작업 메뉴를 사용할 수 있습니다.
7. 작업 메뉴에서 평가를 선택합니다. 확인 대화 상자가 표시됩니다.
8. 대화 상자에서 평가를 선택하여 선택을 확인합니다.

데이터베이스 마이그레이션 평가 보고서 보기

평가 보고서를 작성하면 DMS Schema Conversion은 다음 탭에 정보를 추가합니다.

- 요약
- 작업 항목

요약 탭에는 DMS Schema Conversion이 자동으로 변환할 수 있는 항목 수가 표시됩니다.

작업 항목 탭에는 DMS Schema Conversion이 자동으로 변환할 수 없는 항목이 표시되며 이러한 항목을 관리하는 방법에 대한 권장 사항이 제공됩니다.

평가 보고서 요약

요약 탭에는 데이터베이스 마이그레이션 평가 보고서의 요약 정보가 표시됩니다. 여기에는 DMS Schema Conversion이 데이터베이스 스토리지 객체 및 데이터베이스 코드 객체에 대해 자동으로 변환할 수 있는 항목의 수가 표시됩니다.

대부분의 경우 DMS Schema Conversion은 모든 스키마 항목을 대상 데이터베이스 엔진으로 자동 변환할 수 없습니다. 요약 탭은 대상 DB 인스턴스에서 소스의 스키마 항목과 동일한 스키마 항목을 생성하는 데 필요한 예상 작업량을 제공합니다.

테이블, 시퀀스, 제약 조건, 데이터 형식 등과 같은 데이터베이스 스토리지 객체의 변환 요약을 보려면 데이터베이스 스토리지 객체를 선택합니다.

프로시저, 함수, 뷰, 트리거 등과 같은 데이터베이스 코드 객체의 변환 요약을 보려면 데이터베이스 코드 객체를 선택합니다.

평가 보고서의 범위를 변경하려면 소스 데이터베이스 트리에서 필요한 노드를 선택합니다. DMS Schema Conversion은 선택한 범위와 일치하도록 평가 보고서 요약을 업데이트합니다.

평가 보고서 작업 항목

작업 항목 탭에는 DMS Schema Conversion에서 대상 데이터베이스 엔진과 호환되는 형식으로 자동 변환할 수 없는 항목 목록이 포함되어 있습니다. 각 작업 항목에 대해 DMS Schema Conversion은 문제의 설명 및 권장 조치를 제공합니다. DMS Schema Conversion은 유사한 작업 항목을 그룹화하고 발생 횟수를 표시합니다.

관련 데이터베이스 객체의 코드를 보려면 목록에서 작업 항목을 하나 선택합니다.

데이터베이스 마이그레이션 평가 보고서 저장

데이터베이스 마이그레이션 평가 보고서를 작성한 후 이 보고서의 사본을 PDF 또는 CSV(쉼표로 구분된 값) 파일로 저장할 수 있습니다.

데이터베이스 마이그레이션 평가 보고서를 PDF 파일로 저장하려면

1. 내보내기를 선택한 다음 PDF를 선택합니다. 대화 상자를 검토하고 PDF로 내보내기를 선택합니다.
2. DMS Schema Conversion은 PDF 파일로 아카이브를 생성하고 이 아카이브를 Amazon S3 버킷에 저장합니다. Amazon S3 버킷을 변경하려면 인스턴스 프로파일에서 스키마 변환 설정을 편집하세요.
3. Amazon S3 버킷에서 평가 보고서 파일을 엽니다.

데이터베이스 마이그레이션 평가 보고서를 CSV 파일로 저장하려면

1. 내보내기를 선택한 다음 CSV를 선택합니다. 대화 상자를 검토하고 CSV로 내보내기를 선택합니다.
2. DMS Schema Conversion은 CSV 파일로 아카이브를 생성하고 이 아카이브를 Amazon S3 버킷에 저장합니다. Amazon S3 버킷을 변경하려면 인스턴스 프로파일에서 스키마 변환 설정을 편집하세요.

3. Amazon S3 버킷에서 평가 보고서 파일을 엽니다.

PDF 파일에는 요약과 작업 항목 정보가 모두 들어 있습니다.

평가 보고서를 CSV로 내보내는 경우 DMS Schema Conversion이 세 개의 CSV 파일을 생성합니다.

첫 번째 CSV 파일은 다음 작업 항목에 대한 정보를 포함합니다.

- 범주
- 발생
- 작업 항목
- 제목
- 그룹
- 설명
- 문서 참조
- 권장 조치
- 행
- 위치
- 소스
- 대상
- 서버 IP 주소 및 포트
- 데이터베이스
- Schema

두 번째 CSV 파일은 이름에 `Action_Items_Summary` 접미사가 포함되며 다음 정보를 포함하고 있습니다.

- Schema
- 작업 항목
- 발생 횟수
- 학습 곡선 노력(각 작업 항목을 변환하기 위한 접근 방식을 설계하는 데 필요한 노력의 양)
- 각 작업 항목을 변환하기 위한 작업량(설계된 접근 방식에 따라 각 작업 항목을 변환하는 데 필요한 노력)
- 작업 항목 설명

- 권장 조치

필요한 노력의 수준을 나타내는 값은 낮음(최소)에서 높음(가장)까지의 가중치 척도를 기반으로 합니다.

세 번째 CSV 파일은 이름에 `Summary`가 포함되며 다음 정보를 포함하고 있습니다.

- 범주
- 객체 수
- 자동으로 변환된 객체
- 간단한 작업이 포함된 객체
- 중간 복잡도의 작업이 포함된 객체
- 복잡한 작업이 포함된 객체
- 총 코드 줄 수

DMS Schema Conversion 사용

DMS Schema Conversion은 기존 데이터베이스 스키마와 대부분의 데이터베이스 코드 객체를 대상 데이터베이스와 호환되는 형식으로 변환합니다.

DMS Schema Conversion은 온라인 트랜잭션 처리(OLTP) 데이터베이스 스키마를 Amazon RDS for MySQL 또는 RDS for PostgreSQL로 변환하는 대부분의 프로세스를 자동화합니다. 소스 및 대상 데이터베이스 엔진에는 다양한 특징과 기능이 포함되어 있으며 DMS Schema Conversion은 가능하다면 동일한 스키마를 생성하려고 합니다. 직접 변환이 불가능한 데이터베이스 객체의 경우, DMS Schema Conversion은 수행할 작업 목록을 제공합니다.

데이터베이스 스키마를 변환하려면 다음 프로세스를 사용하십시오.

- 데이터베이스 스키마를 변환하기 전에 변환 중에 데이터베이스 객체의 이름을 변경하는 변환 규칙을 설정하십시오.
- 마이그레이션의 복잡성을 추정하려면 데이터베이스 마이그레이션 평가 보고서를 생성하십시오. 이 보고서는 DMS Schema Conversion이 자동으로 변환할 수 없는 스키마 요소에 관한 세부 정보를 제공합니다.
- 소스 데이터베이스 스토리지와 코드 객체를 변환하세요. DMS Schema Conversion은 변환된 데이터베이스 객체의 로컬 버전을 생성합니다. 마이그레이션 프로젝트에서 이러한 변환된 객체에 액세스할 수 있습니다.

- 변환된 코드를 SQL 파일에 저장하여 변환 작업 항목을 검토, 편집 또는 처리할 수 있습니다. 변환된 코드를 대상 데이터베이스에 직접 적용할 수도 있습니다.

데이터 웨어하우스 스키마를 변환하려면 AWS Schema Conversion Tool 데스크톱을 사용합니다. 자세한 내용은 AWS 스키마 변환 도구 사용 설명서의 데이터 웨어하우스 스키마를 Amazon Redshift로 변환을 참조하십시오.

주제

- DMS Schema Conversion에서 변환 규칙 설정
- DMS Schema Conversion에서 데이터베이스 스키마 변환
- 마이그레이션 프로젝트에 대한 스키마 변환 설정 지정
- DMS Schema Conversion에서 데이터베이스 스키마 새로 고침
- 변환된 코드를 DMS Schema Conversion에 저장 및 적용

DMS Schema Conversion에서 변환 규칙 설정

DMS Schema Conversion으로 데이터베이스 스키마를 변환하기 전에 변환 규칙을 설정할 수 있습니다. 변환 규칙은 객체 이름을 소문자나 대문자로 변경하고 접두사나 접미사를 추가하거나 제거하며 객체 이름을 바꾸는 등의 작업을 수행할 수 있습니다. 예를 들면, 소스 스키마에 `test_TABLE_NAME`이라는 테이블 집합이 있다고 가정해 보십시오. 대상 스키마의 접두사 `test_`를 접두사 `demo_`로 변경하는 규칙을 설정할 수 있습니다.

다음 작업을 수행하는 변환 규칙을 생성할 수 있습니다.

- 접두사 추가, 제거 또는 교체
- 접미사 추가, 제거 또는 교체
- 열의 데이터 형식 변경
- 객체 이름을 소문자나 대문자로 변경
- 객체 이름 변경

다음 객체에 대한 변환 규칙을 생성할 수 있습니다.

- 스키마
- 표
- 열

변환 규칙 생성

DMS Schema Conversion은 변환 규칙을 마이그레이션 프로젝트의 일부로 저장합니다. 마이그레이션 프로젝트를 생성할 때 변환 규칙을 설정하거나 나중에 편집할 수 있습니다.

프로젝트에 여러 변환 규칙을 추가할 수 있습니다. 변환 중에 DMS Schema Conversion은 변환 규칙을 추가한 순서와 동일한 순서로 적용합니다.

변환 규칙을 생성하려면

1. 마이그레이션 프로젝트 생성 페이지에서 변환 규칙 추가를 선택합니다. 자세한 설명은 마이그레이션 프로젝트 생성 섹션을 참조하세요.
2. 규칙 대상에서 이 규칙을 적용할 데이터베이스 객체 유형을 선택합니다.
3. 소스 스키마에서 스키마 입력을 선택합니다. 그런 다음, 이 규칙이 적용되는 소스 스키마, 테이블 및 열의 이름을 입력합니다. 정확한 이름을 입력하여 하나의 객체를 선택하거나 패턴을 입력하여 여러 객체를 선택할 수 있습니다. 백분율(%)을 와일드카드로 사용하여 데이터베이스 객체 이름에 있는 여러 기호를 바꿀 수 있습니다.
4. 작업에서 수행할 작업을 선택합니다.
5. 규칙 유형에 따라 하나 또는 두 개의 추가 값을 입력합니다. 예를 들어, 객체의 이름을 바꾸려면 객체의 새 이름을 입력합니다. 접두사를 바꾸려면 이전 접두사와 새 접두사를 입력합니다.
6. 변환 규칙 추가를 선택하여 다른 변환 규칙을 추가합니다.

   규칙 추가를 완료한 후 마이그레이션 프로젝트 생성을 선택합니다.

기존 변환 규칙을 복제하려면 복제를 선택합니다. 기존 변환 규칙을 편집하려면 목록에서 규칙을 선택합니다. 기존 변환 규칙을 삭제하려면 제거를 선택합니다.

변환 규칙 편집

마이그레이션 프로젝트에서 기존 변환 규칙을 새로 추가, 제거 또는 편집할 수 있습니다. DMS Schema Conversion은 스키마 변환을 시작하는 동안 변환 규칙을 적용하므로 규칙을 편집한 후 스키마 변환을 종료하고 다시 실행해야 합니다.

변환 규칙을 편집하려면

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택한 다음, 마이그레이션 프로젝트를 선택합니다.

3. 스키마 변환을 선택한 다음, 스키마 변환 종료를 선택합니다.
4. 스키마 변환을 AWS DMS 종료한 후 수정을 선택하여 마이그레이션 프로젝트 설정을 편집합니다.
5. 변환 규칙에서 다음 작업 중 하나를 선택합니다.

   - 기존 변환 규칙을 복제하여 목록 끝에 추가하려면 복제를 선택합니다.
   - 제거를 선택하여 기존 변환 규칙을 제거합니다.
   - 기존 변환 규칙을 선택하여 편집합니다.
6. 편집 규칙을 완료한 후 변경 내용 저장을 선택합니다.
7. 마이그레이션 프로젝트 페이지의 목록에서 프로젝트를 선택합니다. 스키마 변환을 선택한 다음, 스키마 변환 시작을 선택합니다.

DMS Schema Conversion에서 데이터베이스 스키마 변환

마이그레이션 프로젝트를 생성하고 소스 및 대상 데이터베이스에 연결한 후 소스 데이터베이스 객체를 대상 데이터베이스와 호환되는 형식으로 변환할 수 있습니다. DMS Schema Conversion은 소스 데이터베이스 스키마를 왼쪽 패널에 트리 보기 형식으로 표시합니다.

데이터베이스 트리의 각 노드가 지연 로드됩니다. 트리 보기에서 노드를 선택하면 DMS Schema Conversion이 이때 소스 데이터베이스의 스키마 정보를 요청합니다. 스키마 정보를 더 빨리 로드하려면 스키마를 선택한 다음, 작업 메뉴에서 메타데이터 로드를 선택합니다. 그런 다음, DMS Schema Conversion은 데이터베이스 메타데이터를 읽고 Amazon S3 버킷에 정보를 저장합니다. 이제 데이터베이스 객체를 더 빠르게 찾아볼 수 있습니다.

전체 데이터베이스 스키마를 변환하거나 소스 데이터베이스에서 변환할 스키마 항목을 선택할 수 있습니다. 선택한 스키마 항목이 상위 항목에 따라 달라지는 경우, DMS Schema Conversion은 상위 항목에 대한 스키마도 생성합니다. 예를 들어, 변환할 테이블을 선택하면 DMS Schema Conversion은 변환된 테이블과 이 테이블이 속한 데이터베이스 스키마를 생성합니다.

데이터베이스 객체 변환

DMS Schema Conversion을 사용하여 전체 데이터베이스 스키마 또는 개별 데이터베이스 스키마 객체를 변환할 수 있습니다.

전체 데이터베이스 스키마를 변환하려면

1. AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.

3. 마이그레이션 프로젝트를 선택한 다음, 스키마 변환을 선택합니다.
4. 스키마 변환 시작을 선택합니다. 스키마 변환 페이지가 열립니다.
5. 소스 데이터베이스 창에서 스키마 이름 확인란을 선택합니다.
6. 마이그레이션 프로젝트의 왼쪽 창에서 이 스키마를 선택합니다. DMS Schema Conversion은 스키마 이름을 파란색으로 강조 표시하고 작업 메뉴를 활성화합니다.
7. 작업에서 변환을 선택합니다. 변환 대화 상자가 나타납니다.
8. 대화 상자에서 변환을 선택하여 선택을 확인합니다.

소스 데이터베이스 객체를 변환하려면

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택한 다음, 스키마 변환을 선택합니다.
4. 스키마 변환 시작을 선택합니다. 스키마 변환 페이지가 열립니다.
5. 소스 데이터베이스 창에서 소스 데이터베이스 객체를 선택합니다.
6. 변환하려는 객체의 확인란을 모두 선택한 후 왼쪽 패널에서 선택한 모든 객체의 상위 노드를 선택합니다.

   DMS Schema Conversion은 상위 노드를 파란색으로 강조 표시하고 작업 메뉴를 활성화합니다.
7. 작업에서 변환을 선택합니다. 변환 대화 상자가 나타납니다.
8. 대화 상자에서 변환을 선택하여 선택을 확인합니다.

예를 들어, 10개 테이블 중 2개를 변환하려면 변환하려는 두 테이블의 확인란을 선택합니다. 작업 메뉴가 비활성 상태인 것을 알 수 있습니다. 테이블 노드를 선택하면 DMS Schema Conversion에서 해당 이름이 파란색으로 강조 표시되고 작업 메뉴가 활성화됩니다. 그런 다음, 이 메뉴에서 변환을 선택할 수 있습니다.

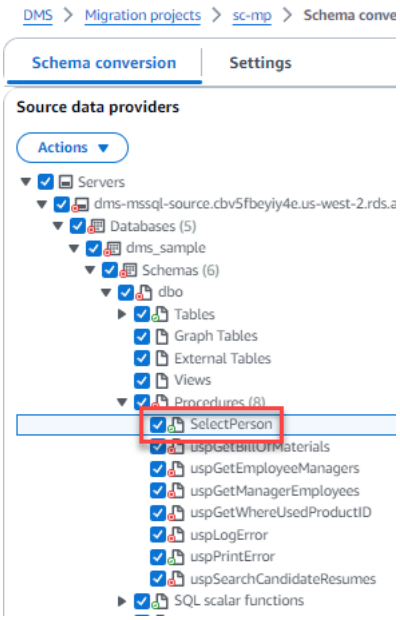
마찬가지로 2개의 테이블과 3개의 프로시저를 변환하려면 객체 이름의 확인란을 선택합니다. 그런 다음, 스키마 노드를 선택하여 작업 메뉴를 활성화하고 스키마 변환을 선택합니다.

변환된 SQL 코드 편집 및 저장

스키마 변환 페이지에서는 데이터베이스 개체의 변환된 SQL 코드를 편집할 수 있습니다. 다음 절차를 사용하여 변환된 SQL 코드를 편집하고 변경 내용을 적용한 다음 저장합니다.

변환된 SQL 코드를 편집하고, 변경 내용을 적용하고 저장하려면

1. 스키마 변환 페이지에서 소스 데이터 공급자 창의 트리 보기를 열어 코드 객체를 표시합니다.



2. 소스 데이터 공급자 창에서 작업, 변환을 선택합니다. 작업을 확인합니다.
3. 변환이 완료되면 변환된 SQL을 보려면 필요에 따라 가운데 창을 확장하십시오. 변환된 SQL을 편집하려면 대상 SQL 창에서 편집 아이콘을 선택합니다.

4. 대상 SQL을 편집한 후 페이지 상단에 있는 확인 아이콘을 선택하여 변경 사항을 확인합니다. 작업을 확인합니다.
5. 대상 데이터 공급자 창에서 작업, 변경 사항 적용을 선택합니다. 작업을 확인합니다.
6. DMS는 편집된 프로시저를 대상 데이터 저장소에 기록합니다.

변환된 데이터베이스 객체 검토

소스 데이터베이스 객체를 변환한 후 프로젝트의 왼쪽 창에서 객체를 선택할 수 있습니다. 그러면 해당 객체의 소스 및 변환된 코드를 볼 수 있습니다. DMS Schema Conversion은 왼쪽 창에서 선택한 객체의 변환된 코드를 자동으로 로드합니다. 선택한 객체의 속성이나 파라미터도 볼 수 있습니다.

DMS Schema Conversion은 변환된 코드를 마이그레이션 프로젝트의 일부로 자동 저장합니다. 대상 데이터베이스에는 이러한 코드 변경 내용이 적용되지 않습니다. 변환된 코드를 대상 데이터베이스에 적용하는 방법에 관한 자세한 내용은 변환된 코드 적용을 참조하세요. 변환된 코드를 마이그레이션 프로젝트에서 제거하려면 오른쪽 창에서 대상 스키마를 선택한 다음, 작업에서 데이터베이스에서 새로 고침을 선택합니다.

소스 데이터베이스 객체를 변환한 후에는 하단 중앙 창에서 변환 요약과 작업 항목을 볼 수 있습니다. 평가 보고서를 생성할 때 동일한 정보를 볼 수 있습니다. 평가 보고서는 DMS Schema Conversion이 변환할 수 없는 스키마 항목을 식별하고 해결하는 데 유용합니다. 평가 보고서 요약과 전환 작업 항목 목록을 CSV 파일에 저장할 수 있습니다. 자세한 설명은 데이터베이스 마이그레이션 평가 보고서 섹션을 참조하세요.

마이그레이션 프로젝트에 대한 스키마 변환 설정 지정

마이그레이션 프로젝트를 생성한 후 DMS Schema Conversion에서 변환 설정을 지정할 수 있습니다. 스키마 변환 설정을 구성하면 변환된 코드의 성능이 향상됩니다.

변환 설정을 편집하려면

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택합니다. 스키마 변환을 선택한 다음, 스키마 변환 시작을 선택합니다.
4. 설정을 선택합니다. 설정 페이지가 열립니다.
5. 변환 섹션에서 설정을 변경합니다.
6. 적용을 선택한 다음, 스키마 변환을 선택합니다.

모든 변환 쌍에 대해 변환된 코드에서 작업 항목이 있는 설명 수를 제한할 수 있습니다. 변환된 코드의 설명 수를 제한하려면 마이그레이션 프로젝트에서 변환 설정을 여십시오.

변환된 SQL 코드의 설명에서 작업 항목의 심각도 수준을 선택합니다. DMS Schema Conversion은 선택한 수준 또는 그보다 높은 수준의 심각도에 해당하는 작업 항목에 대해 변환된 코드에 설명을 추가합니다. 예를 들어, 변환된 코드의 설명 수를 최소화하려면 오류만을 선택하십시오.

변환된 코드의 모든 작업 항목에 대한 설명을 포함하려면 모든 메시지를 선택합니다.

기타 변환 설정은 소스 및 대상 데이터베이스 쌍마다 다릅니다.

주제

- Oracle에서 MySQL로의 변환 설정
- Oracle에서 PostgreSQL로 변환 설정
- SQL Server에서 MySQL로의 변환 설정
- SQL Server에서 PostgreSQL로 변환 설정
- PostgreSQL에서 MySQL로의 변환 설정

Oracle에서 MySQL로의 변환 설정

DMS Schema Conversion의 경우, Oracle에서 MySQL로의 변환 설정에는 다음이 포함됩니다.

- 소스 Oracle 데이터베이스는 `ROWID` 의사 열을 사용할 수 있습니다. MySQL에서는 유사한 기능을 지원하지 않습니다. DMS Schema Conversion은 변환된 코드의 `ROWID` 의사 열을 에뮬레이션할 수 있습니다. 이렇게 하려면 행 ID 생성 옵션을 활성화하십시오.

  소스 Oracle 코드에서 `ROWID` 의사 열을 사용하지 않는 경우, 행 ID 생성 옵션을 비활성화하십시오. 이 경우, 변환된 코드는 더 빠르게 작동합니다.
- 소스 Oracle 코드에는 MySQL이 지원하지 않는 파라미터가 있는 `TO_CHAR`, `TO_DATE`, `TO_NUMBER` 함수가 포함될 수 있습니다. 기본적으로 DMS Schema Conversion은 변환된 코드에서 이러한 파라미터의 사용을 에뮬레이션합니다.

  소스 Oracle 코드에 MySQL에서 지원되지 않는 파라미터가 없는 경우, 네이티브 MySQL `TO_CHAR`, `TO_DATE` 및 `TO_NUMBER` 함수를 사용할 수 있습니다. 이 경우, 변환된 코드는 더 빠르게 작동합니다. 이렇게 하려면 다음 값을 선택합니다.
  - 기본 MySQL TO_CHAR 함수 사용
  - 기본 MySQL TO_DATE 함수 사용
  - 기본 MySQL TO_NUMBER 함수 사용
- 데이터베이스와 애플리케이션은 서로 다른 시간대에 실행할 수 있습니다. 기본적으로 DMS Schema Conversion은 변환된 코드로 시간대를 에뮬레이션합니다. 하지만 데이터베이스와 애플리케이션이

동일한 시간대를 사용하는 경우에는 이 에뮬레이션이 필요하지 않습니다. 이 경우, 데이터베이스와 애플리케이션이 동일한 시간대를 사용하는 경우 변환된 코드의 성능 개선을 선택합니다.

Oracle에서 PostgreSQL로 변환 설정

DMS Schema Conversion의 경우, Oracle에서 PostgreSQL로의 변환 설정에는 다음이 포함됩니다.

- AWS DMS Oracle 구체화된 뷰를 PostgreSQL에서 테이블 또는 구체화된 뷰로 변환할 수 있습니다. 구체화된 뷰의 경우, 소스 구체화된 뷰를 변환하는 방법을 선택합니다.
- 소스 Oracle 데이터베이스는 `ROWID` 의사 열을 사용할 수 있습니다. PostgreSQL에서는 유사한 기능을 지원하지 않습니다. DMS Schema Conversion은 `bigint` 또는 `character varying` 데이터 유형을 사용하여 변환된 코드로 `ROWID` 의사 열을 에뮬레이션할 수 있습니다. 이렇게 하려면 bigint 데이터 유형 사용을 선택하여 ROWID 의사 열을 에뮬레이션을 선택하거나 행 ID에 대해 문자 가변 데이터 유형을 사용하여 ROWID 의사 열을 에뮬레이션을 선택하십시오.

  소스 Oracle 코드에서 `ROWID` 의사 열을 사용하지 않는 경우, 생성 안 함을 선택합니다. 이 경우, 변환된 코드는 더 빠르게 작동합니다.
- 소스 Oracle 코드에는 PostgreSQL에서 지원하지 않는 파라미터가 있는 `TO_CHAR`, `TO_DATE`, `TO_NUMBER` 함수가 포함될 수 있습니다. 기본적으로 DMS Schema Conversion은 이러한 파라미터의 사용을 변환된 코드로 에뮬레이션합니다.

  PostgreSQL에서 지원되지 않는 파라미터가 소스 Oracle 코드에 없는 경우, 기본 PostgreSQL `TO_CHAR`, `TO_DATE` 및 `TO_NUMBER` 함수를 사용할 수 있습니다. 이 경우, 변환된 코드는 더 빠르게 작동합니다. 이렇게 하려면 다음 값을 선택합니다.
  - 기본 PostgreSQL TO_CHAR 함수 사용
  - 기본 PostgreSQL TO_DATE 함수 사용
  - 기본 PostgreSQL TO_NUMBER 함수 사용
- 데이터베이스와 애플리케이션은 서로 다른 시간대에 실행할 수 있습니다. 기본적으로 DMS Schema Conversion은 변환된 코드로 시간대를 에뮬레이션합니다. 하지만 데이터베이스와 애플리케이션이 동일한 시간대를 사용하는 경우에는 이 에뮬레이션이 필요하지 않습니다. 이 경우, 데이터베이스와 애플리케이션이 동일한 시간대를 사용하는 경우 변환된 코드의 성능 개선을 선택합니다.
- 변환된 코드의 시퀀스를 계속 사용하려면 변환된 시퀀스를 소스 측에서 생성된 마지막 값으로 채우기를 선택합니다.
- 경우에 따라 차이는 있겠지만 소스 Oracle 데이터베이스는 `NUMBER` 데이터 유형의 기본 키 또는 외래 키 열에 정수 값만 저장할 수 있습니다. 이러한 경우 이러한 열을 데이터 유형으로 AWS DMS 변환할 수 있습니다. `BIGINT` 이 방식을 적용하면 변환된 코드의 성능이 향상됩니다. 이렇게 하려면

NUMBER 데이터 유형의 기본 키 및 외래 키 열을 BIGINT 데이터 유형으로 변환을 선택합니다. 데이터 손실을 방지하려면 소스에서 이러한 열에 부동 소수점 값이 포함되지 않도록 해야 합니다.

- 소스 코드에서 비활성화된 트리거와 제약 조건을 건너뛰려면 활성 트리거 및 제약 조건만 변환을 선택합니다.
- DMS Schema Conversion을 사용하여 동적 SQL이라고 하는 문자열 변수를 변환할 수 있습니다. 데이터베이스 코드는 이러한 문자열 변수의 값을 변경할 수 있습니다. 이 문자열 변수의 최신 값을 AWS DMS 항상 변환하도록 하려면 호출 루틴에서 만든 동적 SQL 코드 변환을 선택합니다.
- PostgreSQL 버전 10 및 이전 버전은 프로시저를 지원하지 않습니다. PostgreSQL에서 프로시저를 사용하는 데 익숙하지 않은 경우 Oracle 프로시저를 PostgreSQL AWS DMS 함수로 변환할 수 있습니다. 이렇게 하려면 프로시저를 함수로 변환을 선택합니다.
- 확장 팩에 특정 함수를 추가하면 발생한 작업 항목에 관한 추가 정보를 볼 수 있습니다. 이렇게 하려면 사용자 정의 예외를 발생시키는 확장 팩 함수 추가를 선택합니다. 그런 다음, 사용자 정의 예외가 발생하도록 심각도 수준을 선택합니다. 소스 데이터베이스 객체를 변환한 후에는 확장 팩 스키마를 적용해야 합니다. 확장 팩에 관한 자세한 내용은 확장 팩 사용을 참조하세요.
- 소스 Oracle 데이터베이스에는 자동으로 생성된 이름과 함께 제약 조건이 포함될 수 있습니다. 소스 코드에서 이러한 이름을 사용하는 경우, 시스템 생성 제약 조건의 이름 유지를 선택해야 합니다. 소스 코드에서 이러한 제약 조건을 사용하지만 이름을 사용하지 않는 경우 이 옵션을 선택 해제하여 변환 속도를 높이십시오.
- 소스 및 대상 데이터베이스가 서로 다른 시간대에서 실행되는 경우, SYSDATE 내장 Oracle 함수를 에뮬레이션하는 함수는 소스 함수와 비교해 다른 값을 반환합니다. 소스 함수와 대상 함수가 동일한 값을 반환하도록 하려면 소스 데이터베이스의 시간대 설정을 선택합니다.
- orafce 확장의 함수를 변환된 코드에 사용할 수 있습니다. 이렇게 하려면 Oracle 내장 루틴에서 사용할 함수를 선택하십시오. 오라클에 대한 자세한 내용은 orace on을 참조하십시오. GitHub

SQL Server에서 MySQL로의 변환 설정

DMS Schema Conversion의 경우, SQL Server에서 MySQL로의 변환 설정에는 다음이 포함됩니다.

- 소스 SQL Server 데이터베이스는 EXEC의 출력을 테이블에 저장할 수 있습니다. DMS Schema Conversion은 이 기능을 에뮬레이션하기 위한 임시 테이블과 추가 프로시저를 생성합니다. 이 에뮬레이션을 사용하려면 열린 데이터 세트를 처리할 추가 루틴 생성을 선택합니다.

SQL Server에서 PostgreSQL로 변환 설정

DMS Schema Conversion의 경우, SQL Server에서 PostgreSQL로의 변환 설정에는 다음이 포함됩니다.

- SQL Server에서는 서로 다른 테이블에서 같은 이름의 인덱스를 사용할 수 있습니다. 하지만 PostgreSQL에서는 스키마에서 사용하는 모든 인덱스 이름이 고유해야 합니다. DMS Schema Conversion이 모든 인덱스에 대해 고유한 이름을 생성하도록 하려면 인덱스에 고유한 이름 생성을 선택합니다.
- PostgreSQL 버전 10 및 이전 버전은 프로시저를 지원하지 않습니다. PostgreSQL에서 프로시저를 사용하는 데 익숙하지 않은 경우 SQL Server 프로시저를 PostgreSQL AWS DMS 함수로 변환할 수 있습니다. 이렇게 하려면 프로시저를 함수로 변환을 선택합니다.
- 소스 SQL Server 데이터베이스는 EXEC의 출력을 테이블에 저장할 수 있습니다. DMS Schema Conversion은 이 기능을 에뮬레이션하기 위한 임시 테이블과 추가 프로시저를 생성합니다. 이 에뮬레이션을 사용하려면 열린 데이터 세트를 처리할 추가 루틴 생성을 선택합니다.
- 변환된 코드의 스키마 이름에 사용할 템플릿을 정의할 수 있습니다. 스키마 이름에서 다음 옵션 중 하나를 선택합니다.
  - DB – PostgreSQL에서 SQL Server 데이터베이스 이름을 스키마 이름으로 사용합니다.
  - SCHEMA – PostgreSQL에서 SQL Server 스키마 이름을 스키마 이름으로 사용합니다.
  - DB_SCHEMA – PostgreSQL에서 SQL Server 데이터베이스와 스키마 이름의 조합을 스키마 이름으로 사용합니다.
- 소스 객체 이름의 대소문자를 그대로 사용할 수 있습니다. 객체 이름을 소문자로 변환하지 않으려면 객체 이름을 대/소문자로 유지를 선택합니다. 이 옵션은 대상 데이터베이스에서 대소문자 구분 옵션을 활성화한 경우에만 적용됩니다.
- 소스 데이터베이스의 파라미터 이름을 유지할 수 있습니다. DMS Schema Conversion은 변환된 코드의 파라미터 이름에 큰따옴표를 추가할 수 있습니다. 이렇게 하려면 원래 파라미터 이름 유지를 선택합니다.
- 소스 데이터베이스에서 루틴 파라미터 길이를 유지할 수 있습니다. DMS Schema Conversion은 도메인을 생성하고 이를 사용하여 루틴 파라미터 길이를 지정합니다. 이렇게 하려면 파라미터 길이 유지를 선택합니다.

PostgreSQL에서 MySQL로의 변환 설정

DMS 스키마 변환의 PostgreSQL에서 MySQL로의 변환 설정에는 다음이 포함됩니다.

- 변환된 SQL 코드의 주석: 이 설정에는 선택한 심각도 이상의 작업 항목에 대한 변환된 코드의 주석이 포함됩니다. 이 설정은 다음과 같은 값을 지원합니다.
  - 오류만
  - Errors and warnings(오류 및 경고)
  - 모든 메시지

DMS Schema Conversion에서 데이터베이스 스키마 새로 고침

마이그레이션 프로젝트를 생성한 후 DMS Schema Conversion은 소스 및 대상 스키마에 관한 정보를 이 프로젝트에 저장합니다. DMS Schema Conversion은 데이터베이스 트리에서 노드를 선택하는 경우와 같이 필요한 경우에만 지연 로딩을 사용하여 메타데이터를 로드합니다. 즉시 로딩을 사용하면 스키마 정보를 더 빨리 로드할 수 있습니다. 이렇게 하려면 스키마를 선택한 다음, 작업에서 메타데이터 로드를 선택합니다.

객체를 마이그레이션 프로젝트에 자동 또는 수동으로 로드한 후에는 DMS Schema Conversion에서 지연 로딩을 다시 사용하지 않습니다. 따라서 데이터베이스의 테이블 및 프로시저와 같은 객체를 변경할 때는 마이그레이션 프로젝트에서 해당 객체에 대해 새로 고침을 실행해야 합니다.

데이터베이스로부터 스키마를 새로 고치려면 새로 고침을 실행할 해당 객체를 선택한 후 작업에서 데이터베이스에서 새로 고침을 선택합니다. 소스 및 대상 데이터베이스 스키마의 데이터베이스 객체는 다음과 같이 새로 고칠 수 있습니다.

- 소스 – 소스 데이터베이스 스키마를 업데이트하는 경우, 데이터베이스에서 새로 고침을 선택하여 프로젝트의 스키마를 소스 데이터베이스의 최신 스키마로 바꾸십시오.
- 대상 – 대상 데이터베이스의 스키마를 업데이트하면 DMS Schema Conversion이 프로젝트의 스키마를 대상 데이터베이스의 최신 스키마로 바꿉니다. DMS Schema Conversion은 변환된 코드를 대상 데이터베이스의 코드로 바꿉니다. 데이터베이스에서 새로 고침을 선택하기 전에 변환된 코드를 대상 데이터베이스에 적용했는지 확인하십시오. 그렇지 않으면 소스 데이터베이스 스키마를 다시 변환하세요.

변환된 코드를 DMS Schema Conversion에 저장 및 적용

DMS Schema Conversion은 소스 데이터베이스 객체를 변환한 후 변환된 코드를 대상 데이터베이스에 즉시 적용하지 않습니다. 그 대신 DMS Schema Conversion은 대상 데이터베이스에 적용할 준비가 될 때까지 변환된 코드를 프로젝트에 저장합니다.

변환된 코드를 적용하기 전에 소스 데이터베이스 코드를 업데이트하고 업데이트된 객체를 다시 변환하여 기존 작업 항목을 처리할 수 있습니다. DMS Schema Conversion에서 자동으로 변환할 수 없는 항목에 관한 자세한 내용은 DMS Schema Conversion을 사용하여 데이터베이스 마이그레이션 평가 보고서 생성를 참조하십시오. DMS Schema Conversion에 대한 마이그레이션 프로젝트의 소스 데이터베이스 객체를 새로 고치는 방법에 관한 자세한 내용은 데이터베이스 스키마 새로 고침를 참조하십시오.

변환된 코드를 DMS Schema Conversion의 데이터베이스에 직접 적용하는 대신, 이 코드를 SQL 스크립트로 파일에 저장할 수 있습니다. 이러한 SQL 스크립트를 검토하고 필요하다면 편집한 후 대상 데이터베이스에 해당 SQL 스크립트를 수동으로 적용할 수 있습니다.

변환된 코드를 SQL 파일에 저장

변환된 스키마를 텍스트 파일에 SQL 스크립트로 저장할 수 있습니다. 변환된 코드를 수정하여 DMS Schema Conversion에서 자동으로 변환할 수 없는 작업 항목을 처리할 수 있습니다. 그런 다음, 대상 데이터베이스에서 업데이트된 SQL 스크립트를 실행하여 변환된 코드를 대상 데이터베이스에 적용할 수 있습니다.

변환된 스키마를 SQL 스크립트로 저장하려면

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택한 다음, 스키마 변환을 선택합니다.
4. 스키마 변환 시작을 선택합니다. 스키마 변환 페이지가 열립니다.
5. 오른쪽 창에서 대상 데이터베이스 스키마를 선택하거나 저장하려는 변환된 객체를 선택합니다. DMS Schema Conversion이 상위 노드 이름을 파란색으로 강조 표시하고 대상 데이터베이스의 작업 메뉴를 활성화하는지 확인하십시오.
6. 작업에서 SQL로 저장을 선택합니다. 저장 대화 상자가 나타납니다.
7. SQL로 저장을 선택해 선택을 확인합니다.

   DMS Schema Conversion은 SQL 파일로 아카이브를 생성하고 이 아카이브를 Amazon S3 버킷에 저장합니다.
8. (선택 사항) 인스턴스 프로파일에서 스키마 변환 설정을 편집하여 아카이브의 S3 버킷을 변경합니다.
9. S3 버킷에서 SQL 스크립트를 엽니다.

변환된 코드 적용

변환된 코드를 대상 데이터베이스에 적용할 준비가 되면 프로젝트의 오른쪽 창에서 데이터베이스 객체를 선택합니다. 전체 데이터베이스 스키마 또는 선택한 데이터베이스 스키마 객체에 변경 내용을 적용할 수 있습니다.

데이터베이스 객체를 선택하면 DMS Schema Conversion에서는 선택한 노드 또는 상위 노드의 이름이 파란색으로 강조 표시됩니다. 그러면 작업 메뉴가 활성화됩니다. 작업에서 변경 사항 적용을 선택합니다. 화면에 표시되는 대화 상자에서 적용을 선택하여 선택을 확인하고 변환된 코드를 대상 데이터베이스에 적용합니다.

확장 팩 스키마 적용

변환된 스키마를 대상 데이터베이스에 맨 처음 적용하는 경우 DMS Schema Conversion에서 확장 팩 스키마도 적용할 수 있습니다. 확장 팩 스키마는 대상 데이터베이스의 변환된 코드를 실행하는 데 필요한 소스 데이터베이스의 시스템 함수를 에뮬레이션합니다. 변환된 코드에서 확장 팩의 함수를 사용하는 경우, 확장 팩 스키마를 적용해야 합니다.

확장 팩을 대상 데이터베이스에 수동으로 적용하려면 작업에 대해 변경 사항 적용을 선택합니다. 화면에 표시되는 대화 상자에서 확인을 선택하여 대상 데이터베이스에 확장 팩을 적용합니다.

변환된 코드에서 예상치 못한 결과가 발생하지 않도록 확장 팩 스키마를 수정하지 않는 것이 좋습니다.

자세한 내용은 DMS Schema Conversion에서 확장 팩 사용을(를) 참조하세요.

DMS Schema Conversion에서 확장 팩 사용

DMS Schema Conversion의 확장 팩은 대상 데이터베이스에서 지원되지 않는 소스 데이터베이스 함수를 에뮬레이션하는 추가 모듈입니다. 확장 팩을 사용하여 변환된 코드가 소스 코드와 동일한 결과를 생성하는지 확인하십시오. 확장 팩을 설치하기 전에 데이터베이스 스키마를 변환하세요.

확장 팩마다 데이터베이스 스키마가 포함되어 있습니다. 이 스키마에는 소스 데이터베이스를 출처로 한 특정 온라인 트랜잭션 처리(OLTP) 객체 또는 지원되지 않는 내장 함수를 에뮬레이션하기 위한 SQL 함수, 프로시저, 테이블 및 뷰가 포함됩니다.

소스 데이터베이스를 변환할 때 DMS Schema Conversion은 대상 데이터베이스에 추가 스키마를 추가합니다. 이 스키마는 대상 데이터베이스에서 변환된 코드를 실행하는 데 필요한 소스 데이터베이스의 SQL 시스템 함수를 구현합니다. 이 추가 스키마를 일컬어 확장 팩 스키마라고 합니다.

확장 팩 스키마는 소스 데이터베이스에 따라 다음과 같이 이름이 지정됩니다.

- Microsoft SQL Server - `aws_sqlserver_ext`
- Oracle – `aws_oracle_ext`

다음 두 가지 방법으로 확장 팩을 적용할 수 있습니다.

- DMS Schema Conversion은 변환된 코드를 적용할 때 확장 팩을 자동으로 적용할 수 있습니다. DMS Schema Conversion은 다른 모든 스키마 객체를 적용하기 전에 확장 팩을 적용합니다.
- 확장 팩을 수동으로 적용할 수 있습니다. 이렇게 하려면 대상 데이터베이스 트리에서 확장 팩 스키마를 선택한 후 적용, 확장 팩 적용을 차례로 선택합니다.

AWS DMS를 사용하여 데이터베이스를 동등한 Amazon RDS 데이터베이스로 마이그레이션

AWS Database Migration Service(AWS DMS)의 동종 데이터 마이그레이션은 자체 관리형 온프레미스 데이터베이스를 동등한 Amazon Relational Database Service(RDS) 데이터베이스로 마이그레이션하는 과정을 단순화합니다. 예를 들어, 동종 데이터 마이그레이션을 사용하여 온프레미스 PostgreSQL 데이터베이스를 Amazon RDS for PostgreSQL 또는 Aurora PostgreSQL로 마이그레이션할 수 있습니다. 동종 데이터 마이그레이션의 경우 AWS DMS가 기본 데이터베이스 도구를 사용하여 간편하고 성능이 우수한 동종 마이그레이션을 제공합니다.

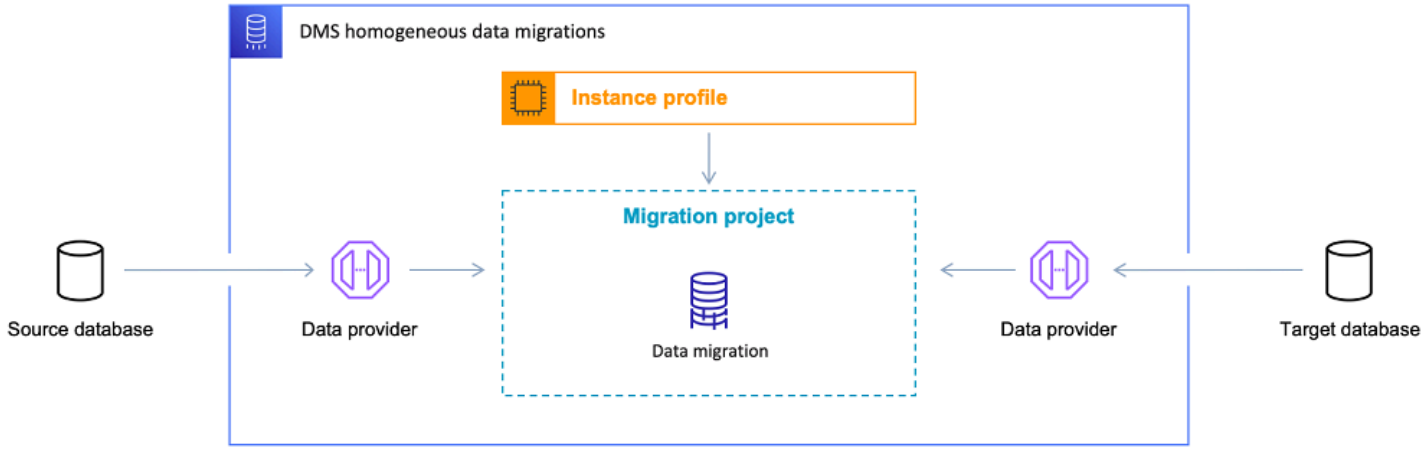
동종 데이터 마이그레이션은 서버리스입니다. 즉, AWS DMS가 마이그레이션에 필요한 리소스를 자동으로 확장합니다. 동종 데이터 마이그레이션을 사용하면 데이터, 테이블 파티션, 데이터 형식 및 보조 객체(예: 함수, 저장 프로시저 등)를 마이그레이션할 수 있습니다.

상위 수준에서, 동종 데이터 마이그레이션은 인스턴스 프로파일, 데이터 공급자 및 마이그레이션 프로젝트를 사용하여 작동합니다. 동일한 유형의 호환 가능한 소스 및 대상 데이터 공급자를 사용하여 마이그레이션 프로젝트를 생성하면 AWS DMS가 데이터 마이그레이션이 실행되는 서버리스 환경을 배포합니다. 그런 다음 AWS DMS는 기본 데이터베이스 도구를 사용하여 소스 데이터 공급자에 연결하고, 소스 데이터를 읽고, 파일을 디스크에 덤프하고, 데이터를 복원합니다. 인스턴스 프로파일, 데이터 공급자, 마이그레이션 프로젝트에 대한 자세한 내용은 의 데이터 제공업체, 인스턴스 프로필 및 마이그레이션 프로젝트와 협력 AWS DMS 섹션을 참조하세요.

지원되는 소스 데이터베이스의 목록은 DMS 동종 데이터 마이그레이션이 지원하는 소스 섹션을 참조하세요.

지원되는 대상 데이터베이스의 목록은 DMS 동종 데이터 마이그레이션이 지원하는 대상 섹션을 참조하세요.

다음 다이어그램은 동종 데이터 마이그레이션의 작동 방식을 설명합니다.



다음 단원에서는 동종 데이터 마이그레이션 사용에 대한 정보를 제공합니다.

주제

- 지원되는 AWS 리전
- 특성
- 동종 데이터 마이그레이션 제한 사항
- AWS DMS의 동종 데이터 마이그레이션 프로세스 개요
- AWS DMS의 동종 데이터 마이그레이션 설정
- 에서 동종 데이터 마이그레이션을 위한 소스 데이터 공급자 생성 AWS DMS
- AWS DMS에서 동종 데이터 마이그레이션을 위한 대상 데이터 공급자 생성
- AWS DMS에서 동종 데이터 마이그레이션 실행
- AWS DMS에서 동종 데이터 마이그레이션 문제 해결

지원되는 AWS 리전

다음 AWS 리전에서 동종 데이터 마이그레이션을 실행할 수 있습니다.

리전 이름	리전
미국 동부(버지니아 북부)	us-east-1
미국 동부(오하이오)	us-east-2
미국 서부(오레곤)	us-west-2

리전 이름	리전
아시아 태평양(도쿄)	ap-northeast-1
아시아 태평양(싱가포르)	ap-southeast-1
아시아 태평양(시드니)	ap-southeast-2
유럽(프랑크푸르트)	eu-central-1
유럽(스톡홀름)	eu-north-1
유럽(아일랜드)	eu-west-1

특성

동종 데이터 마이그레이션은 다음 기능을 제공합니다.

- AWS DMS는 AWS 클라우드에서 동종 데이터 마이그레이션에 필요한 컴퓨팅 및 스토리지 리소스를 자동으로 관리합니다. AWS DMS는 데이터 마이그레이션을 시작할 때 이러한 리소스를 서버리스 환경에 배포합니다.
- AWS DMS는 기본 데이터베이스 도구를 사용하여 동일한 유형의 데이터베이스 간에 완전 자동화된 마이그레이션을 시작합니다.
- 동종 데이터 마이그레이션을 사용하여 데이터는 물론 파티션, 함수, 저장 프로시저 등과 같은 보조 객체도 마이그레이션할 수 있습니다.
- 동종 데이터 마이그레이션은 전체 로드, 지속적 복제 및 지속적 복제 포함 전체 로드의 세 가지 마이그레이션 모드에서 실행할 수 있습니다.
- 동종 데이터 마이그레이션에는 온프레미스, Amazon EC2, Amazon RDS 데이터베이스를 소스로 사용할 수 있습니다. Amazon RDS 또는 Amazon Aurora를 동종 데이터 마이그레이션의 마이그레이션 대상으로 선택할 수 있습니다.

동종 데이터 마이그레이션 제한 사항

동종 데이터 마이그레이션을 사용할 경우 다음 제한 사항이 적용됩니다.

- 동종 데이터 마이그레이션은 선택 규칙을 지원하지 않습니다. 또한 변환 규칙을 사용하여 열의 데이터 형식을 변경하거나, 한 스키마에서 다른 스키마로 객체를 이동하거나, 객체 이름을 변경할 수 없습니다.
- 동종 데이터 마이그레이션에는 데이터 유효성 검사를 위한 기본 제공 도구가 없습니다.
- PostgreSQL에서 동종 데이터 마이그레이션을 사용하는 경우 AWS DMS는 뷰를 테이블로 대상 데이터베이스에 마이그레이션합니다.
- 동종 데이터 마이그레이션은 지속적 데이터 복제 중에 스키마 수준의 변경 사항을 캡처하지 않습니다. 소스 데이터베이스에서 새 테이블을 생성하는 경우 AWS DMS는 이 테이블을 마이그레이션할 수 없습니다. 이 새 테이블을 마이그레이션하려면 데이터 마이그레이션을 다시 시작해야 합니다.
- AWS DMS에서 동종 데이터 마이그레이션을 사용하여 상위 데이터베이스 버전에서 하위 데이터베이스 버전으로 데이터를 마이그레이션할 수 없습니다.
- CLI 또는 API에서는 동종 데이터 마이그레이션을 사용할 수 없습니다.
- 동종 데이터 마이그레이션은 VPC 보조 CIDR 범위의 데이터베이스 인스턴스에 대한 연결 설정을 지원하지 않습니다.
- 데이터 공급자의 동종 마이그레이션에는 8081 포트를 사용할 수 없습니다.

AWS DMS의 동종 데이터 마이그레이션 프로세스 개요

AWS DMS에서 동종 데이터 마이그레이션을 사용하여 동일한 유형의 두 데이터베이스 간에 데이터를 마이그레이션할 수 있습니다. 다음 워크플로를 사용하여 데이터 마이그레이션을 생성하고 실행할 수 있습니다.

1. 필요한 AWS Identity and Access Management(IAM) 정책 및 역할을 생성합니다. 자세한 내용은 IAM 리소스 생성 섹션을 참조하세요.
2. 소스 및 대상 데이터베이스를 구성하고 AWS DMS에서 동종 데이터 마이그레이션에 필요한 최소 권한을 가진 데이터베이스 사용자를 생성하십시오. 자세한 정보는 소스 데이터 공급자 생성 및 대상 데이터 공급자 생성 섹션을 참조하세요.
3. 소스 및 대상 데이터베이스 자격 증명을 AWS Secrets Manager에 저장하십시오. 자세한 정보는 AWS Secrets Manager 사용 설명서의 1단계: 보안 암호 생성을 참조하세요.
4. AWS DMS 콘솔에서 서브넷 그룹, 인스턴스 프로파일, 데이터 공급자를 생성합니다. 자세한 내용은 서브넷 그룹 생성, 인스턴스 프로파일 생성, 데이터 공급자 생성 섹션을 참조하세요.
5. 이전 단계에서 생성한 리소스를 사용하여 마이그레이션 프로젝트를 생성합니다. 자세한 내용은 마이그레이션 프로젝트 생성 섹션을 참조하세요.

6. 데이터 마이그레이션을 생성, 구성 및 시작합니다. 자세한 내용은 데이터 마이그레이션 생성 섹션을 참조하세요.
7. 전체 로드 또는 지속적인 복제를 완료한 후에는 전환하여 새 대상 데이터베이스를 사용할 수 있습니다.
8. 리소스를 정리합니다. Amazon은 마이그레이션 완료 후 3일 이내에 마이그레이션 프로젝트의 데이터 마이그레이션을 종료합니다. 하지만 인스턴스 프로파일, 데이터 공급자, IAM 정책 및 역할, AWS Secrets Manager의 보안 정보와 같은 리소스를 수동으로 삭제해야 합니다.

AWS DMS의 동종 데이터 마이그레이션에 관한 자세한 내용은 PostgreSQL에서 Amazon RDS for PostgreSQL로의 단계별 마이그레이션 안내를 참조하십시오.

이 동영상은 AWS DMS의 동종 데이터 마이그레이션을 소개하고 있으며 이 기능을 익히는 데 도움이 됩니다.

AWS DMS의 동종 데이터 마이그레이션 설정

AWS DMS에서 동종 데이터 마이그레이션을 설정하려면 다음 사전 요구 사항 작업을 완료하십시오.

주제

- AWS DMS에서 동종 데이터 마이그레이션을 위한 필수 IAM 리소스 생성
- AWS DMS의 동종 데이터 마이그레이션을 위한 네트워크 설정

AWS DMS에서 동종 데이터 마이그레이션을 위한 필수 IAM 리소스 생성

동종 데이터 마이그레이션을 실행하려면 계정에 IAM 정책과 IAM 역할을 생성하여 다른 AWS 서비스와 상호 작용해야 합니다. 이 섹션에서는 이러한 필수 IAM 리소스를 생성합니다.

주제

- AWS DMS에서 동종 데이터 마이그레이션을 위한 IAM 정책 생성
- AWS DMS에서 동종 데이터 마이그레이션을 위한 IAM 역할 생성

AWS DMS에서 동종 데이터 마이그레이션을 위한 IAM 정책 생성

데이터베이스에 액세스하고 데이터를 마이그레이션하기 위해 AWS DMS는 동종 데이터 마이그레이션을 위한 서버리스 환경을 생성합니다. 이 환경에서 AWS DMS는 VPC 피어링, 라우팅 테이블, 보안 그룹 및 기타 AWS 리소스에 대한 액세스가 필요합니다. 또한 AWS DMS는 Amazon CloudWatch에 각

데이터 마이그레이션에 대한 로그, 지표 및 진행 상황을 저장합니다. 데이터 마이그레이션 프로젝트를 생성하려면 AWS DMS는 이러한 서비스에 대한 액세스 권한이 필요합니다.

이 단계에서는 Amazon EC2 및 CloudWatch 리소스에 대한 액세스 권한을 AWS DMS에 부여하는 IAM 정책을 생성합니다. 그런 다음, IAM 역할을 생성하고 이 정책을 연결합니다.

AWS DMS에서 동종 데이터 마이그레이션을 위한 IAM 정책을 생성하려면

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 생성 페이지에서 JSON 탭을 선택합니다.
5. 다음 JSON을 편집기에 붙여넣습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeRouteTables",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeVpcPeeringConnections",
                "ec2:DescribeVpcs",
                "ec2:DescribePrefixLists",
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "servicequotas:GetServiceQuota"
            ],
            "Resource": "arn:aws:servicequotas:*:*:vpc/L-0EA8095F"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
```

```
                "logs:DescribeLogStreams"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:dms-data-migration-*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:dms-data-migration-*:log-
stream:dms-data-migration-*"
        },
        {
            "Effect": "Allow",
            "Action": "cloudwatch:PutMetricData",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateRoute",
                "ec2:DeleteRoute"
            ],
            "Resource": "arn:aws:ec2:*:*:route-table/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:security-group-rule/*",
                "arn:aws:ec2:*:*:route-table/*",
                "arn:aws:ec2:*:*:vpc-peering-connection/*",
                "arn:aws:ec2:*:*:vpc/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress"
```

```
            ],
            "Resource": "arn:aws:ec2:*:*:security-group-rule/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:RevokeSecurityGroupEgress",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": "arn:aws:ec2:*:*:security-group/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AcceptVpcPeeringConnection",
                "ec2:ModifyVpcPeeringConnectionOptions"
            ],
            "Resource": "arn:aws:ec2:*:*:vpc-peering-connection/*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:AcceptVpcPeeringConnection",
            "Resource": "arn:aws:ec2:*:*:vpc/*"
        }
    ]
}
```

6. 다음: 태그와 다음: 검토를 선택합니다.
7. 이름*의 경우, **HomogeneousDataMigrationsPolicy**를 입력한 후 정책 생성을 선택합니다.

AWS DMS에서 동종 데이터 마이그레이션을 위한 IAM 역할 생성

이 단계에서는 AWS Secrets Manager, Amazon EC2 및 CloudWatch에 대한 액세스 권한을 AWS DMS에 부여하는 IAM 역할을 생성합니다.

AWS DMS에서 동종 데이터 마이그레이션을 위한 IAM 역할을 생성하려면

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.

3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 선택 페이지의 신뢰할 수 있는 엔터티 유형 아래에서 AWS 서비스를 선택합니다. 기타 AWS 서비스 사용 사례에서 DMS를 선택합니다.
5. DMS 확인란을 선택하고 다음을 선택합니다.
6. 권한 추가 페이지에서 이전에 만든 HomogeneousDataMigrationsPolicy를 선택합니다. 또한 SecretsManagerReadWrite를 선택합니다. 다음을 선택합니다.
7. 이름 지정, 검토 및 생성 페이지에서 역할 이름에 **HomogeneousDataMigrationsRole**을 입력하고 역할 생성을 선택합니다.
8. 역할 페이지에서 역할 이름으로 **HomogeneousDataMigrationsRole**을 입력합니다. HomogeneousDataMigrationsRole을 선택합니다.
9. HomogeneousDataMigrationsRole 페이지에서 신뢰 관계 탭을 선택합니다. 신뢰 정책 편집을 선택합니다.
10. 신뢰 정책 편집 페이지에서 기존 텍스트를 대체하여 다음 JSON을 편집기에 붙여넣습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "dms-data-migrations.amazonaws.com",
                    "dms.your_region.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

이전 예제에서 *your_region*을 AWS 리전의 이름으로 바꾸십시오.

위의 리소스 기반 정책은 AWS 관리형 SecretsManagerReadWrite 및 고객 관리형 HomogeneousDataMigrationsPolicy 정책에 따라 AWS DMS 작업을 수행할 수 있는 권한을 서비스 보안 주체에게 부여합니다.

11. 정책 업데이트를 선택합니다.

AWS DMS의 동종 데이터 마이그레이션을 위한 네트워크 설정

AWS DMS는 Amazon VPC 서비스를 기반으로 Virtual Private Cloud(VPC)에서 동종 데이터 마이그레이션을 위한 서버리스 환경을 생성합니다. 인스턴스 프로파일을 생성할 때 사용할 VPC를 지정합니다. 계정과 AWS 리전에 기본 VPC를 사용하거나 새 VPC를 생성할 수도 있습니다.

각 데이터 마이그레이션에 대해 AWS DMS는 인스턴스 프로파일에 사용하는 VPC와 VPC 피어링 연결을 설정합니다. 그런 다음, AWS DMS는 인스턴스 프로파일과 연결된 보안 그룹에 CIDR 블록을 추가합니다. AWS DMS는 퍼블릭 IP 주소를 인스턴스 프로파일에 연결하므로 동일한 인스턴스 프로파일을 사용하는 모든 데이터 마이그레이션은 동일한 퍼블릭 IP 주소를 갖습니다. 데이터 마이그레이션이 중지되거나 실패할 경우, AWS DMS는 VPC 피어링 연결을 삭제합니다.

CIDR 블록이 인스턴스 프로파일 VPC의 VPC와 겹치지 않도록 AWS DMS는 CIDR 블록 `10.0.0.0/8`, `172.16.0.0/12`, `192.168.0.0/16` 중 하나로부터 `/24` 접두사를 사용합니다. 예를 들어, 세 개의 데이터 마이그레이션을 동시에 실행하는 경우, AWS DMS는 다음 CIDR 블록을 사용하여 VPC 피어링 연결을 설정합니다.

- `192.168.0.0/24` - 첫 번째 데이터 마이그레이션에 사용
- `192.168.1.0/24` - 두 번째 데이터 마이그레이션에 사용
- `192.168.2.0/24` - 세 번째 데이터 마이그레이션에 사용

다양한 네트워크 구성을 사용하여 AWS DMS에서 소스 데이터베이스와 대상 데이터베이스 간의 상호 작용을 설정할 수 있습니다. 또한 지속적인 데이터 복제를 위해서는 소스 데이터베이스와 대상 데이터베이스 간의 상호 작용을 설정해야 합니다. 이러한 구성은 소스 데이터 공급자의 위치와 네트워크 설정에 따라 달라집니다. 다음 섹션에서는 일반적인 네트워크 구성에 대해 설명합니다.

주제

- 소스 및 대상 데이터 공급자를 위한 단일 VPC 사용
- 소스 및 대상 데이터 공급자에 대해 다양한 VPC를 사용
- 온프레미스 소스 데이터 공급자 사용
- 지속적인 데이터 복제 구성

소스 및 대상 데이터 공급자를 위한 단일 VPC 사용

이 구성에서 AWS DMS는 프라이빗 네트워크 내의 소스 및 대상 데이터 공급자에 연결합니다.

소스 및 대상 데이터 공급자가 동일한 VPC에 있을 때 네트워크를 구성하려면

1. AWS DMS 콘솔에서 소스 및 대상 데이터 공급자가 사용하는 VPC와 서브넷으로 서브넷 그룹을 생성합니다. 자세한 내용은 서브넷 그룹 생성 섹션을 참조하세요.
2. VPC 및 생성한 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 또한 소스 및 대상 데이터 공급자가 사용하는 VPC 보안 그룹을 선택하십시오. 자세한 내용은 인스턴스 프로파일 생성 섹션을 참조하세요.

이 구성에서는 데이터 마이그레이션에 퍼블릭 IP 주소를 사용할 필요가 없습니다.

소스 및 대상 데이터 공급자에 대해 다양한 VPC를 사용

이 구성에서 AWS DMS는 프라이빗 네트워크를 사용하여 소스 또는 대상 데이터 공급자에 연결합니다. 다른 데이터 공급자의 경우, AWS DMS는 퍼블릭 네트워크를 사용합니다. 인스턴스 프로파일과 동일한 VPC에 있는 데이터 공급자에 따라 다음 구성 중 하나를 선택하십시오.

소스 데이터 공급자를 위한 프라이빗 네트워크와 대상 데이터 공급자를 위한 퍼블릭 네트워크를 구성하려면

1. AWS DMS 콘솔에서 소스 데이터 공급자가 사용하는 VPC와 서브넷으로 서브넷 그룹을 생성합니다. 자세한 내용은 서브넷 그룹 생성 섹션을 참조하세요.
2. VPC 및 생성한 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 또한 소스 데이터 공급자가 사용하는 VPC 보안 그룹을 선택하십시오. 자세한 내용은 인스턴스 프로파일 생성 섹션을 참조하세요.
3. 마이그레이션 프로젝트를 엽니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 세부 정보 탭의 연결 및 보안에서 퍼블릭 IP 주소를 기록해 둡니다.
4. 대상 데이터베이스 보안 그룹에서 데이터 마이그레이션의 퍼블릭 IP 주소를 통한 액세스를 허용하십시오. 자세한 내용은 Amazon Relational Database Service 사용 설명서의 보안 그룹을 통한 액세스 제어를 참조하세요.

소스 데이터 공급자를 위한 퍼블릭 네트워크와 대상 데이터 공급자를 위한 프라이빗 네트워크를 구성하려면

1. AWS DMS 콘솔에서 대상 데이터 공급자가 사용하는 VPC와 서브넷으로 서브넷 그룹을 생성합니다. 자세한 내용은 서브넷 그룹 생성 섹션을 참조하세요.

2. VPC 및 생성한 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 또한 대상 데이터 공급자가 사용하는 VPC 보안 그룹을 선택하십시오. 자세한 내용은 인스턴스 프로파일 생성 섹션을 참조하세요.
3. 마이그레이션 프로젝트를 엽니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 세부 정보 탭의 연결 및 보안에서 퍼블릭 IP 주소를 기록해 둡니다.
4. 소스 데이터베이스 보안 그룹에서 데이터 마이그레이션의 퍼블릭 IP 주소를 통한 액세스를 허용하십시오. 자세한 내용은 Amazon Relational Database Service 사용 설명서의 보안 그룹을 통한 액세스 제어를 참조하세요.

온프레미스 소스 데이터 공급자 사용

이 구성에서 AWS DMS는 퍼블릭 네트워크 내의 소스 데이터 공급자에 연결합니다. AWS DMS는 프라이빗 네트워크를 사용하여 대상 데이터 공급자에 연결합니다.

소스 온프레미스 데이터 공급자를 위한 네트워크를 구성하려면

1. AWS DMS 콘솔에서 대상 데이터 공급자가 사용하는 VPC와 서브넷으로 서브넷 그룹을 생성합니다. 자세한 내용은 서브넷 그룹 생성 섹션을 참조하세요.
2. VPC 및 생성한 서브넷 그룹을 사용하여 AWS DMS 콘솔에서 인스턴스 프로파일을 생성합니다. 또한 대상 데이터 공급자가 사용하는 VPC 보안 그룹을 선택하십시오. 자세한 내용은 인스턴스 프로파일 생성 섹션을 참조하세요.
3. 마이그레이션 프로젝트를 엽니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 세부 정보 탭의 연결 및 보안에서 퍼블릭 IP 주소를 기록해 둡니다.
4. AWS DMS 내 데이터 마이그레이션의 퍼블릭 IP 주소에서 소스 데이터베이스에 액세스할 수 있도록 허용하세요.

AWS DMS는 VPC 보안 그룹에서 인바운드 또는 아웃바운드 규칙을 생성합니다. 이 작업을 수행하면 데이터 마이그레이션이 실패할 수 있으므로 이러한 규칙을 삭제하지 마십시오. VPC 보안 그룹에서 고유한 규칙을 구성할 수 있습니다. 규칙을 관리할 수 있도록 설명을 규칙에 추가하는 것이 좋습니다.

지속적인 데이터 복제 구성

전체 로드 및 변경 데이터 캡처(CDC) 또는 변경 데이터 캡처(CDC) 유형의 데이터 마이그레이션을 실행하려면 소스 데이터베이스와 대상 데이터베이스 간에 연결을 허용해야 합니다.

공개적으로 액세스할 수 있는 소스 데이터베이스와 대상 데이터베이스 간의 연결을 구성하려면

1. 소스 및 대상 데이터베이스의 퍼블릭 IP 주소를 기록해 두십시오.
2. 대상 데이터베이스의 퍼블릭 IP 주소에서 소스 데이터베이스에 액세스할 수 있도록 허용하십시오.
3. 소스 데이터베이스의 퍼블릭 IP 주소에서 대상 데이터베이스에 액세스할 수 있도록 허용하십시오.

단일 VPC에서 비공개로 액세스할 수 있는 소스 데이터베이스와 대상 데이터베이스 간의 연결을 구성하려면

1. 소스 및 대상 데이터베이스의 프라이빗 IP 주소를 기록해 두십시오.

   > **Important**
   >
   > 소스 및 대상 데이터베이스가 서로 다른 VPC나 네트워크에 있는 경우, 소스 및 대상 데이터베이스에 퍼블릭 IP 주소만 사용할 수 있습니다. 데이터 공급자에서는 퍼블릭 호스트 이름 또는 IP 주소만 사용할 수 있습니다.

2. 대상 데이터베이스의 프라이빗 IP 주소에서 소스 데이터베이스에 액세스할 수 있도록 허용하십시오.
3. 소스 데이터베이스의 프라이빗 IP 주소에서 대상 데이터베이스에 액세스할 수 있도록 허용하십시오.

에서 동종 데이터 마이그레이션을 위한 소스 데이터 공급자 생성 AWS DMS

MySQL 호환, PostgreSQL 및 MongoDB 호환 데이터베이스를 in의 소스 데이터 공급자로 사용할 수 있습니다. 동종 데이터베이스 마이그레이션 AWS DMS

지원되는 데이터베이스 버전은 을 참조하십시오. DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자

소스 데이터 공급자는 온프레미스, Amazon EC2 또는 Amazon RDS 데이터베이스일 수 있습니다.

주제

- MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 AWS DMS
- PostgreSQL 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS

- MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 AWS DMS

MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 AWS DMS

MySQL 호환 데이터베이스(MySQL 또는 MariaDB)를 AWS DMS의 동종 데이터베이스 마이그레이션에 대한 소스로 사용할 수 있습니다. 이 경우, 소스 데이터 공급자는 온프레미스, Amazon EC2, RDS for MySQL 또는 MariaDB 데이터베이스일 수 있습니다.

동종 데이터 마이그레이션을 실행하려면 모든 소스 테이블과 복제용 보조 객체에 대한 SELECT 권한이 있는 데이터베이스 사용자를 사용해야 합니다. 변경 데이터 캡처(CDC) 작업의 경우, 이 사용자는 REPLICATION CLIENT(MariaDB 10.5.2 이후 버전의 경우 BINLOG MONITOR) 및 REPLICATION SLAVE 권한도 있어야 합니다. 전체 로드 데이터 마이그레이션의 경우, 이 두 가지 권한이 필요하지 않습니다.

다음 스크립트를 사용하여 MySQL 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다. 마이그레이션할 대상 모든 데이터베이스에 대해 GRANT 쿼리를 실행합니다. AWS

```
CREATE USER 'your_user'@'%' IDENTIFIED BY 'your_password';

GRANT REPLICATION SLAVE, REPLICATION CLIENT  ON *.* TO 'your_user'@'%';
GRANT SELECT, RELOAD, LOCK TABLES, SHOW VIEW, EVENT, TRIGGER ON *.* TO 'your_user'@'%';

GRANT BACKUP_ADMIN ON *.* TO 'your_user'@'%';
```

이전 예제에서는 자신의 정보로 각각의 *### ## ## ###*를 바꿉니다. 원본 MySQL 데이터베이스 버전이 8.0보다 낮으면 GRANT BACKUP_ADMIN 명령을 건너뛸 수 있습니다.

다음 스크립트를 사용하여 MariaDB 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다. 마이그레이션할 대상 모든 데이터베이스에 대해 GRANT 쿼리를 실행합니다 AWS.

```
CREATE USER 'your_user'@'%' IDENTIFIED BY 'your_password';
GRANT SELECT, RELOAD, LOCK TABLES, REPLICATION SLAVE, BINLOG MONITOR, SHOW VIEW ON  *.*
 TO 'your_user'@'%';
```

다음 예제에서는 자신의 정보로 각각의 *### ## ## ###*를 바꿉니다.

다음 섹션에서는 자체 관리형 및 AWS 관리형 MySQL 데이터베이스의 특정 구성 사전 요구 사항을 설명합니다.

주제

- 자체 관리형 MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용
- AWS관리형 MySQL 호환 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS
- MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 것에 대한 제한 사항

자체 관리형 MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용

이 섹션에서는 온프레미스나 Amazon EC2 인스턴스에서 호스팅되는 MySQL 호환 데이터베이스를 구성하는 방법에 대해 설명합니다.

원본 MySQL 또는 MariaDB 데이터베이스의 버전을 선택합니다. 에 설명된 대로 소스 MySQL 또는 MariaDB 데이터베이스 버전을 AWS DMS 지원하는지 확인하십시오. DMS 동종 데이터 마이그레이션이 지원하는 소스

CDC를 사용하려면 바이너리 로깅을 활성화해야 합니다. 바이너리 로깅을 활성화하려면 MySQL 또는 MariaDB의 `my.ini`(Windows) 또는 `my.cnf`(UNIX) 파일에서 다음 파라미터를 구성해야 합니다.

파라미터	값
`server-id`	이 파라미터 값을 1 이상의 값으로 설정합니다.
`log-bin`	`log-bin=E:\MySql_Logs\BinLog` 과 같은 이진 로그 파일로 이 경로를 설정합니다. 파일 확장자를 포함해서는 안 됩니다.
`binlog_format`	이 파라미터를 `ROW`으로 설정합니다. `binlog_format` 을 `STATEMENT` 로 설정하면 대상에 데이터를 복제할 때 불일치가 발생할 수 있으므로 복제 중에 이 설정을 사용하는 것이 좋습니다. 또한 데이터베이스 엔진이 자동으로 `STATEMENT` 기반 로깅으로 전환되기 때문에 `binlog_format` 이 `MIXED`로 설정된 경우 데이터베이스 엔진은 일관되지 않은 유사 데이터를 대상에 기록합니다.
`expire_logs_days`	이 파라미터 값을 1 이상의 값으로 설정합니다. 디스크 공간의 과사용을 방지하려면 기본값 0을 사용하지 않는 것이 좋습니다.
`binlog_checksum`	이 파라미터를 `NONE`으로 설정합니다.

파라미터	값
binlog_row_image	이 파라미터를 FULL로 설정합니다.
log_slave_updates	MySQL 또는 MariaDB 복제본을 소스로 사용하고 있는 경우 이 파라미터를 TRUE로 설정합니다.

AWS관리형 MySQL 호환 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS

이 섹션에서는 Amazon RDS for MySQL과 Amazon RDS for MariaDB 데이터베이스 인스턴스를 구성하는 방법에 대해 설명합니다.

에서 동종 데이터 마이그레이션을 AWS DMS 위한 소스로 AWS 관리되는 MySQL 또는 MariaDB 데이터베이스를 사용하는 경우 CDC에 대한 다음과 같은 사전 요구 사항이 있는지 확인하십시오.

- RDS for MySQL 및 MariaDB에 대한 바이너리 로그를 활성화하려면 인스턴스 수준에서 자동 백업을 활성화하십시오. Aurora MySQL 클러스터의 바이너리 로그를 활성화하려면 파라미터 그룹에서 binlog_format 변수를 변경하십시오. Aurora MySQL 클러스터에 대해서는 자동 백업을 활성화할 필요는 없습니다.

  그런 다음, binlog_format 파라미터를 ROW로 설정합니다.

  자동 백업 설정에 관한 자세한 내용은 Amazon RDS 사용 설명서의 자동 백업 활성화 섹션을 참조하세요.

  Amazon RDS for MySQL 또는 MariaDB 데이터베이스의 바이너리 로깅 설정에 관한 자세한 내용은 Amazon RDS 사용 설명서의 바이너리 로깅 형식 설정을 참조하십시오.

  Aurora MySQL 클러스터의 바이너리 로깅을 설정하는 방법에 관한 자세한 내용은 Amazon Aurora MySQL 클러스터의 바이너리 로깅을 활성화하려면 어떻게 해야 합니까?를 참조하십시오.
- 에서 바이너리 AWS DMS 로그를 사용할 수 있는지 확인하십시오. AWS-managed MySQL 및 MariaDB 데이터베이스는 가능한 한 빨리 바이너리 로그를 삭제하므로 로그를 사용할 수 있는 기간을 늘려야 합니다. 예를 들어, 로그 보존 시간을 24시간으로 늘리려면 다음 명령을 실행합니다.

  ```
  call mysql.rds_set_configuration('binlog retention hours', 24);
  ```
- binlog_row_image 파라미터를 Full로 설정합니다.

- `binlog_checksum` 파라미터를 `NONE`으로 설정합니다.
- Amazon RDS MySQL 또는 MariaDB 복제본을 소스로 사용하고 있는 경우, 읽기 전용 복제본에서 백업을 활성화하고 `log_slave_updates` 파라미터를 `TRUE`로 설정해야 합니다.

MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 것에 대한 제한 사항

MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 시 적용되는 제한 사항은 다음과 같습니다.

- 시퀀스와 같은 MariaDB 객체는 동종 마이그레이션 작업에서 지원되지 않습니다.
- MariaDB에서 Amazon RDS MySQL/Aurora MySQL로의 마이그레이션은 호환되지 않는 객체 차이로 인해 실패할 수 있습니다.
- 데이터 소스에 연결하는 데 사용하는 사용자 이름에는 다음과 같은 제한이 있습니다.
  - 길이는 2~64자일 수 있습니다.
  - 스페이스를 포함할 수 없습니다.
  - a-z, A-Z, 0-9, 밑줄(_) 문자를 포함할 수 있습니다.
  - a-z 또는 A-Z로 시작해야 합니다.
- 데이터 소스에 연결하는 데 사용하는 암호에는 다음과 같은 제한이 있습니다.
  - 길이는 1~128자일 수 있습니다.
  - 작은따옴표('), 큰따옴표("), 세미콜론(;) 또는 스페이스는 포함할 수 없습니다.

PostgreSQL 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS

PostgreSQL 데이터베이스를 AWS DMS의 동종 데이터베이스 마이그레이션에서 소스로 사용할 수 있습니다. 이 경우, 소스 데이터 공급자는 온프레미스, Amazon EC2, RDS for PostgreSQL 데이터베이스일 수 있습니다.

동종 데이터 마이그레이션을 실행하려면 PostgreSQL 소스 데이터베이스에 지정한 데이터베이스 사용자에게 수퍼유저 권한을 AWS DMS 부여하십시오. 데이터베이스 사용자가 원본의 복제 관련 함수에 액세스하려면 수퍼유저 권한이 필요합니다. 전체 로드 데이터 마이그레이션의 경우, 테이블을 마이그레이션할 수 있는 `SELECT` 권한이 데이터베이스 사용자에게 있어야 합니다.

다음 스크립트를 사용하여 PostgreSQL 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다. 마이그레이션할 대상 모든 데이터베이스에 대해 GRANT 쿼리를 실행합니다. AWS

```
CREATE USER your_user WITH LOGIN PASSWORD 'your_password';
ALTER USER your_user WITH SUPERUSER;
GRANT SELECT ON ALL TABLES IN SCHEMA schema_name TO your_user;
```

이전 예제에서는 자신의 정보로 각각의 *### ## ## ###*를 바꿉니다.

다음 섹션에서는 자체 관리형 및 AWS 관리형 PostgreSQL 데이터베이스의 특정 구성 사전 요구 사항을 설명합니다.

주제

- 자체 관리형 PostgreSQL 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS
- AWS-managed PostgreSQL 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS
- PostgreSQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 것에 대한 제한 사항

자체 관리형 PostgreSQL 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS

이 섹션에서는 온프레미스나 Amazon EC2 인스턴스에서 호스팅되는 PostgreSQL 데이터베이스를 구성하는 방법에 대해 설명합니다.

소스 PostgreSQL 데이터베이스의 버전을 확인하십시오. 에 설명된 대로 소스 PostgreSQL 데이터베이스 버전을 AWS DMS 지원하는지 확인하십시오. DMS 동종 데이터 마이그레이션이 지원하는 소스

동종 데이터 마이그레이션은 논리적 복제를 사용하여 CDC(변경 데이터 캡처)를 지원합니다. 자체 관리형 PostgreSQL 소스 데이터베이스에서 논리적 복제를 활성화하려면 `postgresql.conf` 구성 파일에서 다음 파라미터와 값을 설정합니다.

- `wal_level`을 `logical`로 설정합니다.
- `max_replication_slots`를 1보다 큰 값으로 설정합니다.

  실행할 작업 개수에 따라 `max_replication_slots` 값을 설정합니다. 예를 들어, 5개 작업을 실행하려면 최소 5개의 슬롯을 설정해야 합니다. 작업이 시작된 즉시 슬롯이 자동으로 열리고 작업이 더 이상 실행되지 않더라도 계속 열려 있습니다. 열린 슬롯은 반드시 수동으로 삭제하세요.

- `max_wal_senders`를 1보다 큰 값으로 설정합니다.

  `max_wal_senders` 파라미터는 실행 가능한 동시 작업 개수를 설정합니다.
- `wal_sender_timeout` 파라미터는 지정된 밀리초보다 오래 사용되지 않는 복제 연결을 종료합니다. 기본값은 60,000밀리초(60초)입니다. 값을 0으로 설정하면 시간 제한 메커니즘이 비활성화되며 이는 DMS에 유효한 설정입니다.

일부 파라미터는 정적이며 서버 시작 시에만 설정할 수 있습니다. 구성 파일의 해당 항목에 대한 모든 변경 사항은 서버를 다시 시작할 때까지 무시됩니다. 자세한 내용은 PostgreSQL 설명서를 참조하세요.

AWS-managed PostgreSQL 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS

이 섹션에서는 Amazon RDS for PostgreSQL 데이터베이스 인스턴스를 구성하는 방법에 대해 설명합니다.

PostgreSQL DB 인스턴스의 AWS 마스터 사용자 계정을 PostgreSQL 원본 데이터 공급자의 사용자 계정으로 사용하여 동종 데이터 마이그레이션을 수행할 수 있습니다. AWS DMS 마스터 사용자 계정에는 CDC 설정을 허용하는 데 필요한 역할이 있습니다. 마스터 사용자 계정 이외의 계정을 사용하는 경우, 이 계정에는 `rds_superuser` 역할과 `rds_replication` 역할이 있어야 합니다. `rds_replication` 역할은 논리적 슬롯을 관리하고 논리적 슬롯을 사용하여 데이터를 스트리밍할 수 있는 권한을 부여합니다.

다음 코드 예제를 사용하여 `rds_superuser` 및 `rds_replication` 역할을 부여하십시오.

```
GRANT rds_superuser to your_user;
GRANT rds_replication to your_user;
```

이전 예제에서 *`your_user`*를 데이터베이스 사용자 이름으로 바꾸십시오.

논리적 복제를 활성화하려면 DB 파라미터 그룹의 `rds.logical_replication` 파라미터를 1로 설정합니다. 이 정적 파라미터를 적용하려면 DB 인스턴스를 재부팅해야 합니다.

PostgreSQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용하는 것에 대한 제한 사항

PostgreSQL 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 시 적용되는 제한 사항은 다음과 같습니다.

- 데이터 소스에 연결하는 데 사용하는 사용자 이름에는 다음과 같은 제한이 있습니다.
  - 길이는 2~64자일 수 있습니다.
  - 스페이스를 포함할 수 없습니다.
  - a-z, A-Z, 0-9, 밑줄(_) 문자를 포함할 수 있습니다.
  - a-z 또는 A-Z로 시작해야 합니다.
- 데이터 소스에 연결하는 데 사용하는 암호에는 다음과 같은 제한이 있습니다.
  - 길이는 1~128자일 수 있습니다.
  - 작은따옴표('), 큰따옴표("), 세미콜론(;) 또는 스페이스는 포함할 수 없습니다.

MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 소스로 사용 AWS DMS

에서 MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 원본으로 사용할 수 있습니다. AWS DMS 이 경우 소스 데이터 공급자는 온프레미스, MongoDB용 Amazon EC2 데이터베이스 또는 Amazon DocumentDB (MongoDB 호환 사용) 데이터베이스일 수 있습니다.

지원되는 데이터베이스 버전은 을 참조하십시오. DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자

다음 섹션에서는 자체 관리형 MongoDB 데이터베이스 및 관리형 Amazon DocumentDB 데이터베이스의 특정 구성 사전 요구 사항을 설명합니다. AWS

주제

- 자체 관리형 MongoDB 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS
- Amazon DocumentDB 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS
- MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용하는 기능
- MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 원본으로 사용하는 경우의 제한 사항
- MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 원본으로 사용하는 모범 사례

자체 관리형 MongoDB 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS

이 섹션에서는 온프레미스 또는 Amazon EC2 인스턴스에 호스팅되는 MongoDB 데이터베이스를 구성하는 방법을 설명합니다.

소스 MongoDB 데이터베이스의 버전을 확인하십시오. 에 설명된 대로 소스 MongoDB 데이터베이스 버전을 AWS DMS 지원하는지 확인하십시오. DMS 동종 데이터 마이그레이션이 지원하는 소스 데이터 공급자

MongoDB 소스로 동종 데이터 마이그레이션을 실행하려면 루트 권한이 있는 사용자 계정을 만들거나 마이그레이션할 데이터베이스에만 권한이 있는 사용자를 만들 수 있습니다. 사용자 생성에 대한 자세한 내용은 을 참조하십시오. MongoDB를 AWS DMS 소스로 사용할 때 필요한 권한

MongoDB에서 지속적인 복제 또는 CDC를 사용하려면 MongoDB AWS DMS 작업 로그 (oplog) 에 액세스할 수 있어야 합니다. 자세한 설명은 CDC를 위한 MongoDB 복제본 세트 구성 섹션을 참조하세요.

MongoDB 인증 방법에 대한 자세한 내용은 을 참조하십시오. AWS DMS에서 MongoDB를 소스로 사용하기 위한 보안 요구 사항

MongoDB를 원본으로 사용하는 경우, 동종 데이터 마이그레이션은 Amazon DocumentDB가 지원하는 모든 데이터 유형을 지원합니다.

MongoDB를 소스로 사용하는 경우 Secrets Manager에 사용자 자격 증명을 저장하려면 기타 유형의 비밀 유형을 사용하여 사용자 자격 증명을 일반 텍스트로 제공해야 합니다. 자세한 설명은 보안 암호를 사용하여 AWS Database Migration Service 엔드포인트에 액세스하기 섹션을 참조하세요.

다음 코드 샘플은 일반 텍스트를 사용하여 데이터베이스 암호를 저장하는 방법을 보여줍니다.

```
{
  "username": "dbuser",
  "password": "dbpassword"
}
```

Amazon DocumentDB 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용 AWS DMS

이 단원에서는 Amazon DocumentDB 데이터베이스 인스턴스를 동종 데이터 마이그레이션의 소스로 사용하도록 구성하는 방법을 설명합니다.

Amazon DocumentDB 인스턴스의 마스터 사용자 이름을 MongoDB 호환 소스 데이터 공급자의 사용자 계정으로 사용하여 동종 데이터를 마이그레이션할 수 있습니다. AWS DMS 마스터 사용자 계정에는 CDC 설정을 허용하는 데 필요한 역할이 있습니다. 마스터 사용자 계정이 아닌 계정을 사용하는 경우 계정에 루트 역할이 있어야 합니다. 사용자를 루트 계정으로 생성하는 방법에 대한 자세한 내용은 을 참조하십시오Amazon DocumentDB를 소스로 사용하기 위한 권한 설정.

논리적 복제를 활성화하려면 데이터베이스 change_stream_log_retention_duration 파라미터 그룹의 파라미터를 트랜잭션 워크로드에 적합한 설정으로 설정하십시오. 이 정적 파라미터를 변경하려면 DB 인스턴스를 재부팅해야 적용됩니다. Full Load Only를 포함한 모든 작업 유형에 대한 데이터 마이그레이션을 시작하기 전에, 지정된 데이터베이스 내의 모든 컬렉션 또는 선택한 컬렉션에 대해서만 Amazon DocumentDB 변경 스트림을 활성화하십시오. Amazon DocumentDB의 변경 스트림을 활성화하는 방법에 대한 자세한 내용은 Amazon DocumentDB 개발자 안내서의 변경 스트림 활성화를 참조하십시오.

> **Note**
>
> AWS DMS Amazon DocumentDB 변경 스트림을 사용하여 복제가 진행 중인 동안 변경 사항을 캡처합니다. Amazon DocumentDB가 변경 스트림에서 레코드를 읽기 전에 해당 레코드를 삭제하면 작업이 실패합니다. 변경 사항을 최소 24시간 동안 보존하도록 change_stream_log_retention_duration 파라미터를 설정하는 것이 좋습니다.

Amazon DocumentDB를 사용하여 동종 데이터를 마이그레이션하려면 Secrets Manager의 Amazon DocumentDB 데이터베이스 자격 증명 아래에 사용자 자격 증명을 저장하십시오.

MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션을 위한 소스로 사용하는 기능

- 전체 로드 단계에서 Amazon DocumentDB가 지원하는 모든 보조 인덱스를 마이그레이션할 수 있습니다.
- AWS DMS 컬렉션을 병렬로 마이그레이션합니다. 동종 데이터 마이그레이션은 컬렉션 내 각 문서의 평균 크기를 기준으로 런타임 시 세그먼트를 계산하여 성능을 극대화합니다.
- DMS는 CDC 단계에서 만든 보조 색인을 복제할 수 있습니다. DMS는 MongoDB 버전 6.0에서 이 기능을 지원합니다.
- DMS는 중첩 수준이 97보다 큰 문서를 지원합니다.

MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 원본으로 사용하는 경우의 제한 사항

- 문서에는 $ 접두사가 있는 필드 이름이 있을 수 없습니다.
- AWS DMS 시계열 컬렉션 마이그레이션은 지원하지 않습니다.

- AWS DMS CDC 단계 중의 `createdrop`, 또는 `rename collection` DDL 이벤트는 지원하지 않습니다.
- AWS DMS 필드 컬렉션에서 일치하지 않는 데이터 유형은 지원하지 않습니다. `_id` 예를 들어 다음과 같은 지원되지 않는 컬렉션에는 필드에 대해 여러 데이터 유형이 있습니다. `_id`

```
rs0 [direct: primary] test> db.collection1.aggregate([
...   {
...     $group: {
...       _id: { $type: "$_id" },
...       count: { $sum: 1 }
...     }
...   }
... ])
[ { _id: 'string', count: 6136 }, { _id: 'objectId', count: 848033 } ]
```

- CDC 전용 작업의 경우 시작 AWS DMS 모드만 지원합니다. `immediate`
- AWS DMS 잘못된 UTF8 문자가 있는 문서는 지원하지 않습니다.
- AWS DMS 샤딩된 컬렉션을 지원하지 않습니다.

MongoDB 호환 데이터베이스를 동종 데이터 마이그레이션의 원본으로 사용하는 모범 사례

- 동일한 MongoDB 인스턴스에서 호스팅되는 여러 대규모 데이터베이스 및 컬렉션의 경우 각 데이터베이스 및 컬렉션에 대한 선택 규칙을 사용하여 여러 데이터 마이그레이션 작업과 프로젝트 간에 작업을 분할하는 것이 좋습니다. 데이터베이스 및 컬렉션 부서를 조정하여 성능을 극대화할 수 있습니다.

AWS DMS에서 동종 데이터 마이그레이션을 위한 대상 데이터 공급자 생성

MySQL과 호환되는 PostgreSQL 및 Amazon DocumentDB 데이터베이스를 에서 동종 데이터 마이그레이션을 위한 대상 데이터 공급자로 사용할 수 있습니다. AWS DMS

지원되는 데이터베이스 버전은 DMS 동종 데이터 마이그레이션이 지원하는 대상 데이터 공급자 을 참조하십시오.

대상 데이터 공급자는 Amazon RDS DB 인스턴스 또는 Amazon Aurora DB 클러스터일 수 있습니다. 대상 데이터 공급자의 데이터베이스 버전은 원본 데이터 공급자의 데이터베이스 버전과 같거나 더 높아야 합니다.

주제

- MySQL 호환 데이터베이스를 AWS DMS의 동종 데이터 마이그레이션의 대상으로 사용
- PostgreSQL 데이터베이스를 AWS DMS의 동종 데이터 마이그레이션의 소스로 사용
- Amazon DocumentDB 데이터베이스를 동종 데이터 마이그레이션의 대상으로 사용 AWS DMS

MySQL 호환 데이터베이스를 AWS DMS의 동종 데이터 마이그레이션의 대상으로 사용

MySQL 호환 데이터베이스를 AWS DMS의 동종 데이터 마이그레이션의 대상으로 사용할 수 있습니다.

AWS DMS에서는 데이터를 Amazon RDS for MySQL, MariaDB 또는 Amazon Aurora MySQL 데이터베이스로 마이그레이션하려면 특정 권한이 필요합니다. 다음 스크립트를 사용하여 MySQL 대상 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다.

```
CREATE USER 'your_user'@'%' IDENTIFIED BY 'your_password';

GRANT ALTER, CREATE, DROP, INDEX, INSERT, UPDATE, DELETE, SELECT, CREATE VIEW, CREATE
 ROUTINE, ALTER ROUTINE, EVENT, TRIGGER, EXECUTE, REFERENCES ON *.* TO 'your_user'@'%';
GRANT REPLICATION SLAVE, REPLICATION CLIENT  ON *.* TO 'your_user'@'%';
```

이전 예제에서는 자신의 정보로 각각의 *### ## ## ###*를 바꿉니다.

다음 스크립트를 사용하여 MariaDB 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다. AWS로 마이그레이션할 모든 데이터베이스에 대해 GRANT 쿼리를 실행합니다.

```
CREATE USER 'your_user'@'%' IDENTIFIED BY 'your_password';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER, CREATE VIEW, CREATE
 ROUTINE, ALTER ROUTINE, EVENT, TRIGGER, EXECUTE,SLAVE MONITOR, REPLICATION SLAVE ON
 *.* TO 'your_user'@'%';
```

이전 예제에서는 자신의 정보로 각각의 *### ## ## ###*를 바꿉니다.

> **Note**
>
> Amazon RDS에서 MySQL/Maria 데이터베이스 인스턴스의 자동 백업을 활성화하면 바이너리 로깅도 활성화됩니다. 이러한 설정을 활성화하면 대상 데이터베이스에서 함수, 프로시저, 트리거와 같은 보조 객체를 생성하는 동안 다음 오류가 발생하여 데이터 마이그레이션 작업이 실패할 수 있습니다. 대상 데이터베이스에 바이너리 로깅이 활성화되어 있는 경우 작업을 시작하기 전에 데이터베이스 파라미터 그룹에서 `log_bin_trust_function_creators`를 `true`로 설정하십시오.
>
> ```
> ERROR 1419 (HY000): You don't have the SUPER privilege and binary logging is
>  enabled (you might want to use the less safe log_bin_trust_function_creators
>  variable)
> ```

MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 대상으로 사용하는 것에 대한 제한 사항

MySQL 호환 데이터베이스를 동종 데이터 마이그레이션의 대상으로 사용 시 적용되는 제한 사항은 다음과 같습니다.

- 데이터 소스에 연결하는 데 사용하는 사용자 이름에는 다음과 같은 제한이 있습니다.
  - 길이는 2~64자일 수 있습니다.
  - 스페이스를 포함할 수 없습니다.
  - a-z, A-Z, 0-9, 밑줄(_) 문자를 포함할 수 있습니다.
  - a-z 또는 A-Z로 시작해야 합니다.
- 데이터 소스에 연결하는 데 사용하는 암호에는 다음과 같은 제한이 있습니다.
  - 길이는 1~128자일 수 있습니다.
  - 작은따옴표('), 큰따옴표("), 세미콜론(;) 또는 스페이스는 포함할 수 없습니다.

PostgreSQL 데이터베이스를 AWS DMS의 동종 데이터 마이그레이션의 소스로 사용

PostgreSQL 데이터베이스를 AWS DMS의 동종 데이터 마이그레이션의 대상으로 사용할 수 있습니다.

AWS DMS에서는 데이터를 Amazon RDS for PostgreSQL 또는 Amazon Aurora PostgreSQL 데이터베이스로 마이그레이션하려면 특정 권한이 필요합니다. 다음 스크립트를 사용하여 PostgreSQL 대상 데이터베이스에서 필요한 권한을 가진 데이터베이스 사용자를 생성합니다.

```
CREATE USER your_user WITH LOGIN PASSWORD 'your_password';
GRANT USAGE ON SCHEMA schema_name TO your_user;
GRANT CONNECT ON DATABASE db_name to your_user;
GRANT CREATE ON DATABASE db_name TO your_user;
GRANT CREATE ON SCHEMA schema_name TO your_user;
GRANT UPDATE, INSERT, SELECT, DELETE, TRUNCATE ON ALL TABLES IN SCHEMA schema_name
 TO your_user;
```

이전 예제에서는 자신의 정보로 각각의 *### ## ## ###*를 바꿉니다.

RDS for PostgreSQL 대상에 대한 논리적 복제를 활성화하려면 DB 파라미터 그룹의 `rds.logical_replication` 파라미터를 1로 설정합니다. 이 정적 파라미터를 적용하려면 DB 인스턴스 또는 DB 클러스터를 재부팅해야 합니다. 일부 파라미터는 정적이며 서버 시작 시에만 설정할 수 있습니다. AWS DMS는 서버를 다시 시작할 때까지 DB 파라미터 그룹의 항목 변경 사항을 무시합니다.

PostgreSQL은 트리거를 사용하여 외래 키 제약 조건을 구현합니다. 전체 로드 단계 동안 AWS DMS는 한 번에 하나씩 각 테이블을 로드합니다. 전체 로드 중에는 대상 데이터베이스의 외래 키 제약 조건을 해제하는 것이 좋습니다. 이렇게 하려면 다음 방법 중 하나를 사용하십시오.

- 인스턴스에서 모든 트리거를 임시로 비활성화하고 전체 로드를 완료합니다.
- PostgreSQL 내에서 `session_replication_role` 파라미터의 값을 변경하십시오.

  해당 시점에 트리거는 `origin`, `replica`, `always` 또는 `disabled` 상태 중 하나일 수 있습니다. `session_replication_role` 파라미터를 `replica`로 설정하면 해당 `replica` 상태의 트리거만 활성화됩니다. 그렇지 않으면, 트리거가 비활성 상태로 유지됩니다.

PostgreSQL 호환 데이터베이스를 동종 데이터 마이그레이션의 대상으로 사용하는 것에 대한 제한 사항

PostgreSQL 호환 데이터베이스를 동종 데이터 마이그레이션의 대상으로 사용 시 적용되는 제한 사항은 다음과 같습니다.

- 데이터 소스에 연결하는 데 사용하는 사용자 이름에는 다음과 같은 제한이 있습니다.
  - 길이는 2~64자일 수 있습니다.

  - 스페이스를 포함할 수 없습니다.
  - a-z, A-Z, 0-9, 밑줄(_) 문자를 포함할 수 있습니다.
  - a-z 또는 A-Z로 시작해야 합니다.
- 데이터 소스에 연결하는 데 사용하는 암호에는 다음과 같은 제한이 있습니다.
  - 길이는 1~128자일 수 있습니다.
  - 작은따옴표('), 큰따옴표("), 세미콜론(;) 또는 스페이스는 포함할 수 없습니다.

Amazon DocumentDB 데이터베이스를 동종 데이터 마이그레이션의 대상으로 사용 AWS DMS

Amazon DocumentDB (MongoDB 호환 지원) 데이터베이스 및 DocumentDB Elastic 클러스터를 에서 동종 데이터 마이그레이션을 위한 마이그레이션 대상으로 사용할 수 있습니다. AWS DMS

Amazon DocumentDB 대상에 대해 동종 데이터 마이그레이션을 실행하려면 관리자 권한이 있는 사용자 계정을 생성하거나 마이그레이션할 데이터베이스에 대한 읽기/쓰기 권한만 있는 사용자를 생성할 수 있습니다.

동종 데이터 마이그레이션은 Amazon DocumentDB가 지원하는 모든 BSON 데이터 유형을 지원합니다. 이러한 데이터 유형의 목록은 Amazon DocumentDB 개발자 안내서의 데이터 유형을 참조하십시오.

원본에서 샤딩되지 않은 컬렉션을 마이그레이션하는 데 DocumentDB Elastic 클러스터의 샤드 기능을 사용하려면 데이터 마이그레이션 작업을 시작하기 전에 마이그레이션할 샤드 컬렉션을 생성하십시오. Amazon DocumentDB 엘라스틱 클러스터의 샤드 컬렉션에 대한 자세한 내용은 Amazon DocumentDB 개발자 안내서의 5단계: 컬렉션 샤딩을 참조하십시오.

Amazon DocumentDB 대상의 경우AWS DMS, 또는 SSL 모드를 지원합니다none. `require`

AWS DMS에서 동종 데이터 마이그레이션 실행

을 사용하여 동종 데이터베이스 마이그레이션 원본 데이터베이스의 데이터를 Amazon RDS (Amazon RDS), Amazon Aurora 또는 Amazon DocumentDB의 동등한 엔진으로 마이그레이션할 수 있습니다. AWS DMS AWS DMS원본 및 대상 데이터베이스의 기본 데이터베이스 도구를 사용하여 데이터 마이그레이션 프로세스를 자동화합니다.

동종 데이터 마이그레이션을 위한 인스턴스 프로파일과 호환 가능한 데이터 공급자를 만든 후 마이그레이션 프로젝트를 생성하십시오. 자세한 설명은 마이그레이션 프로젝트 생성 섹션을 참조하세요.

다음 단원에서는 동종 데이터 마이그레이션을 생성하고 구성하며 실행하는 방법에 대해 설명합니다.

주제

- AWS DMS에서 데이터 마이그레이션 생성
- 동종 데이터 마이그레이션을 위한 선택 규칙
- AWS DMS에서 데이터 마이그레이션 관리
- AWS DMS에서 데이터 마이그레이션을 모니터링
- AWS DMS의 동종 데이터 마이그레이션 상태
- AWS DMS에서 동종 데이터 마이그레이션을 통해 MySQL 데이터베이스에서 데이터 마이그레이션
- AWS DMS에서 동종 데이터 마이그레이션을 통해 PostgreSQL 데이터베이스에서 데이터 마이그레이션
- 동종 데이터 마이그레이션을 통해 MongoDB 데이터베이스의 데이터를 마이그레이션합니다. AWS DMS

AWS DMS에서 데이터 마이그레이션 생성

동일한 유형의 호환 가능한 데이터 공급자와 함께 마이그레이션 프로젝트를 만든 후 이 프로젝트를 사용하여 동종 데이터 마이그레이션을 수행할 수 있습니다. 자세한 설명은 마이그레이션 프로젝트 생성 섹션을 참조하세요.

먼저 동종 데이터 마이그레이션을 사용하려면 새 데이터 마이그레이션을 생성하세요. 단일 마이그레이션 프로젝트에서 여러 유형의 동종 데이터 마이그레이션을 여러 개 생성할 수 있습니다.

AWS DMS는 AWS 계정에 대해 생성할 수 있는 최대 수의 동종 데이터 마이그레이션이 있습니다. AWS DMS 서비스 할당량 AWS Database Migration Service에 대한 할당량에 관한 자세한 내용은 다음 섹션을 참조하세요.

데이터 마이그레이션을 생성하기 전에 소스 및 대상 데이터베이스, IAM 정책 및 역할, 인스턴스 프로파일, 데이터 공급자와 같은 필수 리소스를 설정해야 합니다. 자세한 내용은 IAM 리소스 생성, 인스턴스 프로파일 생성, 데이터 공급자 생성 단원을 참조하세요.

또한 상위 데이터베이스 버전에서 하위 데이터베이스 버전으로 데이터를 마이그레이션할 때는 동종 데이터 마이그레이션을 사용하지 않는 것이 좋습니다. 소스 및 대상 데이터 공급자에 사용하는 데이터베이스 버전을 확인하고 필요하다면 대상 데이터베이스 버전을 업그레이드하십시오.

마이그레이션 작업을 생성하려면

1. AWS Management Console로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택하고 데이터 마이그레이션 탭에서 데이터 마이그레이션 생성을 선택합니다.
4. 이름에서 데이터 마이그레이션 이름을 입력합니다. 데이터 마이그레이션을 쉽게 식별할 수 있도록 데이터 마이그레이션에 고유한 이름을 사용해야 합니다.
5. 복제 유형에서 구성하려는 데이터 마이그레이션 유형을 선택합니다. 다음 옵션 중 하나를 선택할 수 있습니다.

   - 전체 로드 - 기존 소스 데이터를 마이그레이션합니다.
   - 전체 로드 및 변경 데이터 캡처(CDC) - 기존 소스 데이터를 마이그레이션하고 지속적인 변경 내용을 복제합니다.
   - 변경 데이터 캡처(CDC) - 지속적인 변경 사항을 복제합니다.
6. Amazon에 데이터 마이그레이션 CloudWatch 로그를 저장하기 위한 로그 활성화 확인란을 선택합니다 CloudWatch. 이 옵션을 선택하지 않은 경우, 데이터 마이그레이션 실패 시 로그 파일을 볼 수 없습니다.
7. (선택 사항) 고급 설정을 확장합니다. 작업 수에는 AWS DMS가 소스 데이터를 대상으로 마이그레이션하는 데 사용할 수 있는 병렬 스레드 수를 입력합니다.
8. IAM 서비스 역할의 경우, 사전 요구 사항에서 생성한 IAM 역할을 선택합니다. 자세한 설명은 AWS DMS에서 동종 데이터 마이그레이션을 위한 IAM 역할 생성 섹션을 참조하세요.
9. 변경 데이터 캡처(CDC) 유형의 데이터 마이그레이션을 위한 시작 모드를 구성합니다. 다음 옵션 중 하나를 선택할 수 있습니다.

   - 즉시 - 데이터 마이그레이션을 시작하면 지속적 복제를 시작합니다.
   - 기본 시작점 사용 - 지정된 시점부터 지속적 복제를 시작합니다.

     PostgreSQL 데이터베이스의 경우, 슬롯 이름에 논리적 복제 슬롯의 이름을 입력하고 기본 시작점에는 트랜잭션 로그 시퀀스 번호를 입력합니다.

     MySQL 데이터베이스의 경우, 로그 시퀀스 번호(LSN)에 트랜잭션 로그 시퀀스 번호를 입력합니다.
10. 변경 데이터 캡처(CDC) 또는 전체 로드 및 변경 데이터 캡처(CDC) 유형의 데이터 마이그레이션에 대해 중지 모드를 구성합니다. 다음 옵션 중 하나를 선택할 수 있습니다.

- CDC 중단 안 함 - AWS DMS는 데이터 마이그레이션을 중지할 때까지 지속적 복제를 계속합니다.
- 서버 시점 사용 - AWS DMS는 지정된 시간에 지속적 복제를 중지합니다.

  이 옵션을 선택하는 경우, 지속적 복제를 자동으로 중지하려는 날짜와 시간을 중지 날짜 및 시간에 입력합니다.

11. 데이터 마이그레이션 생성을 선택합니다.

AWS DMS는 데이터 마이그레이션을 생성하여 마이그레이션 프로젝트의 데이터 마이그레이션 탭에 있는 목록에 추가합니다. 여기에서 데이터 마이그레이션 상태를 확인할 수 있습니다. 자세한 설명은 마이그레이션 상태 섹션을 참조하세요.

> **Important**
>
> 전체 로드 및 전체 로드 및 변경 데이터 캡처(CDC) 유형의 데이터 마이그레이션의 경우, AWS DMS는 대상 데이터베이스의 모든 데이터, 테이블 및 기타 데이터베이스 객체를 삭제합니다. 대상 데이터베이스의 백업이 있는지 확인하십시오.

AWS DMS가 데이터 마이그레이션을 생성한 후 이 데이터 마이그레이션의 상태는 준비됨으로 설정됩니다. 데이터를 마이그레이션하려면 데이터 마이그레이션을 수동으로 시작해야 합니다. 이렇게 하려면 목록에서 데이터 마이그레이션을 선택합니다. 그런 다음, 작업에서 시작을 선택합니다. 자세한 설명은 데이터 마이그레이션 관리 섹션을 참조하세요.

동종 데이터 마이그레이션을 처음 실행하려면 몇 가지 설정이 필요합니다. AWS DMS는 데이터 마이그레이션을 위한 서버리스 환경을 생성합니다. 이 프로세스는 최대 15분이 걸립니다. 데이터 마이그레이션을 중지했다가 다시 시작하면 AWS DMS는 환경을 다시 생성하지 않으므로 데이터 마이그레이션에 더 빠르게 액세스할 수 있습니다.

동종 데이터 마이그레이션을 위한 선택 규칙

선택 규칙을 사용하여 복제에 포함할 스키마, 테이블 또는 둘 다를 선택할 수 있습니다.

> **Note**
>
> AWS DMSMongoDB 호환 데이터베이스를 원본으로 사용하는 경우 동종 데이터 마이그레이션에 대한 선택 규칙만 지원합니다.

데이터 마이그레이션 작업을 생성할 때는 선택 규칙 추가를 선택합니다.

규칙 설정의 경우 다음 값을 제공하십시오.

- 스키마: 스키마 입력을 선택합니다.
- 스키마 이름: 복제하거나 와일드카드로 사용할 % 스키마의 이름을 입력합니다.
- 테이블 이름: 복제하거나 와일드카드로 % 사용할 테이블의 이름을 입력합니다.

기본적으로 DMS에서 지원하는 유일한 규칙-작업은 이며 Include DMS에서 지원하는 유일한 와일드카드 문자는 입니다. %

Example 스키마에서 모든 테이블 마이그레이션

다음은 소스에서 이름이 dmsst인 스키마의 모든 테이블을 대상 엔드포인트로 마이그레이션하는 예입니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-action": "include",
            "object-locator": {
                "schema-name": "dmsst",
                "table-name": "%"
            },
            "filters": [],
            "rule-id": "1",
            "rule-name": "1"
        }
    ]
}
```

Example 스키마에서 일부 테이블 마이그레이션

다음 예제는 소스에서 이름이 지정된 dmsst 스키마에서 로 시작하는 이름을 가진 collectionTest 모든 테이블을 대상 엔드포인트로 마이그레이션합니다.

```
{
    "rules": [
        {
```

```
            "rule-type": "selection",
            "rule-action": "include",
            "object-locator": {
                "schema-name": "dmsst",
                "table-name": "collectionTest%"
            },
            "filters": [],
            "rule-id": "1",
            "rule-name": "1"
        }
    ]
}
```

Example 여러 스키마의 특정 테이블을 마이그레이션합니다.

다음 예제는 원본에 이름이 dmsst 지정되어 있는 여러 스키마의 일부 테이블을 대상 Test 엔드포인트로 마이그레이션합니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-action": "include",
            "object-locator": {
                "schema-name": "dmsst",
                "table-name": "collectionTest1"
            },
            "filters": [],
            "rule-id": "1",
            "rule-name": "1"
        },
        {
            "rule-type": "selection",
            "rule-action": "include",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "products"
            },
            "filters": [],
            "rule-id": "2",
            "rule-name": "2"
        }
    ]
```

```
}
```

AWS DMS에서 데이터 마이그레이션 관리

데이터 마이그레이션을 생성한 후 AWS DMS는 데이터 마이그레이션을 자동으로 시작하지 않습니다. 필요하다면 수동으로 데이터 마이그레이션을 시작합니다.

데이터 마이그레이션을 시작하기 전에 데이터 마이그레이션의 모든 설정을 수정할 수 있습니다. 데이터 마이그레이션을 시작한 후에는 복제 유형을 변경할 수 없습니다. 다른 복제 유형을 사용하려면 새 데이터 마이그레이션을 생성하세요.

데이터 마이그레이션을 시작하려면

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택합니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 데이터 마이그레이션의 요약 페이지가 열립니다.
4. 작업에서 시작을 선택합니다.

   그러면 AWS DMS는 데이터 마이그레이션을 위한 서버리스 환경을 생성합니다. 이 프로세스는 최대 15분이 걸립니다.

데이터 마이그레이션을 시작한 후 AWS DMS는 상태를 시작 중으로 설정합니다. AWS DMS가 데이터 마이그레이션에 사용하는 다음 상태는 데이터 마이그레이션 설정에서 선택한 복제 유형에 따라 달라집니다. 자세한 설명은 마이그레이션 상태 섹션을 참조하세요.

마이그레이션 작업을 수정하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택합니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 데이터 마이그레이션의 요약 페이지가 열립니다.
4. 수정을 선택합니다.
5. 데이터 마이그레이션을 위한 설정을 구성합니다.

> **Important**
>
> 데이터 마이그레이션을 시작한 후에는 복제 유형을 변경할 수 없습니다.

6. CloudWatchAmazon에서 데이터 마이그레이션 로그를 보려면 로그 CloudWatch 켜기 확인란을 선택하십시오.
7. 변경 내용 저장을 선택합니다.

AWS DMS는 데이터 마이그레이션을 시작한 후 중지할 수 있습니다. 중지하려면 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 그런 다음, 작업에서 중지를 선택합니다.

데이터 마이그레이션을 중지한 후 AWS DMS는 상태를 중지 중으로 설정합니다. 그런 다음, AWS DMS는 이 데이터 마이그레이션의 상태를 중지됨으로 설정합니다. AWS DMS가 데이터 마이그레이션을 중지한 후에는 데이터 마이그레이션을 수정, 재개, 재시작 또는 삭제할 수 있습니다.

데이터 복제를 계속하려면 데이터 마이그레이션 탭에서 중지한 데이터 마이그레이션을 선택합니다. 그런 다음, 작업에서 다시 시작 처리 중을 선택합니다.

데이터 로드를 다시 시작하려면 데이터 마이그레이션 탭에서 중지한 데이터 마이그레이션을 선택합니다. 그런 다음, 작업에서 다시 시작을 선택합니다. AWS DMS는 대상 데이터베이스에서 모든 데이터를 삭제하고 데이터 마이그레이션을 처음부터 시작합니다.

중지했거나 아직 시작하지 않은 데이터 마이그레이션을 삭제할 수 있습니다. 데이터 마이그레이션을 삭제하려면 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 그런 다음, 작업에서 삭제를 선택합니다. 마이그레이션 프로젝트를 삭제하려면 모든 데이터 마이그레이션을 중지하고 삭제하세요.

AWS DMS에서 데이터 마이그레이션을 모니터링

동종 데이터 마이그레이션을 시작한 후 상태 및 진행 상황을 모니터링할 수 있습니다. 수백 기가바이트에 달하는 대규모 데이터 세트의 데이터 마이그레이션은 완료하는 데 몇 시간이 걸립니다. 데이터 마이그레이션의 안정성, 가용성 및 고성능을 유지하려면 데이터 마이그레이션 진행 상태를 정기적으로 모니터링하십시오.

데이터 마이그레이션의 상태 및 진행 상황을 확인하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.

2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택하고 데이터 마이그레이션 탭으로 이동합니다.
4. 데이터 마이그레이션에 대해서는 상태 열을 참조하십시오. 이 열에 관한 자세한 내용은 마이그레이션 상태를 참조하세요.
5. 실행 중인 데이터 마이그레이션의 경우, 마이그레이션 진행 상황 열에 마이그레이션된 데이터의 비율(%)이 표시됩니다.

데이터 마이그레이션의 세부 정보를 확인하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택합니다. 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다.
4. 세부 정보 탭에서 마이그레이션 진행 상황을 확인할 수 있습니다. 특히 다음 지표를 확인할 수 있습니다.

   - 퍼블릭 IP 주소 - 데이터 마이그레이션의 퍼블릭 IP 주소입니다. 네트워크를 구성하려면 이 값이 필요합니다. 자세한 설명은 네트워크 설정 섹션을 참조하세요.
   - 로드된 테이블 - 성공적으로 로드된 테이블의 수입니다.
   - 로드 중인 테이블 - 현재 로드 중인 테이블의 수입니다.
   - 대기 중인 테이블 - 현재 로드 대기 중인 테이블의 수입니다.
   - 오류 발생 테이블 - 로드에 실패한 테이블의 수입니다.
   - 경과 시간 - 데이터 마이그레이션이 시작된 후 경과한 시간입니다.
   - CDC 지연 시간 - 소스 테이블에 변경이 발생한 시점과 AWS DMS가 대상 테이블에 변경 내용을 적용하는 시점 간에 경과한 평균 시간입니다.
   - 마이그레이션 시작됨 - 이 데이터 마이그레이션을 시작한 시간입니다.
   - 마이그레이션 중지됨 - 이 데이터 마이그레이션을 중지한 시간입니다.

5. 데이터 마이그레이션에 대한 로그 파일을 보려면 동종 데이터 마이그레이션 설정에서 CloudWatch 로그 보기를 선택합니다. 데이터 마이그레이션을 만들거나 수정할 때 CloudWatch 로그를 켤 수 있습니다. 자세한 내용은 데이터 마이그레이션 생성 및 데이터 마이그레이션 관리 섹션을 참조하세요.

Amazon CloudWatch 알람 또는 이벤트를 사용하여 데이터 마이그레이션을 면밀히 추적할 수 있습니다. 자세한 내용은 Amazon, Amazon CloudWatch 이벤트 및 Amazon CloudWatch CloudWatch 로그란 무엇입니까? 를 참조하십시오. Amazon CloudWatch 사용 설명서에서 확인할 수 있습니다. Amazon 사용 시 요금이 부과된다는 점에 CloudWatch 유의하십시오.

동종 데이터 마이그레이션의 경우 Amazon에 다음 AWS DMS 지표를 포함하십시오. CloudWatch

지표	설명
OverallCDCLatency	CDC 단계 중 전체 지연 시간. MySQL 데이터베이스의 경우, 이 지표는 원본 바이너리 로그의 변경 시점부터 이 변경 사항의 복제 시점까지 경과한 시간(초)을 나타냅니다. PostgreSQL 데이터베이스의 경우, 이 지표는 `pg_stat_subscription` 뷰의 `last_msg_receipt_time` 부터 `last_msg_send_time` 까지 경과한 시간(초)을 나타냅니다. 단위: 초
StorageConsumption	데이터 마이그레이션에 사용되는 스토리지. 단위: 바이트

AWS DMS의 동종 데이터 마이그레이션 상태

실행하는 각 데이터 마이그레이션의 경우, AWS DMS는 AWS DMS 콘솔의 상태를 표시합니다. 다음 목록은 사용 가능한 상태를 포함합니다.

- `Creating` - AWS DMS는 데이터 마이그레이션을 생성하고 있습니다.
- `Ready` - 데이터 마이그레이션을 시작할 준비가 되었습니다.
- `Starting` - AWS DMS는 데이터 마이그레이션을 위한 서버리스 환경을 생성하는 중입니다. 이 프로세스는 최대 15분이 걸립니다.
- `Load running` - AWS DMS는 전체 로드 마이그레이션을 수행하는 중입니다.
- `Load complete, replication ongoing` - AWS DMS는 전체 로드를 완료했으며 이제 지속적인 변경 사항을 복제합니다. AWS DMS는 전체 로드 및 변경 데이터 캡처(CDC) 유형의 데이터 마이그레이션에만 이 상태를 사용합니다.

- `Replication ongoing` - AWS DMS는 지속적인 변경 사항을 복제하는 중입니다. AWS DMS는 변경 데이터 캡처(CDC) 유형의 마이그레이션에만 이 상태를 사용합니다.
- `Reloading target` - AWS DMS는 데이터 마이그레이션을 다시 시작하고 지정된 마이그레이션 유형을 수행합니다.
- `Stopping` - AWS DMS가 데이터 마이그레이션을 중지하고 있습니다. 작업 메뉴에서 데이터 마이그레이션을 중지하도록 선택하면 AWS DMS는 이 상태를 설정합니다.
- `Stopped` - AWS DMS가 데이터 마이그레이션을 중지했습니다.
- `Failed` - 데이터 마이그레이션이 실패했습니다. 자세한 내용은 로그 파일을 참조하십시오.

  로그 파일을 보려면 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 그런 다음 동종 데이터 마이그레이션 CloudWatch 설정에서 로그 보기를 선택합니다.

  > **Important**
  >
  > 데이터 마이그레이션을 생성할 때 로그 설정 확인란을 선택하면 CloudWatch 로그 파일을 볼 수 있습니다.

- `Deleting` - AWS DMS는 데이터 마이그레이션을 삭제하고 있습니다. 작업 메뉴에서 데이터 마이그레이션을 삭제하도록 선택하면 AWS DMS는 이 상태를 설정합니다.

AWS DMS에서 동종 데이터 마이그레이션을 통해 MySQL 데이터베이스에서 데이터 마이그레이션

동종 데이터베이스 마이그레이션를 사용하여 자체 관리형 MySQL 데이터베이스를 RDS for MySQL 또는 RDS for Aurora MySQL로 마이그레이션할 수 있습니다. AWS DMS는 데이터 마이그레이션을 위한 서버리스 환경을 생성합니다. 다양한 유형의 데이터 마이그레이션에서 AWS DMS는 여러 가지 기본 MySQL 데이터베이스 도구를 사용합니다.

전체 로드 유형의 동종 데이터 마이그레이션에서 AWS DMS는 mydumper를 사용하여 소스 데이터베이스에서 데이터를 읽고 이 데이터를 서버리스 환경에 연결된 디스크에 저장합니다. AWS DMS는 소스 데이터를 모두 읽은 후 대상 데이터베이스의 myloader를 사용하여 데이터를 복원합니다.

전체 로드 및 변경 데이터 캡처(CDC) 유형의 동종 데이터 마이그레이션에서 AWS DMS는 mydumper를 사용하여 소스 데이터베이스에서 데이터를 읽고 이 데이터를 서버리스 환경에 연결된 디스크에 저장합니다. AWS DMS는 소스 데이터를 모두 읽은 후 대상 데이터베이스의 myloader를 사용하여 데이터를 복원합니다. AWS DMS는 전체 로드를 완료한 후 binlog 위치를 전체 로드 시작으로 설정한 상태

에서 binlog 복제를 설정합니다. 데이터 불일치를 방지하려면 작업 수를 1로 설정하여 기존 데이터의 일관된 상태를 캡처하십시오. 자세한 설명은 데이터 마이그레이션 생성 섹션을 참조하세요.

변경 데이터 캡처(CDC) 유형의 동종 데이터 마이그레이션에서 AWS DMS는 복제를 시작하기 위한 기본 CDC 시작점이 필요합니다. 기본 CDC 시작점을 제공하면 AWS DMS는 그 시작점부터 변경 내용을 캡처합니다. 또는 데이터 마이그레이션 설정에서 즉시를 선택하여 실제 데이터 마이그레이션이 시작될 때 복제 시작점을 자동으로 캡처할 수도 있습니다.

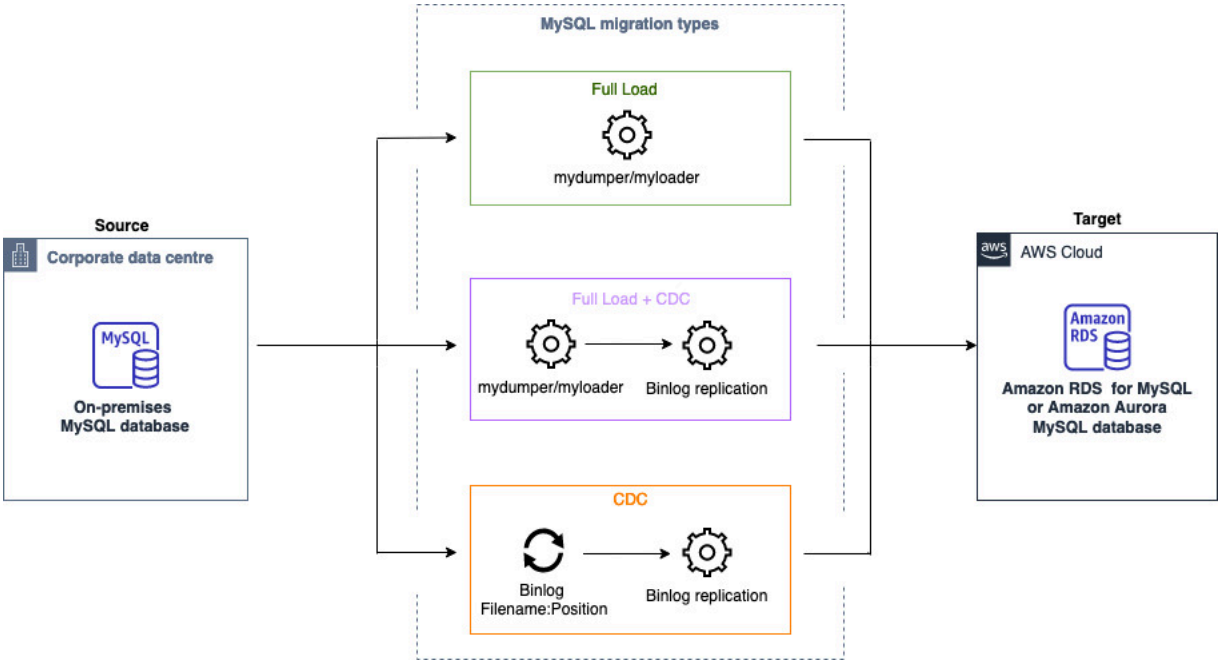
> **Note**
>
> CDC 전용 마이그레이션이 제대로 작동하려면 모든 소스 데이터베이스 스키마와 객체가 대상 데이터베이스에 이미 있어야 합니다. 그러나 대상에는 원본에 없는 객체가 있을 수 있습니다.

다음 코드 예제를 사용하여 MySQL 데이터베이스의 현재 로그 시퀀스 번호(LSN)를 가져올 수 있습니다.

```
show master status
```

이 쿼리는 binlog 파일 이름과 위치를 반환합니다. 기본 시작점의 경우, binlog 파일 이름과 위치의 조합을 사용합니다. 예: `mysql-bin-changelog.000024:373`. 이 예제에서 `mysql-bin-changelog.000024`는 binlog 파일 이름이고 `373`은 AWS DMS가 변경 캡처를 시작하는 위치입니다.

다음 다이어그램은 AWS DMS에서 동종 데이터 마이그레이션을 사용하여 MySQL 데이터베이스를 RDS for MySQL 또는 RDS for Aurora MySQL로 마이그레이션하는 프로세스를 보여줍니다.



AWS DMS에서 동종 데이터 마이그레이션을 통해 PostgreSQL 데이터베이스에서 데이터 마이그레이션

동종 데이터베이스 마이그레이션를 사용하여 자체 관리형 PostgreSQL 데이터베이스를 RDS for PostgreSQL 또는 RDS for Aurora PostgreSQL로 마이그레이션할 수 있습니다. AWS DMS는 데이터 마이그레이션을 위한 서버리스 환경을 생성합니다. 다양한 유형의 데이터 마이그레이션에서 AWS DMS는 여러 가지 기본 PostgreSQL 데이터베이스 도구를 사용합니다.

전체 로드 유형의 동종 데이터 마이그레이션의 경우, AWS DMS는 pg_dump를 사용하여 소스 데이터베이스에서 데이터를 읽고 서버리스 환경에 연결된 디스크에 저장합니다. AWS DMS는 소스 데이터를 모두 읽은 후 대상 데이터베이스의 pg_restore를 사용하여 데이터를 복원합니다.

전체 로드 및 변경 데이터 캡처(CDC) 유형의 동종 데이터 마이그레이션에서 AWS DMS는 pg_dump를 사용하여 소스 데이터베이스에서 데이터를 읽고 이 데이터를 서버리스 환경에 연결된 디스크에 저장합니다. AWS DMS는 소스 데이터를 모두 읽은 후 대상 데이터베이스의 pg_restore를 사용하여 데이터를 복원합니다. AWS DMS가 전체 로드를 완료하면 논리적 복제를 위해 게시자 및 구독자 모델로 자동 전환됩니다. 이 모델에서는 한 명 이상의 구독자가 게시자 노드에 있는 하나 이상의 발행물을 구독합니다.

변경 데이터 캡처(CDC) 유형의 동종 데이터 마이그레이션에서 AWS DMS는 복제를 시작하기 위한 기본 시작점이 필요합니다. 기본 시작점을 제공하면 AWS DMS는 그 시작점부터 변경 내용을 캡처합니다. 또는 데이터 마이그레이션 설정에서 즉시를 선택하여 실제 데이터 마이그레이션이 시작될 때 복제 시작점을 자동으로 캡처할 수도 있습니다.

> **Note**
>
> CDC 전용 마이그레이션이 제대로 작동하려면 모든 소스 데이터베이스 스키마와 객체가 대상 데이터베이스에 이미 있어야 합니다. 그러나 대상에는 원본에 없는 객체가 있을 수 있습니다.

다음 코드 예제를 사용하여 PostgreSQL 데이터베이스의 기본 시작점을 가져올 수 있습니다.

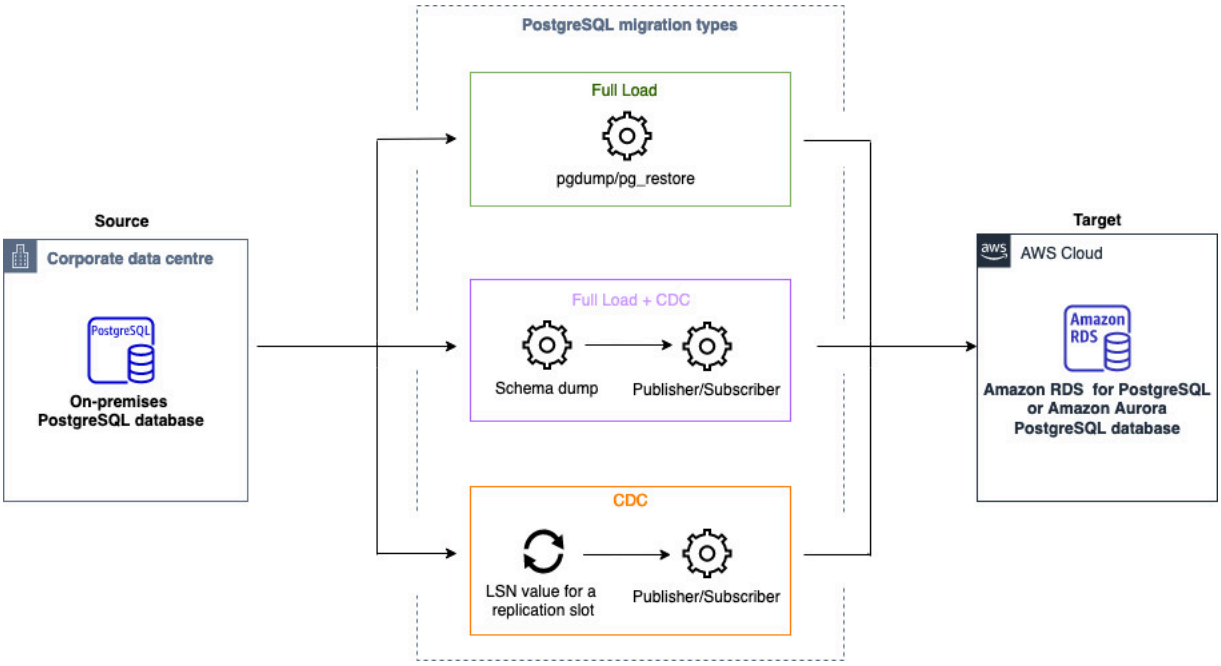
```
select confirmed_flush_lsn from pg_replication_slots where
 slot_name='migrate_to_target';
```

이 쿼리는 PostgreSQL 데이터베이스의 `pg_replication_slots` 뷰를 사용하여 로그 시퀀스 번호(LSN) 값을 캡처합니다.

AWS DMS가 PostgreSQL 동종 데이터 마이그레이션의 상태를 중지됨, 실패 또는 삭제됨으로 설정한 후에는 게시자와 복제가 제거되지 않습니다. 마이그레이션을 재개하지 않으려면 다음 명령을 사용하여 복제 슬롯과 게시자를 삭제하십시오.

```
SELECT pg_drop_replication_slot('migration_subscriber_{ARN}');
            DROP PUBLICATION publication_{ARN};
```

다음 다이어그램은 AWS DMS에서 동종 데이터 마이그레이션을 사용하여 PostgreSQL 데이터베이스를 RDS for PostgreSQL 또는 RDS for Aurora PostgreSQL로 마이그레이션하는 프로세스를 보여줍니다.



동종 데이터 마이그레이션을 통해 MongoDB 데이터베이스의 데이터를 마이그레이션합니다. AWS DMS

를 동종 데이터베이스 마이그레이션 사용하여 자체 관리형 MongoDB 데이터베이스를 Amazon DocumentDB로 마이그레이션할 수 있습니다. AWS DMS데이터 마이그레이션을 위한 서버리스 환경을 만듭니다. 다양한 유형의 데이터 마이그레이션의 경우 다른 기본 MongoDB 데이터베이스 도구를 AWS DMS 사용합니다.

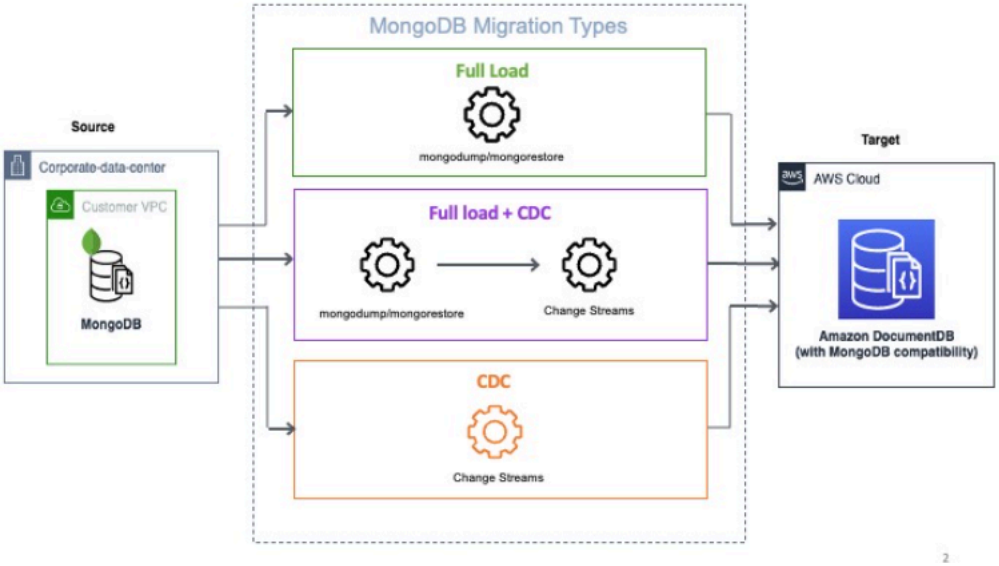
Full load 유형의 동종 데이터 마이그레이션의 경우 mongodump 를 AWS DMS 사용하여 원본 데이터베이스에서 데이터를 읽고 서버리스 환경에 연결된 디스크에 저장합니다. 원본 데이터를 모두 AWS DMS 읽은 후에는 대상 데이터베이스를 사용하여 mongorestore 데이터를 복원합니다.

전체 로드 및 변경 데이터 캡처 (CDC) 유형의 동종 데이터 마이그레이션의 경우 를 AWS DMS mongodump 사용하여 원본 데이터베이스에서 데이터를 읽고 서버리스 환경에 연결된 디스크에 저장합니다. 모든 원본 데이터를 AWS DMS 읽은 후에는 대상 데이터베이스를 사용하여 mongorestore 데이터를 복원합니다. AWS DMS가 전체 로드를 완료하면 논리적 복제를 위해 게시자 및 구독자 모델로 자동 전환됩니다. 이 모델에서는 최소 24시간 동안 변경 내용이 유지되도록 oplog의 크기를 조정하는 것이 좋습니다.

변경 데이터 캡처 (CDC) 유형의 동종 데이터 마이그레이션의 경우 데이터 마이그레이션 설정에서 실제 데이터 마이그레이션이 시작될 때 복제 시작점을 자동으로 캡처하도록 선택하십시오immediately.

> **Note**
>
> 새 컬렉션이나 이름이 바뀐 컬렉션의 경우 해당 컬렉션에 대해 동종 데이터 마이그레이션으로 새 데이터 마이그레이션 작업을 생성해야 합니다. MongoDB 호환 소스의 경우, 및 AWS DMS 작업을 지원하지 create 않습니다. rename drop collection

다음 다이어그램은 에서 동종 데이터 마이그레이션을 사용하여 MongoDB 데이터베이스를 Amazon AWS DMS DocumentDB로 마이그레이션하는 프로세스를 보여줍니다.



AWS DMS에서 동종 데이터 마이그레이션 문제 해결

AWS DMS에서 동종 데이터 마이그레이션과 관련하여 문제가 발생할 때 취해야 할 조치는 다음 목록에서 확인할 수 있습니다.

주제

- AWS DMS에서 동종 데이터 마이그레이션을 생성할 수 없습니다.
- AWS DMS에서 동종 데이터 마이그레이션을 생성할 수 없습니다.
- AWS DMS에서 데이터 마이그레이션을 실행할 때 대상 데이터베이스에 연결할 수 없습니다.
- AWS DMS는 PostgreSQL에서 뷰를 테이블로 마이그레이션합니다.

AWS DMS에서 동종 데이터 마이그레이션을 생성할 수 없습니다.

데이터 마이그레이션 생성을 선택한 후 AWS DMS가 데이터 공급자에 연결할 수 없다는 오류 메시지가 표시되면 필요한 IAM 역할을 구성했는지 확인하세요. 자세한 내용은 IAM 역할 생성 섹션을 참조하세요.

IAM 역할을 구성했는데도 이 오류 메시지가 계속 표시되면 AWS KMS 키 구성에서 이 IAM 역할을 키 사용자에 추가하세요. 자세한 내용은 AWS Key Management Service 개발자 안내서에서 키 사용자가 KMS 키를 사용하도록 허용 섹션을 참조하세요.

AWS DMS에서 동종 데이터 마이그레이션을 생성할 수 없습니다.

마이그레이션 프로젝트에서 데이터 마이그레이션을 시작할 때 `Failed` 상태가 표시되면 소스 및 대상 데이터 공급자의 버전을 확인하세요. 버전을 확인하려면 MySQL 또는 PostgreSQL 데이터베이스에서 `SELECT VERSION();` 쿼리를 실행하십시오. 지원되는 데이터베이스 버전을 사용해야 합니다.

지원되는 소스 데이터베이스의 목록은 DMS 동종 데이터 마이그레이션이 지원하는 소스 섹션을 참조하세요.

지원되는 데이터 형식의 목록은 DMS 동종 데이터 마이그레이션이 지원하는 대상 섹션을 참조하세요.

지원되지 않는 데이터베이스 버전을 사용하는 경우, 소스 또는 대상 데이터베이스를 업그레이드하고 다시 시도하십시오.

AWS DMS 콘솔에서 데이터 마이그레이션에 대한 오류 메시지를 확인하세요. 이렇게 하려면 마이그레이션 프로젝트를 열고 데이터 마이그레이션을 선택합니다. 세부 정보 탭의 일반 아래에서 마지막 실패 메시지를 확인합니다.

마지막으로 CloudWatch 로그를 분석하십시오. 이렇게 하려면 마이그레이션 프로젝트를 열고 데이터 마이그레이션을 선택합니다. 세부 정보 탭에서 CloudWatch 로그 보기를 선택합니다.

AWS DMS에서 데이터 마이그레이션을 실행할 때 대상 데이터베이스에 연결할 수 없습니다.

대상에 연결할 수 없음 오류 메시지가 표시되면 다음 작업을 수행하십시오.

1. 소스 및 대상 데이터베이스에 연결된 보안 그룹에 모든 인바운드 및 아웃바운드 트래픽에 대한 규칙이 포함되어 있는지 확인하십시오. 자세한 내용은 지속적인 데이터 복제 구성 섹션을 참조하세요.
2. 네트워크 액세스 제어 목록(ACL)과 라우팅 테이블 규칙을 확인하십시오.
3. 생성한 VPC에서 데이터베이스에 액세스할 수 있어야 합니다. VPC 보안 그룹에 퍼블릭 IP 주소를 추가하고 방화벽에서 입력 연결을 허용합니다.
4. 마이그레이션 프로젝트의 데이터 마이그레이션 탭에서 데이터 마이그레이션을 선택합니다. 세부 정보 탭의 연결 및 보안에서 퍼블릭 IP 주소를 기록해 둡니다. 그런 다음, 소스 및 대상 데이터베이스에서 데이터 마이그레이션의 퍼블릭 IP 주소를 통한 액세스를 허용하십시오.
5. 지속적인 데이터 복제를 위해서는 소스 및 대상 데이터베이스가 서로 통신할 수 있어야 합니다.

자세한 내용을 알아보려면 Amazon Virtual Private Cloud 사용 설명서의 보안 그룹을 사용하여 리소스에 대한 트래픽 제어를 참조하세요.

AWS DMS는 PostgreSQL에서 뷰를 테이블로 마이그레이션합니다.

동종 데이터 마이그레이션은 PostgreSQL에서 뷰를 뷰로 마이그레이션하는 것을 지원하지 않습니다. PostgreSQL에서 AWS DMS는 뷰를 테이블로 마이그레이션합니다.

의 데이터 제공업체, 인스턴스 프로필 및 마이그레이션 프로젝트와 협력 AWS DMS

에서 AWS Database Migration Service DMS 스키마 변환 및 동종 데이터 마이그레이션을 사용하는 경우 마이그레이션 프로젝트를 진행하게 됩니다. AWS DMS 마이그레이션 프로젝트에서는 서브넷 그룹, 인스턴스 프로파일, 데이터 공급자를 사용하게 됩니다.

서브넷은 VPC의 IP 주소 범위입니다. 복제 서브넷 그룹에는 인스턴스 프로필에서 사용할 수 있는 다양한 가용 영역의 서브넷이 포함됩니다. 복제 서브넷 그룹은 DMS 리소스이며 Amazon VPC 및 Amazon RDS가 사용하는 서브넷 그룹과는 구별된다는 점에 유의하십시오.

인스턴스 프로파일은 마이그레이션 프로젝트가 실행되는 서버리스 환경의 네트워크 및 보안 설정을 지정합니다.

데이터 공급자는 데이터 스토어 유형과 데이터베이스에 대한 위치 정보를 저장합니다. 마이그레이션 프로젝트에 데이터 공급자를 추가한 후 데이터베이스 자격 증명을 제공합니다. AWS Secrets Manager AWS DMS 이 정보를 사용하여 데이터베이스에 연결합니다.

데이터 공급자, 인스턴스 프로필 및 기타 AWS 리소스를 만든 후 마이그레이션 프로젝트를 생성할 수 있습니다. 마이그레이션 프로젝트는 인스턴스 프로필, 원본 및 대상 데이터 제공자, 보안 정보를 설명합니다 AWS Secrets Manager. 다양한 소스 및 대상 데이터 공급자에 대한 여러 마이그레이션 프로젝트를 만들 수 있습니다.

대부분의 작업은 마이그레이션 프로젝트에서 수행합니다. DMS Schema Conversion의 경우 마이그레이션 프로젝트를 사용하여 소스 데이터 공급자의 객체를 평가하고 대상 데이터베이스와 호환되는 형식으로 변환합니다. 그런 다음 변환된 코드를 대상 데이터 공급자에 적용하거나 SQL 스크립트로 저장할 수 있습니다. 동종 데이터 마이그레이션의 경우 마이그레이션 프로젝트를 사용하여 소스 데이터베이스의 데이터를 AWS 클라우드에서 동일한 유형의 대상 데이터베이스로 마이그레이션합니다.

의 AWS DMS 마이그레이션 프로젝트는 서버리스에서만 가능합니다. AWS DMS 마이그레이션 프로젝트를 위한 클라우드 리소스를 자동으로 프로비저닝합니다.

AWS DMS 에는 사용자를 위해 생성할 수 있는 최대 수의 인스턴스 프로필, 데이터 제공자 및 마이그레이션 프로젝트가 있습니다 AWS 계정. AWS DMS 서비스 할당량 AWS Database Migration Service에 대한 할당량에 관한 자세한 내용은 다음 섹션을 참조하세요.

주제

- AWS DMS 마이그레이션 프로젝트를 위한 서브넷 그룹 생성

- 에 대한 인스턴스 프로필 생성 AWS Database Migration Service
- 에서 데이터 제공자 생성 AWS Database Migration Service
- 에서 마이그레이션 프로젝트 생성 AWS Database Migration Service
- 에서 마이그레이션 프로젝트를 관리합니다. AWS Database Migration Service

AWS DMS 마이그레이션 프로젝트를 위한 서브넷 그룹 생성

인스턴스 프로파일을 생성하기 전에 인스턴스 프로파일의 서브넷 그룹을 구성하세요.

서브넷 그룹을 생성하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 서브넷을 선택한 다음 서브넷 생성을 선택합니다.
3. 이름에 서브넷 그룹의 고유한 이름을 입력합니다.
4. 설명에 서브넷 그룹에 대한 설명을 입력합니다.
5. VPC에서 두 개 이상의 가용 영역에 하나 이상의 서브넷이 있는 VPC를 선택합니다.
6. 서브넷 추가에서 서브넷 그룹에 포함할 서브넷을 선택합니다. 두 개 이상의 가용 영역에서 서브넷을 선택해야 합니다.

   Amazon RDS 데이터베이스에 연결하려면 서브넷 그룹에 퍼블릭 서브넷을 추가합니다. 온프레미스 데이터베이스에 연결하려면 서브넷 그룹에 프라이빗 서브넷을 추가합니다.
7. [서브넷 그룹 생성]을 선택합니다.

에 대한 인스턴스 프로필 생성 AWS Database Migration Service

AWS DMS 콘솔에서 여러 인스턴스 프로필을 만들 수 있습니다. AWS DMS에서 생성하는 마이그레이션 프로젝트마다 사용할 인스턴스 프로파일을 하나 선택해야 합니다.

인스턴스 프로파일 생성

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 인스턴스 프로파일을 선택합니다.
3. 인스턴스 프로파일 생성을 선택합니다.

4. 인스턴스 프로파일 생성 페이지에서 인스턴스 프로파일의 이름을 설명하는 값을 입력합니다.
5. 네트워크 유형에서, IPv4 및 IPv6 주소 지정을 지원하는 인스턴스 프로파일을 생성하려면 이중 스택 모드를 선택합니다. IPv4 주소 지정만 지원하는 인스턴스 프로파일을 생성하려면 기본 옵션을 그대로 유지합니다.
6. 다음으로, Virtual Private Cloud(VPC)를 선택하여 선택한 네트워크 유형의 인스턴스를 실행합니다. 그런 다음 인스턴스 프로파일의 서브넷 그룹 및 VPC 보안 그룹을 선택합니다.

   Amazon RDS 데이터베이스에 연결하려면 퍼블릭 서브넷이 포함된 서브넷 그룹을 생성합니다. 온프레미스 데이터베이스에 연결하려면 프라이빗 서브넷이 포함된 서브넷 그룹을 생성하십시오. NAT 게이트웨이의 퍼블릭 IP 주소를 사용하여 원본 온프레미스 데이터베이스에 액세스할 AWS DMS 수 있도록 네트워크를 구성했는지 확인하십시오. 자세한 정보는 Amazon VPC를 기반으로 VPC 생성을 참조하세요.
7. (선택 사항) DMS Schema Conversion용 마이그레이션 프로젝트를 생성한 다음, 스키마 변환 설정 - 선택 사항에서 마이그레이션 프로젝트의 정보를 저장할 Amazon S3 버킷을 선택합니다. 그런 다음 이 Amazon S3 버킷에 대한 액세스를 제공하는 AWS Identity and Access Management (IAM) 역할을 선택합니다. 자세한 정보는 Amazon S3 버킷 생성을 참조하세요.
8. 인스턴스 프로파일 생성을 선택합니다.

생성한 인스턴스 프로파일은 수정하거나 삭제할 수 있습니다.

인스턴스 프로파일을 수정하려면

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 인스턴스 프로파일을 선택합니다. 인스턴스 프로파일 페이지가 열립니다.
3. 인스턴스 프로파일을 선택한 다음 수정을 선택합니다.
4. 인스턴스 프로파일의 이름을 업데이트하고, VPC 또는 Amazon S3 버킷 설정을 편집합니다.
5. 변경 사항 저장을 선택합니다.

인스턴스 프로파일을 삭제하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 인스턴스 프로파일을 선택합니다. 인스턴스 프로파일 페이지가 열립니다.
3. 인스턴스 프로파일을 선택한 다음 삭제를 선택합니다.

4. 삭제를 선택하여 선택을 확인합니다.

에서 데이터 제공자 생성 AWS Database Migration Service

데이터 공급자를 생성하여 AWS DMS 마이그레이션 프로젝트에 사용할 수 있습니다. 데이터베이스 공급자는 온프레미스 또는 Amazon EC2 인스턴스에서 실행되는 자체 관리형 엔진일 수 있습니다. 또한 데이터 공급자는 Amazon Relational DatabaseService(Amazon RDS) 또는 Amazon Aurora와 같은 완전관리형 엔진일 수 있습니다.

각 데이터베이스에 대해 단일 데이터 공급자를 생성할 수 있습니다. 여러 마이그레이션 프로젝트에서 하나의 데이터 공급자를 사용할 수 있습니다.

마이그레이션 프로젝트를 생성하려면 데이터 공급자가 두 개 이상 생성되어 있어야 합니다. 데이터 공급자 중 하나는 AWS 서비스에 있어야 합니다. AWS DMS 를 사용하여 스키마를 변환하거나 데이터를 온프레미스 데이터베이스로 마이그레이션할 수 없습니다.

다음 절차는 AWS DMS 콘솔 마법사에서 데이터 공급자를 만드는 방법을 보여줍니다.

데이터 공급자를 생성하려면

1. 에 로그인한 다음 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다. AWS Management Console
2. 데이터 공급자를 선택합니다. 데이터 공급자 페이지가 열립니다.
3. 데이터 공급자 생성을 선택합니다. 다음 표는 설정에 대한 설명입니다.

옵션	작업
구성	수동으로 데이터 공급자에 대한 정보를 입력할지 아니면 Amazon RDS DB 인스턴스를 사용할지 선택합니다.
이름	데이터 공급자의 이름을 입력합니다. 데이터 공급자를 쉽게 식별할 수 있도록 데이터 공급자에 고유한 이름을 사용해야 합니다.
엔진 유형	데이터 공급자의 데이터베이스 엔진 유형을 선택합니다.
[서버 이름]	데이터베이스 서버의 DNS(Domain Name Service) 이름 또는 IP 주소를 입력합니다. 동종 복제에 사용되는 데이터 공급자의

옵션	작업
	서버 이름은 영숫자로 시작해야 하며 영숫자, 하이픈(-), 마침표(.) 또는 밑줄(_)만 포함할 수 있습니다.
포트	데이터베이스 서버에 연결하는 데 사용되는 포트를 입력합니다.
서비스 ID(SID) 또는 서비스 이름	Oracle SID(SID)를 입력합니다. Oracle SID를 확인하려면 Oracle 데이터베이스에 다음 쿼리를 제출합니다. `SELECT sys_context('userenv','instance_name') AS SID FROM dual;`
데이터베이스 이름	이 데이터 공급자의 데이터베이스 이름을 입력합니다. 동종 복제에 사용되는 데이터 공급자의 데이터베이스 이름은 최대 63자일 수 있으며 스페이스를 포함할 수 없습니다.
Secure Socket Layer(SSL) 모드	이 데이터 공급자에서 연결 암호화를 활성화하려면 SSL 모드를 선택합니다. 선택하는 모드에 따라 인증서와 서버 인증서 정보를 제공해야 할 수도 있습니다. 자세한 내용은 SSL 사용을 참조하십시오 AWS Database Migration Service.
인증 모드	MongoDB 소스의 경우 엔드포인트 연결을 인증하는 데 AWS DMS 사용하는 인증 모드입니다.
인증 소스	MongoDB 소스의 경우 인증을 위한 자격 증명을 검증하는 데 사용할 MongoDB 데이터베이스의 이름입니다.
인증 메커니즘	MongoDB 소스의 경우 MongoDB가 비밀번호를 암호화하는 데 사용하는 인증 방법입니다.

4. 데이터 공급자 생성을 선택합니다.

데이터 공급자를 생성한 후에는 AWS Secrets Manager에서 데이터베이스 연결 보안 인증 정보를 추가해야 합니다.

에서 마이그레이션 프로젝트 생성 AWS Database Migration Service

에서 AWS DMS마이그레이션 프로젝트를 만들기 전에 다음 리소스를 만들어야 합니다.

- 소스 및 대상 데이터베이스를 설명하는 데이터 공급자
- 데이터베이스 자격 증명이 저장된 시크릿 AWS Secrets Manager
- Secrets Manager에 대한 액세스를 제공하는 AWS Identity and Access Management (IAM) 역할
- 네트워크 및 보안 설정이 포함된 인스턴스 프로파일

마이그레이션 프로젝트를 생성하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트 생성을 선택합니다. 다음 표는 설정에 대한 설명입니다.

옵션	작업
이름	마이그레이션 프로젝트의 이름을 입력합니다. 마이그레이션 프로젝트를 쉽게 식별할 수 있도록 데이터 마이그레이션에 고유한 이름을 사용해야 합니다.
인스턴스 프로파일	마이그레이션 프로젝트에 사용할 인스턴스 프로파일을 선택합니다.
소스	찾아보기를 선택한 다음 소스 데이터 공급자를 선택합니다.
비밀 ID	Secrets Manager에서 소스 데이터베이스 보안 인증 정보가 저장된 비밀의 Amazon 리소스 이름(ARN)을 선택합니다.
IAM 역할	Secrets Manager에서 소스 데이터베이스 보안 인증 정보에 대한 액세스를 제공할 IAM 역할을 선택합니다.
대상	찾아보기를 선택한 다음 소스 데이터 공급자를 선택합니다.
비밀 ID	Secrets Manager에서 대상 데이터베이스 보안 인증 정보가 저장된 비밀의 ARN을 선택합니다.

옵션	작업
IAM 역할	Secrets Manager에서 대상 데이터베이스 보안 인증 정보에 대한 액세스를 제공할 IAM 역할을 선택합니다.
변환 규칙	(선택 사항) DMS Schema Conversion용 마이그레이션 프로젝트를 생성하는 경우 변환 규칙 추가를 선택하여 변환 규칙을 설정합니다. 변환 규칙을 사용하면 지정한 규칙에 따라 객체 이름을 변경할 수 있습니다. 자세한 정보는 변환 규칙 설정을 참조하세요.

4. 마이그레이션 프로젝트 생성을 선택합니다.

마이그레이션 프로젝트를 AWS DMS 만든 후 이 프로젝트를 DMS 스키마 변환 또는 동종 데이터 마이그레이션에서 사용할 수 있습니다. 마이그레이션 프로젝트 작업을 시작하려면 마이그레이션 프로젝트 페이지의 목록에서 프로젝트를 선택합니다.

에서 마이그레이션 프로젝트를 관리합니다. AWS Database Migration Service

생성한 마이그레이션 프로젝트는 수정하거나 삭제할 수 있습니다. 예를 들어 소스 또는 대상 데이터 공급자를 변경하려면 마이그레이션 프로젝트를 수정합니다.

스키마 변환 또는 데이터 마이그레이션 작업을 종료한 후에만 마이그레이션 프로젝트를 수정하거나 삭제할 수 있습니다. 이렇게 하려면 목록에서 마이그레이션 프로젝트를 선택하고 스키마 변환 또는 데이터 마이그레이션을 선택합니다. 그런 다음 DMS Schema Conversion에 대해 스키마 변환 종료를 선택하고 선택을 확인합니다. 동종 데이터 마이그레이션의 경우 데이터 마이그레이션을 선택한 다음 작업 메뉴에서 중지를 선택합니다. 마이그레이션 프로젝트를 편집한 후에 스키마 변환을 시작하거나 데이터 마이그레이션을 다시 시작할 수 있습니다.

마이그레이션 프로젝트를 수정하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택한 다음 수정을 선택합니다.

4. 프로젝트의 이름을 업데이트하거나, 인스턴스 프로파일을 편집하거나, 소스 및 대상 데이터 공급자를 변경합니다. 선택적으로, 변환 중에 객체 이름을 변경하는 마이그레이션 규칙을 추가하거나 편집할 수도 있습니다.
5. 변경 사항 저장를 선택합니다.

마이그레이션 프로젝트를 삭제하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 마이그레이션 프로젝트를 선택합니다. 마이그레이션 프로젝트 페이지가 열립니다.
3. 마이그레이션 프로젝트를 선택한 다음 삭제를 선택합니다.
4. 삭제를 선택하여 선택을 확인합니다.

AWS Database Migration Service의 모범 사례

AWS Database Migration Service(AWS DMS)를 가장 효과적으로 사용하려면 데이터를 마이그레이션하는 가장 효율적인 방법에 대한 이 단원의 권장 사항을 참조하세요.

주제

- AWS Database Migration Service 마이그레이션 계획
- 스키마 변환
- AWS DMS 공개 문서 검토
- 개념 증명 실행
- AWS DMS 마이그레이션 성능 개선
- 자체 온프레미스 이름 서버 사용
- 대용량 이진 객체(LOB) 마이그레이션
- 행 필터링을 사용하여 큰 테이블 마이그레이션 시 성능 향상
- 지속적 복제
- 소스 데이터베이스의 로드 감소
- 대상 데이터베이스에서 병목 현상 감소
- 마이그레이션 중에 데이터 유효성 검사 사용
- 지표를 사용하여 AWS DMS 태스크 모니터링
- 이벤트 및 알림
- 마이그레이션 문제 해결을 위해 작업 로그 사용
- Time Travel을 통한 복제 태스크 문제 해결
- Oracle 대상에서 사용자 및 스키마 변경
- Oracle 대상에 대한 테이블 및 인덱스 테이블스페이스 변경
- 복제 인스턴스 버전 업그레이드
- 마이그레이션 비용 이해

AWS Database Migration Service 마이그레이션 계획

AWS Database Migration Service를 사용하는 데이터베이스 마이그레이션을 계획할 때 다음 사항을 고려해야 합니다.

- 소스 및 대상 데이터베이스를 AWS DMS 복제 인스턴스에 연결하려면 네트워크를 구성해야 합니다. 이 작업은 복제 인스턴스와 동일한 Virtual Private Cloud(VPC)에 있는 AWS 리소스 두 개를 연결하는 것만큼 간단할 수도 있고, 가상 프라이빗 네트워크(VPN)을 통해 온프레미스 데이터베이스를 Amazon RDS DB 인스턴스에 연결하는 등 더 복잡한 구성까지 다양할 수 있습니다. 자세한 설명은 데이터베이스 마이그레이션을 위한 네트워크 구성 섹션을 참조하세요.
- 소스 및 대상 엔드포인트 - 대상 데이터베이스로 마이그레이션해야 하는 소스 데이터베이스의 정보와 테이블을 알고 있어야 합니다. AWS DMS는 테이블 및 프라이머리 키 생성을 포함하여 기본 스키마 마이그레이션을 지원합니다. 하지만 AWS DMS는 대상 데이터베이스에 보조 인덱스, 외래 키, 사용자 계정 등을 자동으로 생성하지는 않습니다. 소스 및 대상 데이터베이스 엔진에 따라 소스 또는 대상 데이터베이스에 추가 로깅을 설정하거나 기타 설정을 수정해야 할 수 있습니다. 자세한 내용은 데이터 마이그레이션용 소스 및 마이그레이션에 적합한 대상 섹션을 참조하세요.
- 스키마 및 코드 마이그레이션 - AWS DMS는 스키마 또는 코드 변환을 수행하지 않습니다. Oracle SQL Developer, MySQL Workbench, pgAdmin III와 같은 도구를 사용하여 스키마를 변환할 수 있습니다. 기존 스키마를 다른 데이터베이스 엔진으로 변환하려면 AWS Schema Conversion Tool(AWS SCT)를 사용할 수 있습니다. 대상 스키마를 생성하고 전체 스키마(테이블, 인덱스, 보기 등)을 생성할 수 있습니다. 또한 PL/SQL 또는 TSQL을 PgSQL 및 기타 형식으로 변환하는 도구를 사용할 수도 있습니다. AWS SCT에 대한 자세한 내용은 AWS SCT 사용 설명서를 참조하세요.
- 지원되지 않는 데이터 형식 - 소스 데이터 형식을 대상 데이터베이스의 해당 데이터 형식으로 변환해야 합니다. 지원되는 데이터 형식에 대한 자세한 내용은 해당 데이터 스토어 설명서의 소스 또는 대상 섹션을 참조하세요.
- 진단 지원 스크립트 결과 - 마이그레이션을 계획할 때는 진단 지원 스크립트를 실행하는 것이 좋습니다. 이러한 스크립트의 결과를 통해 잠재적 마이그레이션 실패에 대한 사전 정보를 찾을 수 있습니다.

  데이터베이스에 사용할 수 있는 지원 스크립트가 있는 경우 다음 단원의 해당 스크립트 주제에 있는 링크를 사용하여 다운로드하세요. 스크립트를 확인하고 검토한 후 로컬 환경에서 스크립트 주제에 설명된 절차에 따라 스크립트를 실행할 수 있습니다. 스크립트 실행이 완료되면 결과를 검토할 수 있습니다. 문제 해결의 첫 단계로 이러한 스크립트를 실행하는 것이 좋습니다. 결과는 AWS Support 팀과 함께 작업할 때 유용할 수 있습니다. 자세한 설명은 AWS DMS에서 진단 지원 스크립트 작업 섹션을 참조하세요.
- 마이그레이션 전 평가 - 마이그레이션 전 평가는 데이터베이스 마이그레이션 태스크의 특정 구성 요소를 평가하여 마이그레이션 태스크가 예상대로 실행되는 데 방해가 될 수 있는 문제를 식별하는 데 도움이 됩니다. 이 평가를 사용하면 새 태스크 또는 수정된 태스크를 실행하기 전에 잠재적 문제를 식별할 수 있습니다. 마이그레이션 전 평가 작업에 대한 자세한 내용은 작업에 대한 마이그레이션 전 평가 활성화 및 활용 섹션을 참조하세요.

스키마 변환

AWS DMS에서는 스키마 또는 코드 변환을 수행하지 않습니다. 기존 스키마를 다른 데이터베이스 엔진으로 변환하려는 경우 AWS SCT를 사용할 수 있습니다. AWS SCT는 소스 객체, 테이블, 인덱스, 뷰, 트리거 및 기타 시스템 객체를 대상 데이터 정의 언어(DDL) 형식으로 변환합니다. 또한 AWS SCT를 PL/SQL 또는 TSQL과 같은 대부분의 애플리케이션 코드를 동등한 대상 언어로 변환하는 데 사용할 수 있습니다.

AWS SCT는 AWS에서 무료로 다운로드할 수 있습니다. AWS SCT에 대한 자세한 내용은 AWS SCT 사용 설명서를 참조하세요.

소스 엔드포인트와 대상 엔드포인트가 동일한 데이터베이스 엔진에 있는 경우 Oracle SQL Developer, MySQL Workbench PgAdmin 또는 4와 같은 도구를 사용하여 스키마를 이동할 수 있습니다.

AWS DMS 공개 문서 검토

첫 번째 마이그레이션 전에 소스 및 대상 엔드포인트의 AWS DMS 공개 설명서 페이지를 살펴보는 것이 좋습니다. 이 설명서는 마이그레이션을 시작하기 전에 마이그레이션을 위한 사전 요구 사항을 식별하고 현재 제한 사항을 이해하는 데 도움이 될 수 있습니다. 자세한 설명은 AWS DMS 엔드포인트 작업 섹션을 참조하세요.

마이그레이션하는 동안 공개 설명서를 통해 AWS DMS 관련 문제를 해결할 수 있습니다. 설명서의 문제 해결 페이지는 AWS DMS 및 선택한 엔드포인트 데이터베이스를 모두 사용하여 일반적인 문제를 해결하는 데 도움이 될 수 있습니다. 자세한 설명은 AWS Database Migration Service에서 마이그레이션 작업 문제 해결 섹션을 참조하세요.

개념 증명 실행

데이터베이스 마이그레이션 초기 단계에서 환경 문제를 발견하는 데 도움이 되도록 소규모 테스트 마이그레이션을 실행하는 것이 좋습니다. 이렇게 하면 마이그레이션 일정을 보다 현실적으로 설정하는 데도 도움이 될 수 있습니다. 또한 AWS DMS가 네트워크를 통한 데이터베이스 처리량을 처리할 수 있는지 여부를 측정하기 위해 전체 규모 테스트 마이그레이션을 실행해야 할 수도 있습니다. 이 동안에는 초기 전체 로드 및 지속적 복제를 벤치마킹하고 최적화하는 것이 좋습니다. 이렇게 하면 네트워크 지연 시간을 파악하고 전반적인 성능을 측정하는 데 도움이 될 수 있습니다.

또한 이 시점에서 다음을 포함하여 데이터 프로파일 및 데이터베이스 규모를 파악할 수 있습니다.

- 대형, 중형 및 소형 테이블의 수.
- AWS DMS가 데이터 형식 및 문자 세트 변환을 처리하는 방법.

- 대형 객체(LOB) 열이 있는 테이블의 수
- 테스트 마이그레이션을 실행하는 데 걸리는 시간.

AWS DMS 마이그레이션 성능 개선

AWS DMS 마이그레이션 성능에 영향을 주는 요인은 많습니다.

- 소스의 리소스 가용성.
- 가용 네트워크 처리량.
- 복제 서버의 리소스 용량.
- 대상이 변경 사항을 수집하는 능력.
- 소스 데이터의 형식 및 분포.
- 마이그레이션할 객체의 수.

아래에 언급된 모범 사례 중 일부 또는 전부를 사용하여 성능을 개선할 수 있습니다. 이러한 방법을 사용할 수 있는지 여부는 구체적인 사용 사례에 따라 다릅니다. 다음과 같은 몇 가지 제한 사항을 확인할 수 있습니다.

적절한 복제 서버를 프로비저닝

AWS DMS는 Amazon EC2 인스턴스에서 실행되는 관리형 서비스입니다. 이 서비스는 소스 데이터베이스에 연결하고, 소스 데이터를 읽고, 대상 데이터베이스에서 사용할 수 있도록 데이터 형식을 지정하고, 대상 데이터베이스에 데이터를 로드합니다.

이 절차 대다수는 메모리에서 진행됩니다. 그렇지만, 대규모 트랜잭션은 디스크에서 일부 버퍼링이 필요할 수 있습니다. 캐시된 트랜잭션과 로그 파일도 디스크에 기록됩니다. 다음 단원에서 복제 서버를 선택할 때 고려해야 할 사항을 확인할 수 있습니다.

CPU

AWS DMS는 이기종 마이그레이션용으로 설계되었지만 동종 마이그레이션도 지원합니다. 동종 마이그레이션을 수행하려면 먼저 각 소스 데이터 형식을 동등한 AWS DMS 데이터 형식으로 변환해야 합니다. 그런 다음 각 AWS DMS 형식 데이터를 대상 데이터 형식으로 변환합니다. AWS DMS 사용 설명서에서 각 데이터베이스 엔진의 이러한 변환에 대한 참조를 찾을 수 있습니다.

AWS DMS가 이러한 변환을 최적으로 수행하려면 변환을 실행할 때 CPU를 사용할 수 있어야 합니다. CPU에 과부하가 걸리고 CPU 리소스가 충분하지 않으면 마이그레이션 속도가 느려지고 다른 부작용도 발생할 수 있습니다.

복제 인스턴스 클래스

소형 인스턴스 클래스 중에는 서비스를 테스트하거나 소규모 마이그레이션을 하기에 충분한 것이 있습니다. 마이그레이션에 대량의 테이블이 포함되어 있는 경우 또는 여러 동시 복제 태스크를 실행해야 하는 경우, 대형 인스턴스 중 하나를 사용하는 것이 좋습니다. 서비스가 상당한 양의 메모리 및 CPU를 소비하기 때문에 더 큰 용량의 인스턴스를 사용하는 것이 좋습니다.

T2 유형 인스턴스는 중간 정도의 기본 성능을 발휘하면서 워크로드의 필요에 따라 성능을 크게 높이는 버스트 기능을 제공하도록 설계되었습니다. 이러한 인스턴스는 CPU의 최대 성능을 자주 또는 일관적으로 사용하지 않지만 가끔 순간적인 버스트가 필요한 워크로드에 적합합니다. T2 인스턴스는 웹 서버, 개발자 환경 및 소규모 데이터베이스와 같은 범용 워크로드에 매우 적합합니다. T2 인스턴스 유형을 사용하면서 마이그레이션 속도가 느린 문제를 해결하는 경우 CPU 사용률 호스트 지표를 확인하세요. 이 인스턴스 유형의 기준을 초과하여 버스팅하고 있는지 확인할 수 있습니다.

C4 인스턴스 클래스는 컴퓨터 집약적인 워크로드에 최고 수준의 프로세서 성능을 제공하도록 설계되었습니다. PPS(초당 패킷 수) 성능을 크게 높이고 네트워크 지터를 낮추며 네트워크 지연 시간을 줄였습니다. AWS DMS는 특히 Oracle에서 PostgreSQL로 마이그레이션하는 것과 같은 이기종 마이그레이션 및 복제를 수행할 때 많은 양의 CPU를 소비할 수 있습니다. C4 인스턴스는 이러한 상황에서 훌륭한 선택이 될 수 있습니다.

R4 인스턴스 클래스는 메모리 집약적 워크로드에 최적화된 메모리입니다. AWS DMS를 사용한 고처리량 트랜잭션 시스템의 지속적 마이그레이션 또는 복제 작업에서는 때로 많은 양의 CPU 및 메모리를 소비할 수 있습니다. R4 인스턴스에는 vCPU당 더 많은 메모리가 포함되어 있습니다.

AWS DMS의 R5 및 C5 인스턴스 클래스 지원

R5 인스턴스 클래스는 메모리에서 대규모 데이터를 처리하는 워크로드에 대해 빠른 성능을 제공하도록 설계된 메모리 최적화 인스턴스입니다. AWS DMS를 사용한 고처리량 트랜잭션 시스템의 지속적 마이그레이션 또는 복제 작업에서는 때로 많은 양의 CPU 및 메모리를 소비할 수 있습니다. R5 인스턴스는 R4보다 vCPU당 5%의 추가 메모리를 제공하며 가장 큰 크기는 768GiB의 메모리를 제공합니다. 또한 R5 인스턴스는 R4에 비해 GiB당 가격이 10% 향상되고 CPU 성능이 최대 20% 향상되었습니다.

C5 인스턴스 클래스는 컴퓨팅 집약적 워크로드에 최적화되어 있으며 컴퓨팅 비율당 저렴한 가격으로 비용 효율적인 고성능을 제공합니다. 네트워크 성능이 크게 향상되었습니다. Elastic Network Adapter(ENA)는 최대 25Gbps의 네트워크 대역폭과 최대 14Gbps의 전용 대역폭을 갖춘 C5 인스턴스를 Amazon EBS에 제공합니다. AWS DMS는 특히 Oracle에서 PostgreSQL로 마이그레이션하는 것과 같은 이기종 마이그레이션 및 복제를 수행할 때 CPU를 많이 사용할 수 있습니다. C5 인스턴스는 이러한 상황에서 훌륭한 선택이 될 수 있습니다.

스토리지

인스턴스 클래스에 따라 복제 인스턴스에는 50GB 또는 100GB의 데이터 스토리지가 포함되어 있습니다. 이 스토리지는 로그 파일과 로드 중에 수집되는 캐시된 변경 사항을 저장하는 데 사용됩니다. 소스 시스템이 사용 빈도가 높거나 대규모 트랜잭션을 처리하는 경우 스토리지를 늘려야 할 수 있습니다. 복제 서버에서 여러 태스크를 실행하는 경우 스토리지를 늘려야 할 수도 있습니다. 하지만 일반적으로 기본 용량으로도 충분합니다.

AWS DMS의 모든 스토리지 볼륨은 GP2 또는 범용 SSD(Solid-State Drive) 입니다. GP2 볼륨의 기본 성능은 3IOPS(초당 I/O 작업)이며 크레딧 기준으로 최대 3,000IOPS까지 버스트할 수 있습니다. 경험상 복제 인스턴스의 `ReadIOPS` 및 `WriteIOPS` 지표를 확인하는 것이 좋습니다. 이러한 값의 합계가 해당 볼륨의 기본 성능을 초과하지 않는지 확인하세요.

다중 AZ

다중 AZ 인스턴스를 선택하면 스토리지 장애로부터 마이그레이션을 보호할 수 있습니다. 대부분의 마이그레이션은 일시적이며 장기간 실행되지 않습니다. 지속적 복제 목적으로 AWS DMS를 사용하는 경우 다중 AZ 인스턴스를 선택하면 스토리지 문제 발생 시 가용성을 개선할 수 있습니다.

단일 AZ 또는 다중 AZ 복제 인스턴스를 사용하여 전체 로드를 수행하는 동안 장애 조치 또는 호스트 교체가 발생하는 경우 전체 로드 태스크는 실패할 것으로 예상됩니다. 완료되지 않았거나 오류 상태인 나머지 테이블의 경우 실패 지점부터 태스크를 다시 시작할 수 있습니다.

여러 개의 테이블을 병렬로 로드

기본적으로 AWS DMS는 한 번에 8개의 테이블을 로드합니다. dms.c4.xlarge 이상 인스턴스와 같은 매우 큰 복제 서버를 사용하는 경우 이 값을 약간 높이면 성능이 어느 정도 향상될 수 있습니다. 하지만 어느 시점에서는 이 병렬 처리를 높이면 성능이 저하됩니다. 복제 서버가 dms.t2.medium과 같이 비교적 소형인 경우에는 병렬로 로드되는 테이블 수를 줄이는 것이 좋습니다.

AWS Management Console에서 이 값을 변경하려면 콘솔에서 태스크를 선택하고 태스크를 생성 또는 수정하도록 선택한 다음 고급 설정을 선택합니다. 튜닝 설정에서 병렬로 로드할 최대 테이블 수 옵션을 변경합니다.

AWS CLI를 사용하여 이 값을 변경하려면 `TaskSettings`에서 `MaxFullLoadSubTasks` 파라미터를 변경합니다.

병렬 전체 로드 사용

Oracle, Microsoft SQL Server, MySQL, Sybase 및 IBM Db2 LUW 소스에서 파티션 및 하위 파티션을 기반으로 병렬 로드를 사용할 수 있습니다. 이렇게 하면 전체 로드 기간을 전반적으로 개선할 수 있습니다. 또한 AWS DMS 마이그레이션 태스크를 실행하는 동안 대형 테이블 또는 파티셔닝된 테

이블의 마이그레이션 속도를 높일 수 있습니다. 이렇게 하려면 테이블을 세그먼트로 분할하고 동일한 마이그레이션 태스크에서 세그먼트를 병렬로 로드하세요.

병렬 로드를 사용하려면 `parallel-load` 옵션을 사용하여 `table-settings` 유형의 테이블 매핑 규칙을 생성합니다. `table-settings` 규칙 내에서 병렬로 로드하려는 테이블의 선택 기준을 지정할 수 있습니다. 선택 기준을 지정하려면 `parallel-load`의 `type` 요소를 다음 중 하나로 설정합니다.

- `partitions-auto`
- `subpartitions-auto`
- `partitions-list`
- `ranges`
- `none`

이러한 설정에 대한 자세한 내용은 테이블 및 컬렉션 설정 규칙과 작업 섹션을 참조하세요.

인덱스, 트리거 및 참조 무결성 제약 조건 사용

인덱스, 트리거, 참조 무결성 제약 조건은 마이그레이션 성능에 영향을 미치고 마이그레이션 실패를 초래할 수 있습니다. 이러한 제약 조건이 마이그레이션에 미치는 영향은 복제 태스크가 전체 로드 태스크인지 지속적 복제(변경 데이터 캡처(CDC_) 태스크인지에 따라 달라집니다.

전체 로드 태스크의 경우 프라이머리 키 인덱스, 보조 인덱스, 참조 무결성 제약 조건 및 데이터 조작 언어(DML) 트리거를 삭제하는 것이 좋습니다. 또는 전체 로드 태스크가 완료될 때까지 생성을 연기할 수도 있습니다. 전체 로드 태스크 중에는 인덱스가 필요하지 않으며 인덱스가 있는 경우 유지 관리 오버헤드가 발생합니다. 전체 로드 태스크는 테이블 그룹을 한 번에 로드하므로 참조 무결성 제약 조건이 위반됩니다. 마찬가지로 삽입, 업데이트 및 삭제 트리거로 인해 오류가 발생할 수 있습니다(예: 이전에 대량으로 로드된 테이블에 대해 행 삽입이 트리거되는 경우). 다른 유형의 트리거도 추가 처리로 인해 성능에 영향을 줍니다.

데이터 볼륨이 비교적 작고 추가 마이그레이션 시간이 문제가 되지 않는 경우 전체 로드 태스크 전에 프라이머리 키와 세컨더리 인덱스를 구축할 수 있습니다. 참조 무결성 제약 조건 및 트리거는 항상 비활성화하세요.

전체 로드 + CDC 태스크의 경우 CDC 단계 전에 보조 색인을 추가하는 것이 좋습니다. AWS DMS는 논리적 복제를 사용하므로 전체 테이블 스캔을 방지하기 위해 DML 작업을 지원하는 세컨더리 인덱스를 배치해야 합니다. CDC 단계가 시작되기 전에 복제 태스크를 일시 중지하여 인덱스를 작성하고 참조 무결성 제약 조건을 생성한 다음 태스크를 다시 시작할 수 있습니다.

트리거는 전환 직전에 활성화해야 합니다.

백업 및 트랜잭션 로깅을 비활성화하세요.

Amazon RDS 데이터베이스로 마이그레이션하려면 전환 준비가 될 때까지 대상에서 백업과 다중 AZ를 비활성화하는 것이 좋습니다. 마찬가지로, Amazon RDS가 아닌 다른 시스템으로 마이그레이션할 때는 일반적으로 전환 이후까지 대상에서의 로깅을 비활성화하는 것이 좋습니다.

여러 개의 태스크 사용

경우에 따라 단일 마이그레이션에 여러 태스크를 사용하면 성능이 향상될 수 있습니다. 일반적인 트랜잭션에 참여하지 않는 테이블 집합이 있는 경우 마이그레이션을 여러 태스크로 나눌 수 있습니다. 트랜잭션 일관성은 태스크 내에서 유지되므로 별도의 태스크에 있는 테이블이 일반적인 트랜잭션에 참여하지 않도록 하는 것이 중요합니다. 또한 각 태스크는 독립적으로 트랜잭션 스트림을 읽으므로 소스 데이터베이스에 너무 많은 부하가 걸리지 않도록 주의해야 합니다.

여러 태스크를 사용하여 별도의 복제 스트림을 생성할 수 있습니다. 이렇게 하면 소스에서의 읽기, 복제 인스턴스의 프로세스, 대상 데이터베이스에 대한 쓰기를 병렬화할 수 있습니다.

변경 처리 최적화

기본적으로 AWS DMS는 변경 사항을 트랜잭션 모드에서 처리하므로 트랜잭션 무결성이 유지됩니다. 트랜잭션 무결성이 일시적으로 저하될 여지가 있다면 배치 최적화 적용 옵션을 대신 사용할 수 있습니다. 이 옵션은 효율성을 위해 트랜잭션을 효율적으로 그룹화하고 일괄적으로 적용합니다. 배치 최적화 적용 옵션을 사용하면 거의 항상 참조 무결성 제약 조건을 위반합니다. 따라서 마이그레이션 프로세스 중에 이러한 제약 조건을 해제한 다음, 전환 프로세스의 일부로 다시 설정하는 것이 좋습니다.

자체 온프레미스 이름 서버 사용

일반적으로 AWS DMS 복제 인스턴스는 Amazon EC2 인스턴스의 도메인 이름 시스템(DNS) 해석기를 사용하여 도메인 엔드포인트를 확인합니다. 그러나 Amazon Route 53 Resolver를 사용하는 경우 자체 온프레미스 이름 서버를 사용하여 특정 엔드포인트를 확인할 수 있습니다. 이 도구를 사용하면 온프레미스와 AWS 간에 인바운드 및 아웃바운드 엔드포인트, 전달 규칙, 프라이빗 연결을 사용하여 쿼리할 수 있습니다. 온프레미스 이름 서버를 사용하면 보안이 향상되고 방화벽 뒤에서의 사용 편의성이 향상된다는 이점이 있습니다.

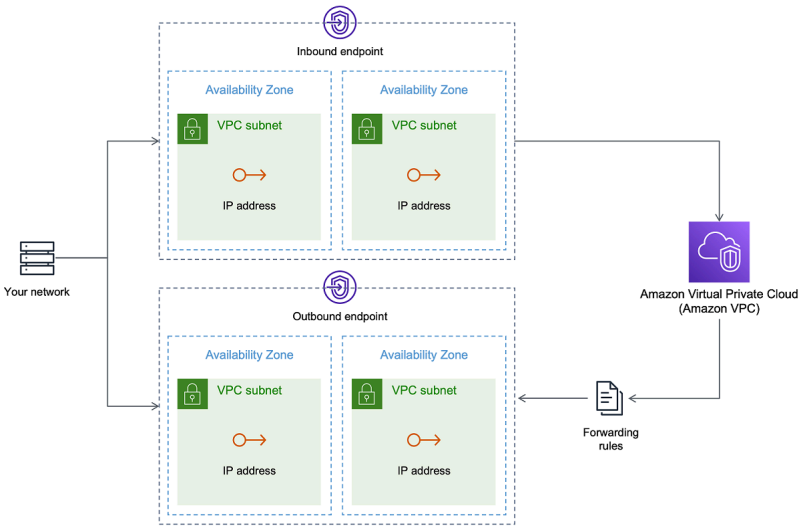
인바운드 엔드포인트가 있는 경우 온프레미스에서 시작되는 DNS 쿼리를 사용하여 AWS 호스팅 도메인을 확인할 수 있습니다. 엔드포인트를 구성하려면 해석기를 제공하려는 각 서브넷의 IP 주소를 할당합니다. 온프레미스 DNS 인프라와 AWS 간에 연결을 설정하려면 AWS Direct Connect 또는 가상 프라이빗 네트워크(VPN)를 사용합니다.

아웃바운드 엔드포인트는 온프레미스 이름 서버에 연결됩니다. 이름 서버는 허용 목록에 포함되고 아웃바운드 엔드포인트에 설정된 IP 주소에만 액세스 권한을 부여합니다. 이름 서버의 IP 주소는 대상 IP 주소입니다. 아웃바운드 엔드포인트에 대한 보안 그룹을 선택할 때는 복제 인스턴스에서 사용하는 것과 동일한 보안 그룹을 선택하세요.

선택된 도메인을 이름 서버로 전달하려면 다음 규칙을 사용합니다. 아웃바운드 엔드포인트는 여러 전달 규칙을 처리할 수 있습니다. 전달 규칙의 범위는 Virtual Private Cloud(VPC)입니다. VPC와 연결된 전달 규칙을 사용하여 AWS 클라우드의 논리적으로 격리된 섹션을 프로비저닝할 수 있습니다. 논리적으로 격리된 이 섹션으로부터 AWS 리소스를 가상 네트워크에서 시작할 수 있습니다.

온프레미스 DNS 인프라 내에서 호스팅되는 도메인을 아웃바운드 DNS 쿼리를 설정하는 조건부 전달 규칙으로 구성할 수 있습니다. 이러한 도메인 중 하나에 대한 쿼리가 수행되면 규칙은 규칙으로 구성된 서버로 DNS 요청을 전달하려는 시도를 트리거합니다. 이 경우에도 AWS Direct Connect 또는 VPN을 통한 프라이빗 연결이 필요합니다.

다음 다이어그램은 Route 53 Resolver 아키텍처입니다.



Route 53 DNS Resolver에 대한 자세한 내용은 Amazon Route 53 개발자 안내서의 Route 53 Resolver 시작하기를 참조하세요.

AWS DMS에서 Amazon Route 53 Resolver 사용

Amazon Route 53 Resolver를 사용하여 엔드포인트를 확인하기 위한 AWS DMS용 온프레미스 이름 서버를 생성할 수 있습니다.

Route 53를 기반으로 AWS DMS용 온프레미스 이름 서버를 생성하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/route53/에서 Route 53 콘솔을 엽니다.
2. Route 53 콘솔에서 Route 53 Resolver를 구성하려는 AWS 리전을 선택합니다. Route 53 Resolver는 리전별 서비스입니다.
3. 쿼리 방향(인바운드, 아웃바운드 또는 둘 다)을 선택합니다.
4. 인바운드 쿼리 구성을 제공합니다.
    a. 엔드포인트 이름을 입력하고 VPC를 선택합니다.
    b. VPC 내에서 하나 이상의 서브넷을 할당합니다(예: 가용성을 위해 2개 선택).
    c. 엔드포인트로 사용할 특정 IP 주소를 할당하거나 Route 53 Resolver가 자동으로 할당하도록 합니다.
5. VPC 내부의 워크로드가 DNS 쿼리를 DNS 인프라로 라우팅할 수 있도록 온프레미스 도메인에 대한 규칙을 생성합니다.
6. 온프레미스 DNS 서버에 대한 IP 주소를 하나 이상 입력합니다.
7. 규칙을 제출합니다.

모든 것이 생성되면 VPC가 인바운드 및 아웃바운드 규칙과 연결되어 트래픽 라우팅을 시작할 수 있습니다.

Route 53 Resolver에 대한 자세한 내용은 Amazon Route 53 개발자 안내서의 Route 53 Resolver 시작하기를 참조하세요.

대용량 이진 객체(LOB) 마이그레이션

일반적으로 AWS DMS는 LOB 데이터를 두 단계로 마이그레이션합니다.

1. AWS DMS는 대상 테이블에 새 행을 생성하고 연결된 LOB 값을 제외한 모든 데이터로 행을 채웁니다.
2. AWS DMS는 대상 테이블의 행을 LOB 데이터로 업데이트합니다.

이 LOB 마이그레이션 프로세스를 수행하려면 마이그레이션 중에 대상 테이블의 모든 LOB 열에 Null을 허용해야 합니다. 이는 소스 테이블에서 LOB 열을 Null로 설정할 수 없는 경우에도 마찬가지입니다. AWS DMS가 대상 테이블을 생성하는 경우 기본적으로 LOB 열을 Null 허용으로 설정합니다. 경우에

따라 가져오기 또는 내보내기와 같은 다른 메커니즘을 사용하여 대상 테이블을 생성할 수 있습니다. 이러한 경우 마이그레이션 태스크를 시작하기 전에 LOB 열이 Null 허용인지 확인하세요.

이 요구 사항에는 한 가지 예외가 있습니다. Oracle 소스에서 Oracle 대상으로 동종 마이그레이션을 수행하고 제한적 LOB 모드를 선택한다고 가정해 보겠습니다. 이 경우 LOB 값을 포함하여 전체 행이 한 번에 채워집니다. 이러한 경우 AWS DMS는 필요하면 Null 허용 안 함 제약 조건을 사용하여 대상 테이블 LOB 열을 생성할 수 있습니다.

제한적 LOB 모드 사용

AWS DMS는 마이그레이션에 LOB 값이 포함된 경우 두 가지 방법을 통해 성능과 편의성의 균형을 맞춥니다.

1. 제한적 LOB 모드에서는 모든 LOB 값을 사용자가 지정한 크기 제한(기본값은 32KB)까지 마이그레이션합니다. 크기 제한보다 큰 LOB 값은 수동으로 마이그레이션해야 합니다. 모든 마이그레이션 태스크의 기본값인 제한적 LOB 모드가 일반적으로 최상의 성능을 제공합니다. 하지만 최대 LOB 크기 파라미터 설정이 올바른지 확인해야 합니다. 이 파라미터를 모든 테이블에서 가장 큰 LOB 크기로 설정하세요.
2. 전체 LOB 모드는 크기에 관계없이 테이블의 모든 LOB 데이터를 마이그레이션합니다. 전체 LOB 모드를 사용하면 테이블의 모든 LOB 데이터를 편리하게 이동할 수 있지만 프로세스가 성능에 상당한 영향을 미칠 수 있습니다.

PostgreSQL과 같은 일부 데이터베이스 엔진의 경우 AWS DMS가 JSON 데이터 형식을 LOB처럼 취급합니다. 제한적 LOB 모드를 선택한 경우 최대 LOB 크기 옵션이 JSON 데이터가 잘리지 않는 값으로 설정되어 있는지 확인하세요.

AWS DMS는 대형 객체 데이터 형식(BLOB, CLOB 및 nCLOB) 사용을 완벽하게 지원합니다. 다음 소스 엔드포인트는 전체 LOB 지원을 제공합니다.

- Oracle
- Microsoft SQL Server
- ODBC

다음 대상 엔드포인트는 전체 LOB 지원을 제공합니다.

- Oracle
- Microsoft SQL Server

다음 대상 엔드포인트는 제한적 LOB 지원을 제공합니다. 이 대상 엔드포인트에는 무제한 LOB 크기를 사용할 수 없습니다.

- Amazon Redshift
- Amazon S3

전체 LOB를 지원하는 엔드포인트의 경우 LOB 데이터 형식에 대한 크기 제한을 설정할 수도 있습니다.

개선된 LOB 성능

LOB 데이터를 마이그레이션하는 동안 다음과 같은 다양한 LOB 최적화 설정을 지정할 수 있습니다.

테이블별 LOB 설정

테이블별 LOB 설정을 사용하여 테이블 일부 또는 전부에 대한 태스크 수준 LOB 설정을 재정의할 수 있습니다. 이렇게 하려면 `table-settings` 규칙에서 `lob-settings`를 정의합니다. 다음은 몇 가지 큰 LOB 값이 포함된 예제 테이블입니다.

```
SET SERVEROUTPUT ON
CREATE TABLE TEST_CLOB
(
ID NUMBER,
C1 CLOB,
C2 VARCHAR2(4000)
);
DECLARE
bigtextstring CLOB := '123';
iINT;
BEGIN
WHILE Length(bigtextstring) <= 60000 LOOP
bigtextstring := bigtextstring || '0000000000000000000000000000000000';
END LOOP;
INSERT INTO TEST_CLOB (ID, C1, C2) VALUES (0, bigtextstring,'AnyValue');
END;
/
SELECT * FROM TEST_CLOB;
COMMIT
```

다음으로 마이그레이션 태스크를 생성하고 새 `lob-settings` 규칙을 사용하여 테이블의 LOB 처리를 수정합니다. `bulk-max-siz` 값에 따라 최대 LOB 크기(KB)가 결정됩니다. 지정된 크기보다 크면 잘립니다.

```
{
 "rules": [{
   "rule-type": "selection",
   "rule-id": "1",
   "rule-name": "1",
   "object-locator": {
    "schema-name": "HR",
    "table-name": "TEST_CLOB"
   },
   "rule-action": "include"
  },
  {
   "rule-type": "table-settings",
   "rule-id": "2",
   "rule-name": "2",
   "object-locator": {
    "schema-name": "HR",
    "table-name": "TEST_CLOB"
   },
   "lob-settings": {
    "mode": "limited",
    "bulk-max-size": "16"
   }
  }
 ]
}
```

이 AWS DMS 태스크가 `FullLobMode : true`로 생성되더라도 테이블별 LOB 설정은 AWS DMS에게 이 특정 테이블의 LOB 데이터를 16,000으로 자르도록 지시합니다. 태스크 로그를 확인하여 이를 확인할 수 있습니다.

```
721331968: 2018-09-11T19:48:46:979532 [SOURCE_UNLOAD] W: The value of column 'C' in
 table
'HR.TEST_CLOB' was truncated to length 16384
```

인라인 LOB 설정

AWS DMS 태스크를 생성할 때 LOB 모드는 LOB 처리 방법을 결정합니다.

전체 LOB 모드와 제한적 LOB 모드는 각각 장단점이 있습니다. 인라인 LOB 모드는 전체 LOB 모드와 제한적 LOB 모드의 장점을 결합합니다.

큰 LOB와 작은 LOB를 모두 복제해야 하지만 대부분의 LOB가 작은 경우 인라인 LOB 모드를 사용할 수 있습니다. 이 옵션을 선택하면 전체 로드 중에 AWS DMS 태스크가 작은 LOB를 인라인으로 전송하므로 더 효율적입니다. AWS DMS 태스크는 소스 테이블에서 조회를 수행하여 큰 LOB를 전송합니다.

변경 처리 중에는 소스 테이블에서 검색을 수행하여 작은 LOB와 큰 LOB를 모두 복제합니다.

인라인 LOB 모드를 사용하는 경우 AWS DMS 태스크는 모든 LOB 크기를 검사하여 인라인으로 전송할 LOB 크기를 결정합니다. 지정된 크기보다 큰 LOB는 전체 LOB 모드를 사용하여 복제됩니다. 따라서 대부분의 LOB가 지정된 설정보다 큰 경우 이 옵션을 사용하지 않는 것이 좋습니다. 대신 LOB 크기를 무제한으로 허용하세요.

이 옵션은 태스크 설정의 `InlineLobMaxSize` 속성을 사용하여 구성합니다. 이 속성은 `FullLobMode`가 `true`로 설정된 경우에만 사용할 수 있습니다. `InlineLobMaxSize`의 기본값은 0이고 범위는 1~102400킬로바이트(100MB)입니다.

예를 들어 다음과 같은 AWS DMS 태스크 설정을 사용할 수 있습니다. 여기서 `InlineLobMaxSize` 값을 5로 설정하면 5KiB(5,120바이트)보다 작거나 같은 모든 LOB가 인라인으로 전송됩니다.

```
{
  "TargetMetadata": {
  "TargetSchema": "",
  "SupportLobs": true,
  "FullLobMode": true,
  "LobChunkSize": 64,
  "LimitedSizeLobMode": false,
  "LobMaxSize": 32,
  "InlineLobMaxSize": 5,
  "LoadMaxFileSize": 0,
  "ParallelLoadThreads": 0,
  "ParallelLoadBufferSize":0,
  "BatchApplyEnabled": false,
  "TaskRecoveryTableEnabled": false},
  . . .
}
```

행 필터링을 사용하여 큰 테이블 마이그레이션 시 성능 향상

대형 테이블을 마이그레이션할 때 성능을 개선하려면 마이그레이션을 두 개 이상의 태스크로 나누세요. 행 필터링을 사용하여 마이그레이션을 여러 태스크로 나누려면 키 또는 파티션 키를 사용합니다. 예를 들어 1에서 8,000,000까지의 정수 프라이머리 키 ID가 있는 경우 행 필터링을 사용하여 8개의 태스크를 생성하여 각각 1백만 개의 레코드를 마이그레이션할 수 있습니다.

콘솔에서 행 필터링을 적용하려면

1. AWS Management Console을 엽니다.
2. 태스크를 선택하고 새 태스크를 생성합니다.
3. 테이블 매핑 탭을 선택하고 선택 규칙을 확장합니다.
4. 새 선택 규칙 추가를 선택합니다. 이제 작거나 같음, 크거나 같음, 같음 또는 두 값 사이의 범위 조건을 사용하여 열 필터를 추가할 수 있습니다. 열 필터링에 대한 자세한 내용은 콘솔에서 테이블 선택 및 변환 규칙 지정 섹션을 참조하세요.

날짜별로 분할된 큰 파티션된 테이블이 있는 경우 날짜를 기준으로 데이터를 마이그레이션할 수 있습니다. 예를 들어, 월별로 분할된 테이블이 있고 현재 월의 데이터만 업데이트된다고 가정합시다. 이 경우 각 고정 월별 파티션에 대해 전체 로드 태스크를 생성하고 현재 업데이트된 파티션에 대해 전체 로드 + CDC 태스크를 생성할 수 있습니다.

테이블에 단일 열 프라이머리 키 또는 고유 인덱스가 있는 경우 AWS DMS 태스크가 범위 유형의 병렬 로드를 사용하여 테이블을 분할하여 데이터를 병렬로 로드하도록 할 수 있습니다. 자세한 설명은 테이블 및 컬렉션 설정 규칙과 작업 섹션을 참조하세요.

지속적 복제

AWS DMS는 소스 및 대상 데이터베이스를 동기화 상태로 유지하면서 지속적 데이터 복제를 제공합니다. 제한된 양의 데이터 정의 언어(DDL) 문만 복제됩니다. AWS DMS는 인덱스, 사용자, 권한, 저장 프로시저, 기타 테이블 데이터와 직접 관련되지 않은 데이터베이스 변경 사항과 같은 항목을 전파하지 않습니다.

지속적 복제를 사용하려는 경우 복제 인스턴스를 생성할 때 다중 AZ 옵션을 설정하세요. 다중 AZ를 선택하면 복제 인스턴스의 고가용성과 장애 조치 기능을 확보할 수 있습니다. 하지만 이 옵션은 성능에 영향을 줄 수 있으며 대상 시스템에 변경 사항을 적용하는 동안 복제 속도가 느려질 수 있습니다.

소스 또는 대상 데이터베이스를 업그레이드하기 전에 이러한 데이터베이스에서 실행 중인 모든 AWS DMS 태스크를 중지하는 것이 좋습니다. 업그레이드가 완료된 후 태스크를 재개하세요.

복제를 진행하는 동안에는 소스 데이터베이스 시스템과 AWS DMS 복제 인스턴스 간의 네트워크 대역폭을 확인하는 것이 중요합니다. 복제가 진행되는 동안 네트워크로 인해 병목 현상이 발생하지 않는지 확인해야 합니다.

소스 데이터베이스 시스템의 시간당 변경 속도 및 아카이브 로그 생성을 파악하는 것도 중요합니다. 이렇게 하면 복제가 진행되는 동안 발생할 수 있는 처리량을 이해하는 데 도움이 될 수 있습니다.

소스 데이터베이스의 로드 감소

AWS DMS는 소스 데이터베이스의 리소스를 일부 사용합니다. 전체 로드 태스크 중에 AWS DMS는 소스 테이블에서 병렬로 처리되는 각 테이블에 대해 전체 테이블 스캔을 수행합니다. 또한 마이그레이션의 일부로 생성하는 각 태스크는 CDC 프로세스의 일환으로 소스에 변경 사항을 쿼리합니다. AWS DMS가 Oracle과 같은 일부 소스에 대해 CDC를 수행하려면 데이터베이스의 변경 로그에 기록되는 데이터의 양을 늘려야 할 수 있습니다.

소스 데이터베이스에 과부하가 걸린다면 마이그레이션할 각 태스크의 태스크 또는 테이블 수를 줄이세요. 각 태스크마다 소스 변경 사항이 독립적으로 적용되므로 태스크를 통합하면 변경 사항 캡처 워크로드를 줄일 수 있습니다.

대상 데이터베이스에서 병목 현상 감소

마이그레이션하는 동안 대상 데이터베이스의 쓰기 리소스를 경합하는 프로세스를 모두 제거해 보세요.

- 불필요한 트리거를 비활성화하세요.
- 초기 로드 중에는 보조 인덱스를 비활성화하고 나중에 복제가 진행되는 동안 다시 활성화합니다.
- Amazon RDS 데이터베이스의 경우 전환이 완료될 때까지 백업 및 다중 AZ를 비활성화하는 것이 좋습니다.
- 비 RDS 시스템으로 마이그레이션할 때는 전환 시까지 대상의 모든 로깅을 비활성화하는 것이 좋습니다.

마이그레이션 중에 데이터 유효성 검사 사용

데이터가 소스에서 대상으로 정확히 마이그레이션되었는지 확인하기 위해 데이터 유효성 검사를 사용하는 것이 좋습니다. 태스크에 대해 유효성 검사를 활성화하면 AWS DMS는 테이블에 대해 전체 로드가 수행된 직후 소스 및 대상 데이터를 비교하기 시작합니다.

데이터 검증은 AWS DMS가 소스 및 대상 엔드포인트로서 다음 데이터베이스를 지원할 때마다 해당 데이터베이스와 연동합니다.

- Oracle
- PostgreSQL
- MySQL
- MariaDB

- Microsoft SQL Server
- Amazon Aurora MySQL 호환 버전
- Amazon Aurora PostgreSQL 호환 에디션
- IBM Db2 LUW
- Amazon Redshift

자세한 설명은 AWS DMS 데이터 검증 섹션을 참조하세요.

지표를 사용하여 AWS DMS 태스크 모니터링

AWS DMS 콘솔을 사용하여 태스크의 지표를 모니터링할 수 있는 몇 가지 옵션이 있습니다.

호스트 지표

특정 복제 인스턴스 각각에 대한 지표 탭에서 호스트 CloudWatch 지표를 찾을 수 있습니다. 여기에서 복제 인스턴스의 크기가 적절한지 모니터링할 수 있습니다.

복제 작업 지표

수신 및 커밋된 변경 사항, 복제 호스트와 소스/대상 데이터베이스 간 지연 시간을 비롯한 복제 작업에 대한 지표는 각 특정 작업의 CloudWatch 지표 탭에서 확인할 수 있습니다.

테이블 지표

개별 테이블 지표는 각 태스크에 대한 테이블 통계 탭에서 확인할 수 있습니다. 이러한 지표에는 다음과 같은 수치가 포함됩니다.

- 전체 로드 중에 로드된 행 수.
- 태스크 시작 이후 삽입, 업데이트 및 삭제 횟수.
- 태스크 시작 이후 DDL 작업 수.

지표 모니터링에 대한 자세한 정보는 AWS DMS 태스크 모니터링 섹션을 참조하세요.

이벤트 및 알림

AWS DMS는 Amazon SNS를 사용하여 AWS DMS 이벤트(복제 인스턴스 생성 또는 삭제 등)가 발생하면 알림을 제공합니다. 특정 AWS 리전의 Amazon SNS에서 지원하는 모든 형식으로 이러한 알림을 사

용할 수 있습니다. 여기에는 이메일 메시지, 문자 메시지 또는 HTTP 엔드포인트에 대한 호출이 포함될 수 있습니다.

자세한 설명은 AWS Database Migration Service에서 Amazon SNS 이벤트 및 알림 사용 섹션을 참조하세요.

마이그레이션 문제 해결을 위해 작업 로그 사용

경우에 따라 AWS DMS에서 해당 경고 또는 오류 메시지가 태스크 로그에만 나타나는 문제가 발생할 수 있습니다. 특히 데이터 잘림 문제나 외래 키 위반으로 인한 행 거부는 태스크 로그에만 기록됩니다. 따라서 데이터베이스를 마이그레이션할 때는 태스크 로그를 검토해야 합니다. 작업 로그를 보려면 Amazon을 작업 생성의 CloudWatch 일부로 구성하십시오.

자세한 내용은 Amazon을 사용한 복제 작업 모니터링을 참조하십시오 CloudWatch.

Time Travel을 통한 복제 태스크 문제 해결

Time Travel을 사용하여 AWS DMS 마이그레이션 문제를 해결할 수 있습니다. Time Travel에 대한 자세한 내용은 Time Travel 작업 설정 섹션을 참조하세요.

Time Travel로 작업할 때 다음 사항을 고려해야 합니다.

- DMS 복제 인스턴스의 오버헤드를 방지하려면 디버깅이 필요한 태스크에만 Time Travel을 활성화합니다.
- Time Travel을 사용하여 며칠 동안 실행될 수 있는 복제 태스크의 문제를 해결할 때는 복제 인스턴스 지표에서 리소스 오버헤드를 모니터링합니다. 이 접근 방식은 특히 소스 데이터베이스에서 장시간 동안 높은 트랜잭션 로드가 실행되는 경우에 적용됩니다. 자세한 내용은 AWS DMS 태스크 모니터링을(를) 참조하세요.
- Time Travel 태스크 설정 `EnableRawData`를 `true`로 설정하면 DMS 복제 중 태스크 메모리 사용량이 Time Travel이 활성화되어 있지 않을 때보다 많을 수 있습니다. Time Travel을 장시간 활성화하는 경우 태스크를 모니터링하세요.
- 현재 Time Travel은 태스크 수준에서만 활성화할 수 있습니다. 모든 테이블의 변경 사항이 Time Travel 로그에 기록됩니다. 트랜잭션 볼륨이 많은 데이터베이스의 특정 테이블에 대한 문제를 해결하려면 별도의 태스크를 생성하세요.

Oracle 대상에서 사용자 및 스키마 변경

Oracle을 대상으로 사용하는 경우 AWS DMS는 데이터를 대상 엔드포인트 사용자가 소유한 스키마로 마이그레이션합니다.

예를 들어 명명된 PERFDATA 스키마를 Oracle 대상 엔드포인트로 마이그레이션하는 중이고 대상 엔드포인트 사용자 이름은 MASTER이라고 가정해 보겠습니다. AWS DMS는 MASTER로 Oracle 대상에 연결하고 MASTER 스키마를 PERFDATA의 데이터베이스 객체로 채웁니다.

이 동작을 재정의하려면 스키마 변환을 제공합니다. 예를 들어 PERFDATA 스키마 객체를 대상 엔드포인트의 PERFDATA 스키마로 마이그레이션하려면 다음 변환을 사용합니다.

```
{
    "rule-type": "transformation",
    "rule-id": "2",
    "rule-name": "2",
    "object-locator": {
        "schema-name": "PERFDATA"
    },
    "rule-target": "schema",
    "rule-action": "rename",
    "value": "PERFDATA"
}
```

변환에 대한 자세한 내용은 JSON을 사용하여 테이블 선택 및 변환 지정 섹션을 참조하세요.

Oracle 대상에 대한 테이블 및 인덱스 테이블스페이스 변경

Oracle을 대상으로 사용하는 경우, AWS DMS는 모든 테이블 및 인덱스를 대상의 기본 테이블스페이스로 마이그레이션합니다. 예를 들어 소스가 Oracle이 아닌 다른 데이터베이스 엔진이라고 가정해 보겠습니다. 모든 대상 테이블 및 인덱스가 동일한 기본 테이블스페이스로 마이그레이션됩니다.

이 동작을 재정의하려면 해당하는 테이블스페이스 변환을 제공합니다. 예를 들어, 테이블 및 인덱스를 소스의 스키마를 따라 명명된 Oracle 대상의 테이블 및 인덱스 테이블스페이스로 마이그레이션하려고 한다고 가정해 보겠습니다. 이 경우 다음과 같은 변형을 사용할 수 있습니다. 여기서 소스의 스키마는 이름이 INVENTORY이고 대상의 해당 테이블 및 인덱스 테이블스페이스는 이름이 각각 INVENTORYTBL 및 INVENTORYIDX입니다.

```
{
```

```
    "rule-type": "transformation",
    "rule-id": "3",
    "rule-name": "3",
    "rule-action": "rename",
    "rule-target": "table-tablespace",
    "object-locator": {
       "schema-name": "INVENTORY",
       "table-name": "%",
       "table-tablespace-name": "%"
    },
    "value": "INVENTORYTBL"
},
{
    "rule-type": "transformation",
    "rule-id": "4
    "rule-name": "4",
    "rule-action": "rename",
    "rule-target": "index-tablespace",
    "object-locator": {
       "schema-name": "INVENTORY",
       "table-name": "%",
       "index-tablespace-name": "%"
    },
    "value": "INVENTORYIDX"
}
```

변환에 대한 자세한 내용은 JSON을 사용하여 테이블 선택 및 변환 지정 섹션을 참조하세요.

Oracle이 소스와 대상인 경우 Oracle 소스 추가 접속 속성인 `enableHomogenousTablespace=true`를 설정하여 기존 테이블 또는 인덱스 테이블스페이스 할당을 유지할 수 있습니다. 자세한 설명은 Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS 섹션을 참조하세요.

복제 인스턴스 버전 업그레이드

AWS는 새로운 기능과 성능 개선 사항을 포함하는 AWS DMS 복제 엔진 소프트웨어의 새 버전을 정기적으로 릴리스합니다. 복제 엔진 소프트웨어의 각 버전마다 고유한 버전 번호가 있습니다. AWS DMS 복제 인스턴스를 최신 버전으로 업그레이드하기 전에 프로덕션 워크로드를 실행하는 복제 인스턴스의 기존 버전을 테스트하는 것이 중요합니다. 사용 가능한 버전 업그레이드에 대한 자세한 내용은 AWS DMS 릴리스 노트 딘원을 참조하세요.

마이그레이션 비용 이해

AWS Database Migration Service를 통해 데이터베이스를 AWS로 간편하고 안전하게 마이그레이션할 수 있습니다. 복제 인스턴스 및 추가 로그 스토리지에 대한 비용만 지불하면 됩니다. 각 데이터베이스 마이그레이션 인스턴스에는 대부분의 복제에 충분한 스왑 공간, 복제 로그 및 데이터 캐시용 스토리지가 포함되어 있으며 인바운드 데이터 전송은 무료입니다.

초기 로드 또는 피크 로드 시간에는 더 많은 리소스가 필요할 수 있습니다. CloudWatch 지표를 사용하여 복제 인스턴스 리소스 사용률을 면밀히 모니터링할 수 있습니다. 그런 다음 사용량에 따라 복제 인스턴스 크기를 늘리거나 줄일 수 있습니다.

마이그레이션 비용 추정에 대한 자세한 내용은 다음을 참조하세요.

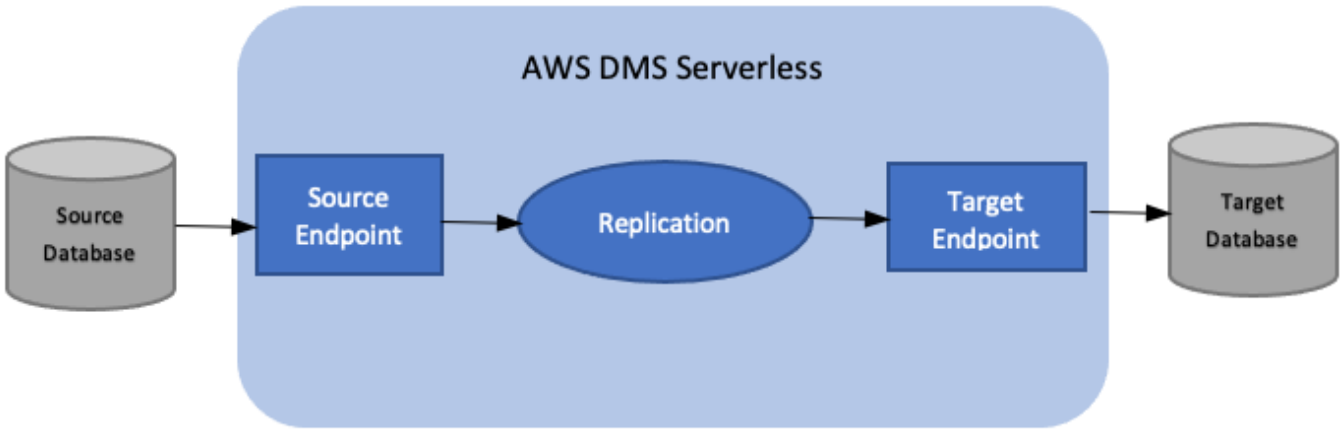
- AWS Database Migration Service 요금
- AWS 요금 계산기

AWS DMS 서버리스로 작업하기

AWS DMS 서버리스는 자동 프로비저닝, 확장, 내장된 고가용성 및 pay-for-use 청구 모델을 제공하여 운영 민첩성을 높이고 비용을 최적화하는 기능입니다. 서버리스 기능을 사용하면 용량 예측, 프로비저닝, 비용 최적화, 복제 엔진 버전 관리, 패치 적용과 같은 복제 인스턴스 관리 태스크가 필요하지 않습니다.

의 현재 기능 AWS DMS (이 문서에서는 AWS DMS 표준이라고 함) 과 마찬가지로 AWS DMS 서버리스를 사용하면 엔드포인트를 사용하여 소스 및 대상 연결을 생성할 수 있습니다. 소스 및 대상 엔드포인트를 생성한 후, 제공된 복제에 대한 구성 설정이 포함된 복제 구성을 생성합니다. 복제를 시작, 중지, 수정 또는 삭제하여 복제를 관리할 수 있습니다. 각 복제 작업에는 데이터베이스 마이그레이션의 요구 사항에 따라 구성할 수 있는 설정이 있습니다. JSON 파일 또는 AWS DMS 섹션을 사용하여 이러한 설정을 지정합니다. AWS Management Console 복제 설정에 대한 자세한 내용은 AWS DMS 엔드포인트 작업을 참조하십시오. 복제가 시작되면 AWS DMS 서버리스는 소스 데이터베이스에 연결하고, 데이터베이스 메타데이터를 수집하여 복제 워크로드를 분석합니다. 이 메타데이터를 사용하여 필요한 용량을 AWS DMS 계산 및 프로비저닝하고 데이터 복제를 시작합니다.

다음 다이어그램은 AWS DMS 서버리스 복제 프로세스를 보여줍니다.



> **Note**
>
> AWS DMS 서버리스는 기본 엔진 버전을 사용합니다. 기본 엔진 버전에 대한 내용은 릴리스 정보 섹션을 참조하세요.

AWS DMS 서버리스에 대한 자세한 내용은 다음 항목을 참조하십시오.

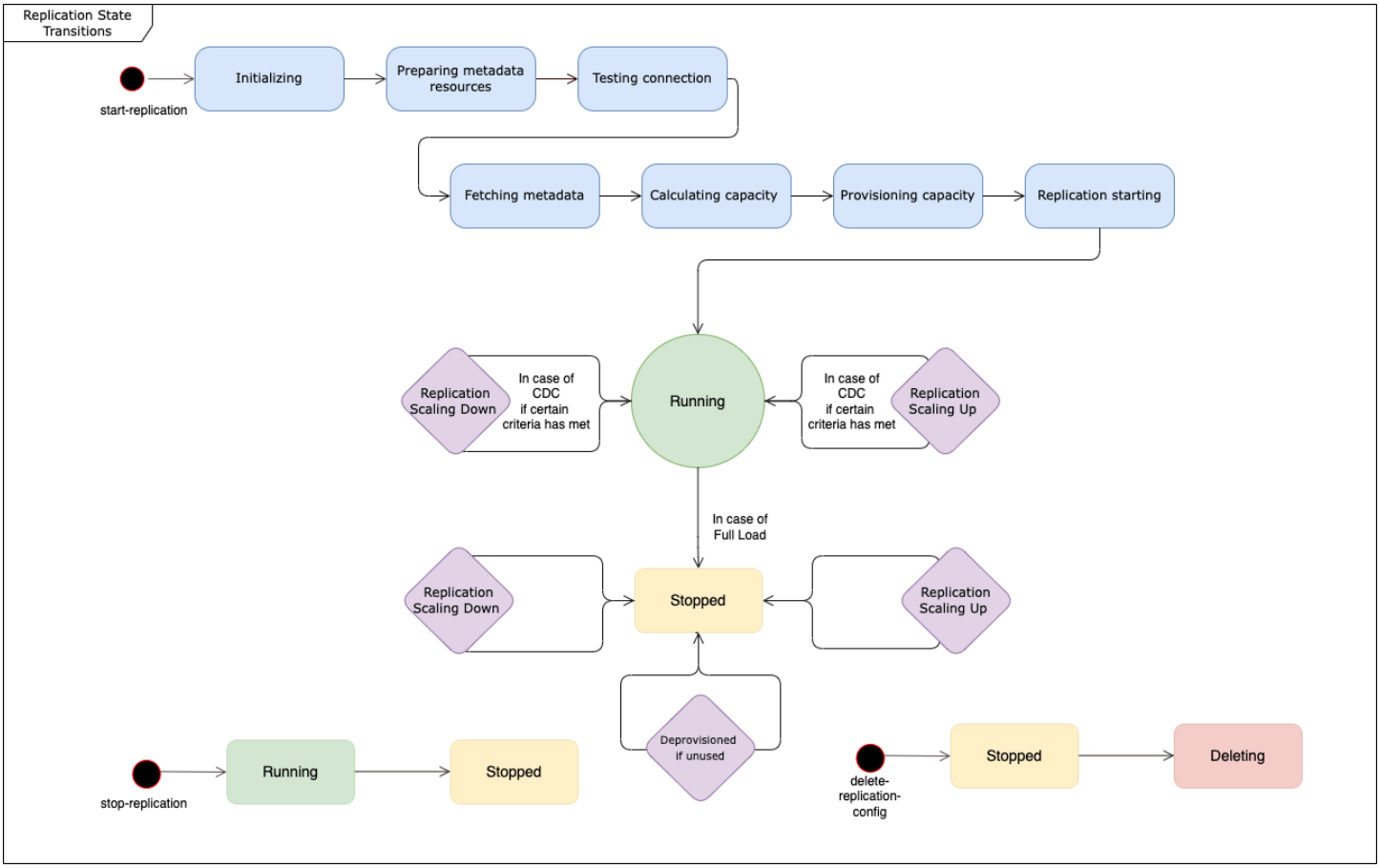
주제

- AWS DMS 서버리스 구성 요소
- AWS DMS 서버리스 제한

AWS DMS 서버리스 구성 요소

복제를 수행하는 데 필요한 리소스를 관리하기 위해 AWS DMS 서버리스는 서비스에서 수행한 다양한 내부 조치를 나타내는 세분화된 상태를 제공합니다. 복제를 시작하면 AWS DMS 서버리스는 용량 부하를 계산하고, 계산된 용량을 프로비저닝한 후 다음 복제 상태에 따라 데이터 복제를 시작합니다.

다음 다이어그램은 서버리스 AWS DMS 복제의 상태 전환을 보여줍니다.



- 복제를 시작한 후 첫 번째 상태는 초기화 중입니다. 이 상태에서는 모든 필수 파라미터가 초기화됩니다.

- 바로 다음 상태에는 메타데이터 리소스 준비 중, 연결 테스트 중, 메타데이터 가져오는 중이 포함됩니다. 이러한 상태에서 AWS DMS 서버리스는 소스 데이터베이스에 연결하여 필요한 용량을 예측하는 데 필요한 정보를 얻습니다.
  - 복제 상태가 연결 테스트인 경우 AWS DMS 서버리스는 원본 및 대상 데이터베이스에 대한 연결이 성공적으로 설정되었는지 확인합니다.
  - 연결 테스트 중 이후의 복제 상태는 메타데이터 가져오는 중입니다. 여기서는 용량을 AWS DMS 계산하는 데 필요한 정보를 검색합니다.
  - 필요한 정보를 AWS DMS 검색한 후 다음 상태는 용량 계산입니다. 이 상태에서 시스템은 복제를 수행하는 데 필요한 기본 리소스의 크기를 계산합니다.
- 용량 계산 중 다음에는 상태가 용량 프로비저닝 중으로 전환됩니다. 복제 작업이 이 상태인 경우, AWS DMS 서버리스는 기본 컴퓨팅 리소스를 초기화합니다.
- 모든 리소스가 성공적으로 프로비저닝된 후의 복제 상태는 복제 시작 중입니다. 이 상태에서 AWS DMS 서버리스는 데이터 복제를 시작합니다.
- 마지막 상태는 실행 중입니다. 실행 중 상태에서는 데이터 복제가 진행되는 중입니다.
- 중지한 복제는 중지됨 상태가 됩니다. 다음과 같은 상황에서 중지된 복제를 다시 시작할 수 있습니다.
  - DMS에서 프로비저닝을 취소한 복제는 다시 시작할 수 없습니다.
  - 작업을 사용하여 중지된 CDC 전용 또는 전체 로드 및 CDC 복제를 다시 시작할 수 있습니다. StartReplication 중지된 복제는 콘솔을 사용하여 다시 시작할 수 없습니다.
  - PostgreSQL을 엔진으로 사용하는 중지된 복제는 다시 시작할 수 없습니다.

이 주제는 다음 섹션을 포함하고 있습니다.

- 지원되는 엔진 버전
- 서버리스 애플리케이션 생성
- 서버리스 복제 수정 AWS DMS
- 컴퓨팅 구성
- 서버리스의 오토스케일링에 대한 이해 AWS DMS
- 서버리스 복제 모니터링 AWS DMS

AWS DMS 서버리스의 경우 AWS DMS 콘솔의 왼쪽 탐색 패널에 서버리스 복제라는 새로운 옵션이 있습니다. 서버리스 복제의 경우 복제 인스턴스 유형이나 태스크 대신, 복제를 지정하여 복제를 정의합니다. 또한 복제를 위해 DMS로 프로비저닝할 최대 및 최소 DMS 용량 단위(DCU)를 지정합니다. DCU는

2GB RAM입니다. AWS DMS 복제에 현재 사용 중인 각 DCU에 대해 계정에 요금이 청구됩니다. AWS DMS 요금에 대한 자세한 내용은 AWS Database Migration Service 요금을 참조하십시오.

AWS DMS 그런 다음 테이블 매핑과 예상 워크로드 크기를 기반으로 복제 리소스를 자동으로 프로비저닝합니다. 이 용량 단위는 지정한 최소 및 최대 용량 단위 값 범위에 속하는 값입니다.

지원되는 엔진 버전

AWS DMS 서버리스를 사용하면 서비스에서 해당 설정을 처리하므로 엔진 버전을 선택하고 관리할 필요가 없습니다. AWS DMS 서버리스는 다음 소스를 지원합니다.

- Microsoft SQL Server
- PostgreSQL 호환 데이터베이스
- MySQL 호환 데이터베이스
- MariaDB
- Oracle
- IBM Db2

AWS DMS 서버리스는 다음 대상을 지원합니다.

- Microsoft SQL Server
- PostgreSQL
- MySQL 호환 데이터베이스
- Oracle
- Amazon S3
- Amazon Redshift
- Amazon DynamoDB
- Amazon Kinesis Data Streams
- Amazon Managed Streaming for Apache Kafka
- 아마존 OpenSearch 서비스
- Amazon DocumentDB(MongoDB 호환)
- Amazon Neptune

AWS DMS 서버리스의 일부로서 서버리스 복제를 생성, 구성, 시작 및 관리할 AWS DMS 수 있는 콘솔 명령에 액세스할 수 있습니다. 콘솔의 서버리스 복제 섹션을 사용하여 이러한 명령을 실행하려면 다음 중 하나를 수행해야 합니다.

- 새 AWS Identity and Access Management (IAM) 정책과 해당 정책을 연결할 IAM 역할을 설정합니다.
- AWS CloudFormation 템플릿을 사용하여 필요한 액세스 권한을 제공하십시오.

AWS DMS 서버리스를 사용하려면 계정에 서비스 연결 역할 (SLR) 이 있어야 합니다. AWS DMS 이 역할의 생성과 사용을 관리합니다. 필요한 SLR이 있는지 확인하는 방법에 대한 자세한 내용은 AWS DMS Serverless의 서비스 연결 역할 섹션을 참조하세요.

서버리스 애플리케이션 생성

두 기존 AWS DMS 엔드포인트 간에 서버리스 복제를 생성하려면 다음을 수행하십시오. AWS DMS 엔드포인트 생성에 대한 자세한 내용은 을 참조하십시오. 소스 및 대상 엔드포인트 생성

서버리스 애플리케이션 생성

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 서버리스 복제를 선택한 다음, 복제 생성을 선택합니다.
3. 복제 생성 페이지에서 서버리스 복제 구성을 지정합니다.

옵션	작업
이름	복제를 식별할 이름을 입력합니다(예: **DMS-replication**).
설명이 포함된 Amazon 리소스 이름 (ARN) - 선택 사항	비워둡니다.
소스 데이터베이스 엔드포인트	계정의 기존 엔드포인트를 선택합니다. 참고로 AWS DMS 서버리스는 AWS DMS 표준에서 지원하는 엔드포인트 유형 중 일부만 지원합니다.

옵션	작업
대상 데이터베이스 엔드포인트	계정의 기존 엔드포인트를 선택합니다. 참고로 AWS DMS 서버리스는 표준에서 지원하는 엔드포인트 유형 중 일부만 지원합니다. AWS DMS
복제 유형	요구 사항에 따라 복제 유형을 선택합니다. • 전체 로드: 기존 데이터만 AWS DMS 마이그레이션합니다. • 전체 로드 및 변경 데이터 캡처 (CDC): 기존 데이터와 복제 중에 발생하는 변경 사항을 AWS DMS 마이그레이션합니다. • 변경 데이터 캡처 (CDC): 복제를 AWS DMS 시작한 후 발생한 변경 사항만 마이그레이션합니다.

설정 섹션에서 복제에 필요한 설정을 지정합니다.

테이블 매핑 섹션에서 테이블 매핑을 설정하여 복제 중인 데이터를 선택하고 필터링하는 규칙을 정의합니다. 매핑을 지정하기 전에 원본 및 대상 데이터베이스의 데이터 형식 매핑에 대한 관련 설명서 단원을 참조하십시오.

컴퓨팅 설정 섹션에서 다음과 같은 설정을 지정합니다. 컴퓨팅 구성 설정에 대한 내용은 컴퓨팅 구성 섹션을 참조하세요.

옵션	작업
VPC	기존 VPC를 선택합니다.
서브넷 그룹	기존 서브넷 그룹을 선택합니다.
VPC 보안 그룹	아직 선택하지 않은 경우 기본값을 선택합니다.
AWS KMS 키	(기본값) aws/dms를 선택한 상태로 둡니다.
배포	그대로 둡니다.
가용 영역	그대로 둡니다.

옵션	작업
최소 DMS 용량 단위(DCU) - (선택 사항)	기본값인 1 DCU를 사용하려면 비워둡니다.
최대 DMS 용량 단위(DCU)	16 DCU를 선택합니다.

유지 관리 설정을 그대로 둡니다.

4. 복제 생성을 선택합니다.

AWS DMS 마이그레이션을 수행할 서버리스 복제본을 생성합니다.

서버리스 복제 수정 AWS DMS

복제 구성을 수정하려면 `modify-replication-config` 작업을 사용합니다. `CREATED`, `STOPPED`, 또는 상태의 AWS DMS 복제 구성만 수정할 수 있습니다. `FAILED`

를 사용하여 서버리스 복제 구성을 수정하려면 AWS Management Console

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 서버리스 복제를 선택합니다.
3. 수정할 복제를 선택합니다. 다음 표에는 복제의 현재 상태에 따라 수정할 수 있는 내용이 설명되어 있습니다.

설정	설명	허용되는 상태
이름	복제의 이름을 변경할 수 있습니다. 인쇄 가능한 ASCII 문자(/, " 및 @ 제외) 8~16자를 포함하는 복제의 이름을 입력합니다. 이름은 선택한 AWS 리전의 계정에서 고유해야 합니다. 수행 중인 AWS 지역 및 작업을 비롯한 몇 가지 세부 정보를 이름에 추가할 수 있습니다. 예를 들면 다음과 같습니다**`west2-mysql2mysql-config1`** .	`ReplicationState` 가 `CREATED`, `STOPPED`, `FAILED`인 경우.

설정	설명	허용되는 상태
소스 데이터베이스 엔드포인트	새 기존 소스 엔드포인트를 복제할 소스로 선택합니다.	ProvisionState 가 null일 때 ReplicationState 가 CREATED 또는 FAILED인 경우.
대상 데이터베이스 엔드포인트	새 기존 대상 엔드포인트를 복제할 대상으로 선택합니다.	ProvisionState 가 null일 때 ReplicationState 가 CREATED 또는 FAILED인 경우.
복제 유형	서버리스 복제 유형을 수정할 수 있습니다.	ProvisionState 가 null일 때 ReplicationState 가 CREATED 또는 FAILED인 경우.
복제 설정	대상 테이블 준비 모드, 복제에 LOB 열을 포함할지 여부, 최대 LOB 크기, 검증, 로깅을 포함한 복제 설정을 수정할 수 있습니다. 자세한 설명은 작업 설정 섹션을 참조하세요.	ReplicationState 가 CREATED, STOPPED, FAILED인 경우.
테이블 매핑	선택 규칙 및 변환 규칙을 포함하여, 서버리스 복제의 테이블 매핑 설정을 수정할 수 있습니다. 자세한 설명은 테이블 매핑 섹션을 참조하세요.	ReplicationState 가 CREATED, STOPPED, FAILED인 경우.

설정	설명	허용되는 상태

컴퓨팅 구성	네트워킹 설정, 크기 조정 설정, 유지 관리 설정을 포함하여 서버리스 복제의 컴퓨팅 구성 설정을 수정할 수 있습니다. 컴퓨팅 구성 설정에 대한 내용은 컴퓨팅 구성 섹션을 참조하세요.	• ReplicationState 가 CREATED, STOPPED 또는 FAILED인 경우 다음과 같은 크기 조정, 유지 관리, 네트워크 설정을 수정할 수 있습니다.
		  • MinCapacityUnits
		  • MaxCapacityUnits
		  • MultiAZ
		  • PreferredMaintenanceWindow
		  • VpcSecurityGroupIds
		• ProvisionState 가 null일 때 ReplicationState 가 CREATED, FAILED 또는 인 경우 다음과 같은 네트워킹 및 보안 설정을 수정할 수 있습니다.
		  • AvailabilityZone
		  • DnsNameServers
		  • KmsKeyId

설정	설명	허용되는 상태
		• ReplicationSubnetGroupId

컴퓨팅 구성

컴퓨팅 구성 파라미터 또는 콘솔 섹션을 사용하여 복제 프로비저닝을 구성합니다. 컴퓨팅 구성 객체의 필드에는 다음 내용이 포함됩니다.

옵션	설명
MinCapacityUnits	프로비저닝할 DMS 용량 단위 (DCU) 의 최소 수입니다. AWS DMS 이는 오토 스케일링으로 스케일 다운할 수 있는 최소 DCU이기도 합니다.
MaxCapacityUnits	복제 용량 예측에 따라 AWS DMS 가 프로비저닝할 수 있는 최대 DMS 용량 단위(DCU)입니다. 이는 오토 스케일링으로 스케일 업할 수 있는 최대 DCU이기도 합니다.
KmsKeyId	복제 스토리지와 연결 정보를 암호화하기 위해 사용할 암호화 키입니다. (기본값) aws/dms를 선택하면 계정과 연결된 기본 KMS 키가 AWS DMS 사용됩니다. AWS 리전 설명 및 계정 번호가 키의 ARN과 함께 표시됩니다. 암호화 키 사용에 대한 자세한 내용은 암호화 키 설정 및 AWS KMS 권한 지정 섹션을 참조하세요. 이 튜토리얼에서는 (기본값) aws/dms를 선택한 상태로 둡니다.
ReplicationSubnetGroupId	복제를 생성할 위치인 선택된 VPC에 있는 복제 서브넷 그룹입니다. 소스 데이터베이스가 VPC에 있는 경우, 소스 데이터베이스가 포함된 서브넷 그룹을 복제할 위치로 선택합니다. 복제 서브넷에 대한 자세한 내용은 복제 서브넷 그룹 생성 섹션을 참조하세요.
VpcSecurityGroupIds	모든 복제 인스턴스가 VPC에 생성됩니다. 소스 데이터베이스가 VPC에 있는 경우, 데이터베이스가 위치한 DB 인

옵션	설명
	스턴스에 대한 액세스 권한을 부여하는 VPC 보안 그룹을 선택합니다.
PreferredMaintenanceWindow	이 파라미터는 시스템 유지 관리를 실행할 수 있는 주 단위 기간(UTC, 협정 세계시)을 정의합니다. 기본값은 요일별로 발생하는 8시간 블록 중에서 무작위로 선택한 30분 AWS 리전 기간입니다.
MultiA	이 선택적 파라미터를 설정하면 다른 가용 영역에 복제의 대기 복제본이 생성되어 장애 조치를 지원합니다. 변경 데이터 캡처(CDC) 또는 지속적 복제를 사용하려면, 이 옵션을 켜야 합니다.

서버리스의 오토스케일링에 대한 이해 AWS DMS

복제본을 프로비저닝하고 RUNNING 상태가 되면 AWS DMS 서비스가 기본 리소스의 용량을 관리하여 변화하는 워크로드에 적응합니다. 이러한 관리 기능은 다음과 같은 복제 설정에 따라 복제 리소스의 크기를 조정합니다.

- `MinCapacityUnits`
- `MaxCapacityUnits`

복제는 사용률 임계값 상한을 초과하는 기간이 지나면 스케일 업되고, 용량 사용률이 오랫동안 최소 용량 사용률 임계값 이하로 떨어지면 스케일 다운됩니다.

서버리스의 자동 크기 조정 AWS DMS

복제 자동 크기 조정 매개변수를 조정하려면 최대값으로 설정하고 리소스 프로비저닝을 `MaxCapacityUnits` AWS DMS 관리하도록 하는 것이 좋습니다. 오토 스케일링의 이점을 최대한 활용하고 트랜잭션 볼륨의 급증에 대비하려면 가장 큰 DCU 최대 용량 설정을 선택하는 것이 좋습니다. 요금 계산기는 복제 시 지속적으로 최대 DCU를 사용할 경우 최대 월별 비용을 보여줍니다. 사용한 용량에 대해서만 비용을 지불하므로, 최대 DCU는 실제 비용을 나타내지 않습니다.

복제가 리소스를 최대 용량으로 사용하지 않는 경우 비용을 절감하기 위해 리소스를 점진적으로 디프로비저닝합니다. AWS DMS 하지만 리소스 프로비저닝 및 프로비저닝 해제에는 시간이 걸리므로, 복제 워크로드가 갑자기 급증할 것으로 예상되는 경우 `MinCapacityUnits` 설정을 이를 처리할 수 있는

값으로 설정하는 것이 좋습니다. 이렇게 하면 복제가 제대로 프로비저닝되지 않도록 하는 동시에 더 높은 워크로드 수준에 맞게 리소스를 AWS DMS 프로비저닝할 수 있습니다.

데이터 요구 사항에 비해 너무 낮은 최대 용량 설정 또는 복제 워크로드의 갑작스러운 급증을 처리하기에 너무 낮은 최소 용량으로 복제를 과소 프로비저닝할 경우, `CapacityUtilization` 지표가 계속 최대값으로 표시될 수 있습니다. 이 경우 복제가 실패할 수 있습니다. 리소스가 제대로 프로비저닝되지 않아 복제가 실패하는 경우 복제 로그에 이벤트가 생성됩니다. AWS DMS out-of-memory 복제 워크로드가 갑자기 급증하여 문제가 발생한 경우 복제가 자동으로 확장되고 다시 시작됩니다. out-of-memory

서버리스 복제 모니터링 AWS DMS

AWS AWS DMS 서버리스 복제를 모니터링하고 잠재적 사고에 대응하기 위한 몇 가지 도구를 제공합니다.

- AWS DMS 서버리스 복제 지표
- AWS DMS 서버리스 복제 로그

AWS DMS 서버리스 복제 지표

서버리스 복제 모니터링에는 다음 통계에 대한 Amazon CloudWatch 지표가 포함됩니다. 이러한 통계는 각 서버리스 복제별로 그룹화됩니다.

지표	단위	설명
CapacityUtilization	%	서버리스 복제에 사용된 메모리 비율
CDC IncomingChanges	%	a에서 대상에 적용되기를 기다리고 point-in-time 있는 변경 이벤트의 총 수입니다. 이 값은 소스 엔드포인트의 트랜잭션 변경 비율 측정치와 같지 않습니다. 일반적으로 이 측정 단위의 수가 많으면 캡처된 변경 사항을 적시에 적용할 수 없어 대상 지연 시간이 길다는 AWS DMS 의미입니다.
CDC LatencySource	초	소스 엔드포인트에서 캡처한 마지막 이벤트와 AWS DMS 인스턴스의 현재 시스템 타임스탬프 간의 간격(초)입니다. CDC는 원본 인스턴스와 복제 인스턴스 간의 지연 시간을 LatencySource 나타냅니다. CDC가 LatencySource 높으면 소스에서 변경 내용을 캡처하는

지표	단위	설명
		프로세스가 지연됩니다. CDC와 함께 이 지표를 검토하여 진행 중인 복제의 지연 시간을 식별할 수 있습니다. LatencyTarget LatencySource CDC와 CDC의 수치가 둘 다 높으면 LatencyTarget 먼저 CDC를 조사하세요. LatencySource 원본과 복제 사이에 복제 지연이 없는 경우 CDC는 0이 될 LatencySource 수 있습니다. 또한 복제가 소스의 트랜잭션 로그에서 다음 이벤트를 읽으려고 시도하고 소스에서 마지막으로 읽은 시점과 비교할 때 새 이벤트가 없는 경우에도 CDC는 0이 될 LatencySource 수 있습니다. 이 경우 복제 시 CDC가 LatencySource 0으로 재설정됩니다.
CDC LatencyTarget	초	대상에 커밋되기를 대기하는 첫 이벤트 타임스탬프와 AWS DMS 인스턴스의 현재 타임스탬프 사이의 간격(초)입니다. 대상 지연 시간은 복제 인스턴스 서버 시간과 대상 구성 요소에 전달된 확인되지 않은 가장 오래된 이벤트 ID 간의 차이입니다. 즉, 대상 지연 시간은 복제 인스턴스와 적용되었지만 TRG 엔드포인트에서 확인되지 않은 가장 오래된 이벤트 간의 타임스탬프 차이입니다(99%). CDC가 LatencyTarget 높으면 대상에 변경 이벤트를 적용하는 프로세스가 지연되었음을 나타냅니다. LatencySourceCDC와 함께 이 지표를 검토하여 진행 중인 복제의 지연 시간을 식별할 수 있습니다. CDC는 높지만 LatencyTarget CDC는 높지 LatencySource 않은 경우 다음을 조사하십시오. • 대상에 프라이머리 키 또는 인덱스가 없음 • 대상 또는 복제 인스턴스에서 리소스 병목 현상이 발생함 • 복제와 대상 사이에 네트워크 문제가 있음

지표	단위	설명
CDC ThroughputBandwidthTarget	초당 KB	대상의 발신 데이터(초당 KB 단위). CDC는 샘플링 ThroughputBandwidth 포인트에서 전송된 발신 데이터를 기록합니다. 어떠한 네트워크 트래픽도 찾을 수 없다면 이 값은 0입니다. CDC는 장시간 트랜잭션을 할 수 없기 때문에 네트워크 트래픽은 기록되지 않을 수 있습니다.
CDC ThroughputRowsSource	초당 행 수	소스의 수신 변경 사항 개수(초당 행 수 단위).
질병통제예방센터 ThroughputRowsTarget	초당 행 수	대상의 발신 변경 사항 개수(초당 행 수 단위).
FullLoadThroughputBandwidthTarget	초당 KB	대상 전체 로드의 발신 데이터(초당 KB 단위).
FullLoadThroughputRowsTarget	초당 행 수	대상 전체 로드의 발신 변경 사항(초당 행 수 단위).

AWS DMS 서버리스 복제 로그

CloudWatch Amazon을 사용하여 AWS DMS 마이그레이션 프로세스 중에 복제 정보를 기록할 수 있습니다. 복제 설정을 선택할 때 로깅을 활성화합니다.

서버리스 복제는 상태 로그를 CloudWatch 계정에 업로드하여 복제 진행 상황을 더 잘 파악하고 문제 해결을 지원합니다.

AWS DMS 서버리스 연결 로그를 접두사가 있는 전용 로그 그룹에 업로드합니다. `dms-serverless-replication-`*`<your replication config resource ID>`* 이 로그 그룹에는 `dms-serverless-replication-orchestrator-`*`<your replication config resource ID>`*라는 로그 스트림이 있습니다. 이 로그 스트림은 복제의 복제 상태를 보고하고, 이 단계에서 진행 중인 작업에 대한 추가 세부 정보를 제공하는 관련 메시지를 보고합니다. 로그 항목의 예는 아래의 서버리스 복제 로그 예제 섹션을 참조하세요.

> **Note**
>
> AWS DMS 복제를 실행할 때까지 로그 그룹이나 스트림을 만들지 않습니다. AWS DMS 복제만 생성하는 경우에는 로그 그룹이나 스트림을 생성하지 않습니다.

실행된 복제의 로그를 보려면 다음 단계를 따르세요.

1. AWS DMS 콘솔을 열고 탐색 창에서 서버리스 복제를 선택합니다. 서버리스 복제 대화 상자가 나타납니다.
2. 구성 섹션으로 이동하고 일반 열에서 서버리스 로그 보기를 선택합니다. CloudWatch 로그 그룹이 열립니다.
3. 마이그레이션 작업 로그 섹션을 찾아 CloudWatch 로그 보기를 선택합니다.

복제가 실패할 경우, 복제 상태가 인 로그 항목과 실패 원인을 설명하는 메시지를 AWS DMS 생성합니다. `failed` 실패한 복제 문제를 해결하기 위한 첫 번째 단계로 CloudWatch 로그를 확인해야 합니다.

> **Note**
>
> AWS DMS Classic과 마찬가지로 데이터 마이그레이션 진행 상태 자체, 즉 기본 복제 작업에서 내보낸 로그에 대한 보다 세분화된 로깅을 활성화할 수 있는 옵션이 있습니다. 아래의 JSON 예제에 나온 것처럼, `Logging` 필드에서 `EnableLogging`을 `true`로 설정하면 복제 설정에서 이러한 로그를 활성화할 수 있습니다.
>
> ```
> {
>   "Logging": {
>     "EnableLogging": true
>   }
> }
> ```
>
> 이러한 로그를 활성화하면 서버리스 복제의 `running` 단계에서만 로그가 나타나기 시작합니다. 로그는 이전 로그 스트림과 동일한 로그 그룹 아래에 표시되지만, 새 로그 스트림인 `dms-serverless-serv-res-id-`*`{unique identifier}`* 아래에 표시됩니다. 서버리스 복제 로그를 해석하는 방법에 대한 내용은 다음 섹션을 참조하세요.

서버리스 복제 로그 예제

이 섹션에는 서버리스 복제에 대한 로그 항목의 예가 포함되어 있습니다.

예: 복제 시작

서버리스 복제를 실행하면 다음과 비슷한 로그 항목이 AWS DMS 생성됩니다.

```
{'replication_state':'initializing', 'message': 'Initializing the replication
 workflow.'}
```

예: 복제 실패

복제의 엔드포인트 중 하나가 올바르게 구성되지 않은 경우 다음과 비슷한 로그 항목이 AWS DMS 생성됩니다.

```
{'replication_state':'failed', 'message': 'Test connection failed for endpoint X.',
 'failure_message': 'X'}
```

실패 후 로그에 이 메시지가 표시되면 지정된 엔드포인트가 정상이고 올바르게 구성되어 있는지 확인하세요.

AWS DMS 서버리스 제한

AWS DMS 서버리스에는 다음과 같은 제한 사항이 있습니다.

- CREATEDSTOPPED, 또는 FAILED 상태의 AWS DMS 복제 구성만 수정할 수 있습니다. 어떤 조건에서 어떤 설정을 변경할 수 있는지에 대한 자세한 내용은 서버리스 복제 수정 AWS DMS 섹션을 참조하세요.
- STOPPED, 또는 FAILED 상태에 있는 AWS DMS 복제 구성만 삭제할 수 있습니다.
- 복제에는 고정 100GB 할당된 스토리지를 사용할 수 있습니다. 장기 실행 트랜잭션 또는 캐싱과 같은 요구 사항으로 인해 복제에 이보다 많은 메모리를 사용할 경우, 워크로드를 별도의 서버리스 복제로 분할하는 것이 좋습니다. LOB와 관련된 모든 복제를 별도의 서버리스 복제에 넣는 등, 테이블 또는 요구 사항별로 워크로드를 분할할 수 있습니다.
- 복제 인스턴스와 달리 AWS DMS 서버리스 복제는 관리 작업을 위한 퍼블릭 IP 주소가 없습니다. 콘솔을 사용하여 서버리스 복제를 관리합니다.
- 이번 AWS DMS 서버리스 릴리스는 표준에서 지원하는 모든 원본 및 대상 엔드포인트 유형을 지원하지는 않습니다. AWS DMS 지원되는 엔진 유형의 목록은 AWS DMS 서버리스 구성 요소 섹션을 참조하세요.

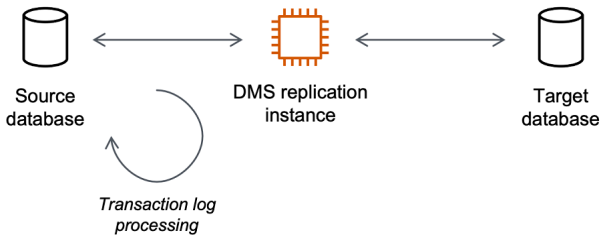
- 서버리스 복제는 VPC 엔드포인트를 사용하여 종속성에 액세스해야 합니다. VPC 엔드포인트를 사용하여 아래와 같은 엔드포인트 유형에 액세스해야 합니다.
  - Amazon Amazon S3
  - Amazon Kinesis
  - AWS Secrets Manager
  - Amazon DynamoDB
  - Amazon Redshift
  - 아마존 OpenSearch 서비스

  VPC 엔드포인트 설정에 대한 내용은 VPC 엔드포인트를 AWS DMS 소스 및 대상 엔드포인트로 구성 섹션을 참조하세요.
- AWS DMS 서버리스는 선택 및 변환 규칙이 있는 뷰를 지원하지 않습니다.
- AWS DMS 서버리스는 AWS 고객 관리 키 사용을 지원하지 않습니다. AWS DMS 서버리스는 기본 DMS 키 사용만 지원합니다. 자세한 내용은 AWS Database Migration Service 데이터 보호을(를) 참조하세요.

AWS DMS 복제 인스턴스 사용

AWS DMS 복제 인스턴스를 생성할 때는 Amazon VPC 서비스를 기반으로 하는 가상 사설 클라우드(VPC) 의 Amazon EC2 인스턴스에 AWS DMS 생성합니다. 이 복제 인스턴스를 사용하여 데이터베이스 마이그레이션을 수행합니다. 복제 인스턴스를 사용하면 다중 AZ 옵션을 선택할 때 다중 AZ 배포로 고가용성 및 장애 조치 지원을 확보할 수 있습니다.

다중 AZ 배포에서는 복제 인스턴스의 동기식 대기 복제본을 다른 가용 영역에 AWS DMS 자동으로 프로비저닝하고 유지 관리합니다. 기본 복제 인스턴스는 가용 영역 전체에서 대기 복제본으로 동기식으로 복제됩니다. 이 접근 방식에서는 데이터 중복을 제공하고 I/O 중지를 없애며 지연 시간 스파이크를 최소화합니다.



AWS DMS 복제 인스턴스를 사용하여 원본 데이터 스토어에 연결하고, 원본 데이터를 읽고, 대상 데이터 저장소에서 사용할 수 있도록 데이터 형식을 지정합니다. 복제 인스턴스는 또한 데이터를 대상 데이터 스토어에 로드합니다. 이 절차 대다수는 메모리에서 진행됩니다. 그렇지만, 대규모 트랜잭션은 디스크에서 일부 버퍼링이 필요할 수 있습니다. 캐시된 트랜잭션과 로그 파일도 디스크에 기록됩니다.

다음 AWS 지역에서 AWS DMS 복제 인스턴스를 생성할 수 있습니다.

리전 이름	리전	엔드포인트	프로토콜	
미국 동부(오하이오)	us-east-2	dms.us-east-2.amazonaws.com	HTTPS	
		dms-fips.us-east-2.amazonaws.com	HTTPS	
미국 동부(버지니아 북부)	us-east-1	dms.us-east-1.amazonaws.com	HTTPS	
		dms-fips.us-east-1.amazonaws.com	HTTPS	
미국 서부(캘리포니아 북부)	us-west-1	dms.us-west-1.amazonaws.com	HTTPS	
		dms-fips.us-west-1.amazonaws.com	HTTPS	

리전 이름	리전	엔드포인트	프로토콜
미국 서부(오레곤)	us-west-2	dms.us-west-2.amazonaws.com dms-fips.us-west-2.amazonaws.com	HTTPS HTTPS
아프리카(케이프타운)	af-south-1	dms.af-south-1.amazonaws.com	HTTPS
아시아 태평양(홍콩)	ap-east-1	dms.ap-east-1.amazonaws.com	HTTPS
아시아 태평양(하이데라바드)	ap-south-2	dms.ap-south-2.amazonaws.com	HTTPS
아시아 태평양(자카르타)	ap-southeast-3	dms.ap-southeast-3.amazonaws.com	HTTPS
아시아 태평양(멜버른)	ap-southeast-4	dms.ap-southeast-4.amazonaws.com	HTTPS
아시아 태평양(뭄바이)	ap-south-1	dms.ap-south-1.amazonaws.com	HTTPS
아시아 태평양(오사카)	ap-northeast-3	dms.ap-northeast-3.amazonaws.com	HTTPS
아시아 태평양(서울)	ap-northeast-2	dms.ap-northeast-2.amazonaws.com	HTTPS

리전 이름	리전	엔드포인트	프로토콜	
아시아 태평양(싱가포르)	ap-southeast-1	dms.ap-southeast-1.amazonaws.com	HTTPS	
아시아 태평양(시드니)	ap-southeast-2	dms.ap-southeast-2.amazonaws.com	HTTPS	
아시아 태평양(도쿄)	ap-northeast-1	dms.ap-northeast-1.amazonaws.com	HTTPS	
캐나다(중부)	ca-central-1	dms.ca-central-1.amazonaws.com	HTTPS	
캐나다 서부(캘거리)	ca-west-1	dms.ca-west-1.amazonaws.com	HTTPS	
유럽(프랑크푸르트)	eu-central-1	dms.eu-central-1.amazonaws.com	HTTPS	
유럽(아일랜드)	eu-west-1	dms.eu-west-1.amazonaws.com	HTTPS	
유럽(런던)	eu-west-2	dms.eu-west-2.amazonaws.com	HTTPS	
유럽(밀라노)	eu-south-1	dms.eu-south-1.amazonaws.com	HTTPS	
유럽(파리)	eu-west-3	dms.eu-west-3.amazonaws.com	HTTPS	
유럽(스페인)	eu-south-2	dms.eu-south-2.amazonaws.com	HTTPS	

리전 이름	리전	엔드포인트	프로토콜	
유럽(스톡홀름)	eu-north-1	dms.eu-north-1.amazonaws.com	HTTPS	
유럽(취리히)	eu-central-2	dms.eu-central-2.amazonaws.com	HTTPS	
이스라엘(텔아비브)	il-central-1	dms.il-central-1.amazonaws.com	HTTPS	
중동(바레인)	me-south-1	dms.me-south-1.amazonaws.com	HTTPS	
중동(UAE)	me-central-1	dms.me-central-1.amazonaws.com	HTTPS	
남아메리카(상파울루)	sa-east-1	dms.sa-east-1.amazonaws.com	HTTPS	
AWS GovCloud(미국 동부)	us-gov-east-1	dms.us-gov-east-1.amazonaws.com	HTTPS	
AWS GovCloud(미국 서부)	us-gov-west-1	dms.us-gov-west-1.amazonaws.com	HTTPS	

AWS DMS 미국 정부 기관 및 고객이 민감한 워크로드를 클라우드로 이전할 수 있도록 설계된 특수 AWS 지역을 지원합니다. AWS GovCloud (US) AWS GovCloud (US) 미국 정부의 특정 규제 및 규정 준수 요구 사항을 해결합니다. 에 대한 AWS GovCloud (US)자세한 내용은 AWS GovCloud (미국) 란 무엇입니까? 를 참조하십시오.

아래에서 복제 인스턴스에 대한 자세한 정보를 확인할 수 있습니다.

주제

- 마이그레이션에 적합한 AWS DMS 복제 인스턴스 선택
- 복제 인스턴스에 가장 적합한 크기 선택
- 복제 엔진 버전 작업
- 퍼블릭 및 프라이빗 복제 인스턴스
- IP 주소 지정 및 네트워크 유형
- 복제 인스턴스용으로 네트워크 설정
- 복제 인스턴스의 암호화 키 설정
- 복제 인스턴스 생성
- 복제 인스턴스 수정
- 복제 인스턴스 재부팅
- 복제 인스턴스 삭제
- AWS DMS 유지 관리 기간 관련 작업

마이그레이션에 적합한 AWS DMS 복제 인스턴스 선택

AWS DMS Amazon EC2 인스턴스에 복제 인스턴스를 생성합니다. AWS DMS 현재 복제 인스턴스에 대해 T2, T3, C4, C5, C6i, R4, R5 및 R6i Amazon EC2 인스턴스 클래스를 지원합니다.

- T2 인스턴스는 성능 버스트 기능이 있는 인스턴스로, 기본 수준의 CPU 성능 외에 필요할 경우 기준 이상으로 높일 수 있는 기능을 제공합니다. 기본 성능과 버스트 기능은 CPU 크레딧에 의해 좌우됩니다. T2 인스턴스는 인스턴스 크기에 따라 정해진 비율의 CPU 크레딧을 지속적으로 받습니다. 유휴 상태일 때는 CPU 크레딧을 누적하고 활성 상태일 때는 CPU 크레딧을 소비합니다.

  T2 인스턴스는 다양한 범용 워크로드에 적합합니다. 이러한 워크로드에는 마이크로서비스, 지연 시간이 짧은 대화형 애플리케이션, 중소 규모 데이터베이스, 가상 데스크톱, 개발, 빌드 및 스테이지 환경, 코드 리포지토리, 제품 프로토타입이 포함됩니다.
- T3 인스턴스는 차세대 버스트 가능 범용 인스턴스 유형입니다. 이 유형은 기본 수준의 성능과 함께, 필요할 때는 언제든지 CPU 사용량을 버스트할 수 있는 기능을 제공합니다. T3 인스턴스는 컴퓨팅, 메모리 및 네트워크 리소스를 균형 있게 제공하고 CUP 사용량이 중간 정도이지만 일시적으로 사용량이 급증하는 워크로드에 적합합니다. T3 인스턴스는 워크로드가 기준 임계값 이하로 작동할 때 CPU 크레딧을 누적합니다. 획득한 각 CPU 크레딧은 T3 인스턴스가 필요할 때 1분 동안 전체 CPU 코어 성능으로 버스트할 수 있는 기회를 제공합니다.

T3 인스턴스는 언제든지 unlimited 모드에서 필요한 기간 동안 버스트할 수 있습니다. unlimited 모드에 대한 자세한 내용은 성능 버스트 기능이 있는 인스턴스의 무제한 모드 사용 단원을 참조하세요.

- C4 인스턴스는 컴퓨팅 집약적 워크로드에 최적화되어 있으며 높은 컴퓨팅 가성비로 매우 비용 효율적인 고성능을 제공합니다. PPS (Packet Per Second) 성능을 크게 높이고 네트워크 지터를 낮추며 네트워크 지연 시간을 줄입니다. AWS DMS 또한 특히 Oracle에서 PostgreSQL로 마이그레이션하는 것과 같은 이기종 마이그레이션 및 복제를 수행할 때 CPU를 많이 사용할 수 있습니다. C4 인스턴스는 이러한 상황에서 훌륭한 선택이 될 수 있습니다.
- C5 인스턴스는 고급 컴퓨팅 집약적 워크로드를 실행하기 위해 높은 컴퓨팅 가성비로 비용 효율적인 고성능을 제공하는 차세대 인스턴스 유형입니다. 여기에는 고성능 웹 서버, 고성능 컴퓨팅(HPC), 배치 처리, 광고 서비스, 고확장성 멀티플레이어 게임, 비디오 인코딩 등의 워크로드가 포함됩니다. C5 인스턴스가 적합한 다른 워크로드에는 과학 모델링, 분산 분석, 기계 학습, 딥 러닝 추론 등이 있습니다. C5 인스턴스는 Intel 및 AMD의 다양한 프로세서를 사용할 수 있습니다.
- C6i 인스턴스는 다양한 워크로드에 대해 동급 5세대 인스턴스보다 최대 15% 더 우수한 컴퓨팅 가격 성능과 상시 메모리 암호화를 제공합니다. C6i 인스턴스는 배치 처리, 분산 분석, 고성능 컴퓨팅(HPC), 광고 서비스, 고확장성 멀티플레이어 게임, 비디오 인코딩과 같은 컴퓨팅 집약적 워크로드에 매우 적합합니다.
- R4 인스턴스는 메모리 최적화되어 메모리 집약적 워크로드에 적합합니다. AWS DMS 를 사용한 고처리량 트랜잭션 시스템의 지속적 마이그레이션 또는 복제 작업에서 많은 양의 CPU 및 메모리를 소비할 수도 있습니다. R4 인스턴스는 이전 세대 인스턴스 유형보다 vCPU당 더 많은 메모리를 제공합니다.
- R5 인스턴스는 Amazon EC2를 위한 차세대 메모리 최적화 인스턴스 유형입니다. R5 인스턴스는 고성능 데이터베이스, 분산형 웹 스케일 인 메모리 캐시, 중간 규모 인 메모리 캐시, 실시간 빅데이터 분석 및 기타 엔터프라이즈 애플리케이션 등 메모리 집약적 애플리케이션에 매우 적합합니다. 를 사용하여 처리량이 높은 트랜잭션 시스템을 지속적으로 마이그레이션하거나 복제하는 경우에도 많은 양의 CPU와 메모리를 소비할 수 있습니다. AWS DMS
- R6i 인스턴스는 다양한 워크로드에 대해 동급 5세대 인스턴스보다 최대 15% 더 우수한 컴퓨팅 가격 성능과 상시 메모리 암호화를 제공합니다. R6i 인스턴스는 SAP 인증을 받았으며 SQL 및 NoSQL 데이터베이스와 같은 워크로드, Memcached 및 Redis와 같은 분산형 웹 스케일 인 메모리 캐시, SAP HANA와 같은 인 메모리 데이터베이스, Hadoop 및 Spark 클러스터와 같은 실시간 빅데이터 분석에 적합합니다.

각 복제 인스턴스에는 특정 메모리 및 vCPU 구성이 있습니다. 다음 표에는 각 복제 인스턴스 유형별 구성이 나와 있습니다. 요금 정보는 AWS Database Migration Service 서비스 요금 페이지를 참조하세요.

범용 복제 인스턴스 유형

유형	vCPU	메모리(GiB)
dms.t2.micro	1	1
dms.t2.small	1	2
dms.t2.medium	2	4
dms.t2.large	2	8
dms.t3.micro	2	1
dms.t3.small	2	2
dms.t3.medium	2	4
dms.t3.large	2	8

컴퓨팅 최적화 복제 인스턴스 유형

유형	vCPU	메모리(GiB)
dms.c4.large	2	3.75
dms.c4.xlarge	4	7.5
dms.c4.2xlarge	8	15
dms.c4.4xlarge	16	30
dms.c5.large	2	4
dms.c5.xlarge	4	8

유형	vCPU	메모리(GiB)
dms.c5.2xlarge	8	16
dms.c5.4xlarge	16	32
dms.c5.9xlarge	36	72
dms.c5.12xlarge	48	96
dms.c5.18xlarge	72	144
dms.c5.24xlarge	96	192
dms.c6i.large	2	4
dms.c6i.xlarge	4	8
dms.c6i.2xlarge	8	16
dms.c6i.4xlarge	16	32
dms.c6i.8xlarge	32	64
dms.c6i.12xlarge	48	96
dms.c6i.16xlarge	64	128
dms.c6i.24xlarge	96	192
dms.c6i.32xlarge	128	256

메모리 최적화 복제 인스턴스 유형

유형	vCPU	메모리(GiB)
dms.r4.large	2	15.25
dms.r4.xlarge	4	30.5

유형	vCPU	메모리(GiB)
dms.r4.2xlarge	8	61
dms.r4.4xlarge	16	122
dms.r4.8xlarge	32	244
dms.r5.large	2	16
dms.r4.xlarge	4	32
dms.r5.2xlarge	8	64
dms.r5.4xLarge	16	128
dms.r5.8xlarge	32	256
dms.r5.12xLarge	48	384
dms.r5.16 x Large	64	512
dms.r5.24 xlarge	96	768
dms.r6i.large	2	16
dms.r6i.xlarge	4	32
dms.r6i.2xlarge	8	64
dms.r6i.4xlarge	16	128
dms.r6i.8xlarge	32	256
dms.r6i.12xlarge	48	384
dms.r6i.16xlarge	64	512
dms.r6i.24xlarge	96	768
dms.r6i.32xlarge	128	1024

위 표에는 모든 AWS DMS 복제 인스턴스 유형이 나열되어 있지만 사용 가능한 유형은 지역에 따라 다를 수 있습니다. 다음 AWS CLI 명령을 실행하여 해당 리전에서 사용 가능한 복제 인스턴스 유형을 확인할 수 있습니다.

```
aws dms describe-orderable-replication-instances --region your_region_name
```

주제

- 사용할 인스턴스 클래스 결정
- 성능 버스트 기능이 있는 인스턴스의 무제한 모드 사용

사용할 인스턴스 클래스 결정

가장 적합한 복제 인스턴스 클래스를 결정하는 데 도움이 되도록 AWS DMS 사용하는 변경 데이터 캡처 (CDC) 프로세스를 살펴보겠습니다.

사용자가 전체 로드 및 CDC 작업(대량 로드 및 지속적 복제)을 실행 중이라고 가정해 봅시다. 이 경우 해당 작업에는 자체 SQLite 리포지토리가 있어 메타데이터 및 기타 정보를 저장할 수 있습니다. 전체 로드를 AWS DMS 시작하기 전에 다음 단계를 수행합니다.

- AWS DMS 소스 엔진의 트랜잭션 로그에서 마이그레이션하는 테이블의 변경 사항 캡처를 시작합니다 (이를 캐시된 변경 내용이라고 함). 전체 로드가 완료되고 나면 이러한 캐시된 변경 사항이 수집되어 대상에 적용됩니다. 캐시된 변경 사항의 볼륨에 따라 이러한 변경 사항은 메모리로부터 직접 적용될 수 있습니다. 이 경우 변경 사항은 적용되기 전에 먼저 설정된 임계값에 이를 때까지 수집됩니다. 또는 디스크로부터 적용될 수 있습니다. 이 경우 변경 사항은 메모리에 유지될 수 없을 때 기록됩니다.
- 캐시된 변경 내용이 적용된 후에는 기본적으로 대상 AWS DMS 인스턴스에서 트랜잭션 적용 프로세스가 시작됩니다.

캐시된 변경 내용을 적용한 단계와 진행 중인 복제 단계에서는 들어오고 나가는 데이터에 각각 하나씩 총 두 개의 스트림 버퍼를 AWS DMS 사용합니다. AWS DMS 또 다른 메모리 버퍼인 분류기라는 중요한 구성 요소도 사용합니다. 다음은 분류기 구성 요소의 두 가지 중요 용도입니다(다른 용도도 있음).

- 분류기는 모든 트랜잭션을 추적하고 발신 버퍼에 관련이 있는 트랜잭션만 전달합니다.
- 이를 통해 트랜잭션이 원본에서와 동일한 커밋 순서로 전달됩니다.

보시다시피 이와 같은 AWS DMS의 CDC용 아키텍처에는 세 가지 중요 메모리 버퍼가 있습니다. 이 버퍼 중에 메모리 부족을 겪는 것이 있다면 해당 마이그레이션에는 장애를 유발할 가능성이 있는 성능 문제가 있을 수 있습니다.

초당 트랜잭션(TPS)이 높은 과중한 워크로드를 이 아키텍처에 플러깅하는 경우 R5 및 R6i 인스턴스에서 제공하는 추가 메모리가 유용할 수 있습니다. R5 및 R6i 인스턴스를 사용하여 메모리에 대규모 트랜잭션을 유지하고 지속적 복제 중에 메모리 부족 문제를 방지할 수 있습니다.

성능 버스트 기능이 있는 인스턴스의 무제한 모드 사용

T3 인스턴스와 같이 `unlimited`로 구성된 성능 버스트 기능이 있는 인스턴스는 필요한 경우 언제든지 원하는 기간 동안 높은 CPU 사용률을 유지할 수 있습니다. 시간당 인스턴스 가격은 모든 CPU 사용량 급증을 자동으로 충당할 수 있습니다. 이는 24시간 또는 인스턴스 수명 중 더 짧은 기간 동안 인스턴스의 평균 CPU 사용률이 기준 이하인 경우에 적용됩니다.

대부분의 범용 워크로드에서 `unlimited`로 구성된 인스턴스는 추가 요금 없이 충분한 성능을 제공합니다. 인스턴스 실행에 장기간 높은 CPU 사용률이 필요한 경우, vCPU-시간당 추가 고정 요금으로 인스턴스를 실행할 수 있습니다. T3 인스턴스 요금에 대한 자세한 내용은 AWS Database Migration Service의 'T3 CPU 크레딧'을 참조하세요.

T3 인스턴스의 `unlimited` 모드에 대한 자세한 내용은 Linux 인스턴스용 Amazon EC2 사용 설명서에서 성능 버스트 기능이 있는 인스턴스의 무제한 모드를 참조하세요.

> **Important**
>
> AWS 프리 티어 혜택이 적용되는 `dms.t3.micro` 인스턴스를 `unlimited` 모드에서 사용하는 경우에는 요금이 부과될 수 있습니다. 특히, 24시간 동안 평균 사용률이 인스턴스의 기준 사용률을 초과하면 요금이 적용될 수 있습니다. 자세한 내용은 Linux 인스턴스용 Amazon EC2 사용 설명서의 기준 사용률을 참조하세요.
>
> T3 인스턴스는 `unlimited`로 시작하도록 기본 설정되어 있습니다. 24시간 동안 평균 CPU 사용량이 기준을 초과하면 잉여 크레딧에 대한 요금이 발생합니다. 경우에 따라 T3 스팟 인스턴스를 `unlimited`로 시작하고 즉각적으로 단기간 사용하려는 경우가 있습니다. CPU 크레딧을 누적할 유휴 시간 없이 이렇게 하면 잉여 크레딧에 대한 요금이 발생합니다. 더 높은 비용을 지불하지 않으려면 표준 모드에서 T3 스팟 인스턴스를 시작하는 것이 좋습니다. 자세한 내용은 Linux 인스턴스용 Amazon EC2 사용 설명서의 잉여 크레딧으로 요금 발생 가능, T3 스팟 인스턴스 및 성능 버스트 기능이 있는 인스턴스의 표준 모드를 참조하세요.

복제 인스턴스에 가장 적합한 크기 선택

적절한 복제 인스턴스를 선택하는 것은 사용 사례의 여러 요소에 따라 달라집니다. 복제 인스턴스 리소스가 사용되는 방식을 이해하는 데 도움이 되도록 다음 설명을 참조하세요. 여기서는 전체 로드 + CDC 작업의 일반적인 시나리오를 다룹니다.

전체 로드 작업 중에는 테이블을 개별적으로 AWS DMS 로드합니다. 기본적으로 한 번에 8개의 테이블이 로드됩니다. AWS DMS 전체 로드 작업 중에 소스에 대한 진행 중인 변경 사항을 캡처하여 나중에 대상 엔드포인트에 변경 내용을 적용할 수 있습니다. 변경 사항은 메모리에 캐시되며, 사용 가능한 메모리가 모두 사용되면 변경 사항이 디스크에 캐시됩니다. 테이블에 대한 전체 로드 작업이 완료되면 캐시된 변경 내용을 대상 테이블에 AWS DMS 즉시 적용합니다.

테이블에 대한 모든 캐시 변경 사항이 적용된 후에는 대상 엔드포인트는 트랜잭션 일치 상태가 됩니다. 이때 타겟은 마지막으로 캐시된 변경 사항과 관련하여 소스 엔드포인트와 동기화됩니다. AWS DMS 그런 다음 소스와 타겟 간의 지속적인 복제를 시작합니다. 이를 위해 는 소스 트랜잭션 로그에서 변경 작업을 AWS DMS 가져와 트랜잭션적으로 일관된 방식으로 대상에 적용합니다. (이 프로세스에서는 일괄 최적화 적용이 선택되지 않은 것으로 가정합니다.) AWS DMS 가능한 경우 복제 인스턴스의 메모리를 통해 진행 중인 변경 사항을 스트리밍합니다. 그렇지 않으면 대상에 적용할 수 있을 때까지 복제 인스턴스의 AWS DMS 디스크에 변경 내용을 기록합니다.

복제 인스턴스의 변경 처리 방식과 해당 프로세스에서 메모리가 사용되는 방식을 어느 정도 제어할 수 있습니다. 변경 처리를 조정하는 방법에 대한 자세한 내용은 변경 처리 튜닝 설정 섹션을 참조하세요.

고려해야 할 요소

메모리 및 디스크 공간은 사용 사례에 적합한 복제 인스턴스를 선택할 때 가장 중요한 요소입니다. 이하에서는 복제 인스턴스를 선택하기 위해 분석해야 할 사용 사례 특성에 대한 설명합니다.

- 데이터베이스 및 테이블 크기

  데이터 볼륨은 전체 로드 성능을 최적화하기 위한 태스크 구성을 결정하는 데 도움이 됩니다. 예를 들어 1TB 스키마 2개에 대해 테이블을 500GB 태스크 4개로 파티션하여 병렬로 실행할 수 있습니다. 가능한 병렬 처리는 복제 인스턴스에서 사용할 수 있는 CPU 리소스에 따라 달라집니다. 따라서 전체 로드 성능을 최적화하려면 데이터베이스와 테이블의 크기를 이해하는 것이 좋습니다. 이는 수행할 수 있는 태스크의 수를 결정하는 데 도움이 됩니다.
- 대형 객체

  마이그레이션 범위에 있는 데이터 형식은 성능에 영향을 줄 수 있습니다. 특히 대형 객체(LOB)는 성능과 메모리 소비에 영향을 미칩니다. LOB 값을 마이그레이션하려면 2단계 프로세스를 AWS DMS

수행합니다. 먼저 LOB 값 없이 대상에 행을 AWS DMS 삽입합니다. 그런 다음 LOB 값으로 행을 AWS DMS 업데이트합니다. 이는 메모리에 영향을 미치므로 소스에서 LOB 열을 식별하고 크기를 분석하는 것이 중요합니다.

- 로드 빈도 및 트랜잭션 크기

  로드 빈도와 초당 트랜잭션 수(TPS)는 메모리 사용량에 영향을 줍니다. TPS 또는 데이터 조작 언어(DML) 활동이 많으면 메모리 사용량이 많아집니다. DMS는 변경 사항이 대상에 적용될 때까지 변경 사항을 캐시하기 때문입니다. CDC 중에 이로 인해 스와핑(메모리 오버플로로 인한 물리적 디스크에 쓰기)이 발생하여 지연 시간이 발생합니다.

- 테이블 키 및 참조 무결성

  테이블의 키에 대한 정보에 따라 데이터 마이그레이션에 사용하는 CDC 모드(배치 적용 또는 트랜잭션 적용)가 결정됩니다. 일반적으로 트랜잭션 적용은 배치 적용보다 느립니다. 장기 실행 트랜잭션의 경우 마이그레이션해야 할 변경 사항이 많을 수 있습니다. 트랜잭션 적용을 사용하는 경우 일괄 적용에 비해 변경 내용을 저장하는 데 더 많은 메모리가 AWS DMS 필요할 수 있습니다. 프라이머리 키가 없는 테이블을 마이그레이션하는 경우 배치 적용이 실패하고 DMS 작업이 트랜잭션 적용 모드로 전환됩니다. CDC 중에 테이블 간 참조 무결성이 활성화되면 기본적으로 트랜잭션 적용을 AWS DMS 사용합니다. 배치 적용과 트랜잭션 적용의 비교에 대한 자세한 내용은 DMS 배치 적용 기능을 사용하여 CDC 복제 성능을 개선하려면 어떻게 해야 합니까?를 참조하세요.

이러한 지표를 사용하여 복제 인스턴스를 컴퓨팅 최적화해야 하는지 아니면 메모리 최적화해야 하는지 결정하세요.

일반적인 문제

마이그레이션 중에 다음과 같이 복제 인스턴스에서 리소스 경합을 유발하는 일반적인 문제가 발생할 수 있습니다. 복제 인스턴스 지표에 대한 자세한 내용은 복제 인스턴스 지표 섹션을 참조하세요.

- 복제 인스턴스의 메모리가 부족해지면 데이터가 디스크에 쓰여집니다. 디스크에서 읽을 때 지연 시간이 발생할 수 있는데, 충분한 메모리로 복제 인스턴스 크기를 조정하면 지연 시간을 피할 수 있습니다.
- 복제 인스턴스에 할당된 디스크 크기는 필요한 것보다 작을 수 있습니다. 디스크 크기는 메모리에서 데이터가 넘칠 때 사용되며 태스크 로그를 저장하는 데도 사용됩니다. 최대 IOPS도 이에 따라 달라집니다.
- 여러 태스크를 실행하거나 병렬성이 높은 태스크를 실행하면 복제 인스턴스의 CPU 사용량에 영향을 줍니다. 이로 인해 태스크 처리 속도가 느려지고 지연 시간이 길어집니다.

모범 사례

복제 인스턴스의 크기를 조정할 때는 다음과 같이 가장 일반적인 2가지 모범 사례를 고려해야 합니다. 자세한 정보는 AWS Database Migration Service의 모범 사례을 참조하세요.

1. 워크로드의 규모를 파악하고 컴퓨팅 집약적인지 메모리 집약적인지 판단하세요. 이를 기반으로 복제 인스턴스의 클래스 및 크기를 결정할 수 있습니다.
   - AWS DMS 메모리에서 LOB를 처리합니다. 이 작업에는 상당한 양의 메모리가 필요합니다.
   - 태스크 수와 스레드 수는 CPU 사용량에 영향을 줍니다. 전체 로드 작업 중에는 `MaxFullLoadSubTasks`를 8개 이상 사용하지 마세요.
2. 전체 로드 중에 워크로드가 많으면 복제 인스턴스에 할당된 디스크 공간을 늘리세요. 이렇게 하면 복제 인스턴스가 할당된 최대 IOPS를 사용할 수 있습니다.

위 가이드라인에서 가능한 시나리오를 모두 다루는 것은 아닙니다. 복제 인스턴스의 크기를 결정할 때 특정 사용 사례의 구체적인 내용을 고려하는 것이 중요합니다.

이전 테스트에서 CPU와 메모리가 워크로드에 따라 달라지는 것으로 나타났습니다. 특히 LOB는 메모리에 영향을 미치고 태스크 수 또는 병렬 처리는 CPU에 영향을 미칩니다. 마이그레이션을 실행한 후에는 복제 인스턴스의 CPU, 여유 메모리, 여유 스토리지, IOPS를 모니터링하세요. 수집한 데이터를 기반으로 필요에 따라 복제 인스턴스의 크기를 늘리거나 줄일 수 있습니다.

복제 엔진 버전 작업

복제 엔진은 복제 인스턴스에서 실행되고 지정된 마이그레이션 작업을 수행하는 핵심 AWS DMS 소프트웨어입니다. AWS 새 기능 및 성능 개선을 포함한 새 버전의 AWS DMS 복제 엔진 소프트웨어를 정기적으로 릴리스합니다. 복제 엔진 소프트웨어의 각 버전마다 고유한 버전 번호가 있어서 버전 간에 구분됩니다.

새 복제 인스턴스를 시작하면 달리 지정하지 않는 한 최신 AWS DMS 엔진 버전이 실행됩니다. 자세한 정보는 AWS DMS 복제 인스턴스 사용을 참조하세요.

현재 실행 중인 복제 인스턴스가 있는 경우 이를 최신 엔진 버전으로 업그레이드할 수 있습니다. (엔진 버전 다운그레이드는AWS DMS 지원하지 않습니다.) 복제 엔진 버전에 대한 자세한 내용은 AWS DMS 릴리스 노트 섹션을 참조하세요.

콘솔을 사용하여 엔진 버전 업그레이드

를 사용하여 AWS DMS 복제 인스턴스를 업그레이드할 수 있습니다. AWS Management Console

콘솔을 사용하여 복제 인스턴스를 업그레이드하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 [Replication instances]를 선택합니다.
3. 복제 엔진을 선택한 다음 [Modify]를 선택합니다.
4. 엔진 버전에서 원하는 버전 번호를 선택한 다음 수정을 선택합니다.

> **Note**
>
> 복제 인스턴스를 업그레이드하기 전에 모든 태스크를 중지하는 것이 좋습니다. 작업을 중지하지 않으면 업그레이드 전에 작업이 자동으로 중지됩니다. AWS DMS 태스크를 수동으로 중지하는 경우 업그레이드가 완료된 후 태스크를 수동으로 시작해야 합니다. 복제 인스턴스를 업그레이드하는 데는 몇 분이 걸립니다. 인스턴스가 준비되면 인스턴스의 상태가 [available]로 변경됩니다.

를 사용하여 엔진 버전 업그레이드 AWS CLI

다음과 같이 를 사용하여 AWS DMS 복제 인스턴스를 업그레이드할 수 있습니다. AWS CLI

를 사용하여 복제 인스턴스를 업그레이드하려면 AWS CLI

1. 다음 명령을 사용하여 복제 인스턴스의 ARN(Amazon 리소스 이름)을 확인합니다.

   ```
   aws dms describe-replication-instances \
   --query "ReplicationInstances[*].
   [ReplicationInstanceIdentifier,ReplicationInstanceArn,ReplicationInstanceClass]"
   ```

   출력에서 업그레이드할 복제 인스턴스에 대한 ARN을 적어둡니다. 예: `arn:aws:dms:us-east-1:123456789012:rep:6EFQQO6U6EDPRCPKLNPL2SCEEY`
2. 다음 명령을 사용하여 사용할 수 있는 복제 인스턴스 버전을 확인합니다.

   ```
   aws dms describe-orderable-replication-instances \
   --query "OrderableReplicationInstances[*].[ReplicationInstanceClass,EngineVersion]"
   ```

   출력에서 복제 인스턴스 클래스에 사용할 수 있는 엔진 버전 번호를 적어둡니다. 1단계의 출력에 이 정보가 표시됩니다.

3. 다음 명령을 사용하여 복제 인스턴스를 업그레이드합니다.

```
aws dms modify-replication-instance \
--replication-instance-arn arn \
--engine-version n.n.n
```

앞의 *arn*을 전 단계에서 확인한 실제 복제 인스턴스 ARN으로 바꿉니다.

*n.n.n*을 원하는 엔진 버전으로 바꿉니다. 예: `3.4.5`

> **Note**
>
> 복제 인스턴스를 업그레이드하는 데는 몇 분이 걸립니다. 다음 명령을 사용하여 복제 인스턴스 상태를 볼 수 있습니다.
>
> ```
> aws dms describe-replication-instances \
> --query "ReplicationInstances[*].
> [ReplicationInstanceIdentifier,ReplicationInstanceStatus]"
> ```
>
> 복제 인스턴스가 준비되면 인스턴스의 상태가 [available]로 변경됩니다.

퍼블릭 및 프라이빗 복제 인스턴스

복제 인스턴스가 원본과 대상 데이터베이스에 연결하는 데 사용할 퍼블릭 또는 프라이빗 IP 주소 여부를 지정할 수 있습니다.

프라이빗 복제 인스턴스에는 복제 네트워크 외부에서 액세스할 수 없는 프라이빗 IP 주소가 있습니다. 소스 및 대상 데이터베이스가 모두 복제 인스턴스의 Virtual Private Cloud(VPC)에 연결된 동일한 네트워크에 있는 경우 프라이빗 인스턴스를 사용합니다. 네트워크는 가상 사설망 (VPN) 또는 VPC 피어링을 사용하여 VPC에 연결할 수 있습니다. AWS Direct Connect

VPC 피어링 연결은 두 VPC 간의 네트워킹 연결입니다. 이를 통해 각 VPC의 프라이빗 IP 주소를 동일한 네트워크에 있는 것처럼 사용하여 라우팅할 수 있습니다. VPC 피어링에 관한 자세한 내용은 Amazon VPC 사용 설명서에서 VPC 피어링을 참조하세요.

퍼블릭 복제 인스턴스는 복제 인스턴스의 VPC 보안 그룹과 복제 인스턴스의 퍼블릭 IP 주소 또는 NAT 게이트웨이의 퍼블릭 IP 주소를 사용할 수 있습니다. 이 연결은 데이터 마이그레이션에서 사용하는 네트워크를 구성합니다.

IP 주소 지정 및 네트워크 유형

AWS DMS 항상 Amazon VPC (가상 사설 클라우드) 에서 복제 인스턴스를 생성합니다. VPC를 생성할 때 사용할 IP 주소 지정(IPv4, IPv6 또는 둘 다)을 결정할 수 있습니다. 그런 다음 복제 인스턴스를 생성하거나 수정할 때 이중 스택 모드를 사용하여 IPv4 주소 프로토콜 또는 IPv6 주소 프로토콜의 사용을 지정할 수 있습니다.

IPv4 주소

VPC를 생성할 때 VPC의 IPv4 주소 범위를 Classless Inter-Domain Routing(CIDR) 블록 형태로 지정할 수 있습니다(예: 10.0.0.0/16). 서브넷 그룹은 이 CIDR 블록에서 IP 주소의 범위를 정의합니다. 이 IP 주소는 프라이빗 또는 퍼블릭일 수 있습니다.

프라이빗 IPv4 주소는 인터넷을 통해 연결할 수 없는 IP 주소입니다. 프라이빗 IPv4 주소는 복제 인스턴스 및 같은 VPC 내의 Amazon EC2 인스턴스와 같은 기타 리소스 간의 통신을 위해 사용될 수 있습니다. 각 복제 인스턴스에는 VPC 내 통신을 위한 프라이빗 IP 주소가 있습니다.

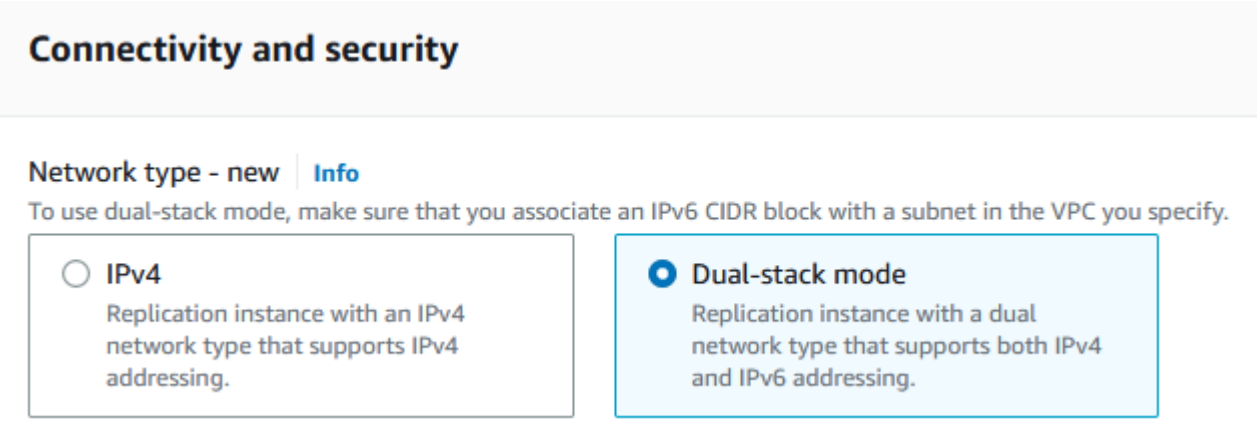
퍼블릭 IP 주소는 인터넷을 통해 연결할 수 있는 IPv4 주소입니다. 퍼블릭 주소는 복제 인스턴스와 인터넷 상의 리소스 사이의 통신을 위해 사용될 수 있습니다. 복제 인스턴스가 퍼블릭 IP 주소를 수신할지 여부를 제어할 수 있습니다.

이중 스택 모드 및 IPv6 주소

IPv6를 통해 복제 인스턴스와 통신해야 하는 리소스가 있는 경우 이중 스택 모드를 사용합니다. 이중 스택 모드를 사용하려면 복제 인스턴스와 연결하는 DB 서브넷 그룹의 각 서브넷에 IPv6 CIDR 블록이 연결되어 있어야 합니다. 새 복제 서브넷 그룹을 생성하거나 기존 복제 서브넷 그룹을 수정하여 이 요구 사항을 충족하도록 수정할 수 있습니다. 각 IPv6 주소는 전역적으로 고유합니다. VPC에 대한 IPv6 CIDR 블록은 Amazon의 IPv6 주소 풀에서 자동으로 할당되므로 범위를 직접 선택할 수는 없습니다.

DMS는 프라이빗 이중 스택 모드 복제 인스턴스의 IPv6 엔드포인트에 대한 인터넷 게이트웨이 액세스를 비활성화합니다. DMS는 IPv6 엔드포인트가 프라이빗이며 VPC 내에서만 액세스할 수 있도록 하기 위해 이 작업을 수행합니다.

AWS DMS 콘솔을 사용하여 복제 인스턴스를 생성 또는 수정하고 네트워크 유형 섹션에서 이중 스택 모드를 지정할 수 있습니다. 다음 이미지는 콘솔의 네트워크 유형 섹션을 보여줍니다.



참조

- IPv4 및 IPv6 주소에 대한 자세한 내용은 Amazon VPC 사용 설명서의 IP 주소 지정을 참조하세요.
- 이중 스택 모드를 사용한 복제 인스턴스 생성에 대한 자세한 내용은 복제 인스턴스 생성 단원을 참조하세요.
- 복제 인스턴스 수정에 대한 자세한 내용은 복제 인스턴스 수정 섹션을 참조하세요.

복제 인스턴스용으로 네트워크 설정

AWS DMS는 항상 Amazon VPC를 기반으로 하는 VPC에 복제 인스턴스를 생성합니다. 복제 인스턴스의 위치할 VPC는 사용자가 지정합니다. 계정 및 AWS 지역에 기본 VPC를 사용하거나 새 VPC를 만들 수 있습니다.

복제 인스턴스의 VPC에 할당된 탄력적 네트워크 인터페이스가 보안 그룹과 연결되어 있는지 확인해야 합니다. 또한 이 보안 그룹의 규칙이 모든 포트에서 모든 트래픽이 VPC를 벗어나는 것(송신)을 허용해야 합니다. 이 접근 방식을 통해 엔드포인트에서 올바른 수신 규칙이 활성화되어 있는 한 통신을 복제 인스턴스에서 소스 및 대상 데이터베이스 엔드포인트로 전달할 수 있습니다. 모든 포트에서 모든 주소로의 송신을 허용할 수 있도록 엔드포인트에서 기본 설정을 사용하는 것이 좋습니다.

원본과 대상 엔드포인트는 VPC에 연결하거나 VPC 내부에 있어 VPC 내부에 있는 복제 인스턴스에 액세스합니다. 데이터베이스 엔드포인트는 복제 인스턴스에서 수신되는 액세스를 허용하는 네트워크 ACL(액세스 제어 목록)과 보안 그룹 규칙(해당하는 경우)을 포함해야 합니다. 설정 방법은 사용하는 네트워크 구성에 따라 다릅니다. 복제 인스턴스 VPC 보안 그룹, 복제 인스턴스의 프라이빗 또는 퍼블릭 IP 주소 또는 NAT 게이트웨이의 퍼블릭 IP 주소를 사용할 수 있습니다. 이 연결은 데이터 마이그레이션에서 사용하는 네트워크를 구성합니다.

> **Note**
>
> 기본 인프라 변경으로 인해 IP 주소가 달라질 수 있으므로 VPC CIDR 범위를 사용하거나 NAT GW 연결 탄력적 IP를 통해 복제 인스턴스 아웃바운드 트래픽을 라우팅하는 것이 좋습니다. CIDR 블록을 포함하여 VPC 생성에 대한 자세한 내용은 Amazon Virtual Private Cloud 사용 설명서의 VPC 및 서브넷 사용을 참조하세요. 탄력적 IP 주소에 대한 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 탄력적 IP 주소를 참조하세요.

데이터베이스 마이그레이션을 위한 네트워크 구성

AWS Database Migration Service에서는 여러 가지 네트워크 구성을 사용할 수 있습니다. 다음은 데이터베이스 마이그레이션에 사용되는 네트워크의 일반적인 구성입니다.

주제

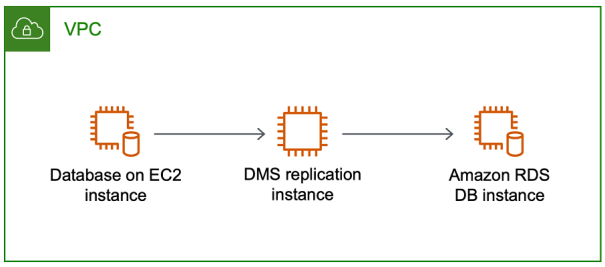
- 하나의 VPC에 모든 데이터베이스 마이그레이션 구성 요소가 있는 구성
- 여러 VPC를 사용한 구성
- 공유 VPC를 사용한 구성
- AWS Direct Connect 또는 VPN을 사용하여 VPC에 대한 네트워크 구성
- 인터넷을 사용한 네트워크와 VPC 연결 구성
- 를 사용하여 VPC에 있지 않은 RDS DB 인스턴스를 VPC의 DB 인스턴스로 구성 ClassicLink

가능하면 대상 엔드포인트와 동일한 리전, 대상 엔드포인트와 동일한 VPC 또는 서브넷에 DMS 복제 인스턴스를 생성하는 것이 좋습니다.

하나의 VPC에 모든 데이터베이스 마이그레이션 구성 요소가 있는 구성

데이터베이스 마이그레이션에서 가장 간단한 네트워크는 원본 엔드포인트, 복제 인스턴스, 대상 엔드포인트가 모두 동일한 VPC에 있도록 하는 것입니다. 이 구성은 소스 및 대상 엔드포인트가 Amazon RDS DB 인스턴스 또는 Amazon EC2 인스턴스에 있는 경우에 잘 맞습니다.

다음 그림은 Amazon EC2 인스턴스의 데이터베이스가 복제 인스턴스에 연결되고 데이터가 Amazon RDS DB 인스턴스에 마이그레이션되는 경우의 구성을 보여줍니다.



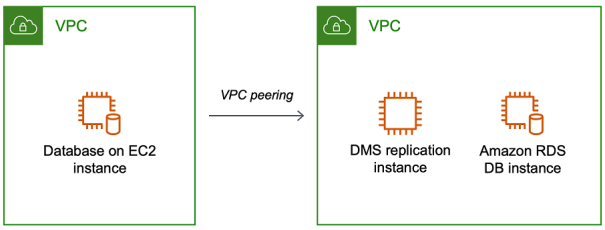
이 구성에서 사용되는 VPC 보안 그룹은 복제 인스턴스의 데이터베이스 포트에서 수신을 허용해야 합니다. 이 작업은 두 가지 방법으로 수행할 수 있습니다. 복제 인스턴스에서 사용하는 보안 그룹이 엔드포인트에서 수신을 허용하도록 할 수 있습니다. 또는 복제 인스턴스의 VPC CIDR 범위, NAT GW 탄력적 IP 또는 프라이빗 IP 주소(사용 중인 경우)를 허용할 수 있습니다. 하지만 복제 인스턴스의 프라이빗 IP 주소는 사용하지 않는 것이 좋습니다. 복제 IP 주소가 변경되면 복제가 중단될 수 있기 때문입니다.

여러 VPC를 사용한 구성

소스 엔드포인트와 대상 엔드포인트가 다른 VPC에 있는 경우, VPC 중 하나에서 복제 인스턴스를 생성할 수 있습니다. 그런 다음 VPC 피어링을 사용하여 두 VPC를 연결할 수 있습니다.

VPC 피어링 연결은 두 VPC가 동일한 네트워크에 있는 것처럼 각 VPC의 프라이빗 IP 주소를 사용하여 라우팅을 지원하는 두 VPC 간 네트워크 연결입니다. VPC 간 AWS , 다른 계정의 VPC 또는 다른 지역의 VPC 간에 VPC 피어링 연결을 생성할 수 있습니다. AWS VPC 피어링에 관한 자세한 내용은 Amazon VPC 사용 설명서에서 VPC 피어링을 참조하세요.

다음 그림에 VPC 피어링을 사용한 구성 예가 나와 있습니다. 여기서, VPC의 Amazon EC2 인스턴스에 있는 소스 데이터베이스는 VPC 피어링에 의해 VPC에 연결됩니다. 이 VPC에는 Amazon RDS DB 인스턴스의 복제 인스턴스 및 대상 데이터베이스가 들어 있습니다.



VPC 피어링을 구현하려면 Amazon Virtual Private Cloud, VPC 피어링 설명서에 있는 VPC 피어링 연결 사용의 지침을 따르세요. 한 VPC의 라우팅 테이블에 다른 VPC의 CIDR 블록이 포함되어야 합니다. 예를 들어 VPC A가 대상 10.0.0.0/16을 사용하고 VPC B가 대상 172.31.0.0을 사용하는 경우 VPC A의 라우팅 테이블에는 172.31.0.0이 포함되고 VPC B의 라우팅 테이블에는 10.0.0.0/16이 포함되어야 합니다. 자세한 내용은 Amazon Virtual Private Cloud, VPC 피어링 설명서의 VPC 피어링 연결에 대한 라우팅 테이블 업데이트를 참조하세요.

이 구성에 사용되는 VPC 보안 그룹은 복제 인스턴스의 데이터베이스 포트에서 수신을 허용하거나 피어링되는 VPC의 CIDR 블록에서 수신을 허용해야 합니다.

공유 VPC를 사용한 구성

AWS DMS 조직의 참여 고객 계정에 공유되는 서브넷을 동일한 계정의 일반 서브넷처럼 취급합니다. 다음은 VPC, 서브넷을 AWS DMS 처리하는 방법 및 공유 VPC를 사용하는 방법에 대한 설명입니다.

ReplicationSubnetGroup 객체를 생성하여 사용자 지정 서브넷 또는 VPC에서 작동하도록 네트워크 구성을 구성할 수 있습니다. ReplicationSubnetGroup을 생성할 때 계정에 있는 특정 VPC의 서브넷을 지정할 수 있습니다. 지정하는 서브넷 목록에는 별도의 가용 영역에 있는 서브넷이 두 개 이상 포함되어야 하며 모든 서브넷은 동일한 VPC에 있어야 합니다. 를 생성할 때 고객은 ReplicationSubnetGroup 서브넷만 지정합니다. AWS DMS 각 서브넷이 정확히 하나의 VPC에 연결되어 있으므로 사용자를 대신하여 VPC를 결정합니다.

또는 AWS DMS ReplicationConfig a를 생성할 때 AWS DMS ReplicationInstance ReplicationInstance 또는 서버리스 복제가 작동하는 VPC 보안 그룹 ReplicationSubnetGroup 및/또는 하나를 지정하도록 선택할 수 있습니다. 지정하지 않는 경우 고객 기본값 ReplicationSubnetGroup (기본 VPC의 모든 서브넷에 대해 지정되지 않은 경우 사용자 대신 AWS DMS 생성) 및 기본 VPC 보안 그룹을 AWS DMS 선택합니다.

지정한 가용 영역 또는 ReplicationSubnetGroup의 가용 영역 중 하나에서 마이그레이션을 실행하도록 선택할 수 있습니다. 복제 인스턴스를 만들거나 서버리스 복제를 AWS DMS 시작하려고 하면 서브넷의 가용 영역이 핵심 서비스 계정의 가용 영역으로 변환되므로 두 계정 간의 가용 영역 매핑이 동일하지 않더라도 올바른 가용 영역에서 인스턴스를 시작할 수 있습니다.

공유 VPC를 사용하는 경우 공유 VPC에서 사용하려는 서브넷에 매핑되는 ReplicationSubnetGroup 객체를 생성해야 합니다. ReplicationInstance 또는 ReplicationConfig를 생성할 때 공유 VPC에 대해 ReplicationSubnetGroup을 지정하고, Create 요청을 통해 공유 VPC용으로 생성한 VPC 보안 그룹을 지정해야 합니다.

공유 VPC 사용 시 다음에 유의하세요.

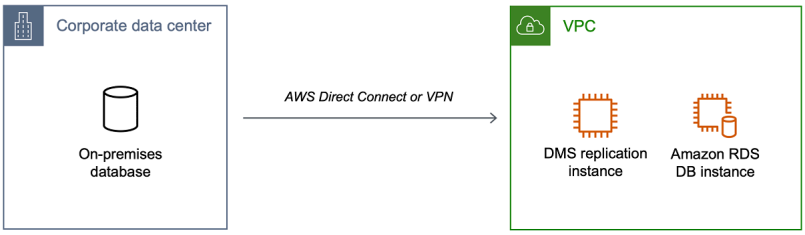
- VPC 소유자는 참여자와 리소스를 공유할 수 없지만 참여자는 소유자의 서브넷에서 서비스 리소스를 생성할 수 있습니다.
- 모든 리소스는 계정별이므로 VPC 소유자는 참여자가 생성한 리소스(예: 복제 인스턴스)에 액세스할 수 없습니다. 하지만 공유 VPC에서 복제 인스턴스를 생성하면 복제 엔드포인트 또는 태스크에 올바른 권한이 있는 한 소유 계정과 관계없이 VPC의 리소스에 액세스할 수 있습니다.

- 리소스는 계정별이므로 다른 참여자는 다른 계정이 소유한 리소스에 액세스할 수 없습니다. 공유 VPC에서 생성된 리소스에 자신의 계정으로 액세스할 수 있도록 다른 계정에 부여할 수 있는 권한은 없습니다.

AWS Direct Connect 또는 VPN을 사용하여 VPC에 대한 네트워크 구성

원격 네트워크는 Direct Connect 또는 소프트웨어 또는 하드웨어 VPN AWS 연결과 같은 여러 옵션을 사용하여 VPC에 연결할 수 있습니다. 이 옵션은 흔히 내부 네트워크를 AWS 클라우드로 확장함으로써 모니터링, 인증, 보안, 데이터 또는 기타 시스템과 같은 기존 현장 서비스를 통합하는 데 사용됩니다. 이러한 유형의 네트워크 확장을 활용하여 VPC와 같이 AWS호스팅 리소스에 완벽하게 연결할 수 있습니다.

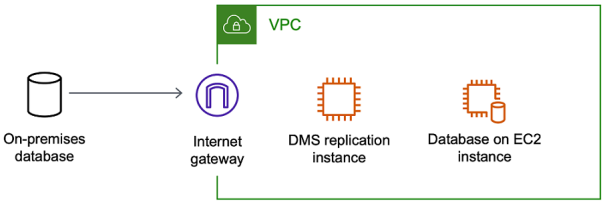
다음 그림은 원본 엔드포인트가 기업 데이터 센터에서 온프레미스 데이터베이스인 구성을 나타냅니다. Amazon RDS DB 인스턴스에서 복제 인스턴스와 대상 데이터베이스를 포함하는 AWS Direct Connect 또는 VPN-VPC를 사용하여 연결합니다.



이 구성에서 VPC 보안 그룹에는 VPC CIDR 범위 또는 특정 IP 주소로 지정된 트래픽을 호스트로 전송하는 라우팅 규칙이 포함되어야 합니다. 이 호스트는 트래픽을 VPC에서 온프레미스 VPN으로 연결할 수 있어야 합니다. 이 경우, NAT 호스트에는 자체 보안 그룹 설정이 포함됩니다. 이러한 설정은 복제 인스턴스의 VPC CIDR 범위, 프라이빗 IP 주소 또는 보안 그룹에서 NAT 인스턴스로 들어오는 트래픽을 허용해야 합니다. 하지만 복제 인스턴스의 프라이빗 IP 주소는 사용하지 않는 것이 좋습니다. 복제 IP 주소가 변경되면 복제가 중단될 수 있기 때문입니다.

인터넷을 사용한 네트워크와 VPC 연결 구성

VPN을 사용하지 않거나 AWS Direct Connect AWS 리소스에 연결하지 않는 경우 인터넷을 사용하여 데이터베이스를 마이그레이션할 수 있습니다. 이 경우 Amazon EC2 인스턴스 또는 Amazon RDS DB 인스턴스로 마이그레이션할 수 있습니다. 이 구성은 대상 엔드포인트와 복제 인스턴스를 포함하는 인터넷 게이트웨이를 지원하는 VPC에 퍼블릭 복제 인스턴스를 포함합니다.



VPC에 인터넷 게이트웨이를 추가하려면 Amazon VPC 사용 설명서의 인터넷 게이트웨이 연결을 참조하세요.

VPC 라우팅 테이블에는 기본적으로 VPC로 향하지 않는 트래픽을 인터넷 게이트웨이로 보내는 라우팅 규칙이 포함되어야 합니다. 이 구성에서 엔드포인트와의 연결이 프라이빗 IP 주소가 아닌 복제 인스턴스의 퍼블릭 IP 주소에서 오는 것으로 나타납니다. 자세한 내용은 Amazon VPC 사용 설명서의 VPC 라우팅 테이블을 참조하세요.

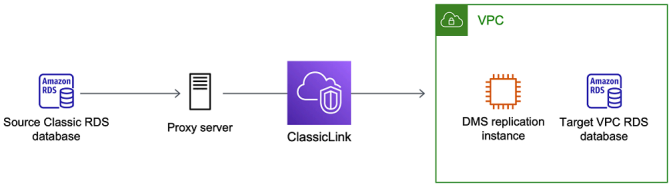
를 사용하여 VPC에 있지 않은 RDS DB 인스턴스를 VPC의 DB 인스턴스로 구성 ClassicLink

> EC2-Classic은 2022년 8월 15일에 사용 중지될 예정입니다. EC2-Classic에서 VPC로 마이그레이션하는 것이 좋습니다. 자세한 내용은 Amazon EC2 사용 설명서의 Migrate from EC2-Classic to a VPC(EC2-Classic에서 VPC로 마이그레이션) 및 EC2-Classic Networking is Retiring – Here's How to Prepare(EC2-Classic 네트워킹 지원 중단에 대비하는 방법) 블로그 게시물을 참조하세요.

VPC에 있지 않은 Amazon RDS DB 인스턴스를 VPC의 DMS 복제 서버 및 DB 인스턴스에 연결하려면 프록시 서버와 함께 사용할 수 있습니다 ClassicLink .

ClassicLink 동일한 지역 내에서 EC2-Classic DB 인스턴스를 계정의 VPC에 연결할 수 있습니다. AWS 링크를 생성하고 난 후 원본 DB 인스턴스는 프라이빗 IP 주소를 사용하여 VPC 내부에 있는 복제 인스턴스와 통신할 수 있습니다.

VPC의 복제 인스턴스는 를 사용하여 EC2-Classic 플랫폼의 원본 DB 인스턴스에 직접 액세스할 수 없으므로 프록시 ClassicLink 서버를 사용합니다. 프록시 서버는 원본 DB 인스턴스를 VPC에 연결하며, 여기에는 복제 인스턴스와 대상 DB 인스턴스가 포함됩니다. 프록시 서버는 ClassicLink VPC에 연결하는 데 사용합니다. 프록시 서버의 포트 전달은 VPC에서 원본 DB 인스턴스와 대상 DB 인스턴스 통신을 허용합니다.



AWS Database Migration ClassicLink Service와 함께 사용

VPC에 있지 않은 Amazon RDS DB 인스턴스를 VPC에 있는 DMS 복제 서버 및 DB 인스턴스에 연결할 수 있습니다. AWS 이를 위해 Amazon EC2를 프록시 ClassicLink 서버와 함께 사용할 수 있습니다.

다음 절차는 이 ClassicLink 용도로 사용하는 방법을 보여줍니다. 이 절차는 VPC에 있지 않은 Amazon RDS 원본 DB 인스턴스를 DMS 복제 인스턴스 및 대상 DB AWS 인스턴스를 포함하는 VPC에 연결합니다.

- VPC에서 AWS DMS 복제 인스턴스를 생성합니다. (모든 복제 인스턴스는 VPC에서 생성됩니다).
- VPC 보안 그룹을 복제 인스턴스와 대상 DB 인스턴스에 연결합니다. 두 인스턴스가 VPC 보안 그룹을 공유할 때에는 기본적으로 이 두 인스턴스가 서로 통신할 수 있습니다.
- EC2 Classic 인스턴스에서 프록시 서버를 설정합니다.
- 프록시 서버와 VPC ClassicLink 사이를 사용하여 연결을 생성합니다.
- 소스 및 대상 데이터베이스를 위한 AWS DMS 엔드포인트를 생성합니다.
- AWS DMS 작업을 생성합니다.

VPC에 있지 않은 DB 인스턴스의 데이터베이스를 VPC에 있는 DB 인스턴스의 데이터베이스로 마이그레이션하는 ClassicLink 데 사용합니다.

1. AWS DMS 복제 인스턴스를 생성하고 VPC 보안 그룹을 할당합니다.

    a. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.

    AWS Identity and Access Management (IAM) 사용자로 로그인한 경우 적절한 액세스 AWS DMS권한이 있는지 확인하세요. 데이터베이스 마이그레이션에 필요한 권한에 대한 자세한 내용은 AWS DMS 사용에 필요한 IAM 권한 섹션을 참조하세요.

    b. [Dashboard] 페이지에서 [Replication Instance]를 선택합니다. 1단계: AWS DMS 콘솔을 사용하여 복제 인스턴스 생성의 지침에 따라 복제 인스턴스를 생성합니다.

    c. AWS DMS 복제 인스턴스를 생성한 후 EC2 서비스 콘솔을 엽니다. 탐색 창에서 네트워크 인스턴스를 선택합니다.

    d. NetworkInterfaceDMS를 선택한 다음 작업 메뉴에서 보안 그룹 변경을 선택합니다.

    e. 복제 인스턴스와 대상 DB 인스턴스에서 사용할 보안 그룹을 선택합니다.

2. 마지막 단계에서 선택한 보안 그룹을 대상 DB 인스턴스와 연결합니다.

    a. Amazon RDS 서비스 콘솔을 엽니다. 탐색 창에서 인스턴스를 선택합니다.

b. 대상 DB 인스턴스를 선택합니다. 인스턴스 작업에서 수정을 선택합니다.

c. 보안 그룹 파라미터에서 이전 단계에서 사용한 보안 그룹을 선택합니다.

d. 계속을 선택하고 DB 인스턴스 수정을 선택합니다.

3. 3단계: NGINX를 사용하여 EC2 Classic 인스턴스에서 프록시 서버를 설정합니다. 선택한 AMI를 사용하여 EC2 Classic 인스턴스를 시작합니다. 다음 예제는 AMI Ubuntu Server 14.04 LTS(HVM)를 기반으로 합니다.

   EC2 Classic 인스턴스에서 프록시 서버를 설정하려면

   a. 다음 명령을 사용하여 EC2 Classic 인스턴스에 연결하고 NGINX를 설치합니다.

```
Prompt> sudo apt-get update
Prompt> sudo wget http://nginx.org/download/nginx-1.9.12.tar.gz
Prompt> sudo tar -xvzf nginx-1.9.12.tar.gz
Prompt> cd nginx-1.9.12
Prompt> sudo apt-get install build-essential
Prompt> sudo apt-get install libpcre3 libpcre3-dev
Prompt> sudo apt-get install zlib1g-dev
Prompt> sudo ./configure --with-stream
Prompt> sudo make
Prompt> sudo make install
```

   b. 다음 코드를 사용하여 NGINX 데몬 파일 `/etc/init/nginx.conf`를 편집합니다.

```
# /etc/init/nginx.conf – Upstart file

description "nginx http daemon"
author "email"

start on (filesystem and net-device-up IFACE=lo)
stop on runlevel [!2345]

env DAEMON=/usr/local/nginx/sbin/nginx
env PID=/usr/local/nginx/logs/nginx.pid

expect fork
respawn
respawn limit 10 5
```

```
pre-start script
        $DAEMON -t
        if [ $? -ne 0 ]
                then exit $?
        fi
end script

exec $DAEMON
```

c. /usr/local/nginx/conf/nginx.conf에 NGINX 구성 파일을 만듭니다. 구성 파일에서 다음을 추가합니다.

```
# /usr/local/nginx/conf/nginx.conf - NGINX configuration file

worker_processes  1;

events {
    worker_connections  1024;
}

stream {
  server {
    listen DB instance port number;
proxy_pass DB instance identifier:DB instance port number;
    }
}
```

d. 명령줄에서 다음 명령을 사용하여 NGINX를 시작합니다.

```
Prompt> sudo initctl reload-configuration
Prompt> sudo initctl list | grep nginx
Prompt> sudo initctl start nginx
```

4. 프록시 서버와 대상 DB 인스턴스 및 복제 인스턴스가 포함된 대상 VPC 간에 ClassicLink 연결을 생성합니다.

a. EC2 콘솔을 열고 프록시 서버를 실행하는 EC2 Classic 인스턴스를 선택합니다.

b. 작업에서 을 선택한 ClassicLink다음 VPC에 연결을 선택합니다.

c. 이 절차에서 앞서 사용한 보안 그룹을 선택합니다.

d. VPC에 연결을 선택합니다.

5. 5단계: 의 절차를 사용하여 AWS DMS 엔드포인트를 생성합니다. 2단계: 소스 및 대상 엔드포인트 지정 소스 엔드포인트를 지정할 때에는 서버 이름으로 프록시의 내부 EC2 DNS 호스트 이름을 사용해야 합니다.

6. 의 절차를 사용하여 AWS DMS 작업을 생성합니다. 3단계: 태스크 생성 및 데이터 마이그레이션

복제 서브넷 그룹 생성

데이터베이스 마이그레이션에서 사용하는 네트워크의 일부로서 Virtual Private Cloud(VPC)에서 사용할 서브넷을 지정해야 합니다. 이 VPC는 Amazon VPC 서비스를 기반으로 해야 합니다. 서브넷은 지정된 가용 영역에 있는 VPC의 IP 주소 범위입니다. 이러한 서브넷은 VPC가 위치한 AWS 지역의 가용 영역에 분산될 수 있습니다.

AWS DMS 콘솔에서 복제 인스턴스 또는 인스턴스 프로필을 생성할 때 선택한 서브넷을 사용할 수 있습니다.

복제 서브넷 그룹을 생성하여 사용할 서브넷을 정의합니다. 가용 영역 2개 이상에서 서브넷을 지정해야 합니다.

복제 서브넷 그룹을 생성하려면

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.

   IAM 사용자로 로그인한 경우 AWS DMS에 액세스하기 위한 적절한 권한이 있어야 합니다. 데이터베이스 마이그레이션에 필요한 권한에 대한 자세한 내용은 AWS DMS 사용에 필요한 IAM 권한 섹션을 참조하세요.

2. 탐색 창에서 서브넷 그룹을 선택합니다.

3. [서브넷 그룹 생성]을 선택합니다.

4. 복제 서브넷 그룹 생성 페이지에서 복제 서브넷 그룹 정보를 지정합니다. 다음 표는 설정에 대한 설명입니다.

옵션	작업
이름	인쇄 가능한 ASCII 문자(/, " 및 @ 제외) 8-16자를 포함하는 복제 서브넷 그룹 이름을 입력합니다. 선택한 AWS 지역의 계정 이름은 고유해야 합니다. 예를 들어 수행 중인 AWS 지역 및 작업을 포함하는 등 이름에 일부 인텔리전스를 추가할 수 **DMS-default-VPC** 있습니다.
설명	복제 인스턴스 그룹에 대한 간략한 설명을 입력합니다.
VPC	데이터베이스 마이그레이션에 사용할 VPC를 선택합니다. VPC는 최소 2개의 가용 영역에 서브넷 1개 이상이 있어야 합니다.
서브넷 추가	복제 서브넷 그룹에 포함할 서브넷을 선택합니다. 두 개 이상의 가용 영역에서 서브넷을 선택해야 합니다.

5. [서브넷 그룹 생성]을 선택합니다.

DNS를 사용하여 도메인 엔드포인트 확인

일반적으로 AWS DMS 복제 인스턴스는 Amazon EC2 인스턴스의 DNS (도메인 이름 시스템) 확인자를 사용하여 도메인 엔드포인트를 확인합니다. DNS 확인이 필요한 경우 Amazon Route 53 Resolver를 사용할 수 있습니다. Route 53 DNS Resolver 사용에 대한 자세한 내용은 Route 53 Resolver 시작하기를 참조하세요.

자체 온프레미스 이름 서버를 사용하여 Amazon Route 53 Resolver로 특정 엔드포인트를 확인하는 방법에 대한 자세한 내용은 자체 온프레미스 이름 서버 사용 섹션을 참조하세요.

복제 인스턴스의 암호화 키 설정

AWS DMS는 복제 인스턴스가 사용하는 스토리지와 엔드포인트 연결 정보를 암호화합니다. 복제 인스턴스가 사용하는 스토리지를 암호화하기 위해 AWS DMS는 사용자 계정에 고유한 스토리지를 AWS KMS key 사용합니다. AWS () 를 사용하여 이 KMS 키를 보고 관리할 수 있습니다 AWS Key Management Service .AWS KMS계정의 기본 KMS 키(aws/dms) 또는 사용자가 생성한 KMS 키를 사용할 수 있습니다. 기존 AWS KMS 암호화 키가 있는 경우 해당 키를 암호화에 사용할 수도 있습니다.

KMS 키 식별자를 제공하여 AWS DMS 리소스를 암호화하여 자체 암호화 키를 지정할 수 있습니다. 자체 암호화 키를 지정할 때 데이터베이스 마이그레이션을 수행하는 데 사용되는 사용자 계정에는 키에 대한 액세스 권한이 있어야 합니다. 자체 암호화 키를 생성하고 암호화 키에 사용자 액세스 권한을 부여하는 방법에 대한 자세한 내용은 AWS KMS 개발자 안내서를 참조하세요.

KMS 키 식별자를 지정하지 않으면 AWS DMS는 기본 암호화 키를 사용합니다. KMS는 계정의 AWS DMS용 기본 암호화 키를 생성합니다. AWS AWS 계정에는 각 AWS 지역마다 다른 기본 암호화 키가 있습니다.

AWS DMS 리소스를 암호화하는 데 사용되는 키를 관리하려면 를 사용합니다. AWS KMS탐색 창에서 AWS Management Console KMS를 검색하면 AWS KMS 에서 찾을 수 있습니다.

AWS KMS 안전한 고가용성 하드웨어와 소프트웨어를 결합하여 클라우드에 맞게 확장된 키 관리 시스템을 제공합니다. 를 사용하여 AWS KMS암호화 키를 생성하고 이러한 키의 사용 방법을 제어하는 정책을 정의할 수 있습니다. AWS KMS 를 AWS CloudTrail지원하므로 키 사용을 감사하여 키가 적절하게 사용되고 있는지 확인할 수 있습니다. AWS KMS 키는 AWS DMS 및 기타 지원 AWS 서비스와 함께 사용할 수 있습니다. 지원되는 AWS 서비스로는 Amazon RDS, Amazon S3, Amazon Elastic Block Store(Amazon EBS), Amazon Redshift가 있습니다.

특정 암호화 키로 AWS DMS 리소스를 만든 경우 해당 리소스의 암호화 키를 변경할 수 없습니다. AWS DMS 리소스를 생성하기 전에 암호화 키 요구 사항을 결정해야 합니다.

복제 인스턴스 생성

데이터베이스를 마이그레이션하는 첫 번째 작업은 복제 인스턴스를 생성하는 것입니다. 이 복제 인스턴스에는 소스 데이터베이스에서 대상 데이터베이스로 데이터를 할당하고 마이그레이션하는 작업을 수행하는 데 충분한 스토리지 및 처리 능력이 필요합니다. 이 인스턴스의 필요한 크기는 마이그레이션해야 하는 데이터 양과 인스턴스 실행에 필요한 작업에 따라 달라집니다. 복제 인스턴스에 대한 자세한 내용은 AWS DMS 복제 인스턴스 사용 섹션을 참조하세요.

AWS 콘솔을 사용하여 복제 인스턴스를 만들려면

1. AWS DMS 콘솔의 탐색 창에서 복제 인스턴스를 선택한 다음 복제 인스턴스 생성을 선택합니다.
2. [Create replication instance] 페이지에서 복제 인스턴스 정보를 지정합니다. 다음 표는 사용자가 지정할 수 있는 설정에 대한 설명입니다.

옵션	작업
이름	인쇄 가능한 ASCII 문자(/, " 및 @ 제외) 8-16자를 포함하는 복제 인스턴스의 이름을 입력합니다. 이름은 선택한 AWS 리전의 계정에서 고유해야 합니다. 예를 들어 수행 중인 AWS 지역 및 작업을 포함하는 등 이름에 일부 인텔리전스를 추가하도록 선택할 수 **`west2-mysql2mysql-instance1`** 있습니다.
설명이 포함된 Amazon 리소스 이름(ARN) - 선택 사항	기본 DMS ARN을 재정의하기 위한 친숙한 이름입니다. 생성 후에는 수정할 수 없습니다.
설명	복제 인스턴스에 대한 간략한 설명을 입력합니다.
인스턴스 클래스	마이그레이션에 필요한 구성과 함께 인스턴스 클래스를 선택합니다. 마이그레이션을 성공적으로 완료할 수 있도록 인스턴스의 스토리지, 네트워크 및 처리 능력은 충분해야 합니다. 마이그레이션에 최적인 인스턴스 클래스를 결정하는 방법에 대한 자세한 내용은 AWS DMS 복제 인스턴스 사용 섹션을 참조하세요.
엔진 버전	AWS DMS 콘솔에서 지원되는 엔진 버전 중 원하는 버전을 선택할 수 있습니다. 에서 다른 엔진 버전을 지정하지 않는 한 AWS DMS 복제 인스턴스는 최신 비베타 버전의 복제 엔진을 실행합니다. AWS CLI AWS CLI
고가용성	이 선택적 파라미터를 사용하여 다른 가용 영역에 복제 인스턴스의 대기 복제본을 생성하여 장애 조치를 지원합니다. 변경 데이터 캡처(CDC) 또는 지속적 복제를 사용하려면, 이 옵션을 활성화해야 합니다.

옵션	작업
할당된 스토리지(GiB)	스토리지는 주로 로그 파일과 캐시된 트랜잭션에서 소모합니다. 캐시 트랜잭션에서 스토리지는 캐시된 트랜잭션을 디스크에 기록해야 할 때에만 사용됩니다. 따라서 AWS DMS는 스토리지를 많이 사용하지 않습니다. 일부 예외 사항은 다음과 같습니다. • 상당한 트랜잭션 로드를 발생하는 매우 큰 테이블. 큰 테이블을 로드하면 다소 시간이 걸릴 수 있으므로, 캐시된 트랜잭션은 큰 테이블 로드 중에 디스크에 기록될 가능성이 더 높습니다. • 캐시된 트랜잭션을 로드하기 전에 일시 중지되도록 구성되는 작업입니다. 이 경우에 모든 테이블에서 전체 로드가 완료될 때까지 모든 트랜잭션이 캐시됩니다. 이 구성에서 상당량의 스토리지가 캐시된 트랜잭션에서 소모될 수 있습니다. • Amazon Redshift로 로드되고 있는 테이블을 사용하여 구성된 작업입니다. 그렇지만, 이 구성은 Amazon Aurora가 대상일 때는 문제가 아닙니다. 대다수의 경우에 스토리지의 기본 할당량은 충분합니다. 하지만 항상 스토리지 관련 지표에 주의를 기울이는 것이 좋습니다. 기본 할당량보다 많이 소비하는 경우 스토리지를 스케일 업해야 합니다.
네트워크 유형	DMS는 IPv4 주소 지정 프로토콜 네트워크 유형을 지원하고 이중 스택 모드에서 IPv4 및 IPv6 주소 지정 프로토콜 네트워크 유형을 모두 지원합니다. IPv6 주소 지정 프로토콜 네트워크 유형을 사용하여 복제 인스턴스와 통신해야 하는 리소스가 있는 경우 이중 스택 모드를 사용합니다. 이중 스택 모드의 제한 사항에 대한 자세한 내용은 Amazon Relational Database Service 사용 설명서의 이중 스택 네트워크 DB 인스턴스 제한 사항을 참조하세요.

옵션	작업
VPC	사용할 VPC를 선택합니다. 원본 또는 대상 데이터베이스가 VPC에 있는 경우, 해당 VPC를 선택합니다. 소스 데이터베이스와 대상 데이터베이스가 다른 VPC에 있으면, 모두 퍼블릭 서브넷에 있고 공개적으로 액세스 가능해야 합니다. 그런 다음 복제 인스턴스를 배치할 VPC를 선택합니다. 복제 인스턴스는 원본 VPC에 있는 데이터에 액세스할 수 있어야 합니다. 소스 또는 대상 데이터베이스가 VPC에 없으면, 복제 인스턴스가 배치될 VPC를 선택합니다.
[Replication Subnet Group]	복제 인스턴스를 생성할 위치에 있는 선택된 VPC에서 복제 서브넷 그룹을 선택합니다. 원본 데이터베이스가 VPC에 있는 경우, 원본 데이터베이스가 포함된 서브넷 그룹을 복제 인스턴스 위치로 선택합니다. 복제 서브넷에 대한 자세한 내용은 복제 서브넷 그룹 생성 섹션을 참조하세요.
공개적으로 액세스할 수 있음(Publicly accessible)	인터넷에서 복제 인스턴스에 액세스하려면 이 옵션을 선택합니다. 기본값은 공개적으로 액세스할 수 있으며, 옵션을 선택한 후에는 생성한 복제 인스턴스를 수정할 수 없습니다.

3. 고급 탭을 선택하고 네트워크와 암호화 설정이 필요하면 값을 설정합니다. 다음 표는 설정에 대한 설명입니다.

옵션	작업
가용 영역	원본 데이터베이스가 있는 가용 영역을 선택합니다.
[VPC Security group(s)]	모든 복제 인스턴스가 VPC에 생성됩니다. 소스 데이터베이스가 VPC에 있는 경우, 데이터베이스가 상주하는 곳에 있는 DB 인스턴스에 대한 액세스 권한을 부여하는 VPC 보안 그룹을 선택합니다.
KMS 키	복제 스토리지와 연결 정보를 암호화하기 위해 사용할 암호화 키를 선택합니다. (기본값) aws/dms를 선

옵션	작업
	택하면 계정 및 AWS 지역과 관련된 기본 AWS Key Management Service (AWS KMS) 키가 사용됩니다. 설명 및 계정 번호가 키의 ARN과 함께 표시됩니다. 암호화 키 사용에 대한 자세한 내용은 암호화 키 설정 및 AWS KMS 권한 지정를 참조하세요.

4. [Maintenance] 설정을 지정합니다. 다음 표는 설정에 대한 설명입니다. 유지 관리 설정에 대한 자세한 내용은 AWS DMS 유지 관리 기간 관련 작업 섹션을 참조하세요.

옵션	작업
자동 버전 업그레이드	AWS DMS 메이저 버전과 마이너 버전을 구분하지 않습니다. 예를 들어 버전 3.4.x에서 3.5.x로 업그레이드하는 것은 메이저 업그레이드로 간주되지 않으며, 따라서 모든 변경 사항은 이전 버전과 호환됩니다. 자동 버전 업그레이드가 활성화된 경우 DMS는 유지 관리 기간 동안 더 이상 사용되지 않는 복제 인스턴스 버전을 자동으로 업그레이드합니다. AutoMinorVersionUpgrade가 활성화되면 DMS는 복제 인스턴스를 만들 때 현재 기본 엔진 버전을 사용합니다. 예를 들어 엔진 버전을 현재 기본 버전보다 낮은 버전 번호로 설정하면 DMS는 기본 버전을 사용합니다. 복제 인스턴스를 만들 때 AutoMinorVersionUpgrade활성화되지 않은 경우 DMS는 엔진 버전 매개변수로 지정된 엔진 버전을 사용합니다.
유지보수 윈도우	시스템 유지 관리를 실행할 수 있는 주 단위 기간(UTC, 협정 세계시)을 선택합니다. 기본값: AWS 지역별 8시간 블록 중에서 무작위로 선택한 30분 기간으로, 요일마다 무작위로 나타납니다.

5. [Create replication instance]를 선택합니다.

복제 인스턴스 수정

복제 인스턴스에 대한 설정을 수정하여 예컨대 인스턴스 클래스를 변경하거나 스토리지를 늘릴 수 있습니다.

복제 인스턴스를 수정하는 경우 변경 사항을 즉시 적용할 수 있습니다. 변경 사항을 즉시 적용하려면 AWS Management Console에서 변경 사항 즉시 적용 옵션을 선택합니다. 또는 를 호출할 때 `--apply-immediately` 파라미터를 사용하거나 AWS CLI, DMS API를 사용할 `true` 때는 `ApplyImmediately` 파라미터를 로 설정하십시오.

변경 사항을 즉시 적용하지 않기로 선택하면 변경 사항이 보류 중 수정 사항 대기열로 보내집니다. 다음 유지 관리 기간에 대기열에 있는 보류 중 변경 사항이 적용됩니다.

> **Note**
>
> 변경 사항을 즉시 적용하기로 선택하면 보류 중 수정 사항 대기열에 있는 변경 사항까지 모두 적용됩니다. 보류 중 수정 사항이 하나라도 가동 중지를 필요로 하는 경우 [Apply changes immediately]을 선택하면 예기치 못한 가동 중지가 발생할 수 있습니다.

콘솔을 AWS 사용하여 복제 인스턴스를 수정하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 [Replication instances]를 선택합니다.
3. 수정할 복제 인스턴스를 선택합니다. 다음 표에서는 수행할 수 있는 수정 사항을 설명합니다.

옵션	작업
이름	복제 인스턴스의 이름을 변경할 수 있습니다. 인쇄 가능한 ASCII 문자(/, " 및 @ 제외) 8-16자를 포함하는 복제 인스턴스의 이름을 입력합니다. 이름은 선택한 AWS 리전의 계정에서 고유해야 합니다. 예를 들어 수행 중인 AWS 지역 및 작업을 포함하는 등 이름에 일부 인텔리전스를 추가하도록 선택할 수 **`west2-mysql2mysql-instance1`** 있습니다.

옵션	작업
설명	복제 인스턴스에 대한 간략한 설명을 수정하거나 입력합니다.
인스턴스 클래스	인스턴스 클래스를 변경할 수 있습니다. 마이그레이션에 필요한 구성과 함께 인스턴스 클래스를 선택합니다. 인스턴스 클래스를 변경하면 복제 인스턴스가 재부팅됩니다. 이 재부팅은 다음 유지 관리 기간 중에 발생하지만 변경 사항 즉시 적용 옵션을 선택하는 경우 즉시 발생할 수 있습니다. 마이그레이션에 최적인 인스턴스 클래스를 결정하는 방법에 대한 자세한 내용은 AWS DMS 복제 인스턴스 사용 섹션을 참조하세요.
엔진 버전	복제 인스턴스에 사용되는 엔진 버전을 업그레이드할 수 있습니다. 복제 엔진 버전을 업그레이드하면 업그레이드 중에 복제 인스턴스가 종료됩니다.
다중 AZ	이 옵션을 변경하여 다른 가용 영역에 장애 조치를 지원할 복제 인스턴스의 예비 복제본을 생성하거나 이 옵션을 제거합니다. 데이터 캡처(CDC) 또는 지속적 복제를 사용하려면, 이 옵션을 활성화해야 합니다.

옵션	작업
할당된 스토리지(GiB)	스토리지는 주로 로그 파일과 캐시된 트랜잭션에서 소모합니다. 캐시 트랜잭션에서 스토리지는 캐시된 트랜잭션을 디스크에 기록해야 할 때에만 사용됩니다. 따라서 AWS DMS는 많은 양의 스토리지를 사용하지 않습니다. 일부 예외 사항은 다음과 같습니다. • 상당한 트랜잭션 로드를 발생하는 매우 큰 테이블. 큰 테이블을 로드하면 다소 시간이 걸릴 수 있으므로, 캐시된 트랜잭션은 큰 테이블 로드 중에 디스크에 기록될 가능성이 더 높습니다. • 캐시된 트랜잭션을 로드하기 전에 일시 중지되도록 구성되는 작업입니다. 이 경우에 모든 테이블에서 전체 로드가 완료될 때까지 모든 트랜잭션이 캐시됩니다. 이 구성에서 상당량의 스토리지가 캐시된 트랜잭션에서 소모될 수 있습니다. • Amazon Redshift로 로드되고 있는 테이블을 사용하여 구성된 작업입니다. 그렇지만, 이 구성은 Amazon Aurora가 대상일 때는 문제가 아닙니다. 대다수의 경우에 스토리지의 기본 할당량은 충분합니다. 그렇지만, 항상 스토리지 관련 지표에 유의하고 기본 할당량보다 더 많이 소모하게 되면 스토리지를 확장하는 것이 좋습니다.
네트워크 유형	DMS는 IPv4 주소 지정 프로토콜 네트워크 유형을 지원하고 이중 스택 모드에서 IPv4 및 IPv6 주소 지정 프로토콜 네트워크 유형을 모두 지원합니다. IPv6 주소 지정 프로토콜 네트워크 유형을 사용하여 복제 인스턴스와 통신해야 하는 리소스가 있는 경우 이중 스택 모드를 선택합니다. 이중 스택 모드의 제한 사항에 대한 자세한 내용은 Amazon Relational Database Service 사용 설명서의 이중 스택 네트워크 DB 인스턴스 제한 사항을 참조하세요.

옵션	작업
VPC 보안 그룹	모든 복제 인스턴스가 VPC에 생성됩니다. 소스 데이터베이스가 VPC에 있는 경우, 데이터베이스가 상주하는 곳에 있는 DB 인스턴스에 대한 액세스 권한을 부여하는 VPC 보안 그룹을 선택합니다.
자동 버전 업그레이드	AWS DMS 메이저 버전과 마이너 버전을 구분하지 않습니다. 예를 들어 버전 3.4.x에서 3.5.x로 업그레이드하는 것은 메이저 업그레이드로 간주되지 않으며, 따라서 모든 변경 사항은 이전 버전과 호환됩니다. 자동 버전 업그레이드가 활성화된 경우 DMS는 유지 관리 기간 동안 더 이상 사용되지 않는 복제 인스턴스 버전을 자동으로 업그레이드합니다. 자동 버전 업그레이드가 활성화된 경우 DMS는 복제 인스턴스를 생성할 때 현재 기본 엔진 버전을 사용합니다. 예를 들어 엔진 버전을 현재 기본 버전보다 낮은 버전 번호로 설정하면 DMS는 기본 버전을 사용합니다. 자동 버전 업그레이드가 활성화되지 않은 상태에서 복제 인스턴스를 생성하면 DMS는 엔진 버전 파라미터로 지정된 엔진 버전을 사용합니다.
유지보수 윈도우	시스템 유지 관리를 실행할 수 있는 주 단위 기간(UTC, 협정 세계시)을 선택합니다. 기본값: AWS 지역별 8시간 블록 중에서 무작위로 선택한 30분 기간으로, 요일마다 무작위로 나타납니다.
Apply changes immediately	변경 내용을 즉시 적용하려면 이 옵션을 선택합니다. 선택하는 설정에 따라 이 옵션을 선택하면 복제 인스턴스가 즉시 재부팅될 수 있습니다. AWS DMS 가 변경 사항을 적용하는 동안 연결 테스트를 선택하면 오류 메시지가 표시됩니다. 복제 인스턴스에 변경 내용을 AWS DMS 적용한 후 연결 테스트를 다시 선택합니다.

옵션	작업
다음 유지 관리 기간에 변경 사항 적용	DMS가 다음 예정된 유지 관리 기간까지 기다려 변경 사항을 적용하도록 하려면 이 옵션을 선택합니다.

복제 인스턴스 재부팅

AWS DMS 복제 인스턴스를 재부팅하여 복제 엔진을 다시 시작할 수 있습니다. 재부팅하면 인스턴스 상태가 재부팅 중으로 설정되면서 복제 인스턴스가 잠시 중단됩니다. AWS DMS 인스턴스가 다중 AZ 용으로 구성된 경우 장애 조치를 통해 재부팅을 수행할 수 있습니다. 재부팅이 완료되면 AWS DMS 이벤트가 생성됩니다.

다중 AZ 배포인 AWS DMS 인스턴스의 경우 재부팅 시 한 가용 영역에서 다른 가용 AWS 영역으로 계획된 장애 조치를 강제로 적용할 수 있습니다. 계획된 AWS DMS 인스턴스 장애 조치를 강제로 적용하면 다른 가용 영역의 대기 인스턴스로 자동 전환하기 전에 현재 인스턴스의 활성 연결이 끊어집니다. AWS DMS 계획된 장애 조치로 재부팅하면 복제 AWS DMS 인스턴스 클래스를 확장할 때와 같이 인스턴스의 계획된 장애 조치 이벤트를 시뮬레이션하는 데 도움이 됩니다.

> **Note**
>
> 재부팅할 때 한 가용 영역에서 다른 가용 영역으로 장애 조치를 강제로 실시하면 가용 영역 변경 사항이 몇 분 동안 반영되지 않을 수 있습니다. 이 지연은 과 및 API AWS Management Console호출에서 나타납니다. AWS CLI AWS DMS

복제 인스턴스에서 마이그레이션 태스크가 실행 중일 때 재부팅이 발생하면 데이터 손실은 발생하지 않지만 태스크가 중지되고 태스크 상태는 오류 상태로 바뀝니다.

마이그레이션 태스크의 테이블이 대량 로드 중에 있고(전체 로드 단계) 아직 시작되지 않은 경우 오류 상태로 전환됩니다. 하지만 그 시점에 완료된 테이블은 완료 상태로 남아 있습니다. 전체 로드 단계에서 재부팅이 발생하는 경우 아래 단계 중 하나를 수행하는 것이 좋습니다.

- 태스크에서 완료 상태인 테이블을 제거하고 나머지 테이블을 사용하여 태스크를 다시 시작합니다.
- 오류 상태의 테이블과 보류 중인 테이블로 새 태스크를 생성합니다.

마이그레이션 작업의 테이블이 지속적인 복제 단계에 있는 경우 재부팅이 완료된 후 작업이 다시 시작됩니다.

상태가 Available 상태가 아닌 경우 AWS DMS 복제 인스턴스를 재부팅할 수 없습니다. 이전에 요청한 수정 또는 유지 관리 기간 작업 등 여러 가지 이유로 AWS DMS 인스턴스를 사용할 수 없을 수 있습니다. AWS DMS 복제 인스턴스를 재부팅하는 데 걸리는 시간은 일반적으로 짧습니다 (5분 미만).

콘솔을 사용하여 복제 인스턴스 재부팅 AWS

복제 인스턴스를 재부팅하려면 콘솔을 사용합니다. AWS

AWS 콘솔을 사용하여 복제 인스턴스를 재부팅하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 [Replication instances]를 선택합니다.
3. 재부팅할 복제 인스턴스를 선택합니다.
4. 재부팅을 선택합니다. 복제 인스턴스 재부팅 대화 상자가 열립니다.
5. 다중 AZ 배포에 대해 복제 인스턴스를 구성한 경우 다른 AWS 가용 영역으로 장애 조치하려면 계획된 장애 조치로 재부팅하시겠습니까? 확인란을 선택합니다.
6. 재부팅을 선택합니다.

CLI를 사용하여 복제 인스턴스 재부팅

복제 인스턴스를 재부팅하려면 AWS CLI `reboot-replication-instance`명령을 다음 파라미터와 함께 사용합니다.

- `--replication-instance-arn`

Example 간단한 재부팅 예제

다음 AWS CLI 예제에서는 복제 인스턴스를 재부팅합니다.

```
aws dms reboot-replication-instance \
--replication-instance-arn arn of my rep instance
```

Example 장애 조치로 간단한 재부팅 예제

다음 AWS CLI 예제에서는 페일오버를 사용하여 복제 인스턴스를 재부팅합니다.

```
aws dms reboot-replication-instance \
```

```
--replication-instance-arn arn of my rep instance \
--force-planned-failover
```

API를 사용하여 복제 인스턴스 재부팅

복제 인스턴스를 재부팅하려면 다음 파라미터와 함께 AWS DMS API RebootReplicationInstance작업을 사용합니다.

- ReplicationInstanceArn = *arn of my rep instance*

Example 간단한 재부팅 예제

다음 코드 예제에서는 복제 인스턴스를 재부팅합니다.

```
https://dms.us-west-2.amazonaws.com/
?Action=RebootReplicationInstance
&DBInstanceArn=arn of my rep instance
&SignatureMethod=HmacSHA256
&SignatureVersion=4
&Version=2014-09-01
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=AKIADQKE4SARGYLE/20140425/us-east-1/dms/aws4_request
&X-Amz-Date=20140425T192732Z
&X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
&X-Amz-Signature=1dc9dd716f4855e9bdf188c70f1cf9f6251b070b68b81103b59ec70c3e7854b3
```

Example 장애 조치로 간단한 재부팅 예제

다음 코드 예제는 복제 인스턴스를 재부팅하고 다른 AWS 가용 영역으로 장애 조치합니다.

```
https://dms.us-west-2.amazonaws.com/
?Action=RebootReplicationInstance
&DBInstanceArn=arn of my rep instance
&ForcePlannedFailover=true
&SignatureMethod=HmacSHA256
&SignatureVersion=4
&Version=2014-09-01
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=AKIADQKE4SARGYLE/20140425/us-east-1/dms/aws4_request
&X-Amz-Date=20140425T192732Z
&X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
```

```
&X-Amz-Signature=1dc9dd716f4855e9bdf188c70f1cf9f6251b070b68b81103b59ec70c3e7854b3
```

복제 인스턴스 삭제

AWS DMS 복제 인스턴스 사용이 끝나면 복제 인스턴스를 삭제할 수 있습니다. 복제 인스턴스를 사용하는 마이그레이션 작업이 있는 경우 복제 인스턴스를 삭제하기 전에 작업을 중지한 후 삭제해야 합니다.

계정을 폐쇄하면 2일 후에 AWS 계정과 관련된 모든 AWS DMS 리소스 및 구성이 삭제됩니다. 이 리소스에는 모든 복제 인스턴스, 원본 및 대상 엔드포인트 구성, 복제 작업, SSL 인증서가 포함됩니다. 이틀 후에 AWS DMS 다시 사용하기로 결정하면 필요한 리소스를 다시 생성합니다.

복제 인스턴스가 모든 삭제 기준을 충족하고 장기간 `DELETING` 상태를 유지하는 경우 지원 팀에 문의하여 문제를 해결하세요.

콘솔을 AWS 사용하여 복제 인스턴스 삭제

복제 인스턴스를 삭제하려면 AWS 콘솔을 사용합니다.

AWS 콘솔을 사용하여 복제 인스턴스를 삭제하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 [Replication instances]를 선택합니다.
3. 삭제할 복제 인스턴스를 선택합니다.
4. 삭제를 선택합니다.
5. 대화 상자에서 삭제를 선택합니다.

CLI를 사용하여 복제 인스턴스 삭제

복제 인스턴스를 삭제하려면 AWS CLI `delete-replication-instance`명령을 다음 파라미터와 함께 사용합니다.

- `--replication-instance-arn`

Example 삭제 예제

다음 AWS CLI 예제에서는 복제 인스턴스를 삭제합니다.

```
aws dms delete-replication-instance \
--replication-instance-arn arn of my rep instance
```

API를 사용하여 복제 인스턴스 삭제

복제 인스턴스를 삭제하려면 다음 파라미터와 함께 AWS DMS API [DeleteReplicationInstance](#)작업을 사용합니다.

- `ReplicationInstanceArn = arn of my rep instance`

Example 삭제 예제

다음 코드 예시에서는 복제 인스턴스를 삭제합니다.

```
https://dms.us-west-2.amazonaws.com/
?Action=DeleteReplicationInstance
&DBInstanceArn=arn of my rep instance
&SignatureMethod=HmacSHA256
&SignatureVersion=4
&Version=2014-09-01
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=AKIADQKE4SARGYLE/20140425/us-east-1/dms/aws4_request
&X-Amz-Date=20140425T192732Z
&X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
&X-Amz-Signature=1dc9dd716f4855e9bdf188c70f1cf9f6251b070b68b81103b59ec70c3e7854b3
```

AWS DMS 유지 관리 기간 관련 작업

모든 AWS DMS 복제 인스턴스에는 사용 가능한 시스템 변경 사항이 적용되는 주간 유지 관리 기간이 있습니다. 유지 관리 기간은 수정 사항 및 소프트웨어 패치 적용 시점을 조절할 수 있는 기회로 생각하면 좋습니다.

특정 주에 유지 관리가 필요하다고 AWS DMS 판단되면 복제 인스턴스를 생성할 때 선택한 30분 유지 관리 기간 동안 유지 관리가 이루어집니다. AWS DMS 30분의 유지 관리 기간 동안 대부분의 유지 관리를 완료합니다. 단, 대규모 변경 사항의 경우 보다 긴 시간이 필요할 수 있습니다.

유지 관리가 기존 마이그레이션 작업에 미치는 영향

인스턴스에서 AWS DMS 마이그레이션 작업을 실행하는 경우 패치를 적용하면 다음과 같은 이벤트가 발생합니다.

- 마이그레이션 작업의 테이블이 진행 중 변경 사항 복제 단계(CDC)에 있는 경우 AWS DMS 는 태스크를 잠시 동안 중지했다가 패치가 적용된 후에 다시 시작합니다. 그러고 나면 패치가 적용될 때 중단되었던 지점부터 마이그레이션이 계속됩니다.
- 기존 데이터 이전 또는 기존 데이터 마이그레이션 및 진행 중인 변경 사항 복제 작업의 일부로 테이블을 마이그레이션하는 경우 AWS DMS DMS는 패치가 적용되는 동안 전체 로드 단계에 있는 모든 테이블에 대해 마이그레이션을 중지했다가 다시 시작합니다. 또한 DMS는 패치가 적용되는 동안 CDC 단계에 있는 모든 테이블을 중지했다가 다시 시작합니다.

유지 관리 기간 설정 변경

AWS Management Console, 또는 API를 사용하여 유지 관리 기간 기간을 변경할 수 있습니다 AWS CLI. AWS DMS

콘솔을 사용하여 유지 관리 기간 설정 변경

AWS Management Console을 사용하여 유지 관리 기간 프레임을 변경할 수 있습니다.

콘솔을 사용하여 기본 유지 관리 기간을 변경하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 [Replication instances]를 선택합니다.
3. [Modify]를 수정 및 선택할 복제 인스턴스를 선택합니다.
4. 유지 관리 섹션을 확장하고 유지 관리 기간의 날짜 및 시간을 선택합니다.
5. [Apply changes immediately]를 선택합니다.
6. 수정을 선택합니다.

CLI를 사용하여 유지 관리 기간 설정 변경

기본 유지 관리 기간을 조정하려면 AWS CLI `modify-replication-instance`명령을 다음 매개 변수와 함께 사용합니다.

- `--replication-instance-identifier`
- `--preferred-maintenance-window`

Example

다음 AWS CLI 예에서는 유지 관리 기간을 화요일 오전 4:00 - 4:30 으로 설정합니다. UTC 기준입니다.

```
aws dms modify-replication-instance \
--replication-instance-identifier myrepinstance \
--preferred-maintenance-window Tue:04:00-Tue:04:30
```

API를 사용하여 유지 관리 기간 설정 변경

기본 유지 관리 기간을 조정하려면 다음 매개 변수와 함께 AWS DMS API [ModifyReplicationInstance](#)작업을 사용하십시오.

- `ReplicationInstanceIdentifier = myrepinstance`
- `PreferredMaintenanceWindow = Tue:04:00-Tue:04:30`

Example

다음은 유지 관리 기간을 화요일 오전 4:00~4:30으로 설정하는 코드 예제입니다. UTC 기준입니다.

```
https://dms.us-west-2.amazonaws.com/
?Action=ModifyReplicationInstance
&DBInstanceIdentifier=myrepinstance
&PreferredMaintenanceWindow=Tue:04:00-Tue:04:30
&SignatureMethod=HmacSHA256
&SignatureVersion=4
&Version=2014-09-01
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=AKIADQKE4SARGYLE/20140425/us-east-1/dms/aws4_request
&X-Amz-Date=20140425T192732Z
&X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
&X-Amz-Signature=1dc9dd716f4855e9bdf188c70f1cf9f6251b070b68b81103b59ec70c3e7854b3
```

AWS DMS 엔드포인트 작업

엔드포인트는 데이터 스토어에 대한 연결, 데이터 스토어 유형, 위치 정보를 제공합니다. AWS Database Migration Service는 이 정보를 사용하여 데이터 스토어에 연결하고 소스 엔드포인트에서 대상 엔드포인트로 데이터를 마이그레이션합니다. 엔드포인트 설정을 사용하여 엔드포인트에 대한 추가 연결 속성을 지정할 수 있습니다. 이러한 설정은 로깅, 파일 크기 및 기타 파라미터를 제어할 수 있습니다. 엔드포인트 설정에 대한 자세한 내용은 해당 데이터 스토어의 설명서 딘원을 참조하세요.

아래에서 엔드포인트에 대한 자세한 정보를 확인할 수 있습니다.

주제

- 소스 및 대상 엔드포인트 생성
- 데이터 마이그레이션용 소스
- 마이그레이션에 적합한 대상
- VPC 엔드포인트를 AWS DMS 소스 및 대상 엔드포인트로 구성
- AWS DMS에 의해 지원되는 DDL 문

소스 및 대상 엔드포인트 생성

복제 인스턴스를 생성할 때 원본 및 대상 엔드포인트를 생성할 수 있습니다. 또는 복제 인스턴스가 생성된 후에 엔드포인트를 생성할 수 있습니다. 원본과 대상 데이터 스토어는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, Amazon Relational Database Service(Amazon RDS) DB 인스턴스 또는 온프레미스 데이터베이스에 있을 수 있습니다. (단, 엔드포인트 중 하나는 AWS 서비스에 있어야 합니다. AWS DMS를 사용하여 온프레미스 데이터베이스에서 다른 온프레미스 데이터베이스로 마이그레이션할 수는 없습니다.)

다음 절차에서는 AWS DMS 콘솔 마법사를 선택했다고 가정합니다. 참고로 AWS DMS 콘솔의 탐색 창에서 엔드포인트를 선택한 다음 엔드포인트 생성을 선택하여 이 단계를 수행할 수도 있습니다. 콘솔 마법사를 사용할 때에는 동일한 페이지에서 원본과 대상 엔드포인트를 모두 생성합니다. 콘솔 마법사를 사용하지 않을 때에는 각 엔드포인트를 별도로 생성합니다.

AWS 콘솔을 사용하여 소스 또는 대상 데이터베이스 엔드포인트를 지정하려면

1. [소스 및 타겟 데이터베이스 엔드포인트 연결] 페이지에서 원본 또는 대상 데이터베이스에 대한 연결 정보를 지정합니다. 다음 표는 설정에 대한 설명입니다.

옵션	조치
[엔드포인트 유형]	이 엔드포인트가 원본 엔드포인트인지, 아니면 대상 엔드포인트인지 선택합니다.
Select RDS DB Instance(RDS DB 인스턴스 선택)	엔드포인트가 Amazon RDS DB 인스턴스인 경우 이 옵션을 선택합니다.
[엔드포인트 식별자]	엔드포인트를 식별하는 데 사용할 이름을 입력합니다. 이름에 **`oracle-source`** 또는 **`PostgreSQL-target`**과 같은 엔드포인트 유형을 포함해야 합니다. 모든 복제 인스턴스에서 이 이름은 고유해야 합니다.
[소스 엔진] 및 [타겟 엔진]	엔드포인트가 되는 데이터베이스 엔진의 유형을 선택합니다.
엔드포인트 데이터베이스에 대한 액세스	엔드포인트 데이터베이스 보안 인증 정보 지정에 사용할 옵션을 선택합니다. • AWS Secrets Manager 선택 - 다음과 같이 AWS Secrets Manager에 정의된 비밀을 사용하여 보안 인증 정보를 비밀리에 제공합니다. 이러한 비밀 및 AWS DMS가 이러한 비밀에 액세스할 수 있는 비밀 액세스 역할을 생성하는 방법에 대한 자세한 내용은 보안 암호를 사용하여 AWS Database Migration Service 엔드포인트에 액세스하기 섹션을 참조하세요. • 수동으로 액세스 정보 제공 - 다음과 같이 직접 입력하는 일반 텍스트 보안 인증 정보를 사용합니다.
AWS Secrets Manager 선택	다음 비밀 보안 인증 정보를 설정합니다.
비밀 ID	AWS Secrets Manager에서 엔드포인트 데이터베이스 액세스를 위해 생성한 비밀의 전체 Amazon 리소스 이름(ARN), 부분 ARN 또는 기억하기 쉬운 이름을 입력합니다.

옵션	조치
IAM 역할	AWS DMS가 사용자를 대신하여 비밀 ID로 식별되는 비밀에 액세스할 권한을 부여하기 위해 IAM에서 생성한 비밀 액세스 역할의 ARN을 입력합니다. 비밀 액세스 역할 생성에 대한 자세한 내용은 보안 암호를 사용하여 AWS Database Migration Service 엔드포인트에 액세스하기 단원을 참조하세요.
Oracle Automatic Storage Management(ASM)용 비밀 ID	(Oracle ASM을 사용하는 Oracle 소스 엔드포인트에만 해당) Oracle ASM 액세스를 위해 AWS Secrets Manager에서 생성한 비밀의 전체 Amazon 리소스 이름(ARN), 부분 ARN 또는 기억하기 쉬운 이름을 입력합니다. 이 비밀은 일반적으로 보안 ID로 식별되는 비밀과 동일한 서버에서 Oracle ASM에 액세스하기 위해 생성됩니다.
Oracle ASM용 IAM 역할	(Oracle ASM을 사용하는 Oracle 소스 엔드포인트에만 해당) AWS DMS가 사용자를 대신하여 Oracle Automatic Storage Management(ASM)용 비밀 ID로 식별되는 비밀에 액세스할 권한을 부여하기 위해 IAM에서 생성한 비밀 액세스 역할의 ARN을 입력합니다.
수동으로 액세스 정보 제공	다음과 같은 일반 텍스트 보안 인증 정보를 설정합니다.
[서버 이름]	서버 이름을 입력합니다. 온프레미스 데이터베이스의 경우, 이것은 IP 주소 또는 퍼블릭 호스트 이름이 될 수 있습니다. Amazon RDS DB 인스턴스의 경우, 이것은 **`mysqlsrvinst.abcd12345678.us-west-2.rds.amazonaws.com`** 과 같이 DB 인스턴스의 엔드포인트(DNS 이름)가 될 수 있습니다.
포트	데이터베이스가 사용하는 포트를 입력합니다.
Secure Socket Layer(SSL) 모드	이 엔드포인트에서 연결 암호화를 활성화해야 하는 경우 SSL 모드를 선택합니다. 선택하는 모드에 따라 인증서와 서버 인증서 정보를 제공해야 할 수도 있습니다.

옵션	조치
사용자 이름	데이터 마이그레이션을 허용하는 데 필요한 권한을 가진 사용자 이름을 입력합니다. 필요한 권한에 대한 정보는 이 사용 설명서의 원본 또는 대상 데이터베이스 엔진의 보안 섹션을 참조하세요.
암호	필요한 권한을 가진 계정의 암호를 입력합니다. AWS DMS 소스 및 대상 엔드포인트의 암호에는 데이터베이스 엔진에 따라 문자 제한이 있습니다. 자세한 정보는 다음 표를 참조하세요.
데이터베이스 이름	일부 데이터베이스 엔진의 경우, 엔드포인트 데이터베이스로 사용하려는 데이터베이스의 이름.

다음 표에는 나열된 데이터베이스 엔진의 엔드포인트 암호 및 Secret Manager 암호에서 지원되지 않는 문자가 나열되어 있습니다. 엔드포인트 암호에 쉼표(,)를 사용하려면 AWS DMS에서 제공하는 Secrets Manager 지원을 사용하여 AWS DMS 인스턴스에 대한 액세스를 인증합니다. 자세한 설명은 보안 암호를 사용하여 AWS Database Migration Service 엔드포인트에 액세스하기 섹션을 참조하세요.

이 데이터베이스 엔진의 경우	엔드포인트 암호 및 Secret Manager 암호에는 다음 문자가 지원되지 않습니다.
모두	`{ }`
Microsoft Azure(소스로 사용되는 경우에만)	`;`
Microsoft SQL Server	`, ;`
MySQL 호환(MySQL, MariaDB, Amazon Aurora MySQL 포함)	`;`
Oracle	`,`

이 데이터베이스 엔진의 경우	엔드포인트 암호 및 Secret Manager 암호에는 다음 문자가 지원되지 않습니다.
PostgreSQL, Amazon Aurora PostgreSQL 호환 에디션, Amazon Aurora Serverless(Aurora PostgreSQL 호환 에디션용 대상으로 사용되는 경우에만)	; + %
Amazon Redshift(대상으로 사용되는 경우에만)	, ;

2. 엔드포인트 설정을 선택하고 필요한 경우 AWS KMS key을 선택합니다. [테스트 실행]을 선택하여 엔드포인트 연결을 테스트할 수 있습니다. 다음 표는 설정에 대한 설명입니다.

옵션	조치
엔드포인트 설정	여기에서 추가 연결 파라미터를 선택합니다. 엔드포인트 설정에 대한 자세한 내용은 1단계에서 지정한 소스 엔진 또는 대상 엔진의 설명서를 참조하세요. Oracle ASM을 사용하는 Oracle 소스 엔드포인트의 경우 1단계에서 수동으로 액세스 정보 제공을 선택한 경우 엔드포인트 설정에 입력하여 Oracle ASM 사용자 보안 인증 정보를 지정해야 할 수도 있습니다. 이러한 Oracle ASM 엔드포인트 설정에 대한 자세한 내용은 CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용 섹션을 참조하세요.
AWS KMS key	복제 스토리지와 연결 정보를 암호화하기 위해 사용할 암호화 키를 선택합니다. (기본값) aws/dms를 선택하면, 계정 및 AWS 리전과 연결된 기본 AWS Key Management Service(AWS KMS) 키가 사용됩니다. 암호화 키 사용에 대한 자세한 내용은 암호화 키 설정 및 AWS KMS 권한 지정를 참조하세요.
엔드포인트 연결 테스트(선택 사항)	VPC 및 복제 인스턴스 이름을 추가합니다. 연결을 테스트하려면 테스트 실행을 선택합니다.

데이터 마이그레이션용 소스

AWS Database Migration Service(AWS DMS)는 데이터 복제용 소스로 가장 인기 있는 데이터 엔진을 다수 사용할 수 있습니다. 데이터베이스 소스는 Amazon EC2 인스턴스 또는 온프레미스 데이터베이스에서 실행되는 자체 관리형 엔진이 될 수 있습니다. 또는 Amazon RDS 혹은 Amazon S3와 같은 AWS 서비스에 있는 데이터 소스가 될 수도 있습니다.

유효한 소스의 전체 목록은 AWS DMS용 소스 섹션을 참조하세요.

주제

- Oracle 데이터베이스를 AWS DMS에서 소스로 사용
- Microsoft SQL Server 데이터베이스를 원본으로 사용 AWS DMS
- AWS DMS용 소스로 Microsoft Azure SQL 데이터베이스 사용
- Microsoft Azure SQL 관리형 인스턴스를 AWS DMS용 소스로 사용
- Microsoft Azure Database for PostgreSQL 유연한 서버를 AWS DMS용 소스로 사용
- Microsoft Azure Database for MySQL 유연한 서버를 AWS DMS용 소스로 사용
- OCI MySQL Heatwave를 AWS DMS용 소스로 사용
- Google Cloud for MySQL을 AWS DMS용 소스로 사용
- Google Cloud for PostgreSQL을 AWS DMS용 소스로 사용
- PostgreSQL 데이터베이스를 AWS DMS 소스로 사용
- AWS DMS에서 MySQL 호환 데이터베이스를 소스로 사용
- AWS DMS에서 SAP ASE 데이터베이스를 소스로 사용
- MongoDB를 AWS DMS에서 소스로 사용
- Amazon DocumentDB(MongoDB 호환)를 AWS DMS 소스로 사용
- Amazon S3를 AWS DMS 소스로 사용
- 리눅스, 유닉스, 윈도우용 IBM Db2 및 아마존 RDS 데이터베이스 (Db2 LUW) 를 소스로 사용 AWS DMS
- IBM Db2 for z/OS 데이터베이스를 AWS DMS 소스로 사용

Oracle 데이터베이스를 AWS DMS에서 소스로 사용

를 사용하여 하나 이상의 Oracle 데이터베이스에서 데이터를 마이그레이션할 수 AWS DMS있습니다. Oracle 데이터베이스를 소스로 사용하여 AWS DMS가 지원하는 대상으로 데이터를 마이그레이션할 수 있습니다.

AWS DMS 다음 Oracle 데이터베이스 에디션을 지원합니다.

- Oracle Enterprise Edition
- Oracle Standard Edition
- Oracle Express Edition
- Oracle Personal Edition

소스로 AWS DMS 지원하는 Oracle 데이터베이스 버전에 대한 자세한 내용은 을 참조하십시오출처: AWS DMS.

Secure Sockets Layer(SSL)를 사용하여 Oracle 엔드포인트와 복제 인스턴스 간의 연결을 암호화할 수 있습니다. Oracle 엔드포인트에서 SSL을 사용하는 방법에 대한 자세한 내용은 Oracle 엔드포인트용 SSL 지원 섹션을 참조하십시오.

AWS DMS Oracle TDE (투명 데이터 암호화) 를 사용하여 원본 데이터베이스에 저장된 데이터를 암호화할 수 있습니다. Oracle 소스 엔드포인트에서 Oracle TDE를 사용하는 방법에 대한 자세한 내용은 Oracle을 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS 단원을 참조하십시오.

AWS Oracle 엔드포인트 (및 기타 모든 엔드포인트 유형) 에서 TLS 버전 1.2 이상 사용을 지원하며 TLS 버전 1.3 이상을 사용할 것을 권장합니다.

다음 단계에 따라 Oracle 데이터베이스를 원본 엔드포인트로 AWS DMS 구성하십시오.

1. Oracle 소스 데이터베이스에 액세스할 수 있는 적절한 권한을 가진 Oracle 사용자를 생성하십시오. AWS DMS
2. 선택한 Oracle 데이터베이스 구성을 준수하는 Oracle 소스 엔드포인트를 생성합니다. full-load-only 작업을 생성하기 위해 추가 구성은 필요하지 않습니다.
3. 변경 데이터 캡처를 처리하는 작업 (CDC 전용 또는 전체 로드 및 CDC 작업) 을 생성하려면 Oracle LogMiner 또는 AWS DMS Binary Reader를 선택하여 데이터 변경 사항을 캡처하십시오. LogMiner 또는 바이너리 리더를 선택하면 이후의 일부 권한 및 구성 옵션이 결정됩니다. LogMiner 와 바이너리 리더를 비교하려면 다음 섹션을 참조하십시오.

> **Note**
>
> 전체 로드 작업, CDC 전용 작업, 전체 로드 및 CDC 작업에 대한 자세한 내용은 작업 생성을 참조하세요.

Oracle 소스 데이터베이스 및 AWS DMS사용에 대한 자세한 내용은 다음 섹션을 참조하십시오.

주제

- CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용
- 자체 관리형 또는 AWS관리형 Oracle 원본 데이터베이스를 구성하기 위한 워크플로 AWS DMS Oracle 소스 데이터베이스 구성
- 자체 관리형 Oracle 데이터베이스를 원본으로 사용 AWS DMS
- AWS-Managed Oracle 데이터베이스를 원본으로 사용하여 AWS DMS
- Oracle을 원본으로 사용하는 데 따른 제한 사항 AWS DMS
- Oracle 엔드포인트용 SSL 지원
- Oracle을 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS
- Oracle을 원본으로 사용하기 위해 지원되는 압축 방법 AWS DMS
- Oracle을 원본으로 사용하여 중첩된 테이블을 복제하기 AWS DMS
- Oracle을 소스로 사용하는 경우 Oracle ASM에 REDO를 저장합니다. AWS DMS
- Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS
- Oracle용 소스 데이터 형식

CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용

에서는 AWS DMS Oracle을 소스로 사용하여 CDC (변경 데이터 캡처) 를 수행할 때 리두 로그를 읽는 방법에는 LogMiner Oracle과 AWS DMS Binary Reader라는 두 가지가 있습니다. LogMiner 온라인 리두 로그와 아카이브된 리두 로그 파일을 읽기 위한 Oracle API입니다. Binary Reader는 원시 리두 로그 파일을 직접 읽고 파싱하는 AWS DMS 방법입니다. 이 메서드의 특징은 다음과 같습니다.

기능	LogMiner	Binary Reader
간편한 구성	예	아니요
소스 시스템 I/O 및 CPU에 미치는 영향 감소	아니요	예
향상된 CDC 성능	아니요	예
Oracle 테이블 클러스터 지원	예	아니요
모든 유형의 Oracle HCC(Hybrid Columnar Compression) 지원	예	부분적으로

기능	LogMiner	Binary Reader
		Binary Reader는 CDC 작업에서 QUERY LOW를 지원하지 않습니다. 다른 모든 HCC 유형은 완벽하게 지원됩니다.
Oracle 12c에서만 LOB 열 지원	아니요 (Oracle LogMiner 12c에서는 LOB 지원을 사용할 수 없습니다.)	예
LOB 열에만 영향을 미치는 UPDATE 문 지원	아니요	예
Oracle 투명한 데이터 암호화(TDE) 지원	부분적으로 Oracle을 사용하는 경우 LogMiner, Oracle용 Amazon RDS에 대한 열 수준의 TDE 암호화를 AWS DMS 지원하지 않습니다.	부분적으로 Binary Reader는 자체 관리형 Oracle 데이터베이스에서만 TDE를 지원합니다.
모든 Oracle 압축 방법 지원	예	아니요
XA 트랜잭션 지원	아니요	예
RAC	예 권장되지 않음	예 적극적으로 권장됨

> **Note**
>
> 기본적으로 (CDC) LogMiner 용 오라클을 AWS DMS 사용합니다.
> AWS DMS Oracle 원본 데이터베이스로 작업할 때 투명한 데이터 암호화 (TDE) 방법을 지원합니다. 지정한 TDE 자격 증명이 올바르지 않아도 AWS DMS 마이그레이션 작업이 실패하지 않으므로 암호화된 테이블의 지속적인 복제에 영향을 미칠 수 있습니다. TDE 보안 인증에 대한 자세한 내용은 Oracle을 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS을 참조하세요.

와 LogMiner 함께 사용하면 얻을 수 있는 주요 이점은 다음과 AWS DMS 같습니다.

- LogMiner 암호화 옵션 및 압축 옵션과 같은 대부분의 Oracle 옵션을 지원합니다. Binary Reader는 모든 Oracle 옵션을 지원하지 않고, 특히 압축과 대부분의 암호화 옵션을 지원합니다.
- LogMiner 특히 바이너리 리더 직접 액세스 설정이나 Oracle ASM (자동 저장 장치 관리) 을 사용하여 리두 로그를 관리하는 경우와 비교할 때 더 간단한 구성을 제공합니다.
- LogMiner 에서 사용할 테이블 클러스터를 지원합니다. AWS DMS Binary Reader는 테이블 클러스터를 지원하지 않습니다.

바이너리 리더와 함께 사용할 때의 주요 이점은 다음과 AWS DMS 같습니다.

- 변경 내용이 많은 마이그레이션의 경우 Oracle 원본 데이터베이스를 호스팅하는 컴퓨터에 I/O 또는 CPU에 어느 정도 영향을 미칠 LogMiner 수 있습니다. Binary Reader는 여러 데이터베이스 쿼리를 수행하는 대신 로그를 직접 마이닝하기 때문에 I/O 또는 CPU에 영향을 미칠 가능성이 적습니다.
- 변경 내용이 많은 마이그레이션의 경우 일반적으로 Oracle을 사용하는 것보다 Binary Reader를 사용할 때 CDC 성능이 훨씬 더 좋습니다. LogMiner
- 바이너리 리더는 오라클 버전 12c에서 LOB용 CDC를 지원합니다. LogMiner그렇지 않습니다.

일반적으로 다음 상황 중 하나가 아닌 한 Oracle 데이터베이스를 마이그레이션할 때는 Oracle을 사용하십시오. LogMiner

- 원본 Oracle 데이터베이스에서 여러 마이그레이션 작업을 실행해야 합니다.
- 소스 Oracle 데이터베이스의 변경 사항 볼륨 또는 다시 실행 로그 볼륨이 크거나 변경 사항이 있고 Oracle ASM도 사용 중입니다.

> **Note**
>
> LogMiner 오라클과 AWS DMS 바이너리 리더 사용 사이를 변경하는 경우 CDC 작업을 다시 시작해야 합니다.

Oracle 소스 데이터베이스에서의 CDC 구성

Oracle 소스 엔드포인트가 변경 데이터 캡처(CDC) 작업을 위해 데이터베이스에 연결하려면 추가 연결 속성을 지정해야 할 수 있습니다. 이는 전체 로드 및 CDC 작업이나 CDC 전용 작업에도 해당될 수 있습니다. 지정하는 추가 연결 속성은 리두 로그에 액세스하는 데 사용하는 방법 (Oracle LogMiner 또는 AWS DMS Binary Reader) 에 따라 다릅니다.

소스 엔드포인트를 생성할 때 추가 연결 속성을 지정합니다. 연결 속성 설정이 여러 개인 경우, 추가 공백 없이 세미콜론으로 구분하십시오(예: `oneSetting;thenAnother`).

AWS DMS LogMiner 기본적으로 를 사용합니다. LogMiner를 사용하기 위해 추가 연결 속성을 지정할 필요는 없습니다.

Binary Reader를 사용하여 다시 실행 로그에 액세스하려면 다음 추가 연결 속성을 추가합니다.

```
useLogMinerReader=N;useBfile=Y;
```

다음 형식을 추가 연결 속성에 사용하여 Binary Reader와 함께 ASM을 사용하는 서버에 액세스합니다.

```
useLogMinerReader=N;useBfile=Y;asm_user=asm_username;asm_server=RAC_server_ip_address:port_numb
+ASM;
```

소스 엔드포인트 `Password` 요청 파라미터를 다음과 같이 쉼표로 구분된 Oracle 사용자 암호 및 ASM 암호로 설정합니다.

```
oracle_user_password,asm_user_password
```

Oracle 소스가 ASM을 사용하는 경우, 대규모 트랜잭션 처리를 위해 Binary Reader의 고성능 옵션을 사용할 수 있습니다. 이러한 옵션에는 병렬 스레드 수(`parallelASMReadThreads`) 와 미리 읽기 버퍼 수(`readAheadBlocks`) 를 지정하는 추가 연결 속성이 포함됩니다. 이러한 속성을 함께 설정하면

CDC 작업의 성능을 크게 개선할 수 있습니다. 다음 설정은 대부분의 ASM 구성에서 양호한 결과를 제공합니다.

```
useLogMinerReader=N;useBfile=Y;asm_user=asm_username;asm_server=RAC_server_ip_address:port_numb
+ASM;
    parallelASMReadThreads=6;readAheadBlocks=150000;
```

추가 연결 속성이 지원하는 값에 대한 자세한 내용은 Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS 단원을 참조하십시오.

또한 ASM을 사용하는 Oracle 소스가 있는 CDC 작업의 성능은 선택하는 다른 설정에 따라 달라집니다. 이러한 설정에는 AWS DMS 추가 연결 속성 및 Oracle 소스를 구성하기 위한 SQL 설정이 포함됩니다. ASM을 사용하는 Oracle 소스의 추가 연결 속성에 대한 자세한 내용은 Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS을 참조하세요.

또한 적절한 CDC 시작점을 선택해야 합니다. 일반적으로 이 경우, CDC를 시작할 가장 이른 열린 트랜잭션을 캡처하는 트랜잭션 처리 지점을 식별해야 합니다. 그렇지 않으면 CDC 작업에서 이전의 열린 트랜잭션이 누락될 수 있습니다. Oracle 소스 데이터베이스의 경우, Oracle SCN(시스템 변경 번호)을 기준으로 CDC 네이티브 시작점을 선택하면 가장 이른 열린 트랜잭션을 식별할 수 있습니다. 자세한 정보는 CDC 시작 지점에서 복제를 시작을 참조하세요.

소스로 사용되는 자체 관리형 Oracle 데이터베이스에서 CDC를 구성하는 방법에 대한 자세한 내용은 LogMiner Oracle을 사용하여 리두 로그에 액세스할 때 필요한 계정 권한, AWS DMS 바이너리 리더를 사용하여 리두 로그에 액세스할 때 필요한 계정 권한, Oracle ASM과 함께 Binary Reader를 사용할 때 필요한 추가 계정 권한 섹션을 참조하세요.

CDC를 AWS관리형 Oracle 데이터베이스를 원본으로 구성하는 방법에 대한 자세한 내용은 및 을 참조하십시오 Oracle용 RDS와 함께 바이너리 리더를 사용하도록 CDC 작업을 구성하는 방법: AWS DMS. AWS DMS에서 CDC용 Binary Reader와 함께 Amazon RDS Oracle Standby(읽기 전용 복제본)를 소스로 사용

자체 관리형 또는 AWS관리형 Oracle 원본 데이터베이스를 구성하기 위한 워크플로 AWS DMS

자체 관리형 또는 AWS관리형 Oracle 원본 데이터베이스를 구성하기 위한 워크플로 AWS DMS

자체 관리형 소스 데이터베이스 인스턴스를 구성하려면 CDC 수행 방식에 따라 다음 워크플로 단계를 사용하세요.

이 워크플로 단계의 경우	를 사용하여 LogMiner CDC를 수행하는 경우 다음과 같이 하십시오.	Binary Reader를 사용하여 CDC를 수행하는 경우 다음을 수행
Oracle 계정 권한을 부여합니다.	자체 관리형 Oracle 소스에 필요한 사용자 계정 권한은 다음과 같습니다. AWS DMS을(를) 참조하세요.	자체 관리형 Oracle 소스에 필요한 사용자 계정 권한은 다음과 같습니다. AWS DMS을(를) 참조하세요.
CDC를 사용하여 복제할 소스 데이터베이스를 준비합니다.	를 사용하여 CDC를 위한 Oracle 자체 관리형 원본 데이터베이스 준비 AWS DMS을(를) 참조하세요.	를 사용하여 CDC를 위한 Oracle 자체 관리형 원본 데이터베이스 준비 AWS DMS을(를) 참조하세요.
CDC에 필요한 추가 Oracle 사용자 권한을 부여합니다.	LogMiner Oracle을 사용하여 리두 로그에 액세스할 때 필요한 계정 권한을(를) 참조하세요.	AWS DMS 바이너리 리더를 사용하여 리두 로그에 액세스할 때 필요한 계정 권한을(를) 참조하세요.
ASM을 사용하는 Oracle 인스턴스의 경우, CDC를 위해 ASM에 액세스하는 데 필요한 추가 사용자 계정 권한을 부여합니다.	추가 조치는 없습니다. AWS DMS 추가 계정 권한 없이 Oracle ASM을 지원합니다.	Oracle ASM과 함께 Binary Reader를 사용할 때 필요한 추가 계정 권한 섹션을 참조하십시오.
아직 설정하지 않았다면 CDC용 바이너리 LogMiner 리더를 사용하도록 작업을 구성하십시오.	CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용을(를) 참조하세요.	CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용을(를) 참조하세요.
Oracle Standby를 CDC의 소스로 구성합니다.	AWS DMS Oracle 스탠바이를 소스로 지원하지 않습니다.	AWS DMS에서 CDC용 Binary Reader와 함께 자체 관리형 Oracle Standby를 소스로 사용 섹션을 참조하십시오.

다음 워크플로우 단계를 사용하여 AWS-managed Oracle 소스 데이터베이스 인스턴스를 구성하십시오.

이 워크플로 단계의 경우	를 사용하여 LogMiner CDC를 수행하는 경우 다음과 같이 하십시오.	Binary Reader를 사용하여 CDC를 수행하는 경우 다음을 수행
Oracle 계정 권한을 부여합니다.	자세한 정보는 AWS-Managed Oracle 소스에 필요한 사용자 계정 권한은 다음과 같습니다. AWS DMS을 참조하세요.	자세한 정보는 AWS-Managed Oracle 소스에 필요한 사용자 계정 권한은 다음과 같습니다. AWS DMS을 참조하세요.
CDC를 사용하여 복제할 소스 데이터베이스를 준비합니다.	자세한 정보는 AWS관리형 Oracle 소스 구성 대상 AWS DMS을 참조하세요.	자세한 정보는 AWS관리형 Oracle 소스 구성 대상 AWS DMS을 참조하세요.
CDC에 필요한 추가 Oracle 사용자 권한을 부여합니다.	추가 계정 권한은 필요하지 않습니다.	자세한 정보는 Oracle용 RDS와 함께 바이너리 리더를 사용하도록 CDC 작업을 구성하는 방법: AWS DMS을 참조하세요.
아직 설정하지 않았다면, 사용할 작업을 LogMiner 구성하거나 CDC용 바이너리 리더를 구성하십시오.	자세한 정보는 CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용을 참조하세요.	자세한 정보는 CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용을 참조하세요.
Oracle Standby를 CDC의 소스로 구성합니다.	AWS DMS Oracle 스탠바이를 소스로 지원하지 않습니다.	자세한 정보는 AWS DMS에서 CDC용 Binary Reader와 함께 Amazon RDS Oracle Standby(읽기 전용 복제본)를 소스로 사용을 참조하세요.

자체 관리형 Oracle 데이터베이스를 원본으로 사용 AWS DMS

자체 관리형 데이터베이스는 사용자가 구성 및 제어하는 데이터베이스로, 로컬 온프레미스 데이터베이스 인스턴스이거나 Amazon EC2의 데이터베이스입니다. 다음에서 자체 관리형 Oracle 데이터베이스를 사용할 때 필요한 권한 및 구성에 대해 확인할 수 있습니다. AWS DMS

자체 관리형 Oracle 소스에 필요한 사용자 계정 권한은 다음과 같습니다. AWS DMS

Oracle 데이터베이스를 원본으로 사용하려면 Oracle 엔드포인트 연결 설정에 지정된 Oracle 사용자에게 다음 권한을 부여하십시오. AWS DMS

> **Note**
>
> 권한을 부여할 때는 각 객체의 동의어가 아닌 객체의 실제 이름을 사용하십시오. 예를 들어 밑줄이 없는 V$OBJECT가 아니라 밑줄을 포함하여 V_$OBJECT를 사용하십시오.

```
GRANT CREATE SESSION TO db_user;
GRANT SELECT ANY TRANSACTION TO db_user;
GRANT SELECT ON V_$ARCHIVED_LOG TO db_user;
GRANT SELECT ON V_$LOG TO db_user;
GRANT SELECT ON V_$LOGFILE TO db_user;
GRANT SELECT ON V_$LOGMNR_LOGS TO db_user;
GRANT SELECT ON V_$LOGMNR_CONTENTS TO db_user;
GRANT SELECT ON V_$DATABASE TO db_user;
GRANT SELECT ON V_$THREAD TO db_user;
GRANT SELECT ON V_$PARAMETER TO db_user;
GRANT SELECT ON V_$NLS_PARAMETERS TO db_user;
GRANT SELECT ON V_$TIMEZONE_NAMES TO db_user;
GRANT SELECT ON V_$TRANSACTION TO db_user;
GRANT SELECT ON V_$CONTAINERS TO db_user;
GRANT SELECT ON ALL_INDEXES TO db_user;
GRANT SELECT ON ALL_OBJECTS TO db_user;
GRANT SELECT ON ALL_TABLES TO db_user;
GRANT SELECT ON ALL_USERS TO db_user;
GRANT SELECT ON ALL_CATALOG TO db_user;
GRANT SELECT ON ALL_CONSTRAINTS TO db_user;
GRANT SELECT ON ALL_CONS_COLUMNS TO db_user;
GRANT SELECT ON ALL_TAB_COLS TO db_user;
GRANT SELECT ON ALL_IND_COLUMNS TO db_user;
GRANT SELECT ON ALL_ENCRYPTED_COLUMNS TO db_user;
GRANT SELECT ON ALL_LOG_GROUPS TO db_user;
GRANT SELECT ON ALL_TAB_PARTITIONS TO db_user;
GRANT SELECT ON SYS.DBA_REGISTRY TO db_user;
GRANT SELECT ON SYS.OBJ$ TO db_user;
GRANT SELECT ON DBA_TABLESPACES TO db_user;
GRANT SELECT ON DBA_OBJECTS TO db_user; -- Required if the Oracle version is earlier
 than 11.2.0.3.
```

```
GRANT SELECT ON SYS.ENC$ TO db_user; -- Required if transparent data encryption (TDE)
 is enabled. For more information on using Oracle TDE with AWS DMS, see Oracle# ### ###
# ## #### ### ## AWS DMS.
GRANT SELECT ON GV_$TRANSACTION TO db_user; -- Required if the source database is
 Oracle RAC in AWS DMS versions 3.4.6 and higher.
GRANT SELECT ON V_$DATAGUARD_STATS TO db_user; -- Required if the source database is
 Oracle Data Guard and Oracle Standby is used in the latest release of DMS version
 3.4.6, version 3.4.7, and higher.
```

특정 테이블 목록을 사용하는 경우, 복제된 각 테이블에 대해 다음 추가 권한을 부여합니다.

```
GRANT SELECT on any-replicated-table to db_user;
```

검증 기능으로 LOB 열을 검증하려면 다음 추가 권한을 부여합니다.

```
GRANT EXECUTE ON SYS.DBMS_CRYPTO TO db_user;
```

대신 바이너리 리더를 사용하는 경우 다음 권한을 추가로 부여하십시오. LogMiner

```
GRANT SELECT ON SYS.DBA_DIRECTORIES TO db_user;
```

뷰를 노출할 수 있는 다음 추가 권한을 부여합니다.

```
GRANT SELECT on ALL_VIEWS to dms_user;
```

뷰를 노출하려면 소스 엔드포인트에 exposeViews=true 추가 연결 속성도 추가해야 합니다.

서버리스 복제를 사용할 경우, 다음 추가 권한을 부여합니다.

```
GRANT SELECT on dba_segments to db_user;
```

서버리스 복제에 대해서는 AWS DMS 서버리스로 작업하기을 참조하세요.

Oracle 관련 마이그레이션 전 평가를 사용할 경우, 다음 추가 권한을 부여합니다.

```
GRANT SELECT on gv_$parameter  to dms_user;
```

```
GRANT SELECT on v_$instance to dms_user;
GRANT SELECT on v_$version to dms_user;
GRANT SELECT on gv_$ASM_DISKGROUP to dms_user;
GRANT SELECT on gv_$database to dms_user;
GRANT SELECT on dba_db_links to dms_user;
GRANT SELECT on gv_$log_History to dms_user;
GRANT SELECT on gv_$log to dms_user;
GRANT SELECT ON DBA_TYPES TO db_user;
```

Oracle 관련 마이그레이션 전 평가에 대해서는 오라클 평가를 참조하세요.

Oracle Standby 열린 트랜잭션 처리를 위한 사전 요구 사항

AWS DMS 버전 3.4.6 이상을 사용하는 경우 다음 단계를 수행하여 Oracle Standby의 미결 트랜잭션을 처리하십시오.

1. 기본 데이터베이스에 AWSDMS_DBLINK라는 데이터베이스 링크를 생성합니다. *DMS_USER*는 이 데이터베이스 링크를 사용하여 기본 데이터베이스에 연결합니다. 이 데이터베이스 링크는 대기 인스턴스에서 실행되어 기본 데이터베이스에서 실행 중인 열린 트랜잭션을 쿼리합니다. 다음 예를 참조하세요.

   ```
   CREATE PUBLIC DATABASE LINK AWSDMS_DBLINK
      CONNECT TO DMS_USER IDENTIFIED BY DMS_USER_PASSWORD
      USING '(DESCRIPTION=
               (ADDRESS=(PROTOCOL=TCP)(HOST=PRIMARY_HOST_NAME_OR_IP)(PORT=PORT))
               (CONNECT_DATA=(SERVICE_NAME=SID))
             )';
   ```

2. 다음 예제와 같이 *DMS_USER*를 사용하여 데이터베이스 링크에 연결되었는지 확인합니다.

   ```
   select 1 from dual@AWSDMS_DBLINK
   ```

를 사용하여 CDC를 위한 Oracle 자체 관리형 원본 데이터베이스 준비 AWS DMS

다음을 수행하여 자체 관리형 Oracle 데이터베이스를 CDC 작업을 실행할 소스로 준비합니다.

- 원본 데이터베이스 버전을 AWS DMS 지원하는지 확인.
- ARCHIVELOG 모드가 켜져 있는지 확인.

- 보충 로깅 설정.

원본 데이터베이스 버전을 AWS DMS 지원하는지 확인

다음과 같은 쿼리를 실행하여 AWS DMS가 현재 버전의 Oracle 데이터베이스를 지원하는지 확인합니다.

```
SELECT name, value, description FROM v$parameter WHERE name = 'compatible';
```

여기서 name, value 및 description는 name의 값을 기준으로 쿼리되는 데이터베이스의 어딘가에 있는 열입니다. 이 쿼리가 오류 없이 실행되면 데이터베이스의 현재 버전을 AWS DMS 지원하는 것이므로 마이그레이션을 계속할 수 있습니다. 쿼리에서 오류가 발생하면 데이터베이스의 현재 버전을 AWS DMS 지원하지 않는 것입니다. 마이그레이션을 진행하려면 먼저 Oracle 데이터베이스를 에서 지원하는 버전으로 AWS DMS변환하십시오.

ARCHIVELOG 모드가 켜져 있는지 확인

Oracle은 ARCHIVELOG 모드와 NOARCHIVELOG 모드 두 가지 모드로 실행할 수 있습니다. CDC 작업을 실행하려면 데이터베이스를 ARCHIVELOG 모드에서 실행합니다. 데이터베이스가 ARCHIVELOG 모드 상태인지 확인하려면 다음 쿼리를 실행합니다.

```
SQL> SELECT log_mode FROM v$database;
```

NOARCHIVELOG 모드가 반환되면 Oracle 지침에 따라 데이터베이스를 ARCHIVELOG로 설정합니다.

보충 로깅 설정

진행 중인 변경 사항을 AWS DMS 캡처하려면 Oracle 소스 데이터베이스에서 최소한의 보충 로깅을 활성화해야 합니다. 또한 데이터베이스의 복제된 각 테이블에서 보충 로깅을 활성화해야 합니다.

기본적으로 모든 복제된 AWS DMS 테이블에 PRIMARY KEY 보충 로깅을 추가합니다. PRIMARY KEY보충 AWS DMS 로깅을 추가할 수 있게 하려면 복제된 각 테이블에 다음 권한을 부여하십시오.

```
ALTER on any-replicated-table;
```

추가 연결 속성을 AWS DMS 사용하여 추가된 기본 PRIMARY KEY 보충 로깅을 비활성화할 수 있습니다. addSupplementalLogging 자세한 정보는 Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS을 참조하세요.

복제 작업에서 WHERE 절을 사용하여 프라이머리 키 열을 참조하지 않는 테이블을 업데이트하는 경우, 보충 로깅을 켜야 합니다.

수동으로 보충 로깅을 설정하려면

1. 다음 쿼리를 실행하여 데이터베이스에 보충 로깅이 이미 활성화되어 있는지 확인합니다.

   ```
   SELECT supplemental_log_data_min FROM v$database;
   ```

   반환된 결과가 YES 또는 IMPLICIT인 경우, 데이터베이스에 보충 로깅이 활성화되어 있습니다.

   활성화되어 있지 않다면 다음 명령을 실행하여 데이터베이스에 보충 로깅을 활성화합니다.

   ```
   ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;
   ```

2. 복제된 각 테이블에 필요한 보충 로깅이 추가되었는지 확인합니다.

   다음을 고려하세요.

   - 테이블에 ALL COLUMNS 보충 로깅이 추가되면 로깅을 더 추가할 필요가 없습니다.
   - 프라이머리 키가 있는 경우 프라이머리 키에 대한 보충 로깅을 추가합니다. 형식을 사용하여 프라이머리 키 자체에 보충 로깅을 추가하거나 데이터베이스에서 프라이머리 키 열에 보충 로깅을 추가하여 이 작업을 수행할 수 있습니다.

     ```
     ALTER TABLE Tablename ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;
     ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;
     ```

   - 프라이머리 키가 없고 테이블에 고유 인덱스가 하나만 있는 경우, 고유 인덱스의 모든 열을 보충 로그에 추가합니다.

     ```
     ALTER TABLE TableName ADD SUPPLEMENTAL LOG GROUP LogGroupName
      (UniqueIndexColumn1[, UniqueIndexColumn2] ...) ALWAYS;
     ```

     SUPPLEMENTAL LOG DATA (UNIQUE INDEX) COLUMNS를 사용해도 고유 인덱스 열이 로그에 추가되지 않습니다.

   - 기본 키가 없고 테이블에 여러 개의 고유 인덱스가 있는 경우 알파벳순 오름차순 목록에서 첫 번째 고유 인덱스를 AWS DMS 선택합니다. 이전 항목에서처럼 선택한 인덱스의 열에 보충 로깅을 추가해야 합니다.

SUPPLEMENTAL LOG DATA (UNIQUE INDEX) COLUMNS를 사용해도 고유 인덱스 열이 로그에 추가되지 않습니다.

- 프라이머리 키가 없고 고유 인덱스가 없는 경우 모든 열에 보충 로깅을 추가합니다.

```
ALTER TABLE TableName ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;
```

일부의 경우, 대상 테이블 프라이머리 키 또는 고유 색인이 원본 테이블 프라이머리 키 또는 고유 색인과 다릅니다. 이러한 경우, 대상 테이블 프라이머리 키 또는 고유 인덱스를 구성하는 소스 테이블 열에 보충 로깅을 수동으로 추가합니다.

또한 대상 테이블 프라이머리 키를 변경하는 경우, 원래 프라이머리 키 또는 고유 인덱스 대신 대상 고유 인덱스에 보충 로깅을 추가해야 합니다.

테이블에 대해 필터 또는 변환이 정의된 경우 추가 로깅을 활성화해야 할 수 있습니다.

다음을 고려하세요.

- 테이블에 ALL COLUMNS 보충 로깅이 추가되면 로깅을 더 추가할 필요가 없습니다.
- 테이블에 고유 인덱스 또는 프라이머리 키가 있다면 필터 또는 변환에 포함된 각 열에 보충 로깅을 추가합니다. 하지만 해당 열이 프라이머리 키 또는 고유 인덱스 열과 다른 경우에만 그렇게 하세요.
- 변환에 하나의 열만 포함되는 경우, 이 열을 보충 로깅 그룹에 추가하지 마세요. 예를 들어 A+B 변환의 경우 열 A 및 B 모두에 보충 로깅을 추가합니다. 그러나 substring(A,10) 변환의 경우 열 A에 보충 로깅을 추가하지 마십시오.
- 프라이머리 키 또는 고유 인덱스 열과 필터링되거나 변환되는 그 밖의 열에 보충 로깅을 설정하려면 USER_LOG_GROUP 보충 로깅을 설정하면 됩니다. 프라이머리 키 또는 고유 인덱스 열과 필터링되거나 변환되는 그 밖의 특정 열 모두에 이 보충 로깅을 추가합니다.

  예를 들어, 프라이머리 키 ID와 열 NAME 기준 필터를 사용하여 TEST.LOGGING이라는 테이블을 복제하려면 다음과 비슷한 명령을 실행하여 로그 그룹 보충 로깅을 만들 수 있습니다.

```
ALTER TABLE TEST.LOGGING ADD SUPPLEMENTAL LOG GROUP TEST_LOG_GROUP (ID, NAME) ALWAYS;
```

LogMiner Oracle을 사용하여 리두 로그에 액세스할 때 필요한 계정 권한

Oracle을 사용하여 리두 로그에 액세스하려면 Oracle LogMiner 엔드포인트 연결 설정에 지정된 Oracle 사용자에게 다음 권한을 부여하십시오.

```
GRANT EXECUTE on DBMS_LOGMNR to db_user;
GRANT SELECT on V_$LOGMNR_LOGS to db_user;
GRANT SELECT on V_$LOGMNR_CONTENTS to db_user;
GRANT LOGMINING to db_user; -- Required only if the Oracle version is 12c or higher.
```

AWS DMS 바이너리 리더를 사용하여 리두 로그에 액세스할 때 필요한 계정 권한

AWS DMS 바이너리 리더를 사용하여 리두 로그에 액세스하려면 Oracle 엔드포인트 연결 설정에 지정된 Oracle 사용자에게 다음 권한을 부여하십시오.

```
GRANT SELECT on v_$transportable_platform to db_user;   -- Grant this privilege if the
 redo logs are stored in Oracle Automatic Storage Management (ASM) and AWS DMS accesses
 them from ASM.
GRANT CREATE ANY DIRECTORY to db_user;                  -- Grant this privilege to
 allow AWS DMS to use Oracle BFILE read file access in certain cases. This access is
 required when the replication instance doesn't have file-level access to the redo logs
 and the redo logs are on non-ASM storage.
GRANT EXECUTE on DBMS_FILE_TRANSFER to db_user;         -- Grant this privilege to copy
 the redo log files to a temporary folder using the CopyToTempFolder method.
GRANT EXECUTE on DBMS_FILE_GROUP to db_user;
```

Binary Reader는 Oracle 디렉터리를 포함하는 Oracle 파일 기능과 함께 작동합니다. 각 Oracle 디렉터리 객체에는 처리할 다시 실행 로그 파일이 들어 있는 폴더의 이름이 포함됩니다. 이러한 Oracle 디렉터리는 파일 시스템 수준에서 표시되지 않습니다. 대신, Oracle 데이터베이스 수준에서 생성되는 논리적 디렉터리입니다. Oracle `ALL_DIRECTORIES` 뷰에서 볼 수 있습니다.

이러한 Oracle 디렉토리를 AWS DMS 생성하려면 앞에 지정된 `CREATE ANY DIRECTORY` 권한을 부여하십시오. AWS DMS 접두사를 사용하여 디렉토리 이름을 생성합니다. `DMS_ CREATE ANY DIRECTORY` 권한을 부여하지 않는 경우 수동으로 해당 디렉터리를 생성해야 합니다. 수동으로 Oracle 디렉터리를 만들 때 Oracle 소스 엔드포인트에 지정된 Oracle 사용자가 이러한 디렉터리를 만든 사용자가 아닌 경우도 있습니다. 이런 경우에도 `READ on DIRECTORY` 권한을 부여하십시오.

Oracle 소스 엔드포인트가 ADG (Active Dataguard Standby) 상태인 경우 데이터베이스 블로그의 ADG와 함께 바이너리 리더를 사용하는 방법 게시물을 참조하십시오. AWS

> **Note**
>
> AWS DMS CDC는 자동 리두 전송 서비스를 사용하도록 구성되지 않은 액티브 Dataguard 스탠바이를 지원하지 않습니다.

경우에 따라 Oracle Managed Files(OMF)를 사용하여 로그를 저장할 수 있습니다. 또는 소스 엔드포인트가 ADG에 있고 따라서 CREATE ANY DIRECTORY 권한을 부여할 수 없습니다. 이러한 경우에는 복제 작업을 시작하기 전에 가능한 모든 로그 위치가 포함된 디렉토리를 수동으로 생성하십시오. AWS DMS AWS DMS 가 미리 만들어진 디렉터리를 찾지 못하면 작업이 중지됩니다. 또한 AWS DMS는 ALL_DIRECTORIES 뷰에서 생성한 항목을 삭제하지 않으므로 수동으로 삭제하십시오.

Oracle ASM과 함께 Binary Reader를 사용할 때 필요한 추가 계정 권한

Binary Reader를 사용하여 ASM(Automatic Storage Management)의 다시 실행 로그에 액세스하려면 Oracle 엔드포인트 연결 설정에 지정된 Oracle 사용자에게 다음 권한을 부여합니다.

```
SELECT ON v_$transportable_platform
SYSASM -- To access the ASM account with Oracle 11g Release 2 (version 11.2.0.2) and
 higher, grant the Oracle endpoint user the SYSASM privilege. For older supported
 Oracle versions, it's typically sufficient to grant the Oracle endpoint user the
 SYSDBA privilege.
```

명령 프롬프트를 열고 앞서 지정한 Oracle 버전에 따라 다음 명령문 중 하나를 간접적으로 호출하여 ASM 계정 액세스를 검증할 수 있습니다.

SYSDBA 권한이 필요한 경우 다음을 사용하십시오.

```
sqlplus asmuser/asmpassword@+asmserver as sysdba
```

SYSASM 권한이 필요한 경우 다음을 사용하십시오.

```
sqlplus asmuser/asmpassword@+asmserver as sysasm
```

AWS DMS에서 CDC용 Binary Reader와 함께 자체 관리형 Oracle Standby를 소스로 사용

CDC용 Binary Reader를 사용할 때 Oracle Standby 인스턴스를 소스로 구성하려면 다음 사전 요구 사항부터 시작합니다.

- AWS DMS 현재는 오라클 액티브 데이터 가드 스탠바이만 지원합니다.
- Oracle Data Guard 구성에서 다음을 사용하는지 확인하세요.
  - 다시 실행 데이터의 자동 전송을 위한 다시 실행 전송 서비스.
  - 대기 데이터베이스에 다시 실행을 자동으로 적용하는 서비스 적용.

이러한 요구 사항이 충족되는지 확인하려면 다음 쿼리를 실행합니다.

```
SQL> select open_mode, database_role from v$database;
```

해당 쿼리의 출력에서 대기 데이터베이스가 읽기 전용 모드로 열리고 다시 실행이 자동으로 적용되는지 확인합니다. 예:

```
OPEN_MODE            DATABASE_ROLE
-------------------  ----------------
READ ONLY WITH APPLY  PHYSICAL STANDBY
```

CDC용 Binary Reader를 사용할 때 Oracle Standby 인스턴스를 소스로 구성하려면

1. 대기 로그 파일에 액세스하는 데 필요한 추가 권한을 부여합니다.

   ```
   GRANT SELECT ON v_$standby_log TO db_user;
   ```

2. AWS Management Console 또는 AWS CLI를 사용하여 Oracle Standby의 소스 엔드포인트를 생성합니다. 엔드포인트를 생성할 때 다음과 같은 추가 연결 속성을 지정합니다.

   ```
   useLogminerReader=N;useBfile=Y;
   ```

   > **Note**
   >
   > AWS DMS에서는 추가 연결 속성을 사용하여 리두 로그 대신 아카이브 로그에서 마이그레이션할지 여부를 지정할 수 있습니다. 자세한 정보는 Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS을 참조하세요.

3. 아카이브된 로그 대상을 구성합니다.

   ASM이 없는 Oracle 소스용 DMS Binary Reader는 Oracle 디렉토리를 사용하여 아카이빙된 다시 실행 로그에 액세스합니다. 데이터베이스가 FRA(Fast Recovery Area)를 아카이브 로그 대상으로 사용하도록 구성된 경우, 아카이브 다시 실행 파일의 위치는 일정하지 않습니다. 아카이빙된 다시 실행 로그가 생성되는 날마다 FRA에 디렉터리 이름 형식 YYYY_MM_DD를 사용하여 새 디렉터리가 생성됩니다. 예:

   ```
   DB_RECOVERY_FILE_DEST/SID/archivelog/YYYY_MM_DD
   ```

DMS가 새로 생성된 FRA 디렉터리의 아카이빙된 다시 실행 파일에 액세스해야 하고 기본 읽기-쓰기 데이터베이스를 소스로 사용하는 경우, DMS는 다음과 같이 새 Oracle 디렉터리를 생성하거나 기존 Oracle 디렉터리를 대체합니다.

```
CREATE OR REPLACE DIRECTORY dmsrep_taskid AS 'DB_RECOVERY_FILE_DEST/SID/archivelog/
YYYY_MM_DD';
```

대기 데이터베이스를 소스로 사용하는 경우, 데이터베이스가 읽기 전용 모드이기 때문에 DMS는 Oracle 디렉터리를 생성하거나 바꿀 수 없습니다. 하지만 다음 추가 단계 중 하나를 수행할 수 있습니다.

a. Oracle이 일일 하위 디렉터리를 생성하지 않는 구성에서는 FRA 대신 실제 경로를 사용하도록 `log_archive_dest_id_1`을 수정합니다.

```
ALTER SYSTEM SET log_archive_dest_1='LOCATION=full directory path'
```

그런 다음 DMS가 사용할 Oracle 디렉터리 객체를 생성합니다.

```
CREATE OR REPLACE DIRECTORY dms_archived_logs AS 'full directory path';
```

b. 추가 아카이브 로그 대상과 해당 대상을 가리키는 Oracle 디렉터리 객체를 생성합니다. 예:

```
ALTER SYSTEM SET log_archive_dest_3='LOCATION=full directory path';
CREATE DIRECTORY dms_archived_log AS 'full directory path';
```

그런 다음 작업 소스 엔드포인트에 추가 연결 속성을 추가합니다.

```
archivedLogDestId=3
```

c. DMS가 사용할 Oracle 디렉터리 객체를 수동으로 사전 생성합니다.

```
CREATE DIRECTORY dms_archived_log_20210301 AS 'DB_RECOVERY_FILE_DEST/SID/
archivelog/2021_03_01';
CREATE DIRECTORY dms_archived_log_20210302 AS 'DB_RECOVERY_FILE_DEST>/SID>/
archivelog/2021_03_02';
...
```

d. 매일 실행되고 필요한 디렉터리를 생성하는 Oracle 스케줄러 작업을 생성합니다.

AWS DMS에서 OCI(Oracle Cloud Infrastructure)의 사용자 관리형 데이터베이스를 CDC의 소스로 사용

사용자 관리형 데이터베이스는 가상 머신(VM), 베어 메탈 또는 Exadata 서버에서 만든 Oracle 데이터베이스와 같이 사용자가 구성하고 제어하는 데이터베이스입니다. 또는 OCI(Oracle Cloud Infrastructure)와 같은 전용 인프라에서 실행되고 사용자가 구성하고 제어하는 데이터베이스인 경우도 있습니다. 다음 정보는 AWS DMS에서 OCI의 Oracle 사용자 관리형 데이터베이스를 변경 데이터 캡처(CDC) 소스로 사용할 때 필요한 권한과 구성을 설명합니다.

OCI에서 호스팅하는 사용자 관리형 Oracle 데이터베이스를 변경 데이터 캡처의 소스로 구성하려면

1. OCI의 사용자 관리형 Oracle 소스 데이터베이스에 필요한 사용자 계정 권한을 부여합니다. 자세한 내용은 자체 관리형 Oracle 소스 엔드포인트에 대한 계정 권한을 참조하세요.
2. Binary Reader를 사용하여 다시 실행 로그에 액세스할 때 필요한 계정 권한을 부여합니다. 자세한 내용은 Binary Reader를 사용할 때 필요한 계정 권한을 참조하세요.
3. Oracle ASM(Automatic Storage Management)에서 Binary Reader를 사용할 때 필요한 계정 권한을 추가합니다. 자세한 내용은 Binary Reader를 Oracle ASM과 함께 사용할 때 필요한 추가 계정 권한을 참조하세요.
4. 보충 로깅을 설정합니다. 자세한 내용은 보충 로깅 설정을 참조하세요.
5. TDE 암호화를 설정합니다. 자세한 내용은 Oracle 데이터베이스를 소스 엔드포인트로 사용할 때 암호화 방법을 참조하세요.

OCI(Oracle Cloud Infrastructure)의 Oracle 소스 데이터베이스에서 데이터를 복제할 때는 다음과 같은 제한 사항이 적용됩니다.

제한 사항

- DMS는 Oracle을 사용하여 리두 LogMiner 로그에 액세스하는 것을 지원하지 않습니다.
- DMS는 자율 DB를 지원하지 않습니다.

AWS-Managed Oracle 데이터베이스를 원본으로 사용하여 AWS DMS

AWS관리형 데이터베이스는 Amazon RDS, Amazon Aurora 또는 Amazon S3와 같은 아마존 서비스에 있는 데이터베이스입니다. 다음에서 AWS관리형 Oracle 데이터베이스를 사용할 때 설정해야 하는 권한 및 구성을 확인할 수 있습니다. AWS DMS

AWS-Managed Oracle 소스에 필요한 사용자 계정 권한은 다음과 같습니다. AWS DMS

Oracle 소스 엔드포인트 정의에 지정된 Oracle 사용자 계정에 다음 권한을 부여합니다.

> **Important**
>
> *db_user*와 *any-replicated-table* 같은 모든 파라미터 값에서 대소문자를 구분하는 식별자로 값을 지정하지 않는 한 Oracle은 값이 모두 대문자인 것으로 가정합니다. 예를 들어 `CREATE USER myuser` 또는 `CREATE USER MYUSER`에서처럼 따옴표를 사용하지 않고 *db_user* 값을 생성한다고 가정해 보겠습니다. 이 경우 Oracle은 값을 모두 대문자(MYUSER)로 식별하고 저장합니다. `CREATE USER "MyUser"` 또는 `CREATE USER 'MyUser'`에서처럼 따옴표를 사용하면 Oracle은 사용자가 지정한 대소문자를 구별하는 값(MyUser)을 식별하고 저장합니다.

```
GRANT CREATE SESSION to db_user;
GRANT SELECT ANY TRANSACTION to db_user;
GRANT SELECT on DBA_TABLESPACES to db_user;
GRANT SELECT ON any-replicated-table to db_user;
GRANT EXECUTE on rdsadmin.rdsadmin_util to db_user;
 -- For Oracle 12c or higher:
GRANT LOGMINING to db_user; - Required only if the Oracle version is 12c or higher.
```

또한 아래 나온 것처럼 Amazon RDS 절차 `rdsadmin.rdsadmin_util.grant_sys_object`를 사용하여 SYS 객체에 대한 SELECT 및 EXECUTE 권한을 부여합니다. 자세한 내용은 SYS 객체에 SELECT 또는 EXECUTE 권한 부여를 참조하십시오.

```
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_VIEWS', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_TAB_PARTITIONS', 'db_user',
 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_INDEXES', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_OBJECTS', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_TABLES', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_USERS', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_CATALOG', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_CONSTRAINTS', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_CONS_COLUMNS', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_TAB_COLS', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_IND_COLUMNS', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_LOG_GROUPS', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$ARCHIVED_LOG', 'db_user', 'SELECT');
```

```
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOG', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOGFILE', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$DATABASE', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$THREAD', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$PARAMETER', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$NLS_PARAMETERS', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$TIMEZONE_NAMES', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$TRANSACTION', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$CONTAINERS', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_REGISTRY', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('OBJ$', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('ALL_ENCRYPTED_COLUMNS', 'db_user',
 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOGMNR_LOGS', 'db_user', 'SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$LOGMNR_CONTENTS','db_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_LOGMNR', 'db_user', 'EXECUTE');

-- (as of Oracle versions 12.1 and higher)
exec rdsadmin.rdsadmin_util.grant_sys_object('REGISTRY$SQLPATCH', 'db_user', 'SELECT');

-- (for Amazon RDS Active Dataguard Standby (ADG))
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$STANDBY_LOG', 'db_user', 'SELECT');

-- (for transparent data encryption (TDE))

exec rdsadmin.rdsadmin_util.grant_sys_object('ENC$', 'db_user', 'SELECT');

-- (for validation with LOB columns)
exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_CRYPTO', 'db_user', 'EXECUTE');

-- (for binary reader)
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_DIRECTORIES','db_user','SELECT');

-- Required when the source database is Oracle Data guard, and Oracle Standby is used
 in the latest release of DMS version 3.4.6, version 3.4.7, and higher.

exec rdsadmin.rdsadmin_util.grant_sys_object('V_$DATAGUARD_STATS', 'db_user',
 'SELECT');
```

AWS DMS 에서 Amazon RDS Active Dataguard Standby(ADG)를 사용하는 방법에 대한 자세한 내용은 AWS DMS에서 CDC용 Binary Reader와 함께 Amazon RDS Oracle Standby(읽기 전용 복제본)를 소스로 사용 섹션을 참조하세요.

Oracle TDE와 함께 AWS DMS사용하는 방법에 대한 자세한 내용은 을 참조하십시오. Oracle을 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS

Oracle Standby 열린 트랜잭션 처리를 위한 사전 요구 사항

AWS DMS 버전 3.4.6 이상을 사용하는 경우 다음 단계를 수행하여 Oracle Standby의 미결 트랜잭션을 처리하십시오.

1. 기본 데이터베이스에 AWSDMS_DBLINK라는 데이터베이스 링크를 생성합니다. *DMS_USER*는 이 데이터베이스 링크를 사용하여 기본 데이터베이스에 연결합니다. 이 데이터베이스 링크는 대기 인스턴스에서 실행되어 기본 데이터베이스에서 실행 중인 열린 트랜잭션을 쿼리합니다. 다음 예를 참조하세요.

```
CREATE PUBLIC DATABASE LINK AWSDMS_DBLINK
   CONNECT TO DMS_USER IDENTIFIED BY DMS_USER_PASSWORD
   USING '(DESCRIPTION=
            (ADDRESS=(PROTOCOL=TCP)(HOST=PRIMARY_HOST_NAME_OR_IP)(PORT=PORT))
            (CONNECT_DATA=(SERVICE_NAME=SID))
          )';
```

2. 다음 예제와 같이 *DMS_USER*를 사용하여 데이터베이스 링크에 연결되었는지 확인합니다.

```
select 1 from dual@AWSDMS_DBLINK
```

AWS관리형 Oracle 소스 구성 대상 AWS DMS

AWS-managed Oracle 데이터베이스를 원본으로 사용하기 전에 Oracle 데이터베이스에 대해 AWS DMS다음 작업을 수행하십시오.

- 자동 백업을 활성화합니다. 자동 백업 활성화에 대한 자세한 내용은 Amazon RDS 사용 설명서의 자동 백업 활성화 섹션을 참조하세요.
- 보충 로깅을 설정합니다.
- 보관을 설정합니다. Oracle DB 인스턴스용 Amazon RDS의 리두 로그를 보관하면 Oracle 또는 바이너리 AWS DMS 리더를 사용하여 로그 정보를 검색할 수 있습니다. LogMiner

보관을 설정하려면

1. `rdsadmin.rdsadmin_util.set_configuration` 명령을 실행하여 보관을 설정합니다.

   예를 들어 보관된 다시 실행 로그를 24시간 동안 유지하려면 다음 명령을 실행합니다.

   ```
   exec rdsadmin.rdsadmin_util.set_configuration('archivelog retention hours',24);
   commit;
   ```

   > **Note**
   >
   > 변경 사항을 적용하려면 커밋해야 합니다.

2. 지정된 보존 기간 동안 보관된 다시 실행 로그를 위한 스토리지 공간이 충분한지 확인해야 합니다. 예를 들어 보존 기간이 24시간인 경우, 일반적인 트랜잭션 처리 시간 동안 보관된 다시 실행 로그의 누적된 총 크기를 계산하고 그 합계에 24를 곱합니다. 계산된 이 24시간 합계를 사용 가능한 스토리지 공간과 비교하여 전체 24시간 트랜잭션 처리를 처리할 수 있는 충분한 스토리지 공간이 있는지 결정합니다.

보충 로깅을 설정하려면

1. 데이터베이스 수준에서 보충 로깅을 활성화하려면 다음 명령을 실행합니다.

   ```
   exec rdsadmin.rdsadmin_util.alter_supplemental_logging('ADD');
   ```

2. 다음 명령을 실행하여 프라이머리 키 보충 로깅을 활성화합니다.

   ```
   exec rdsadmin.rdsadmin_util.alter_supplemental_logging('ADD','PRIMARY KEY');
   ```

3. (선택 사항) 테이블 수준에서 키 수준 보충 로깅을 활성화합니다.

   키 수준 보충 로깅을 활성화하면 소스 데이터베이스에 약간의 오버헤드가 발생합니다. 따라서 테이블 하위 집합만을 마이그레이션할 경우, 테이블 수준에서 키 수준 보충 로깅을 활성화하는 것이 좋습니다. 테이블 수준에서 키 수준 보충 로깅을 활성화하려면 다음 명령을 실행합니다.

   ```
   alter table table_name add supplemental log data (PRIMARY KEY) columns;
   ```

Oracle용 RDS와 함께 바이너리 리더를 사용하도록 CDC 작업을 구성하는 방법: AWS DMS

CDC용 바이너리 리더를 사용하여 원본 Amazon RDS for Oracle용 인스턴스 다시 실행 로그에 AWS DMS 액세스하도록 구성할 수 있습니다.

> **Note**
>
> LogMinerOracle을 사용하려면 필요한 최소 사용자 계정 권한이면 충분합니다. 자세한 정보는 AWS-Managed Oracle 소스에 필요한 사용자 계정 권한은 다음과 같습니다. AWS DMS을 참조하세요.

AWS DMS Binary Reader를 사용하려면 AWS DMS 버전에 따라 Oracle 소스 엔드포인트에 대한 추가 설정 및 추가 연결 속성을 지정하십시오.

Binary Reader는 다음 버전의 Amazon RDS for Oracle에서 지원됩니다.

- Oracle 11.2 – 버전 11.2.0.4V11 이상
- Oracle 12.1 – 버전 12.1.0.2.V7 이상
- Oracle 12.2 – 모든 버전
- Oracle 18.0 – 모든 버전
- Oracle 19.0 – 모든 버전

Binary Reader를 사용하여 CDC를 구성하려면

1. Amazon RDS for Oracle 소스 데이터베이스에 마스터 사용자로 로그인하고 다음 저장 프로시저를 실행하여 서버 수준 디렉터리를 생성합니다.

   ```
   exec rdsadmin.rdsadmin_master_util.create_archivelog_dir;
   exec rdsadmin.rdsadmin_master_util.create_onlinelog_dir;
   ```

2. Oracle 소스 엔드포인트에 액세스하는 데 사용되는 Oracle 사용자 계정에 다음 권한을 부여합니다.

   ```
   GRANT READ ON DIRECTORY ONLINELOG_DIR TO db_user;
   GRANT READ ON DIRECTORY ARCHIVELOG_DIR TO db_user;
   ```

3. Amazon RDS Oracle 소스 엔드포인트에서 다음 추가 연결 속성을 설정합니다.

- RDS Oracle 버전 11.2 및 12.1의 경우, 다음을 설정합니다.

```
useLogminerReader=N;useBfile=Y;accessAlternateDirectly=false;useAlternateFolderForOnline=
oraclePathPrefix=/rdsdbdata/db/{$DATABASE_NAME}_A/;usePathPrefix=/rdsdbdata/
log/;replacePathPrefix=true;
```

- RDS Oracle 버전 12.2, 18.0, 19.0의 경우, 다음을 설정합니다.

```
useLogminerReader=N;useBfile=Y;
```

> **Note**
>
> 여러 속성 설정의 경우, 세미콜론 구분자(;) 뒤에 공백이 오면 안 됩니다(예: `oneSetting;thenAnother`).

CDC 작업 구성에 대한 자세한 내용은 Oracle 소스 데이터베이스에서의 CDC 구성 섹션을 참조하세요.

AWS DMS에서 CDC용 Binary Reader와 함께 Amazon RDS Oracle Standby(읽기 전용 복제본)를 소스로 사용

AWS DMS에서 CDC용 Binary Reader를 사용할 때 Amazon RDS for Oracle Standby를 소스로 사용하려면 다음 사전 요구 사항을 확인하세요.

- Oracle 마스터 사용자를 사용하여 Binary Reader를 설정합니다.
- AWS DMS 현재 Oracle 액티브 데이터 가드 스탠바이만 사용할 수 있는지 확인하십시오.

그런 다음 CDC용 Binary Reader를 사용할 때 다음 절차를 사용하여 RDS for Oracle Standby를 소스로 사용합니다.

CDC용 Binary Reader를 사용할 때 RDS for Oracle Standby를 소스로 구성하려면

1. RDS for Oracle 기본 인스턴스에 마스터 사용자로 로그인합니다.
2. Amazon RDS 사용 설명서에 설명된 대로 다음 저장 프로시저를 실행하여 서버 수준 디렉터리를 생성합니다.

```
exec rdsadmin.rdsadmin_master_util.create_archivelog_dir;
```

```
exec rdsadmin.rdsadmin_master_util.create_onlinelog_dir;
```

3. 2단계에서 생성한 디렉터리를 식별합니다.

```
SELECT directory_name, directory_path FROM all_directories
WHERE directory_name LIKE ( 'ARCHIVELOG_DIR_%' )
        OR directory_name LIKE ( 'ONLINELOG_DIR_%' )
```

예를 들어 위 코드는 다음과 같은 디렉터리 목록을 표시합니다.

DIRECTORY_NAME	DIRECTORY_PATH
ARCHIVELOG_DIR_A	/rdsdbdata/db/ORCL_A/arch
ARCHIVELOG_DIR_B	/rdsdbdata/db/ORCL_B/arch
ONLINELOG_DIR_A	/rdsdbdata/db/ORCL_A/onlinelog
ONLINELOG_DIR_B	/rdsdbdata/db/ORCL_B/onlinelog

4. Oracle Standby에 액세스하는 데 사용되는 Oracle 사용자 계정에 이전 디렉터리에 대한 Read 권한을 부여합니다.

```
GRANT READ ON DIRECTORY ARCHIVELOG_DIR_A TO db_user;
GRANT READ ON DIRECTORY ARCHIVELOG_DIR_B TO db_user;
GRANT READ ON DIRECTORY ONLINELOG_DIR_A TO db_user;
GRANT READ ON DIRECTORY ONLINELOG_DIR_B TO db_user;
```

5. 기본 인스턴스에서 아카이브 로그 스위치를 수행합니다. 이렇게 하면 ALL_DIRECTORIES에 대한 변경 사항이 Oracle Standby에도 포팅되도록 할 수 있습니다.
6. Oracle Standby에서 ALL_DIRECTORIES 쿼리를 실행하여 변경 사항이 적용되었는지 확인합니다.
7. AWS DMS 관리 콘솔 또는 AWS Command Line Interface (AWS CLI) 를 사용하여 Oracle Standby의 소스 엔드포인트를 생성합니다. 엔드포인트를 생성할 때 다음과 같은 추가 연결 속성을 지정합니다.

```
useLogminerReader=N;useBfile=Y;archivedLogDestId=1;additionalArchivedLogDestId=2
```

8. 엔드포인트를 생성한 후 콘솔의 엔드포인트 생성 페이지에서 엔드포인트 연결 테스트 또는 AWS CLI test-connection 명령을 사용하여 연결이 설정되었는지 확인합니다.

Oracle을 원본으로 사용하는 데 따른 제한 사항 AWS DMS

AWS DMS용 소스로 Oracle 데이터베이스 사용 시 다음 제한 사항이 적용됩니다.

- AWS DMS AWS DMS 버전 3.5.0 이상에서 Oracle 확장 데이터 유형을 지원합니다.
- AWS DMS 긴 객체 이름 (30바이트 초과) 은 지원하지 않습니다.
- AWS DMS 함수 기반 인덱스는 지원하지 않습니다.
- 보충 로깅을 관리하고 임의의 열에서 변환을 수행하는 경우, 모든 필드 및 열에 대해 보충 로깅이 활성화되어 있어야 합니다. 보충 로깅 설정에 대한 자세한 내용은 다음 주제를 참조하세요.
  - 자체 관리형 Oracle 소스 데이터베이스는 보충 로깅 설정 섹션을 참조하세요.
  - AWS관리형 Oracle 소스 데이터베이스의 경우 을 참조하십시오. AWS관리형 Oracle 소스 구성 대상 AWS DMS
- AWS DMS 멀티테넌트 컨테이너 루트 데이터베이스 (CDB$ROOT) 를 지원하지 않습니다. DMS는 Binary Reader를 사용하여 PDB를 지원합니다.
- AWS DMS 지연된 제약 조건을 지원하지 않습니다.
- 보안 LOB는 LOB 조회를 수행하여 전체 LOB 모드를 사용해야만 지원됩니다.
- AWS DMS 지원되는 모든 Oracle 버전 11 이상의 `rename table` *`table-name`* `to` *`new-table-name`* 구문을 지원합니다. 이 구문은 Oracle 버전 10 소스 데이터베이스에서는 지원되지 않습니다.
- AWS DMS DDL 문의 결과를 복제하지 않습니다. `ALTER TABLE ADD` *`column data_type`* `DEFAULT` *`default_value default_value`*를 대상에 복제하는 대신 새 열을 `NULL`로 설정합니다.
- AWS DMS 버전 3.4.7 이상을 사용하는 경우 파티션 또는 하위 파티션 작업으로 인한 변경 내용을 복제하려면 DMS 작업을 시작하기 전에 다음을 수행하십시오.
  - 파티셔닝된 테이블 구조(DDL)를 수동으로 생성합니다.
  - Oracle 소스와 Oracle 대상 모두에서 DDL이 동일해야 합니다.
  - 추가 연결 속성 `enableHomogenousPartitionOps=true`를 설정합니다.

  `enableHomogenousPartitionOps`에 대한 자세한 정보는 Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS 섹션을 참조하십시오. 또한 FULL+CDC 작업에서 DMS는 캐시된 변경 사항의 일부로 캡처된 데이터 변경 사항을 복제하지 않는다는 점에 유의하세요. 이 사용 사례에서는 Oracle 대상에서 테이블 구조를 다시 만들고 해당 테이블을 다시 로드하세요.

  버전 3.4.7 이전: AWS DMS

DMS는 파티션 또는 하위 파티션 작업(ADD, DROP, EXCHANGE, TRUNCATE)으로 인해 발생하는 데이터 변경 사항을 복제하지 않습니다. 이러한 업데이트로 인해 복제 중에 다음과 같은 오류가 발생할 수 있습니다.

- ADD 작업의 경우 추가된 데이터에 대한 업데이트 및 삭제로 인해 "적용된 행 없음" 경고가 발생할 수 있습니다.
- DROP 및 TRUNCATE 작업의 경우 새 삽입은 "중복" 오류를 발생시킬 수 있습니다.
- EXCHANGE 작업을 수행하면 "적용된 행 없음" 경고와 "중복" 오류가 발생할 수 있습니다.

파티션 또는 하위 파티션 작업으로 인해 발생하는 변경 사항을 복제하려면 해당 테이블을 다시 로드합니다. 빈 파티션을 새로 추가한 후 새로 추가된 파티션에 대한 작업이 정상적으로 대상에 복제됩니다.

- AWS DMS 3.4 이전 버전은 소스에서 CREATE TABLE AS 명령문을 실행하여 발생한 타겟의 데이터 변경을 지원하지 않습니다. 하지만, 새 테이블이 대상에서 생성됩니다.
- AWS DMS Oracle DBMS_REDEFINITION 패키지에서 변경한 내용 (예: 테이블 메타데이터 및 OBJECT_ID 필드) 은 캡처하지 않습니다.
- AWS DMS 빈 BLOB 및 CLOB 열을 NULL 대상에 매핑합니다.
- Oracle LogMiner 11에서 변경 사항을 캡처하는 경우 문자열 길이가 1982보다 큰 CLOB 열에 대한 업데이트가 손실되고 대상이 업데이트되지 않습니다.
- 변경 데이터 캡처 (CDC) 중에는 기본 키로 정의된 숫자 열에 대한 일괄 업데이트가 지원되지 AWS DMS 않습니다.
- AWS DMS 특정 UPDATE 명령을 지원하지 않습니다. 다음은 지원되지 않는 UPDATE 명령의 예입니다.

```
UPDATE TEST_TABLE SET KEY=KEY+1;
```

여기서, TEST_TABLE은 테이블 이름이고 KEY는 프라이머리 키로 정의 된 숫자 열입이다.

- AWS DMS LONG RAW 열 및 LONG RAW 열을 로드하기 위한 전체 LOB 모드를 지원하지 않습니다. 대신 제한된 LOB 모드를 사용하여 이러한 데이터 형식을 Oracle 대상으로 마이그레이션할 수 있습니다. 제한된 LOB 모드에서는 64KB보다 긴 긴 RAW 열 또는 긴 RAW 열로 설정한 모든 데이터를 64KB로 AWS DMS 잘라냅니다.
- AWS DMS XMLTYPE 열을 로드하기 위한 전체 LOB 모드를 지원하지 않습니다. 대신 제한된 LOB 모드를 사용하여 XMLTYPE 열을 Oracle 대상으로 마이그레이션할 수 있습니다. 제한된 LOB 모드에서 DMS는 사용자가 정의한 '최대 LOB 크기' 변수보다 큰 데이터를 모두 잘라냅니다. '최대 LOB 크기'의 최대 권장 값은 100MB입니다.

- AWS DMS 이름에 아포스트로피가 포함된 테이블을 복제하지 않습니다.
- AWS DMS 구체화된 뷰에서 CDC를 지원합니다. 하지만 DMS는 그 밖의 뷰에서는 CDC를 지원하지 않습니다.
- AWS DMS 오버플로우 세그먼트가 있는 인덱스로 구성된 테이블에 대해서는 CDC를 지원하지 않습니다.
- AWS DMS ~로 설정된 참조로 파티셔닝된 테이블에 대한 `Drop Partition` 작업은 지원하지 않습니다. `enableHomogenousPartitionOps true`
- LogMiner Oracle을 사용하여 리두 로그에 액세스할 때는 다음과 같은 AWS DMS 제한이 있습니다.
  - Oracle 12의 경우에만 LOB 열에 대한 변경 사항을 복제하지 AWS DMS 않습니다.
  - 모든 Oracle 버전에서는 `XMLTYPE` 및 LOB 열에 대한 `UPDATE` 작업 결과를 복제하지 AWS DMS 않습니다.
  - AWS DMS Oracle을 사용하는 동안에는 복제에서 XA 트랜잭션을 지원하지 않습니다. LogMiner
  - LogMiner Oracle은 플러그형 데이터베이스 (PDB) 에 대한 연결을 지원하지 않습니다. PDB에 연결하려면 Binary Reader를 사용하여 다시 실행 로그에 액세스합니다.
  - SHRINK SPACE 작업은 지원되지 않습니다.
- 바이너리 리더를 사용하는 경우 다음과 같은 제한이 AWS DMS 있습니다.
  - 테이블 클러스터를 지원하지 않습니다.
  - 테이블 수준 `SHRINK SPACE` 작업만 지원합니다. 이 수준에는 전체 테이블, 파티션, 하위 파티션이 포함됩니다.
  - 키 압축을 사용하는 인덱스 구성 테이블에 대한 변경 사항은 지원하지 않습니다.
  - 원시 디바이스에서 온라인 다시 실행 로그 구현은 지원하지 않습니다.
  - RDS for Oracle이 TDE 암호화 키를 위한 지갑 암호 검색을 지원하지 않으므로 Binary Reader는 자체 관리형 Oracle 데이터베이스에서만 TDE를 지원합니다.
- AWS DMS Oracle 자동 스토리지 관리 (ASM) 프록시를 사용하는 Amazon RDS Oracle 소스에 연결할 수 없습니다.
- AWS DMS 가상 컬럼은 지원하지 않습니다.
- AWS DMS ROWID 열을 기반으로 하는 `ROWID` 데이터 유형이나 구체화된 뷰는 지원하지 않습니다.

  AWS DMS Oracle 구체화된 뷰를 부분적으로 지원합니다. 전체 로드의 경우, DMS는 Oracle 구체화된 뷰의 전체 로드 복사를 수행할 수 있습니다. DMS는 구체화된 뷰를 대상 시스템에 기본 테이블로 복사하고 구체화된 뷰의 모든 ROWID 열을 무시합니다. 지속적 복제(CDC) 의 경우 DMS는 구체화된 뷰 데이터에 대한 변경 사항을 복제하려고 시도하지만 결과는 이상적이지 않을 수 있습니다. 특히, 구체화된 뷰가 완전히 새로 고쳐지면 DMS는 모든 행에 개별 삭제를 복제한 후 모든 행에 개별

삽입을 복제합니다. 이는 리소스를 매우 많이 사용하는 작업이며, 행 수가 많은 구체화된 뷰에서는 성능이 떨어질 수 있습니다. 구체화된 뷰가 빠른 새로 고침을 수행하는 지속적인 복제의 경우, DMS는 빠른 새로 고침 데이터 변경을 처리하고 복제하려고 합니다. 어느 경우든 DMS는 구체화된 뷰의 ROWID 열을 모두 건너뜁니다.

- AWS DMS 글로벌 임시 테이블을 로드하거나 캡처하지 않습니다.
- 복제를 사용하는 S3 대상의 경우, 소스 행 업데이트가 모든 열 값을 캡처할 수 있도록 모든 열에서 보충 로깅을 활성화하세요. 예제는 다음과 같습니다. `alter table yourtablename add supplemental log data (all) columns;`.
- `null`을 포함하는 복합 고유 키가 있는 행에 대한 업데이트는 대상에서 복제될 수 없습니다.
- AWS DMS 동일한 소스 엔드포인트에서 여러 Oracle TDE 암호화 키를 사용할 수 없습니다. 각 엔드포인트는 TDE 암호화 키 이름 "`securityDbEncryptionName`"에 대한 속성 하나와 이 키에 대한 하나의 TDE 암호만 가질 수 있습니다.
- Amazon RDS for Oracle에서 복제하는 경우 TDE는 암호화된 테이블스페이스와 Oracle을 사용하는 경우에만 지원됩니다. LogMiner
- AWS DMS 여러 테이블 이름 바꾸기 작업을 빠르게 연속해서 수행할 수 없습니다.
- Oracle 19.0을 소스로 사용하는 경우 다음 기능을 AWS DMS 지원하지 않습니다.
  - 데이터가드 DML 리디렉션
  - 파티셔닝된 하이브리드 테이블
  - 스키마 전용 Oracle 계정
- AWS DMS 또는 유형의 `BIN$` 테이블 또는 뷰 마이그레이션을 지원하지 않습니다. `DR$`
- Oracle 18.x부터는 오라클 익스프레스 에디션 (오라클 데이터베이스 XE) 의 변경 데이터 캡처 (CDC) 를 AWS DMS 지원하지 않습니다.
- CHAR 열에서 데이터를 마이그레이션할 때 DMS는 모든 후행 공백을 잘라냅니다.
- AWS DMS 애플리케이션 컨테이너에서의 복제를 지원하지 않습니다.
- AWS DMS Oracle 플래시백 데이터베이스 및 복원 지점 수행은 지원되지 않습니다. 이러한 작업은 Oracle Redo 로그 파일의 일관성에 영향을 미치기 때문입니다.
- 다음과 같은 경우에는 병렬 실행 옵션을 사용하는 직접 로드 `INSERT` 프로시저가 지원되지 않습니다.
  - 열이 255개를 넘는 압축되지 않은 테이블
  - 행 크기가 8K를 초과
  - Exadata HCC 테이블
  - 빅 엔디안 플랫폼에서 실행되는 데이터베이스

- 프라이머리 키도 고유 키도 없는 소스 테이블에는 ALL COLUMN 보충 로깅을 활성화해야 합니다. 이로 인해 더 많은 다시 실행 로그 활동이 생성되고 DMS CDC 대기 시간이 늘어날 수 있습니다.
- AWS DMS 소스 데이터베이스의 보이지 않는 열에서 데이터를 마이그레이션하지 않습니다. 이러한 열을 마이그레이션 범위에 포함하려면 `ALTER TABLE` 문을 사용하여 이러한 열을 표시하세요.

Oracle 엔드포인트용 SSL 지원

AWS DMS 오라클 엔드포인트는 none 및 `verify-ca` SSL 모드에 대해 SSL V3을 지원합니다. Oracle 엔드포인트에서 SSL을 사용하려면, .pem 인증서 파일이 아닌 엔드포인트용 Oracle Wallet을 업로드합니다.

주제

- Oracle SSL용 기존 인증서 사용
- Oracle SSL용 자체 서명 인증서 사용

Oracle SSL용 기존 인증서 사용

기존 Oracle 클라이언트 설치를 사용하여 CA 인증서 파일에서 Oracle wallet 파일을 생성하려면, 다음 단계를 진행합니다.

AWS DMS에서 Oracle SSL에 대해 기존 Oracle 클라이언트 설치를 사용하려면

1. 다음 명령을 실행하여 `ORACLE_HOME` 시스템 변수를 dbhome_1 디렉터리 위치로 설정합니다.

```
prompt>export ORACLE_HOME=/home/user/app/user/product/12.1.0/dbhome_1
```

2. `LD_LIBRARY_PATH` 시스템 변수에 `$ORACLE_HOME/lib`을 추가합니다.

```
prompt>export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME/lib
```

3. `$ORACLE_HOME/ssl_wallet`에 Oracle Wallet을 위한 디렉터리를 만듭니다.

```
prompt>mkdir $ORACLE_HOME/ssl_wallet
```

4. CA 인증서 `.pem` 파일을 `ssl_wallet` 디렉터리에 배치합니다. Amazon RDS를 사용하는 경우, Amazon RDS에서 호스팅되는 `rds-ca-2015-root.pem` 루트 CA 인증서 파일을 다운로드할 수 있습니다. 이 파일 다운로드에 자세한 내용은 Amazon RDS 사용 설명서에서 SSL/TLS를 사용하여 DB 인스턴스에 대한 연결 암호화를 참조하세요.
5. 다음 명령을 실행하여 Oracle Wallet을 만듭니다.

```
prompt>orapki wallet create -wallet $ORACLE_HOME/ssl_wallet -auto_login_only
prompt>orapki wallet add -wallet $ORACLE_HOME/ssl_wallet -trusted_cert -cert
   $ORACLE_HOME/ssl_wallet/ca-cert.pem -auto_login_only
```

이전 단계를 완료했으면, certificate-wallet 매개변수를 지정하여 `ImportCertificate` API 호출을 사용하여 wallet 파일을 가져올 수 있습니다. 그런 다음 Oracle 엔드포인트를 생성하거나 수정할 때 SSL 모드로서 `verify-ca`를 선택하면 가져온 wallet 인증서를 사용할 수 있습니다.

> **Note**
>
> 오라클 지갑은 바이너리 파일입니다. AWS DMS는 이러한 파일을 있는 그대로 받아들입니다.

Oracle SSL용 자체 서명 인증서 사용

Oracle SSL용 자체 서명 인증서를 사용하려면 Oracle wallet 암호가 `oracle123`이라고 가정하고 다음 단계를 수행하세요.

다음과 같이 Oracle SSL용 자체 서명된 인증서를 사용하려면 AWS DMS

1. 자체 서명 인증서를 사용할 디렉터리를 생성합니다.

```
mkdir -p /u01/app/oracle/self_signed_cert
```

2. 이전 단계에서 생성한 디렉터리로 변경합니다.

```
cd /u01/app/oracle/self_signed_cert
```

3. 루트 키를 생성합니다.

```
openssl genrsa -out self-rootCA.key 2048
```

4. 이전 단계에서 생성한 루트 키를 루트 인증서를 자체 서명합니다.

```
openssl req -x509 -new -nodes -key self-rootCA.key
        -sha256 -days 3650 -out self-rootCA.pem
```

다음과 같은 입력 파라미터를 사용하세요.

- `Country Name (2 letter code) [XX]`, 예: `AU`
- `State or Province Name (full name) []`, 예: `NSW`
- `Locality Name (e.g., city) [Default City]`, 예: `Sydney`
- `Organization Name (e.g., company) [Default Company Ltd]`, 예: `AmazonWebService`
- `Organizational Unit Name (e.g., section) []`, 예: `DBeng`
- `Common Name (e.g., your name or your server's hostname) []`, 예: `aws`
- `Email Address []`, 예: abcd.efgh@amazonwebservice.com

5. Oracle 데이터베이스용 Oracle wallet 디렉터리를 만듭니다.

```
mkdir -p /u01/app/oracle/wallet
```

6. 새 Oracle wallet을 만듭니다.

```
orapki wallet create -wallet "/u01/app/oracle/wallet" -pwd oracle123 -
auto_login_local
```

7. 루트 인증서를 Oracle wallet에 추가합니다.

```
orapki wallet add -wallet "/u01/app/oracle/wallet" -pwd oracle123 -trusted_cert
-cert /u01/app/oracle/self_signed_cert/self-rootCA.pem
```

8. Oracle wallet의 콘텐츠를 나열합니다. 이 목록에는 루트 인증서가 포함되어야 합니다.

```
orapki wallet display -wallet /u01/app/oracle/wallet -pwd oracle123
```

예를 들어 다음과 비슷하게 표시될 수 있습니다.

```
Requested Certificates:
User Certificates:
Trusted Certificates:
```

```
Subject:        CN=aws,OU=DBeng,O= AmazonWebService,L=Sydney,ST=NSW,C=AU
```

9. ORAPKI 유틸리티를 사용한 인증서 서명 요청(CSR)을 생성합니다.

```
orapki wallet add -wallet "/u01/app/oracle/wallet" -pwd oracle123
-dn "CN=aws" -keysize 2048 -sign_alg sha256
```

10. 다음 명령을 실행합니다.

```
openssl pkcs12 -in /u01/app/oracle/wallet/ewallet.p12 -nodes -out /u01/app/oracle/
wallet/nonoracle_wallet.pem
```

이 명령의 출력은 다음과 같습니다.

```
Enter Import Password:
MAC verified OK
Warning unsupported bag type: secretBag
```

11. 'dms'를 일반 이름으로 둡니다.

```
openssl req -new -key /u01/app/oracle/wallet/nonoracle_wallet.pem -out certdms.csr
```

다음과 같은 입력 파라미터를 사용하세요.

- `Country Name (2 letter code) [XX]`, 예: AU
- `State or Province Name (full name) []`, 예: NSW
- `Locality Name (e.g., city) [Default City]`, 예: Sydney
- `Organization Name (e.g., company) [Default Company Ltd]`, 예: AmazonWebService
- `Organizational Unit Name (e.g., section) []`, 예: aws
- `Common Name (e.g., your name or your server's hostname) []`, 예: aws
- `Email Address []`, 예: abcd.efgh@amazonwebservice.com

이 단계는 4단계와 달라야 합니다. 예를 들어 다음과 같이 조직 단위 이름을 다른 이름으로 변경하면 됩니다.

인증서 요청과 함께 전송될 다음 추가 속성을 입력합니다.

- A challenge password [], 예: oracle123
- An optional company name [], 예: aws

12. 인증서 서명을 가져옵니다.

```
openssl req -noout -text -in certdms.csr | grep -i signature
```

이 게시물의 서명 키는 sha256WithRSAEncryption입니다.

13. 다음 명령을 실행하여 인증서(.crt) 파일을 생성합니다.

```
openssl x509 -req -in certdms.csr -CA self-rootCA.pem -CAkey self-rootCA.key
-CAcreateserial -out certdms.crt -days 365 -sha256
```

그러면 다음과 같은 출력이 표시됩니다.

```
Signature ok
subject=/C=AU/ST=NSW/L=Sydney/O=awsweb/OU=DBeng/CN=aws
Getting CA Private Key
```

14. 인증서를 wallet에 추가합니다.

```
orapki wallet add -wallet /u01/app/oracle/wallet -pwd oracle123 -user_cert -cert
 certdms.crt
```

15. wallet을 봅니다. 두 항목이 있어야 합니다. 다음 코드를 확인합니다.

```
orapki wallet display -wallet /u01/app/oracle/wallet -pwd oracle123
```

16. sqlnet.ora 파일($ORACLE_HOME/network/admin/sqlnet.ora)을 구성합니다.

```
WALLET_LOCATION =
   (SOURCE =
     (METHOD = FILE)
     (METHOD_DATA =
       (DIRECTORY = /u01/app/oracle/wallet/)
     )
   )

SQLNET.AUTHENTICATION_SERVICES = (NONE)
SSL_VERSION = 1.0
```

```
SSL_CLIENT_AUTHENTICATION = FALSE
SSL_CIPHER_SUITES = (SSL_RSA_WITH_AES_256_CBC_SHA)
```

17. Oracle 리스너를 중지합니다.

```
lsnrctl stop
```

18. `listener.ora` 파일(`$ORACLE_HOME/network/admin/listener.ora`)에 SSL에 대한 항목을 추가합니다.

```
SSL_CLIENT_AUTHENTICATION = FALSE
WALLET_LOCATION =
  (SOURCE =
    (METHOD = FILE)
    (METHOD_DATA =
      (DIRECTORY = /u01/app/oracle/wallet/)
    )
  )

SID_LIST_LISTENER =
 (SID_LIST =
  (SID_DESC =
   (GLOBAL_DBNAME = SID)
   (ORACLE_HOME = ORACLE_HOME)
   (SID_NAME = SID)
  )
 )

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = localhost.localdomain)(PORT = 1521))
      (ADDRESS = (PROTOCOL = TCPS)(HOST = localhost.localdomain)(PORT = 1522))
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC1521))
    )
  )
```

19. `tnsnames.ora` 파일(`$ORACLE_HOME/network/admin/tnsnames.ora`)을 구성합니다.

```
<SID>=
(DESCRIPTION=
        (ADDRESS_LIST =
```

```
                (ADDRESS=(PROTOCOL = TCP)(HOST = localhost.localdomain)(PORT =
 1521))
        )
        (CONNECT_DATA =
                (SERVER = DEDICATED)
                (SERVICE_NAME = <SID>)
        )
)

<SID>_ssl=
(DESCRIPTION=
        (ADDRESS_LIST =
                (ADDRESS=(PROTOCOL = TCPS)(HOST = localhost.localdomain)(PORT =
 1522))
        )
        (CONNECT_DATA =
                (SERVER = DEDICATED)
                (SERVICE_NAME = <SID>)
        )
)
```

20. Oracle 리스너를 다시 시작합니다.

```
lsnrctl start
```

21. Oracle 리스너 상태를 표시합니다.

```
lsnrctl status
```

22. sqlplus 및 SSL tnsnames 항목을 사용하여 로컬 호스트에서 데이터베이스와의 SSL 연결을 테스트합니다.

```
sqlplus -L ORACLE_USER@SID_ssl
```

23. SSL을 사용하여 잘 연결되었는지 확인합니다.

```
SELECT SYS_CONTEXT('USERENV', 'network_protocol') FROM DUAL;

SYS_CONTEXT('USERENV','NETWORK_PROTOCOL')
--------------------------------------------------------------------------------
tcps
```

24. 자체 서명 인증서를 사용하여 디렉터리를 이 디렉터리로 변경합니다.

```
cd /u01/app/oracle/self_signed_cert
```

25. 사용할 새 클라이언트 Oracle AWS DMS 지갑을 생성하십시오.

```
orapki wallet create -wallet ./ -auto_login_only
```

26. Oracle wallet에 자체 서명 루트 인증서를 추가합니다.

```
orapki wallet add -wallet ./ -trusted_cert -cert self-rootCA.pem -auto_login_only
```

27. 사용할 Oracle 지갑의 내용을 나열하십시오. AWS DMS 이 목록에는 자체 서명 루트 인증서가 포함되어야 합니다.

```
orapki wallet display -wallet ./
```

이 명령의 출력은 다음과 같습니다.

```
Trusted Certificates:
Subject:        CN=aws,OU=DBeng,O=AmazonWebService,L=Sydney,ST=NSW,C=AU
```

28. 방금 생성한 Oracle 지갑을 AWS DMS업로드하십시오.

Oracle을 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS

다음 표에는 Oracle 원본 데이터베이스를 사용할 때 AWS DMS 지원되는 투명한 데이터 암호화 (TDE) 방법이 나와 있습니다.

다시 실행 로그 액세스 방법	TDE 테이블스페이스	TDE 열
오라클 LogMiner	예	예
Binary Reader	예	예

AWS DMS 열 레벨과 테이블스페이스 레벨 모두에서 바이너리 리더를 사용할 때 Oracle TDE를 지원합니다. TDE 암호화를 함께 AWS DMS사용하려면 먼저 TDE 암호화 키와 TDE 비밀번호가 저장되어 있는 Oracle 지갑 위치를 식별하십시오. 그런 다음 Oracle 소스 엔드포인트의 올바른 TDE 암호화 키와 암호를 식별합니다.

TDE 암호화를 위한 암호화 키와 암호를 식별하고 지정하려면

1. 다음 쿼리를 실행하여 Oracle 데이터베이스 호스트에서 Oracle 암호화 wallet을 찾습니다.

```
SQL> SELECT WRL_PARAMETER FROM V$ENCRYPTION_WALLET;

WRL_PARAMETER
--------------------------------------------------------------------------------
/u01/oracle/product/12.2.0/dbhome_1/data/wallet/
```

   여기서는 `/u01/oracle/product/12.2.0/dbhome_1/data/wallet/`이 wallet 위치입니다.

2. 이 값을 반환하는 옵션에 따라 다음 암호화 옵션 중 하나를 사용하여 마스터 키 ID를 가져옵니다.

   a. 테이블 또는 열 수준 암호화의 경우 다음 쿼리를 실행합니다.

```
SQL> SELECT OBJECT_ID FROM ALL_OBJECTS
WHERE OWNER='DMS_USER' AND OBJECT_NAME='TEST_TDE_COLUMN' AND
 OBJECT_TYPE='TABLE';

OBJECT_ID
---------------
81046
SQL> SELECT MKEYID FROM SYS.ENC$ WHERE OBJ#=81046;

MKEYID
------------
AWGDC9glSk8Xv+3bVveiVSgAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

   여기서는 `AWGDC9glSk8Xv+3bVveiVSg`가 마스터 키 ID(`MKEYID`)입니다. `MKEYID`의 값을 얻으면 3단계로 진행할 수 있습니다. 그렇지 않은 경우 2.2단계로 진행합니다.

> **Note**
>
> 후행 문자열 `'A'` 문자(`AAA...`)는 값의 일부가 아닙니다.

   b. 테이블스페이스 수준 암호화의 경우, 다음 쿼리를 실행합니다.

```
SQL> SELECT TABLESPACE_NAME, ENCRYPTED FROM dba_tablespaces;
TABLESPACE_NAME                ENC
------------------------------ ---
SYSTEM                         NO
```

```
SYSAUX                         NO
UNDOTBS1                       NO
TEMP                           NO
USERS                          NO
TEST_ENCRYT                    YES
SQL> SELECT name,utl_raw.cast_to_varchar2( utl_encode.base64_encode('01'||
substr(mkeyid,1,4))) ||
 utl_raw.cast_to_varchar2( utl_encode.base64_encode(substr(mkeyid,5,length(mkeyid))))
 masterkeyid_base64
FROM (SELECT t.name, RAWTOHEX(x.mkid) mkeyid FROM v$tablespace t, x$kcbtek x
 WHERE t.ts#=x.ts#)
WHERE name = 'TEST_ENCRYT';

NAME                           MASTERKEYID_BASE64
------------------------------ --------------------------------
TEST_ENCRYT                    AWGDC9glSk8Xv+3bVveiVSg=
```

여기서는 AWGDC9glSk8Xv+3bVveiVSg가 마스터 키 ID(TEST_ENCRYT)입니다. 2.1단계와 2.2단계 모두 값을 반환하는 경우, 항상 동일합니다.

후행 '=' 문자는 값의 일부가 아닙니다.

3. 명령줄을 사용하여 소스 Oracle 데이터베이스 호스트에서 암호화 wallet 항목을 나열합니다.

```
$ mkstore -wrl /u01/oracle/product/12.2.0/dbhome_1/data/wallet/ -list
Oracle Secret Store entries:
ORACLE.SECURITY.DB.ENCRYPTION.AWGDC9glSk8Xv+3bVveiVSgAAAAAAAAAAAAAAAAAAAAAAAAAAAA
ORACLE.SECURITY.DB.ENCRYPTION.AY1mRA8OXU9Qvzo3idU4OH4AAAAAAAAAAAAAAAAAAAAAAAAAAAA
ORACLE.SECURITY.DB.ENCRYPTION.MASTERKEY
ORACLE.SECURITY.ID.ENCRYPTION.
ORACLE.SECURITY.KB.ENCRYPTION.
ORACLE.SECURITY.KM.ENCRYPTION.AY1mRA8OXU9Qvzo3idU4OH4AAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

2단계에서 찾은 마스터 키 ID(AWGDC9glSk8Xv+3bVveiVSg)가 포함된 항목을 찾습니다. 이 항목이 TDE 암호화 키 이름입니다.

4. 이전 단계에서 찾은 항목의 세부 정보를 확인합니다.

```
$ mkstore -wrl /u01/oracle/product/12.2.0/dbhome_1/data/wallet/ -viewEntry
 ORACLE.SECURITY.DB.ENCRYPTION.AWGDC9glSk8Xv+3bVveiVSgAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Oracle Secret Store Tool : Version 12.2.0.1.0
Copyright (c) 2004, 2016, Oracle and/or its affiliates. All rights reserved.
Enter wallet password:
```

```
ORACLE.SECURITY.DB.ENCRYPTION.AWGDC9glSk8Xv+3bVveiVSgAAAAAAAAAAAAAAAAAAAAAAAAAAA
 = AEMAASAASGYs0phWHfNt9J5mEMkkegGFiD4LLfQszDojgDzbfoYDEACv0x3pJC+UGD/
PdtE2jLIcBQcAeHgJChQGLA==
```

결과를 보려면 wallet 암호를 입력합니다.

여기서는 '='의 오른쪽에 있는 값이 TDE 암호입니다.

5. securityDbEncryptionName 추가 연결 속성을 설정하여 Oracle 소스 엔드포인트의 TDE 암호화 키 이름을 지정합니다.

```
securityDbEncryptionName=ORACLE.SECURITY.DB.ENCRYPTION.AWGDC9glSk8Xv
+3bVveiVSgAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

6. 콘솔에서 이 키에 연결된 TDE 암호를 Oracle 소스 암호 값의 일부로 제공합니다. TDE 암호 값으로 끝나는 쉼표로 구분된 암호 값의 형식을 지정하려면 다음 순서를 사용합니다.

```
Oracle_db_password,ASM_Password,AEMAASAASGYs0phWHfNt9J5mEMkkegGFiD4LLfQszDojgDzbfoYDEACv0x3
+UGD/PdtE2jLIcBQcAeHgJChQGLA==
```

Oracle 데이터베이스 구성에 관계없이 이 순서로 암호 값을 지정합니다. 예를 들어 TDE를 사용하고 있지만 Oracle 데이터베이스가 ASM을 사용하지 않는 경우, 암호 값을 다음과 같은 쉼표로 구분된 순서로 지정합니다.

```
Oracle_db_password,,AEMAASAASGYs0phWHfNt9J5mEMkkegGFiD4LLfQszDojgDzbfoYDEACv0x3pJC
+UGD/PdtE2jLIcBQcAeHgJChQGLA==
```

지정한 TDE 자격 증명이 올바르지 않아도 AWS DMS 마이그레이션 작업은 실패하지 않습니다. 하지만 이 작업은 진행 중인 대상 데이터베이스에 대한 복제 변경 사항을 읽거나 적용하지도 않습니다. 작업을 시작한 후에 콘솔 마이그레이션 작업 페이지에서 테이블 통계를 모니터링하여 변경 사항이 복제되었는지 확인하세요.

작업이 실행되는 동안 DBA가 Oracle 데이터베이스의 TDE 보안 인증 정보 값을 변경하면 작업이 실패합니다. 오류 메시지에는 새 TDE 암호화 키 이름이 포함됩니다. 새 값을 지정하고 작업을 다시 시작하려면 이전 절차를 사용하세요.

Important

`cp`, `mv`, `orapki`, `mkstore` 같은 OS 수준 명령으로 인해 ASM 위치에 저장된 wallet 파일이 손상되므로 Oracle ASM(Automatic Storage Management) 위치에 생성된 TDE wallet을 조작할 수 없습니다. 이 제한은 ASM 위치에 저장된 TDE wallet 파일에만 적용되며, 로컬 OS 디렉터리에 저장된 TDE wallet 파일에는 적용되지 않습니다.
OS 수준 명령으로 ASM에 저장된 TDE wallet을 조작하려면 다음과 같이 로컬 키스토어를 생성하고 ASM 키스토어를 로컬 키스토어에 병합하세요.

1. 로컬 키스토어를 생성합니다.

```
ADMINISTER KEY MANAGEMENT create keystore file system wallet location
 identified by wallet password;
```

2. ASM 키스토어를 로컬 키스토어에 병합합니다.

```
ADMINISTER KEY MANAGEMENT merge keystore ASM wallet location identified
 by wallet password into existing keystore file system wallet location
 identified by wallet password with backup;
```

그런 다음 암호화 wallet 항목과 TDE 암호를 나열하려면 로컬 키스토어에 대해 3단계와 4단계를 실행합니다.

Oracle을 원본으로 사용하기 위해 지원되는 압축 방법 AWS DMS

다음 표에는 Oracle 원본 데이터베이스에서 작업할 때 AWS DMS 지원되는 압축 방법이 나와 있습니다. 표에서 볼 수 있듯이 압축 지원은 Oracle 데이터베이스 버전과 DMS가 Oracle을 사용하여 리두 로그에 LogMiner 액세스하도록 구성되었는지 여부에 따라 달라집니다.

버전	기본	OLTP	HCC(Oracle 11g R2 이상부터)	기타
Oracle 10	아니요	N/A	N/A	아니요

버전	기본	OLTP	HCC(Oracle 11g R2 이상부터)	기타
오라클 11 이상 — 오라클 LogMiner	예	예	예	예 — LogMiner 오라클에서 지원하는 모든 압축 방법.
Oracle 11 이상 – Binary Reader	예	예	예 - 자세한 내용은 다음 참고 사항을 참조하세요.	예

> **Note**
>
> Oracle 소스 엔드포인트가 Binary Reader를 사용하도록 구성된 경우 HCC 압축 방법의 Query Low 수준은 전체 로드 작업에 대해서만 지원됩니다.

Oracle을 원본으로 사용하여 중첩된 테이블을 복제하기 AWS DMS

AWS DMS 중첩된 테이블 또는 정의된 유형인 열을 포함하는 Oracle 테이블의 복제를 지원합니다. 이 기능을 활성화하려면 Oracle 소스 엔드포인트에 다음 추가 연결 속성 설정을 추가합니다.

```
allowSelectNestedTables=true;
```

AWS DMS Oracle 중첩 테이블에서 대상 테이블을 고유한 제약 조건 없이 대상의 일반 상위 및 하위 테이블로 생성합니다. 대상에서 올바른 데이터에 액세스하려면 상위 및 하위 테이블을 조인합니다. 이렇게 하려면 먼저 대상 하위 테이블의 `NESTED_TABLE_ID` 열에 고유하지 않은 인덱스를 수동으로 만듭니다. 그런 다음 조인 `ON` 절의 `NESTED_TABLE_ID` 열을 자식 테이블 이름에 해당하는 상위 열과 함께 사용할 수 있습니다. 또한 이러한 인덱스를 생성하면 에서 대상 하위 테이블 데이터를 업데이트하거나 삭제할 때 성능이 향상됩니다. AWS DMS예시는 대상의 상위 및 하위 테이블에 대한 조인 예을 확인하세요.

전체 로드가 완료되면 작업을 중지하도록 구성하는 것이 좋습니다. 그런 다음 대상의 복제된 모든 하위 테이블에 대해 고유하지 않은 인덱스를 만들고 작업을 재개합니다.

캡처된 중첩 테이블이 기존 상위 테이블에 추가되는 경우 (캡처되거나 캡처되지 않음) 는 해당 테이블을 올바르게 AWS DMS 처리합니다. 그러나 해당 대상 테이블에 대한 고유하지 않은 인덱스는 생성되

지 않습니다. 이 경우 대상 하위 테이블이 매우 커지면 성능이 영향을 받을 수 있습니다. 이런 경우에는 작업을 중지하고 인덱스를 만든 다음 작업을 재개하는 것이 좋습니다.

중첩 테이블이 대상에 복제된 후 DBA가 상위 및 해당 하위 테이블에 대한 조인을 실행하여 데이터를 병합하도록 합니다.

Oracle 중첩 테이블을 소스로 복제하기 위한 사전 요구 사항

복제된 모든 중첩 테이블의 상위 테이블을 복제해야 합니다. 상위 테이블 (중첩된 테이블 열을 포함하는 테이블) 과 하위 (즉, 중첩된) 테이블을 모두 테이블 매핑에 포함하십시오 AWS DMS .

소스로 지원되는 Oracle 중첩 테이블 유형

AWS DMS 다음과 같은 Oracle 중첩 테이블 유형을 원본으로 지원합니다.

- 데이터 유형
- 사용자 정의 객체

AWS DMS 가 Oracle 중첩 테이블을 소스로 지원하는 경우 제한 사항

AWS DMS Oracle 중첩 테이블을 원본으로 지원하는 데에는 다음과 같은 제한이 있습니다.

- AWS DMS 한 수준의 테이블 중첩만 지원합니다.
- AWS DMS 테이블 매핑은 부모 테이블과 하위 테이블 또는 테이블이 모두 복제 대상으로 선택되었는지 확인하지 않습니다. 즉, 하위 테이블이 없는 상위 테이블 또는 상위 테이블이 없는 하위 테이블을 선택할 수 있습니다.

AWS DMS 가 Oracle 중첩 테이블을 소스로 복제하는 방법

AWS DMS 다음과 같이 상위 테이블과 중첩 테이블을 대상에 복제합니다.

- AWS DMS 원본과 동일한 부모 테이블을 생성합니다. 그런 다음 상위 테이블의 중첩 열을 `RAW(16)`로 정의하고 상위 테이블의 중첩 테이블에 대한 참조를 해당 `NESTED_TABLE_ID` 열에 포함합니다.
- AWS DMS 는 중첩된 원본과 동일하지만 이름이 `NESTED_TABLE_ID` 추가된 추가 열을 포함하는 하위 테이블을 만듭니다. 이 열은 해당 상위 중첩 열과 동일한 유형 및 값을 가지며 동일한 의미를 갖습니다.

대상의 상위 및 하위 테이블에 대한 조인 예

상위 테이블을 병합하려면 다음 예제와 같이 상위 테이블과 하위 테이블 간에 조인을 실행합니다.

1. Type 테이블을 만듭니다.

```
CREATE OR REPLACE TYPE NESTED_TEST_T AS TABLE OF VARCHAR(50);
```

2. 앞서 정의한 유형 NESTED_TEST_T의 열을 사용하여 상위 테이블을 만듭니다.

```
CREATE TABLE NESTED_PARENT_TEST (ID NUMBER(10,0) PRIMARY KEY, NAME NESTED_TEST_T)
 NESTED TABLE NAME STORE AS NAME_KEY;
```

3. CHILD.NESTED_TABLE_ID가 PARENT.NAME과 일치하는 NAME_KEY 하위 테이블과의 조인을 사용하여 테이블 NESTED_PARENT_TEST를 병합합니다.

```
SELECT … FROM NESTED_PARENT_TEST PARENT, NAME_KEY CHILD WHERE CHILD.NESTED_
TABLE_ID = PARENT.NAME;
```

Oracle을 소스로 사용하는 경우 Oracle ASM에 REDO를 저장합니다. AWS DMS

REDO가 많이 생성되는 Oracle 소스의 경우, Oracle ASM에 REDO를 저장하면 특히 RAC 구성에서 성능이 향상될 수 있습니다. ASM REDO 읽기를 모든 ASM 노드에 분산하도록 DMS를 구성할 수 있기 때문입니다.

이 구성을 활용하려면 asmServer 연결 속성을 사용하세요. 예를 들어 다음 연결 문자열은 DMS REDO 읽기를 ASM 노드 3개에 분산합니다.

```
asmServer=(DESCRIPTION=(CONNECT_TIMEOUT=8)(ENABLE=BROKEN)(LOAD_BALANCE=ON)(FAILOVER=ON)
(ADDRESS_LIST=
(ADDRESS=(PROTOCOL=tcp)(HOST=asm_node1_ip_address)(PORT=asm_node1_port_number))
(ADDRESS=(PROTOCOL=tcp)(HOST=asm_node2_ip_address)(PORT=asm_node2_port_number))
(ADDRESS=(PROTOCOL=tcp)(HOST=asm_node3_ip_address)(PORT=asm_node3_port_number)))
(CONNECT_DATA=(SERVICE_NAME=+ASM)))
```

NFS를 사용하여 Oracle REDO를 저장하는 경우, 해당 DNFS(Direct NFS) 클라이언트 패치, 특히 Oracle 버그 25224242를 해결하는 모든 패치를 적용해야 합니다. 자세한 내용은 Direct NFS 클라이언트 관련 패치에 관한 다음 Oracle 간행물, Recommended Patches for Direct NFS Client를 참조하세요.

또한 NFS 읽기 성능을 향상시키려면 다음 예와 같이 NFS 볼륨의 fstab에서 rsize 및 wsize의 값을 늘리는 것이 좋습니다.

```
NAS_name_here:/ora_DATA1_archive /u09/oradata/DATA1 nfs
 rw,bg,hard,nointr,tcp,nfsvers=3,_netdev,
timeo=600,rsize=262144,wsize=262144
```

또한 다음과 같이 tcp-max-xfer-size 값을 조정합니다.

```
vserver nfs modify -vserver vserver -tcp-max-xfer-size 262144
```

Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Oracle 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 의 create-endpoint 명령을 --oracle-settings '{"*EndpointSetting*": *"value"*, *...*}' JSON 구문으로 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다. AWS CLI

Oracle을 소스로 하여 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

명칭	설명
AccessAlternateDirectly	Amazon RDS for Oracle을 소스로 하여 변경 데이터 캡처에 Binary Reader를 사용하기 위해서는 이 속성을 false로 설정합니다. 그러면 DMS 인스턴스가 직접 파일 액세스를 사용하여 지정된 경로 접두사 교체를 통해 다시 실행 로그에 액세스하지 않습니다. 자세한 정보는 Oracle용 RDS와 함께 바이너리 리더를 사용하도록 CDC 작업을 구성하는 방법: AWS DMS을 참조하세요. 기본값: true 유효값: true/false 예제: --oracle-settings '{"AccessAlternateDirectly": false}'
AdditionalArchivedLogDestId	프라이머리-스탠바이 설정에서 ArchivedLogDestId 로 이 속성을 설정합니다. 이 설정은 Oracle Data Guard 데이터베이스가 소스로 사용될 때 전환에 유용합니다. 이 경우 AWS DMS 변경 내용을 읽으려면 아카이브 리

명칭	설명
	두 로그를 가져올 대상을 알아야 합니다. 이것은 전환 후에 이전 프라이머리가 이제는 대기 인스턴스이기 때문입니다. Oracle RESETLOGS 옵션을 사용하여 데이터베이스를 열 수 있지만 AWS DMS 필요한 RESETLOGS 경우가 아니면 사용하지 마십시오. RESETLOGS 에 대한 자세한 내용을 알아보려면 Oracle® Database Backup and Recovery User's Guide의 RMAN Data Repair Concepts를 참조하세요. 유효한 값: 아카이브 대상 ID 예제: `--oracle-settings '{"AdditionalArchivedLogDestId": 2}'`
`AddSupplementalLogging`	이 속성을 설정하면 Oracle 데이터베이스의 테이블 수준 보충 로깅을 설정할 수 있습니다. 이 속성은 테이블 메타데이터에 따라 마이그레이션 작업을 위해 선택한 모든 테이블에서 다음 중 하나를 활성화합니다. • PRIMARY KEY COLUMNS 보충 로깅 • UNIQUE KEY COLUMNS 보충 로깅 • ALL COLUMNS 보충 로깅 기본값: false 유효값: true/false 예제: `--oracle-settings '{"AddSupplementalLogging": false}'` **Note** 이 옵션을 사용할 경우, 앞에서 설명한 대로 데이터베이스 수준 보충 로깅을 계속 활성화해야 합니다.

명칭	설명
`AllowSelectNestedTables`	이 속성을 true로 설정하면 중첩 테이블 또는 정의된 유형인 열을 포함하는 Oracle 테이블의 복제를 활성화합니다. 자세한 정보는 Oracle을 원본으로 사용하여 중첩된 테이블을 복제하기 AWS DMS을 참조하세요. 기본값: false 유효값: true/false 예제: `--oracle-settings '{"AllowSelectNestedTables": true}'`
`ArchivedLogDestId`	보관된 다시 실행 로그의 대상 ID를 지정합니다. 이 값은 v$archived_log 보기의 dest_id 열의 숫자와 같아야 합니다. 추가 다시 실행 로그 대상으로 작업하는 경우, `AdditionalArchivedLogDestId` 속성을 사용하여 추가 대상 ID를 지정하는 것이 좋습니다. 그러면 시작 단계부터 올바른 로그에 액세스하도록 보장하여 성능이 개선됩니다. 기본값: 1 유효한 값: 숫자 예제: `--oracle-settings '{"ArchivedLogDestId": 1}'`
`ArchivedLogsOnly`	이 필드를 Y로 설정하면 아카이브된 리두 AWS DMS 로그에만 액세스합니다. 보관된 리두 로그를 Oracle ASM에만 저장하는 경우 AWS DMS 사용자 계정에 ASM 권한을 부여해야 합니다. 기본값: N 유효한 값: Y/N 예제: `--oracle-settings '{"ArchivedLogsOnly": Y}'`

명칭	설명
asmUsePLSQLArray (ECA만 해당)	를 사용하여 소스 변경 내용을 캡처할 때는 이 추가 연결 속성 (ECA)을 사용하십시오. BinaryReader 이 설정을 통해 DMS는 parallelASMReadThread 속성을 사용하여 스레드 수를 제어하면서 단일 읽기 스레드당 ASM 수준에서 50개의 읽기를 버퍼링할 수 있습니다. 이 속성을 설정하면 AWS DMS 바이너리 리더는 익명 PL/SQL 블록을 사용하여 리두 데이터를 캡처하고 이를 대용량 버퍼로 복제 인스턴스에 다시 보냅니다. 이렇게 하면 소스로의 왕복 횟수가 줄어듭니다. 이렇게 하면 소스 캡처 성능이 크게 향상될 수 있지만 ASM 인스턴스의 PGA 메모리 사용량이 증가합니다. 메모리 대상이 충분하지 않을 경우, 안정성 문제가 발생할 수 있습니다. 다음 수식을 사용하여 단일 DMS 작업의 총 ASM 인스턴스 PGA 메모리 사용량을 추정할 수 있습니다. number_of_redo_threads * parallelASMReadThreads * 7 MB 기본값: false 유효값: true/false ECA 예제: asmUsePLSQLArray=true;
ConvertTimestampWithZoneToUTC	'TIMESTAMP WITH TIME ZONE' 및 'TIMESTAMP WITH LOCAL TIME ZONE' 열의 타임스탬프 값을 UTC로 변환하려면 이 속성을 true로 설정합니다. 기본적으로 이 속성의 값은 'false'이며, 데이터는 소스 데이터베이스 시간대를 사용하여 복제됩니다. 기본값: false 유효값: true/false 예제: --oracle-settings '{"ConvertTimestampWithZoneToUTC": true}'

명칭	설명
`EnableHomogenousPartitionOps`	Oracle 동종 마이그레이션을 위한 Oracle 파티션 및 하위 파티션 DDL 작업의 복제를 활성화하려면 이 속성을 `true`로 설정합니다. 참고로 이 기능 및 개선 사항은 버전 3.4.7에 도입되었습니다. AWS DMS 기본값: false 유효값: true/false 예제: `--oracle-settings '{"EnableHomogenousPartitionOps": true}'`
`EnableHomogenousTablespace`	동종 테이블스페이스 복제를 활성화하고 대상의 동일한 테이블스페이스 아래에 기존 테이블이나 인덱스를 생성하려면 이 속성을 설정합니다. 기본값: false 유효값: true/false 예제: `--oracle-settings '{"EnableHomogenousTablespace": true}'`
`EscapeCharacter`	이 속성을 이스케이프 문자로 설정합니다. 이 이스케이프 문자를 사용하면 단일 와일드카드 문자가 테이블 매핑 표현식에서 일반 문자처럼 동작하게 할 수 있습니다. 자세한 정보는 테이블 매핑의 와일드카드을 참조하세요. 기본값: Null 유효한 값: 와일드카드 문자를 제외한 모든 문자 예제: `--oracle-settings '{"EscapeCharacter": "#"}'` **Note** `escapeCharacter` 는 테이블 이름에만 사용할 수 있습니다. 스키마 이름이나 열 이름에서 문자를 이스케이프 처리하지 않습니다.

명칭	설명
ExposeViews	이 속성은 뷰에서 데이터를 한 번 끌어오는 데 사용하세요. 지속적 복제에는 사용할 수 없습니다. 뷰에서 데이터를 추출하면 뷰가 대상 스키마에서 테이블로 표시됩니다. 기본값: false 유효값: true/false 예제: `--oracle-settings '{"ExposeViews": true}'`
ExtraArchivedLogDestIds	하나 이상의 아카이브된 다시 실행 로그에 대한 하나 이상의 대상 ID를 지정합니다. 이 ID는 v$archived_log 뷰에서 dest_id 열의 값입니다. primary-to-single 설치 또는 설치 ArchivedLogDestId 시 추가 연결 속성과 함께 이 설정을 사용하십시오. primary-to-multiple-standby 이 설정은 Oracle Data Guard 데이터베이스를 소스로 사용할 때 전환에 유용합니다. 이 경우 변경 내용을 읽기 위해 아카이브 리두 로그를 가져올 대상에 대한 정보가 AWS DMS 필요합니다. AWS DMS 전환 후에는 이전 기본 인스턴스가 대기 인스턴스가 되기 때문에 이 정보가 필요합니다. 유효한 값: 아카이브 대상 ID 예제: `--oracle-settings '{"ExtraArchivedLogDestIds": 1}'`
FailTasksOnLobTruncation	`true`로 설정할 경우 LOB 열의 실제 크기가 지정된 `LobMaxSize` 보다 큰 경우 작업이 실패합니다. 작업이 제한된 LOB 모드로 설정되어 있고 이 옵션이 `true`로 설정된 경우, LOB 데이터가 잘리지 않는 대신에 작업이 실패합니다. 기본값: false 유효 값: 불 예제: `--oracle-settings '{"FailTasksOnLobTruncation": true}'`

명칭	설명
`filterTransactionsOfUser` (ECA만 해당)	을 사용할 때 Oracle에서 데이터를 복제할 때 DMS가 지정된 사용자의 트랜잭션을 무시하도록 하려면 이 추가 연결 속성 (ECA) 을 사용하십시오. LogMiner 쉼표로 구분된 사용자 이름 값을 전달할 수 있지만 값은 모두 대문자여야 합니다. ECA 예제: `filterTransactionsOfUser=` *`USERNAME`*`;`
`NumberDataTypeScale`	숫자 크기를 지정합니다. 스케일을 최대 38로 선택하거나 FLOAT의 경우 -1, VARCHAR의 경우 -2를 선택할 수 있습니다. 기본적으로 NUMBER 데이터 형식은 정밀도 38, 크기 10으로 변환됩니다. 기본값: 10 유효한 값: -2~38(VARCHAR의 경우 -2, FLOAT의 경우 -1) 예제: `--oracle-settings '{"NumberDataTypeScale": 12}'` **Note** 정밀도-스케일 조합, -1(FLOAT) 또는 -2(VARCHAR)를 선택합니다. DMS는 Oracle에서 지원하는 모든 정밀도-스케일 조합을 지원합니다. 정밀도가 39 이상인 경우, -2(VARCHAR)를 선택합니다. Oracle 데이터베이스의 NumberDataTypeScale 설정은 NUMBER 데이터 유형에만 사용됩니다 (명시적 정밀도 및 배율 정의 제외).
`OpenTransactionWindow`	CDC 전용 작업을 위해 진행 중인 트랜잭션이 있는지 확인할 수 있는 분 단위 기간을 제공합니다. 기본값: 0 유효한 값: 0~240 사이의 정수 예제: `openTransactionWindow=15;`

명칭	설명
OraclePathPrefix	Amazon RDS for Oracle을 소스로 하여 변경 데이터 캡처에 Binary Reader를 사용하려면 이 문자열 속성을 필수 값으로 설정하세요. 이 값은 다시 실행 로그에 액세스하는 데 사용되는 기본 Oracle 루트를 지정합니다. 자세한 정보는 Oracle용 RDS와 함께 바이너리 리더를 사용하도록 CDC 작업을 구성하는 방법: AWS DMS을 참조하세요. 기본값: none 유효한 값: /rdsdbdata/db/ORCL_A/ 예제: `--oracle-settings '{"OraclePathPrefix": "  /rdsdbdata/db/ORCL_A/  "}'`
ParallelASMReadThreads	Oracle ASM(Automatic Storage Management)을 사용하여 변경 데이터 캡처(CDC)를 수행하기 위해 DMS가 구성하는 스레드 수를 변경하려면 이 속성을 설정합니다. 2(기본값) ~ 8(최대값) 사이의 정수 값을 지정할 수 있습니다. 이 속성을 ReadAheadBlocks 속성과 함께 사용합니다. 자세한 정보는 Oracle용 RDS와 함께 바이너리 리더를 사용하도록 CDC 작업을 구성하는 방법: AWS DMS을 참조하세요. 기본값: 2 유효한 값: 2~8 사이의 정수 예제: `--oracle-settings '{"ParallelASMReadThreads": 6;}'`

명칭	설명
ReadAheadBlocks	Oracle ASM(Automatic Storage Management) 및 ASM이 아닌 NAS 스토리지를 사용하여 CDC를 수행하기 위해 DMS가 구성하는 미리 읽기 블록 수를 변경하려면 이 속성을 설정합니다. 1000(기본값) ~ 200,000(최대값) 사이의 정수 값을 지정할 수 있습니다. 이 속성을 ParallelASMReadThreads 속성과 함께 사용합니다. 자세한 정보는 Oracle용 RDS와 함께 바이너리 리더를 사용하도록 CDC 작업을 구성하는 방법: AWS DMS을 참조하세요. 기본값: 1000 유효한 값: 1000~200,000 사이의 정수 예제: `--oracle-settings '{"ReadAheadBlocks": 150000}'`
ReadTableSpaceName	true로 설정할 경우, 이 속성에서 테이블스페이스 복제를 지원합니다. 기본값: false 유효 값: 불 예제: `--oracle-settings '{"ReadTableSpaceName": true}'`
ReplacePathPrefix	Amazon RDS for Oracle을 소스로 하여 변경 데이터 캡처에 Binary Reader를 사용하기 위해서는 이 속성을 true로 설정합니다. 이렇게 설정하면 DMS 인스턴스가 다시 실행 로그에 액세스하기 위해 지정된 UsePathPrefix 설정을 기본 Oracle 루트로 대체합니다. 자세한 정보는 Oracle용 RDS와 함께 바이너리 리더를 사용하도록 CDC 작업을 구성하는 방법: AWS DMS을 참조하세요. 기본값: false 유효값: true/false 예제: `--oracle-settings '{"ReplacePathPrefix": true}'`

명칭	설명
RetryInterval	시스템이 쿼리를 다시 전송하기까지 대기하는 시간(초)을 지정합니다. 기본값: 5 유효한 값: 1부터 시작하는 숫자 예제: `--oracle-settings '{"RetryInterval": 6}'`
SecurityDbEncryptionName	Oracle 원본 데이터베이스의 열 및 테이블스페이스의 TDE(Transparent Data Encryption)에 사용되는 키의 이름을 지정합니다. Oracle 소스 엔드포인트에서 이 속성 및 연결된 암호를 설정하는 방법에 대한 자세한 내용은 Oracle을 소스로 사용하기 위해 지원되는 암호화 방법 AWS DMS 단원을 참조하십시오. 기본값: "" 유효값: 문자열 예제: `--oracle-settings '{"SecurityDbEncryptionName": "ORACLE.SECURITY.DB.ENCRYPTION.Adg8m2dhkU/0v/m5QUaaNJEAAAAAAAAAAAAAAAAAAAAAAAAAAA"}'`
SpatialSdo2GeoJsonFunctionName	PostgreSQL 대상으로 마이그레이션하는 Oracle 버전 12.1 이전 소스의 경우 이 속성을 사용하여 SDO_GEOMETRY를 GEOJSON 형식으로 변환합니다. 기본적으로 는 사용자 정의 함수를 AWS DMS 호출합니다. SDO2GEOJSON 사용자 정의 함수는 존재하고 AWS DMS 사용자가 액세스할 수 있어야 합니다. 또는 SDOGEOJSON 의 작업을 모방하는 자체 사용자 정의 함수를 생성하여 SpatialSdo2GeoJsonFunctionName 이 대신 호출하도록 설정할 수 있습니다. 기본값: SDO2GEOJSON 유효값: 문자열 예제: `--oracle-settings '{"SpatialSdo2GeoJsonFunctionName": "myCustomSDO2GEOJSONFunction"}'`

명칭	설명
StandbyDelayTime	사용 이 속성을 사용하여 스탠바이 동기화에 대한 시간(분)을 지정합니다. 소스가 Active Data Guard 대기 데이터베이스인 경우, 이 속성을 사용하여 프라이머리 데이터베이스와 대기 데이터베이스 간의 시간 지연을 지정합니다. AWS DMS에서는 Active Data Guard 대기 인스턴스를 소스로 사용하여 진행 중인 변경 사항을 복제하는 Oracle CDC 작업을 생성할 수 있습니다. 이렇게 하면 프로덕션 단계에 있을 수 있는 활성 데이터베이스에 연결할 필요가 없습니다. 기본값: 0 유효한 값: 숫자 예제: `--oracle-settings '{"StandbyDelayTime": 1}'` 참고: DMS 3.4.6, 3.4.7 및 그 이상을 사용하는 경우 이 연결 설정의 사용은 선택 사항입니다. DMS 3.4.6 최신 버전과 버전 3.4.7에서 *dms_user*이 V_$DATAGUARD_STATS 에 select 권한이 있어야 DMS가 대기 지연 시간을 계산할 수 있습니다.
UseAlternateFolderForOnline	Amazon RDS for Oracle을 소스로 하여 변경 데이터 캡처에 Binary Reader를 사용하기 위해서는 이 속성을 true로 설정합니다. 그러면 DMS 인스턴스가 지정된 접두사 교체를 사용해 모든 온라인 다시 실행 로그에 액세스합니다. 자세한 정보는 Oracle용 RDS와 함께 바이너리 리더를 사용하도록 CDC 작업을 구성하는 방법: AWS DMS을 참조하세요. 기본값: false 유효값: true/false 예제: `--oracle-settings '{"UseAlternateFolderForOnline": true}'`

명칭	설명
UseBfile	Binary Reader 유틸리티를 사용하여 변경 데이터를 캡처하려면 이 속성을 Y로 설정합니다. 이 속성을 Y로 설정하려면 UseLogminerReader 를 N으로 설정합니다. Amazon RDS for Oracle을 소스로 하여 Binary Reader를 사용하려면 추가 속성을 설정합니다. 이 설정 및 Oracle Automatic Storage Management(ASM) 사용에 대한 자세한 내용은 CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용 단원을 참조하십시오. 참고: 이 값을 추가 연결 속성(ECA)으로 설정할 때 유효한 값은 'Y'와 'N'입니다. 이 값을 엔드포인트 설정으로 설정할 때 유효한 값은 true와 false입니다. 기본값: N 유효한 값: Y/N(이 값을 ECA로 설정하는 경우), true/false(이 값을 엔드포인트 설정으로 설정하는 경우). 예제: --oracle-settings '{"UseBfile": Y}'
UseLogminerReader	LogMiner 유틸리티를 사용하여 변경 데이터를 캡처하려면 이 속성을 Y로 설정합니다 (기본값). AWS DMS 가 이진 파일에서 다시 실행 로그에 액세스하도록 하려면 이 옵션을 N으로 설정합니다. 이 옵션을 N으로 설정하는 경우, useBfile=Y 설정도 추가하세요. 이 설정 및 Oracle ASM(Automatic Storage Management) 사용에 대한 자세한 내용은 CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용 섹션을 참조하세요. 참고: 이 값을 추가 연결 속성(ECA)으로 설정할 때 유효한 값은 'Y'와 'N'입니다. 이 값을 엔드포인트 설정으로 설정할 때 유효한 값은 true와 false입니다. 기본 값: Y 유효한 값: Y/N(이 값을 ECA로 설정하는 경우), true/false(이 값을 엔드포인트 설정으로 설정하는 경우). 예제: --oracle-settings '{"UseLogminerReader": Y}'

명칭	설명
UsePathPrefix	Amazon RDS for Oracle을 소스로 하여 변경 데이터 캡처에 Binary Reader를 사용하려면 이 문자열 속성을 필수 값으로 설정하세요. 이 값은 다시 실행 로그에 액세스하기 위해 기본 Oracle 루트를 대체하는 데 사용되는 경로 접두사를 지정합니다. 자세한 정보는 Oracle용 RDS와 함께 바이너리 리더를 사용하도록 CDC 작업을 구성하는 방법: AWS DMS을 참조하세요. 기본값: none 유효한 값: /rdsdbdata/log/ 예제: `--oracle-settings '{"UsePathPrefix": "  /rdsdbdata/log/ "}'`

Oracle용 소스 데이터 형식

의 Oracle 엔드포인트는 대부분의 Oracle 데이터 유형을 AWS DMS 지원합니다. 다음 표에는 사용 AWS DMS 시 지원되는 Oracle 소스 데이터 유형과 데이터 유형에 대한 기본 매핑이 나와 있습니다. AWS DMS

> **Note**
>
> LONG 및 LONG RAW 데이터 형식을 제외하고 Oracle 소스에서 Oracle 대상으로 복제(동종 복제)하는 경우, 모든 소스 및 대상 데이터 형식이 동일합니다. 하지만 LONG 데이터 형식은 CLOB에 매핑되고 LONG RAW 데이터 형식은 BLOB에 매핑됩니다.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하십시오.

AWS DMS 데이터 유형에 대한 자세한 내용은 을 참조하십시오AWS Database Migration Service에서 사용되는 데이터 형식.

Oracle 데이터 형식	AWS DMS 데이터 유형
BINARY_FLOAT	REAL4

Oracle 데이터 형식	AWS DMS 데이터 유형
BINARY_DOUBLE	REAL8
BINARY	BYTES
FLOAT (P)	정밀도가 24 이하인 경우, REAL4를 사용합니다. 정밀도가 24보다 큰 경우, REAL8을 사용합니다.
NUMBER (P,S)	스케일이 0보다 크면 NUMERIC을 사용합니다. 크기가 0인 경우: • 그리고 정밀도가 2 이하인 경우, INT1을 사용합니다. • 그리고 정밀도가 2보다 크고 4 이하인 경우, INT2를 사용합니다. • 그리고 정밀도가 4보다 크고 9 이하인 경우, INT4를 사용합니다. • 그리고 정밀도가 9보다 큰 경우, NUMERIC을 사용합니다. • 그리고 정밀도가 크기 이상인 경우, NUMERIC을 사용합니다. 스케일이 0 미만인 경우, REAL8을 사용합니다.
날짜	DATETIME
INTERVAL_YEAR TO MONTH	STRING(간격 year_to_month 표시 포함)
INTERVAL_DAY TO SECOND	STRING(간격 day_to_second 표시 포함)
TIMESTAMP	DATETIME
TIMESTAMP(시간대 사용)	STRING(timestamp_with_timezone 표시 포함)
TIMESTAMP WITH LOCAL TIME ZONE	STRING(timestamp_with_local_ timezone 표시 포함)
CHAR	STRING

Oracle 데이터 형식	AWS DMS 데이터 유형
VARCHAR2	STRING
NCHAR	WSTRING
NVARCHAR2	WSTRING
RAW	BYTES
REAL	REAL8
BLOB	BLOB 에서 이 데이터 유형을 사용하려면 특정 작업에 BLOB 데이터 유형을 사용할 수 있도록 설정해야 합니다. AWS DMS AWS DMS 기본 키가 포함된 테이블에서만 BLOB 데이터 유형을 지원합니다.
CLOB	CLOB 에서 이 데이터 유형을 사용하려면 특정 AWS DMS작업에 CLOB 데이터 유형을 사용할 수 있도록 설정해야 합니다. CDC 중에는 기본 키가 포함된 테이블에서만 CLOB 데이터 유형을 AWS DMS 지원합니다.
NCLOB	NCLOB 에서 이 데이터 유형을 사용하려면 특정 AWS DMS작업에 NCLOB 데이터 유형을 사용할 수 있도록 설정해야 합니다. CDC 중에는 기본 키가 포함된 테이블에서만 NCLOB 데이터 유형을 AWS DMS 지원합니다.

Oracle 데이터 형식	AWS DMS 데이터 유형
LONG	CLOB LONG 데이터 유형은 일괄 최적화 적용 모드 (CDC 모드) 에서는 지원되지 않습니다. TurboStream 이 데이터 유형을 함께 AWS DMS사용하려면 특정 작업에 LOB를 사용할 수 있도록 설정하세요. CDC 또는 전체 로드 중에는 기본 키가 있는 테이블에서만 LOB 데이터 유형을 AWS DMS 지원합니다. 또한 LONG 열을 로드하기 위한 전체 LOB 모드를 AWS DMS 지원하지 않습니다. 대신 제한된 LOB 모드를 사용하여 LONG 열을 Oracle 대상으로 마이그레이션할 수 있습니다. 제한된 LOB 모드에서는 64KB보다 긴 LONG 열로 설정한 모든 데이터를 64KB로 AWS DMS 잘라냅니다. 의 LOB 지원에 대한 자세한 내용은 을 참조하십시오. AWS DMS태스크의 소스 데이터베이스에 대한 LOB 지원 설정 AWS DMS
LONG RAW	BLOB LONG RAW 데이터 유형은 일괄 최적화 적용 모드 (TurboStream CDC 모드) 에서 지원되지 않습니다. 이 데이터 유형을 함께 AWS DMS사용하려면 특정 작업에 LOB를 사용할 수 있도록 설정하세요. CDC 또는 전체 로드 중에는 기본 키가 있는 테이블에서만 LOB 데이터 유형을 AWS DMS 지원합니다. 또한 LONG RAW 열을 로드하기 위한 전체 LOB 모드를 AWS DMS 지원하지 않습니다. 대신 제한된 LOB 모드를 사용하여 LONG RAW 열을 Oracle 대상으로 마이그레이션할 수 있습니다. 제한된 LOB 모드에서는 64KB보다 긴 LONG RAW 열로 설정한 모든 데이터를 64KB로 AWS DMS 잘라냅니다. 의 LOB 지원에 대한 자세한 내용은 을 참조하십시오. AWS DMS태스크의 소스 데이터베이스에 대한 LOB 지원 설정 AWS DMS

Oracle 데이터 형식	AWS DMS 데이터 유형
XMLTYPE	CLOB
SDO_GEOMETRY	BLOB(Oracle에서 Oracle로 마이그레이션하는 경우) CLOB(Oracle에서 PostgreSQL로 마이그레이션하는 경우)

다음 데이터 형식의 열과 함께 소스로 사용되는 Oracle 테이블은 지원되지 않으며 복제될 수도 없습니다. 이 데이터 형식을 사용하는 열을 복제하면 null 열이 됩니다.

- BFILE
- ROWID
- REF
- UROWID
- 사용자 정의 데이터 형식
- ANYDATA
- VARRAY

> **Note**
>
> 가상 열은 지원되지 않습니다.

Oracle 공간 데이터 형식 마이그레이션

공간 데이터는 공간에서 객체 또는 위치의 지오메트리 정보를 식별합니다. Oracle 데이터베이스에서 공간 객체의 기하학적 설명은 SDO_GEOMETRY 형식 객체에 저장됩니다. 이 객체 내에서 기하학적 설명은 사용자 정의 테이블의 단일 열에 있는 단일 행에 저장됩니다.

AWS DMS Oracle 유형 SDO_GEOMETRY를 오라클 소스에서 오라클 또는 PostgreSQL 타겟으로 마이그레이션하는 것을 지원합니다.

를 사용하여 AWS DMS Oracle 공간 데이터 유형을 마이그레이션할 때는 다음 고려 사항에 유의하십시오.

- Oracle 대상으로 마이그레이션하는 경우 형식 정보를 포함하는 USER_SDO_GEOM_METADATA 항목을 수동으로 전송해야 합니다.
- Oracle 소스 엔드포인트에서 PostgreSQL 대상 엔드포인트로 마이그레이션할 때 대상 열을 생성합니다. AWS DMS 이러한 열에는 2D 차원과 SRID(공간 참조 식별자)가 0인 기본 geometry 및 geography 형식 정보가 있습니다. 예를 들면, `GEOMETRY, 2, 0`입니다.
- PostgreSQL 대상으로 마이그레이션하는 Oracle 버전 12.1 이전 소스의 경우 `SDO2GEOJSON` 함수 또는 `spatialSdo2GeoJsonFunctionName` 추가 연결 속성을 사용하여 `SDO_GEOMETRY` 객체를 `GEOJSON` 형식으로 변환합니다. 자세한 정보는 Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS을 참조하세요.
- AWS DMS 전체 LOB 모드에 대해서만 Oracle 공간 열 마이그레이션을 지원합니다. AWS DMS 제한된 LOB 또는 인라인 LOB 모드를 지원하지 않습니다. LOB 모드에 대한 자세한 내용은 태스크의 소스 데이터베이스에 대한 LOB 지원 설정 AWS DMS 섹션을 참조하세요.
- Oracle Spatial Columns 마이그레이션을 위한 전체 LOB AWS DMS 모드만 지원하므로 열 테이블에는 기본 키와 고유 키가 필요합니다. 테이블에 프라이머리 키와 고유 키가 없는 경우 마이그레이션에서 테이블을 건너뜁니다.

Microsoft SQL Server 데이터베이스를 원본으로 사용 AWS DMS

를 사용하여 하나 이상의 Microsoft SQL Server 데이터베이스에서 데이터를 AWS DMS 마이그레이션합니다. SQL Server 데이터베이스를 원본으로 사용하면 데이터를 다른 SQL Server 데이터베이스 또는 AWS DMS 지원되는 다른 데이터베이스 중 하나로 마이그레이션할 수 있습니다.

원본으로 AWS DMS 지원하는 SQL Server 버전에 대한 자세한 내용은 을 참조하십시오출처: AWS DMS.

원본 SQL Server 데이터베이스는 네트워크 상의 어느 컴퓨터에나 설치할 수 있습니다. AWS DMS에서 사용하려면 사용자가 선택한 작업 유형에 적절한 원본 데이터베이스 액세스 권한이 있는 SQL Server 계정이 필요합니다. 계정에는 `view definition` 및 `view server state` 권한이 있어야 합니다. 다음 명령을 사용하여 이 권한을 추가합니다.

```
grant view definition to [user]
grant view server state to [user]
```

AWS DMS SQL Server의 명명된 인스턴스에서 데이터를 마이그레이션하는 것을 지원합니다. 원본 엔드포인트를 생성하는 경우 서버 이름에 다음 표기법을 사용할 수 있습니다.

```
IPAddress\InstanceName
```

예를 들어 올바른 원본 엔드포인트 서버 이름은 다음과 같습니다. 여기서, 이름의 첫 번째 부분은 서버의 IP 주소이며, 두 번째 부분은 SQL Server 인스턴스 이름(이 예에서는 SQLTest)입니다.

```
10.0.0.25\SQLTest
```

또한 SQL Server의 명명된 인스턴스가 수신하는 포트 번호를 얻고 이를 사용하여 AWS DMS 원본 엔드포인트를 구성하십시오.

> **Note**
>
> 포트 1433은 Microsoft SQL Server의 기본값입니다. 그러나 SQL Server가 시작될 때마다 변경되는 동적 포트와 방화벽을 통해 SQL Server에 연결하는 데 사용되는 특정 정적 포트 번호도 자주 사용됩니다. 따라서 AWS DMS 원본 엔드포인트를 생성할 때 지정된 SQL Server 인스턴스의 실제 포트 번호를 알고 싶을 것입니다.

SSL을 사용하여 SQL Server 엔드포인트와 복제 인스턴스 사이의 연결을 암호화할 수 있습니다. SQL Server 엔드포인트에서 SSL을 사용하는 방법에 대한 자세한 내용은 AWS Database Migration Service에서 SSL 사용 섹션을 참조하십시오.

SQL Server 원본 데이터베이스 작업에 대한 추가 세부 정보는 AWS DMS 다음을 참조하십시오.

주제

- SQL Server를 원본으로 사용할 때의 제한 사항 AWS DMS
- 전체 로드 전용 작업에 대한 권한
- SQL Server 소스에서 지속적 복제(CDC) 사용을 위한 사전 요구 사항
- 온프레미스 또는 Amazon EC2에서 자체 관리형 SQL Server의 데이터 변경 캡처
- 클라우드 SQL Server DB 인스턴스에서 지속적 복제 설정
- SQL Server용 Amazon RDS를 원본으로 사용할 때의 권장 설정 AWS DMS
- SQL Server에 지원되는 압축 방법
- 자체 관리형 SQL Server 가용성 그룹을 사용한 작업 AlwaysOn
- SQL Server를 원본으로 사용할 때의 보안 요구 사항 AWS Database Migration Service
- SQL Server를 원본으로 사용할 때의 엔드포인트 설정 AWS DMS
- SQL Server용 소스 데이터 형식

SQL Server를 원본으로 사용할 때의 제한 사항 AWS DMS

다음 제한 사항은 SQL Server 데이터베이스를 AWS DMS용 소스로 사용 시 적용됩니다.

- 열의 자격 증명 속성은 대상 데이터베이스 열로 마이그레이션되지 않습니다.
- SQL Server 엔드포인트는 스파스 열이 있는 테이블의 사용을 지원하지 않습니다.
- Windows 인증은 지원되지 않습니다.
- SQL Server에서 컴퓨팅된 필드의 변경 사항은 복제되지 않습니다.
- 임시 테이블은 지원되지 않습니다.
- SQL Server 파티션 전환은 지원되지 않습니다.
- WRITETEXT 및 UPDATETEXT 유틸리티를 사용하는 경우 원본 데이터베이스에 적용된 이벤트를 캡처하지 AWS DMS 않습니다.
- 다음 데이터 조작 언어(DML) 패턴은 지원되지 않습니다.

```
SELECT * INTO new_table FROM existing_table
```

- SQL Server를 소스로 사용할 경우, 열 수준 암호화는 지원되지 않습니다.
- AWS DMS SQL Server 2008 또는 SQL Server 2008 R2를 원본으로 사용하는 경우 서버 수준 감사를 지원하지 않습니다. 이는 SQL Server 2008 및 2008 R2의 알려진 문제 때문입니다. 예를 들어, 다음 명령을 실행하면 오류가 발생합니다 AWS DMS .

```
USE [master]
GO
ALTER SERVER AUDIT [my_audit_test-20140710] WITH (STATE=on)
GO
```

- SQL Server를 소스로 사용하는 경우, Geometry 열은 전체 LOB 모드에서 지원되지 않습니다. 대신, 제한적 LOB 모드를 사용하거나 인라인 LOB 모드를 사용하도록 `InlineLobMaxSize` 작업 설정을 설정하세요.
- 복제 작업에서 Microsoft SQL Server 소스 데이터베이스를 사용하는 경우, 작업을 제거하면 SQL Server Replication Publisher 정의가 제거되지 않습니다. Microsoft SQL Server 시스템 관리자는 Microsoft SQL Server에서 해당 정의를 삭제해야 합니다.
- 전체 로드 전용 작업의 경우 스키마 바운드 및 non-schema-bound 뷰에서 데이터를 마이그레이션할 수 있습니다.
- sp_rename을 사용하여 테이블 이름을 바꾸는 것은 지원되지 않습니다(예: `sp_rename 'Sales.SalesRegion', 'SalesReg;`)).

- sp_rename을 사용하여 열 이름을 바꾸는 것은 지원되지 않습니다(예: `sp_rename 'Sales.Sales.Region', 'RegID', 'COLUMN';`).
- AWS DMS 열 기본값을 설정 및 해제하기 위한 변경 처리를 지원하지 않습니다 (명령문과 함께 절 사용). `ALTER COLUMN SET DEFAULT` `ALTER TABLE`
- AWS DMS 열 무효화 허용 여부를 설정하기 위한 변경 처리 (명령문과 함께 절 사용) 를 지원하지 않습니다. `ALTER COLUMN [SET|DROP] NOT NULL` `ALTER TABLE`
- SQL Server 2012 및 SQL Server 2014에서는 가용성 그룹과 함께 DMS 복제를 사용하는 경우, 배포 데이터베이스를 가용성 그룹에 배치할 수 없습니다. SQL 2016은 병합, 양방향 또는 복제 토폴로지에 사용되는 배포 데이터베이스를 제외하고 배포 데이터베이스를 가용성 그룹에 배치하는 것을 지원합니다. peer-to-peer
- 분할된 테이블의 경우 파티션마다 다른 데이터 압축 설정을 AWS DMS 지원하지 않습니다.
- SQL Server 공간 데이터 유형(GEOGRAPHY 및 GEOMETRY)에 값을 삽입할 때 공간 참조 시스템 식별자(SRID) 속성을 무시하거나 다른 숫자를 지정할 수 있습니다. 공간 데이터 유형이 있는 테이블을 복제하는 경우 SRID를 기본 SRID (지오메트리의 경우 0, GEOGRAPHY의 경우 4326) 로 AWS DMS 대체합니다.
- 데이터베이스가 MS-REPLICATION 또는 MS-CDC에 대해 구성되지 않은 경우에도 프라이머리 키가 없는 테이블을 캡처할 수 있지만 INSERT/DELETE DML 이벤트만 캡처됩니다. UPDATE 및 TRUNCATE TABLE 이벤트는 무시됩니다.
- Columnstore 인덱스는 지원되지 않습니다.
- 메모리 최적화 테이블(인 메모리 OLTP 사용)은 지원되지 않습니다.
- 여러 열로 구성된 프라이머리 키가 있는 테이블을 복제할 때 전체 로드 중에 프라이머리 키 열을 업데이트하는 것은 지원되지 않습니다.
- 지연된 지속성은 지원되지 않습니다.
- `readBackup0nly=Y` 엔드포인트 설정(추가 연결 속성)은 RDS의 백업 수행 방식 때문에 RDS for SQL Server 소스 인스턴스에서 작동하지 않습니다.
- RDS 사용자에게는 SQL Server 저장 프로시저 `sp_repldone`을 실행할 수 있는 액세스 권한이 없기 때문에 `EXCLUSIVE_AUTOMATIC_TRUNCATION`는 Amazon RDS SQL Server 소스 인스턴스에서는 작동하지 않습니다.
- AWS DMS 잘라내기 명령은 캡처하지 않습니다.
- AWS DMS ADR (가속 데이터베이스 복구) 이 설정된 데이터베이스에서의 복제는 지원하지 않습니다.
- AWS DMS 는 단일 트랜잭션 내에서 DDL (데이터 정의 언어) 및 DML (데이터 조작 언어) 문을 캡처하는 것을 지원하지 않습니다.

- AWS DMS 데이터 계층 응용 프로그램 패키지 (DACPAC) 의 복제를 지원하지 않습니다.
- 프라이머리 키나 고유 인덱스를 포함하고 여러 데이터 행을 업데이트하는 UPDATE 문은 대상 데이터베이스에 변경 내용을 적용할 때 충돌을 일으킬 수 있습니다. 예를 들어 대상 데이터베이스가 업데이트를 단일 UPDATE 문 대신 INSERT 문과 DELETE 문으로 적용하는 경우, 이런 일이 발생할 수 있습니다. 일괄 최적화 적용 모드에서는 테이블이 무시될 수 있습니다. 트랜잭션 적용 모드에서는 UPDATE 작업 시 제약 조건 위반이 발생할 수 있습니다. 이 문제를 방지하려면 관련 테이블을 다시 로드하세요. 또는 Apply Exceptions 제어 테이블(`dmslogs.awsdms_apply_exceptions`)에서 문제가 되는 레코드를 찾아 대상 데이터베이스에서 수동으로 편집합니다. 자세한 설명은 변경 처리 튜닝 설정 섹션을 참조하세요.
- AWS DMS 이름에 다음 세트의 특수 문자가 포함된 테이블 및 스키마의 복제를 지원하지 않습니다.

  ```
  \\ -- \n \" \b \r ' \t ;
  ```
- 데이터 마스킹은 지원되지 않습니다. AWS DMS 마스킹 없이 마스킹된 데이터를 마이그레이션합니다.
- AWS DMS 기본 키가 있는 최대 32,767개의 테이블과 각 테이블에 대해 최대 1,000개의 열을 복제합니다. 이는 복제된 각 테이블에 대해 SQL Server 복제 아티클을 AWS DMS 만들며 SQL Server 복제 아티클에는 이러한 제한이 있기 때문입니다.
- CDC(변경 데이터 캡처)를 사용할 때는 고유 인덱스를 구성하는 모든 열을 `NOT NULL`로 정의해야 합니다. 이 요구 사항이 충족되지 않으면 SQL Server 시스템 오류 22838이 발생합니다.

백업 트랜잭션 로그에 액세스할 때는 다음 제한 사항이 적용됩니다.

- 암호화된 백업은 지원되지 않습니다.
- URL 또는 Windows Azure에 저장된 백업은 지원되지 않습니다.
- AWS DMS 대체 공유 폴더의 파일 수준에서 트랜잭션 로그 백업을 직접 처리하는 것은 지원하지 않습니다.

전체 로드 전용 작업에 대한 권한

전체 로드 전용 작업을 수행하려면 다음 권한이 필요합니다. 단, 이렇게 해도 `dms_user` 로그인이 생성되지는 AWS DMS 않습니다. SQL Server 로그인 생성에 대한 자세한 내용은 Microsoft SQL Server를 사용하여 데이터베이스 사용자 생성 섹션을 참조하세요.

```
USE db_name;

                CREATE USER dms_user FOR LOGIN dms_user;
```

```
ALTER ROLE [db_datareader] ADD MEMBER dms_user;
GRANT VIEW DATABASE STATE to dms_user ;

USE master;

GRANT VIEW SERVER STATE TO dms_user;
```

SQL Server 소스에서 지속적 복제(CDC) 사용을 위한 사전 요구 사항

온프레미스 또는 Amazon EC2의 자체 관리형 SQL Server 데이터베이스 또는 Amazon RDS나 Microsoft Azure SQL 관리형 인스턴스 같은 클라우드 데이터베이스에 지속적 복제(변경 데이터 캡처, 즉 CDC)를 사용할 수 있습니다.

특히 SQL Server 데이터베이스를 AWS DMS의 소스로 이용하는 지속적 복제를 사용할 때 다음 요구 사항이 적용됩니다.

- SQL Server는 전체 백업에 맞게 구성되어야 하며, 사용자는 데이터 복제를 시작하기 전에 백업을 수행해야 합니다.
- 복구 모델은 [Bulk logged] 또는 [Full]로 설정되어야 합니다.
- 여러 디스크로의 SQL Server 백업은 지원되지 않습니다. 서로 다른 디스크에 있는 여러 파일에 데이터베이스 백업을 쓰도록 백업이 정의된 경우 데이터를 읽을 AWS DMS 수 없어 AWS DMS 작업이 실패합니다.
- 자체 관리형 SQL Server 소스의 경우, 작업을 제거할 때 DMS CDC 작업에 사용되는 소스의 SQL Server Replication Publisher 정의는 제거되지 않습니다. SQL Server 시스템 관리자는 자체 관리형 원본을 위해 SQL Server에서 이 정의를 삭제해야 합니다.
- CDC 중에는 변경 내용을 읽으려면 SQL Server 트랜잭션 로그 백업을 찾아봐야 합니다. AWS DMS AWS DMS 기본 형식이 아닌 타사 백업 소프트웨어를 사용하여 만든 SQL Server 트랜잭션 로그 백업은 지원되지 않습니다. 네이티브 형식이고 타사 백업 소프트웨어를 사용하여 생성된 트랜잭션 로그 백업을 지원하려면 소스 엔드포인트에 `use3rdPartyBackupDevice=Y` 연결 속성을 추가하세요.
- 자체 관리형 SQL Server 원본의 경우, SQL Server는 변경 사항이 게시될 때까지 새로 생성된 테이블에서 변경 사항을 캡처하지 않음에 유의하십시오. 테이블이 SQL Server 원본에 추가되면 발행물 만들기를 AWS DMS 관리합니다. 그렇지만, 이 프로세스에는 몇 분이 걸릴 수 있습니다. 이 지연 시간 동안 새로 생성된 테이블에 적용된 작업은 대상에 캡처되거나 복제되지 않습니다.
- AWS DMS 변경 데이터를 캡처하려면 SQL Server에서 전체 트랜잭션 로깅을 활성화해야 합니다. SQL Server에서 전체 트랜잭션 로깅을 켜려면 MS-REPLICATION 또는 CHANGE DATA CAPTURE(CDC)를 활성화하세요.

- SQL Server tlog 항목은 MS CDC 캡처 작업이 해당 변경 내용을 처리할 때까지는 재사용으로 표시되지 않습니다.
- CDC 작업은 메모리 최적화된 테이블에서 지원되지 않습니다. 이 제한 사항은 이 기능이 처음 도입된 SQL Server 2014 이상에 적용됩니다.
- AWS DMS 변경 데이터를 캡처하려면 기본적으로 Amazon EC2 또는 온프레미스 SQL 서버의 배포 데이터베이스를 소스로 사용해야 합니다. 따라서 프라이머리 키가 있는 테이블의 MS 복제를 구성하는 동안 배포자를 활성화했는지 확인하세요.

온프레미스 또는 Amazon EC2에서 자체 관리형 SQL Server의 데이터 변경 캡처

소스 Microsoft SQL Server 데이터베이스에서 변경 내용을 캡처하려면 데이터베이스가 전체 백업을 수행하도록 구성되어 있어야 합니다. 데이터베이스를 전체 복구 모드 또는 대량 로그 모드로 구성합니다.

자체 관리형 SQL Server 원본의 경우 다음을 AWS DMS 사용합니다.

MS-REPLICATION

프라이머리 키가 있는 테이블의 변경 사항을 캡처합니다. 원본 SQL Server 인스턴스의 AWS DMS 엔드포인트 사용자에게 sysadmin 권한을 제공하여 이를 자동으로 구성할 수 있습니다. 또는 이 섹션의 단계에 따라 원본을 준비하고 엔드포인트에 대한 sysadmin 권한이 없는 사용자를 사용할 수 있습니다. AWS DMS

MS-CDC

프라이머리 키가 없는 테이블의 변경 사항을 캡처합니다. 데이터베이스 수준에서 모든 테이블에 대해 개별적으로 MS-CDC를 활성화합니다.

지속적 복제(CDC)를 위해 SQL Server 데이터베이스를 설정할 때 다음 중 하나를 수행할 수 있습니다.

- sysadmin 역할을 사용하여 지속적 복제를 설정합니다.
- sysadmin 역할을 사용하지 않도록 지속적 복제를 설정합니다.

자체 관리형 SQL Server에서 지속적 복제 설정

이 섹션에는 sysadmin 역할을 사용하거나 사용하지 않고 자체 관리형 SQL 서버에서 지속적 복제를 설정하는 방법에 대한 정보가 나와 있습니다.

주제

- 자체 관리형 SQL Server에서 지속적 복제 설정: sysadmin 역할 사용
- 독립 실행형 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음

자체 관리형 SQL Server에서 지속적 복제 설정: sysadmin 역할 사용

AWS DMS SQL Server의 지속적 복제는 기본 키가 있는 테이블에는 기본 SQL Server 복제를 사용하고, 기본 키가 없는 테이블에는 CDC (변경 데이터 캡처) 를 사용합니다.

지속적 복제를 설정하기 전에 SQL Server 소스에서 지속적 복제(CDC) 사용을 위한 사전 요구 사항 섹션을 참조하세요.

기본 키가 있는 테이블의 경우 일반적으로 AWS DMS 원본에 필요한 아티팩트를 구성할 수 있습니다. 하지만 자체 관리형인 SQL Server 소스 인스턴스의 경우, 먼저 SQL Server 배포를 수동으로 구성해야 합니다. 이렇게 하면 sysadmin 권한이 있는 AWS DMS 원본 사용자가 기본 키가 있는 테이블에 대한 게시를 자동으로 만들 수 있습니다.

배포가 이미 구성되어 있는지 확인하려면 다음 명령을 실행합니다.

```
sp_get_distributor
```

열 배포 결과가 NULL인 경우 배포가 구성되지 않은 것입니다. 다음 절차에 따라 배포를 설정할 수 있습니다.

배포를 설정하려면

1. SQL Server Management Studio(SSMS) 도구를 사용하여 SQL Server 소스 데이터베이스에 연결합니다.
2. 복제 폴더의 컨텍스트 메뉴(마우스 오른쪽 버튼 클릭)를 열고 배포 구성을 선택합니다. 배포 구성 마법사가 나타납니다.
3. 마법사에 따라 기본값을 입력하고 배포를 생성합니다.

CDC를 설정하려면

AWS DMS 버전 3.4.7 이상에서는 읽기 전용 복제본을 사용하지 않는 경우 데이터베이스 및 모든 테이블에 대해 MS CDC를 자동으로 설정할 수 있습니다. 이 기능을 사용하려면 `SetUpMsCdcForTables` ECA를 true로 설정합니다. ECA에 대한 자세한 내용은 엔드포인트 설정 섹션을 참조하세요.

3.4.7 AWS DMS 이전 버전이거나 읽기 전용 복제본을 원본으로 사용하는 경우 다음 단계를 수행하십시오.

1. 프라이머리 키가 없는 테이블의 경우, 데이터베이스에 MS-CDC를 설정합니다. 이렇게 하려면 sysadmin 역할이 할당된 계정을 사용하고 다음 명령을 실행합니다.

```
use [DBname]
EXEC sys.sp_cdc_enable_db
```

2. 다음으로, 각 소스 테이블마다 MS-CDC를 설정합니다. 고유 키는 있지만 프라이머리 키가 없는 각 테이블마다 다음 쿼리를 실행하여 MS-CDC를 설정합니다.

```
exec sys.sp_cdc_enable_table
@source_schema = N'schema_name',
@source_name = N'table_name',
@index_name = N'unique_index_name',
@role_name = NULL,
@supports_net_changes = 1
GO
```

3. 프라이머리 키가 없거나 고유 키가 없는 각 테이블마다 다음 쿼리를 실행하여 MS-CDC를 설정합니다.

```
exec sys.sp_cdc_enable_table
@source_schema = N'schema_name',
@source_name = N'table_name',
@role_name = NULL
GO
```

특정 테이블에서 MS-CDC를 설정하는 방법에 대한 자세한 내용은 SQL Server 설명서를 참조하십시오.

독립 실행형 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음

sysadmin 역할 없이 독립 실행형 SQL Server에서 지속적 복제를 설정하는 방법에 대한 자세한 내용은 독립 실행형 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음 섹션을 참조하세요.

클라우드 SQL Server DB 인스턴스에서 지속적 복제 설정

이 섹션에서는 클라우드에 호스팅된 SQL Server 데이터베이스 인스턴스에 CDC를 설정하는 방법을 설명합니다. 클라우드에 호스팅된 SQL 서버 인스턴스는 Amazon RDS for SQL Server, Azure SQL Managed Instance 또는 기타 관리형 클라우드 SQL Server 인스턴스에서 실행되는 인스턴스입니다.

각 데이터베이스 유형에 따른 지속적 복제 제한 사항에 대한 자세한 내용은 SQL Server를 원본으로 사용할 때의 제한 사항 AWS DMS 섹션을 참조하세요.

지속적 복제를 설정하기 전에 SQL Server 소스에서 지속적 복제(CDC) 사용을 위한 사전 요구 사항 섹션을 참조하세요.

자체 관리형 Microsoft SQL Server 소스와 달리 Amazon RDS for SQL Server는 MS-Replication을 지원하지 않습니다. 따라서 기본 키가 AWS DMS 있거나 없는 테이블에는 MS-CDC를 사용해야 합니다.

Amazon RDS는 원본 SQL Server 인스턴스의 지속적인 변경에 AWS DMS 사용되는 복제 아티팩트를 설정할 수 있는 시스템 관리자 권한을 부여하지 않습니다. 다음 절차와 같이 Amazon RDS 인스턴스에서 MS-CDC를 켜야 합니다(마스터 사용자 권한 사용).

클라우드 SQL Server DB 인스턴스에서 MS-CDC를 켜려면

1. 데이터베이스 수준에서 다음 쿼리 중 하나를 실행합니다.

   RDS for SQL Server DB 인스턴스에는 다음 쿼리를 사용하세요.

   ```
   exec msdb.dbo.rds_cdc_enable_db 'DB_name'
   ```

   Azure SQL 관리형 DB 인스턴스에는 다음 쿼리를 사용하세요.

   ```
   USE DB_name
   GO
   EXEC sys.sp_cdc_enable_db
   GO
   ```

2. 프라이머리 키가 있는 각 테이블마다 다음 쿼리를 실행하여 MS-CDC를 켭니다.

   ```
   exec sys.sp_cdc_enable_table
   @source_schema = N'schema_name',
   @source_name = N'table_name',
   @role_name = NULL,
   @supports_net_changes = 1
   GO
   ```

   고유 키는 있지만 프라이머리 키가 없는 각 테이블마다 다음 쿼리를 실행하여 MS-CDC를 켭니다.

   ```
   exec sys.sp_cdc_enable_table
   @source_schema = N'schema_name',
   ```

```
@source_name = N'table_name',
@index_name = N'unique_index_name',
@role_name = NULL,
@supports_net_changes = 1
GO
```

프라이머리 키도 없고 고유 키도 없는 각 테이블마다 다음 쿼리를 실행하여 MS-CDC를 켭니다.

```
exec sys.sp_cdc_enable_table
@source_schema = N'schema_name',
@source_name = N'table_name',
@role_name = NULL
GO
```

3. 다음 명령을 사용하여 원본에서 사용할 수 있도록 변경 사항 보존 기간을 설정합니다.

```
use dbname
EXEC sys.sp_cdc_change_job @job_type = 'capture' ,@pollinginterval = 86399
exec sp_cdc_stop_job 'capture'
exec sp_cdc_start_job 'capture'
```

`@pollinginterval` 파라미터는 초 단위로 측정되며 권장 값은 86,399로 설정되어 있습니다. 즉, `@pollinginterval = 86399`일 경우 트랜잭션 로그는 86,399초(하루) 동안 변경 사항을 보존합니다. 프로시저 `exec sp_cdc_start_job 'capture'`에서 설정을 시작합니다.

> **Note**
>
> 일부 버전의 SQL Server에서는 `pollinginterval` 값을 3,599초 이상으로 설정하면 값이 기본값인 5초로 재설정됩니다. 이 경우 T-Log 항목은 읽을 수 있기 전에 삭제됩니다. AWS DMS 이 알려진 문제의 영향을 받는 SQL Server 버전을 확인하려면 이 Microsoft KB 문서를 참조하세요.

다중 AZ에서 Amazon RDS를 사용하는 경우, 장애 조치 시 보조 AZ도 올바른 값을 갖도록 설정해야 합니다.

```
exec rdsadmin..rds_set_configuration 'cdc_capture_pollinginterval' , 86399
```

SQL Server 원본에 대한 지속적인 변경 사항을 캡처하는 AWS DMS 복제 작업이 1시간 이상 중지된 경우 다음 절차를 사용하십시오.

AWS DMS 복제 작업 중에 보존 기간을 유지하려면

1. 다음 명령을 사용하여 트랜잭션 로그를 잘라내는 작업을 중지합니다.

```
exec sp_cdc_stop_job 'capture'
```

2. AWS DMS 콘솔에서 작업을 찾아 작업을 재개하십시오.
3. 모니터링 탭을 선택하고 CDCLatencySource 지표를 확인합니다.
4. CDCLatencySource 지표가 0이고 그대로 유지되면 다음 명령을 사용하여 트랜잭션 로그를 잘라내는 작업을 다시 시작합니다.

```
exec sp_cdc_start_job 'capture'
```

SQL Server 트랜잭션 로그를 잘라내는 작업을 시작해야 한다는 점을 명심하세요. 그렇지 않으면 SQL Server 인스턴스의 스토리지가 꽉 찰 수 있습니다.

클라우드 SQL Server DB 인스턴스에서의 지속적 복제 제한 사항

- AWS DMS 활성 트랜잭션 로그만 포함하는 지속적 복제 (CDC) 를 지원합니다. CDC에서는 백업 로그를 사용할 수 없습니다.
- 이벤트를 활성 트랜잭션 로그에서 백업 로그로 옮기거나 활성 트랜잭션 로그에서 잘라내면 이벤트가 손실될 수 있습니다.

SQL Server용 Amazon RDS를 원본으로 사용할 때의 권장 설정 AWS DMS

Amazon RDS for SQL Server를 소스로 사용하는 경우, 캡처 작업은 maxscans 및 maxtrans 파라미터를 사용합니다. 이러한 파라미터는 캡처가 트랜잭션 로그에서 수행하는 최대 스캔 수와 각 스캔별로 처리되는 트랜잭션 수를 제어합니다.

트랜잭션 수가 maxtrans*maxscans보다 많은 데이터베이스의 경우, polling_interval 값을 늘리면 활성 트랜잭션 로그 레코드가 누적될 수 있습니다. 결과적으로 이러한 누적으로 인해 트랜잭션 로그 크기가 증가할 수 있습니다.

단, MS-CDC 캡처 작업에는 의존하지 AWS DMS 않습니다. MS-CDC 캡처 작업은 트랜잭션 로그 항목을 처리된 것으로 표시합니다. 이렇게 하면 트랜잭션 로그 백업 작업이 트랜잭션 로그에서 항목을 제거할 수 있습니다.

트랜잭션 로그의 크기와 MS-CDC 작업의 성공 여부를 모니터링하는 것이 좋습니다. MS-CDC 작업이 실패하면 트랜잭션 로그가 과도하게 증가하여 복제 실패가 발생할 수 있습니다. AWS DMS 소스 데이터베이스의 `sys.dm_cdc_errors` 동적 관리 뷰를 사용하여 MS-CDC 캡처 작업 오류를 모니터링할 수 있습니다. `DBCC SQLPERF(LOGSPACE)` 관리 명령을 사용하여 트랜잭션 로그 크기를 모니터링할 수 있습니다.

MS-CDC로 인한 트랜잭션 로그 증가를 해결하려면

1. 에서 데이터베이스가 `Log Space Used %` AWS DMS 복제되고 있는지 확인하고 데이터베이스가 계속 증가하는지 확인하십시오.

   ```
   DBCC SQLPERF(LOGSPACE)
   ```

2. 트랜잭션 로그 백업 프로세스를 방해하는 요인을 식별합니다.

   ```
   Select log_reuse_wait, log_reuse_wait_desc, name from sys.databases where name =
    db_name();
   ```

   `log_reuse_wait_desc` 값이 `REPLICATION`과 같으면 MS-CDC의 지연 시간으로 인해 로그 백업 보존이 발생합니다.

3. `maxtrans` 및 `maxscans` 파라미터 값을 늘려 캡처 작업에서 처리되는 이벤트 수를 늘립니다.

   ```
   EXEC sys.sp_cdc_change_job @job_type = 'capture' ,@maxtrans = 5000, @maxscans = 20
   exec sp_cdc_stop_job 'capture'
   exec sp_cdc_start_job 'capture'
   ```

이 문제를 해결하려면 `maxscans` 및 `maxtrans` 값을 매일 원본 데이터베이스에서 `maxtrans*maxscans` AWS DMS 복제되는 테이블에 대해 생성되는 평균 이벤트 수와 같도록 설정하십시오.

이러한 파라미터를 권장 값보다 높게 설정하면 캡처 작업은 트랜잭션 로그의 모든 이벤트를 처리합니다. 이러한 파라미터를 권장 값보다 낮게 설정하면 MS-CDC 지연 시간이 늘어나고 트랜잭션 로그가 증가합니다.

워크로드의 변경으로 이벤트 수가 달라지기 때문에 적절한 maxscans 값과 maxtrans 값을 식별하는 것이 어려울 수 있습니다. 이 경우 MS-CDC 지연 시간에 대한 모니터링을 설정하는 것이 좋습니다. 자세한 내용은 SQL Server 설명서에서 Monitor the process를 참조하세요. 그런 다음 모니터링 결과를 기반으로 maxtrans과 maxscans를 동적으로 구성합니다.

AWS DMS 작업을 재개하거나 계속하는 데 필요한 LSN (로그 시퀀스 번호) 을 찾을 수 없는 경우 작업이 실패하여 완전히 다시 로드해야 할 수 있습니다.

> **Note**
>
> 를 AWS DMS 사용하여 SQL Server용 RDS에서 데이터를 복제할 때 Amazon RDS 인스턴스의 중지 시작 이벤트 이후 복제를 재개하려고 하면 오류가 발생할 수 있습니다. 이는 중지-시작 이벤트 후 다시 시작할 때 SQL Server Agent 프로세스가 캡처 작업 프로세스를 다시 시작하기 때문입니다. 이는 MS-CDC 폴링 간격을 우회합니다.
> 이로 인해 MS-CDC 캡처 작업 처리보다 트랜잭션 볼륨이 낮은 데이터베이스에서는 데이터가 중지된 지점부터 재개되기 전에 데이터가 처리되거나 복제 및 백업된 것으로 표시되어 다음과 같은 오류가 발생할 AWS DMS 수 있습니다.
>
> ```
> [SOURCE_CAPTURE ]E: Failed to access LSN '0000dbd9:0006f9ad:0003' in
>  the backup log sets since BACKUP/LOG-s are not available. [1020465]
>  (sqlserver_endpoint_capture.c:764)
> ```
>
> 이 문제를 완화하려면 앞서 권장한 대로 maxtrans 및 maxscans 값을 설정하세요.

SQL Server에 지원되는 압축 방법

AWS DMS에서의 SQL Server 압축 방법 지원에 대해서는 다음을 참고하세요.

- AWS DMS SQL Server 버전 2008 이상에서는 행/페이지 압축을 지원합니다.
- AWS DMS Vardecimal 저장 형식을 지원하지 않습니다.
- AWS DMS 스파스 열 및 열 구조 압축은 지원하지 않습니다.

자체 관리형 SQL Server 가용성 그룹을 사용한 작업 AlwaysOn

SQL Server AlwaysOn 가용성 그룹은 데이터베이스 미러링의 엔터프라이즈 수준 대안으로서 고가용성과 재해 복구를 제공합니다.

AWS DMS에서는 단일 기본 또는 보조 가용성 그룹 복제본에서 변경 내용을 마이그레이션할 수 있습니다.

기본 가용성 그룹 복제본 사용

에서 기본 가용성 그룹을 원본으로 사용하려면 다음과 같이 하십시오. AWS DMS

1. 가용성 복제본의 모든 SQL Server 인스턴스에서 배포 옵션을 켭니다. 자세한 설명은 자체 관리형 SQL Server에서 지속적 복제 설정 섹션을 참조하세요.
2. AWS DMS 콘솔에서 SQL Server 원본 데이터베이스 설정을 엽니다. 서버 이름에서 가용성 그룹 리스너에 맞게 구성된 DNS(Domain Name Service) 이름이나 IP 주소를 지정합니다.

AWS DMS 작업을 처음 시작하는 경우 시작하는 데 평소보다 시간이 더 걸릴 수 있습니다. 이렇게 느려지는 이유는 가용성 그룹 서버가 테이블 항목 생성을 복제하고 있기 때문입니다.

보조 가용성 그룹 복제본 사용

에서 보조 가용성 그룹을 원본으로 사용하려면 다음과 같이 하십시오. AWS DMS

1. 개별 복제본에 연결할 때는 AWS DMS 원본 엔드포인트 사용자가 사용하는 것과 동일한 자격 증명을 사용하십시오.
2. AWS DMS 복제 인스턴스가 모든 기존 복제본의 DNS 이름을 확인하고 연결할 수 있는지 확인하십시오. 다음 SQL 쿼리를 사용하여 모든 복제본의 DNS 이름을 가져올 수 있습니다.

```
select ar.replica_server_name, ar.endpoint_url from sys.availability_replicas ar
JOIN sys.availability_databases_cluster adc
ON adc.group_id = ar.group_id AND adc.database_name = '<source_database_name>';
```

3. 소스 엔드포인트를 만들 때 엔드포인트의 서버 이름 또는 엔드포인트 암호의 서버 주소에 가용성 그룹 리스너의 DNS 이름을 지정합니다. 가용성 그룹 리스너에 대한 자세한 내용은 SQL Server 설명서의 What is an availability group listner?를 참조하세요.

   공용 DNS 서버 또는 온프레미스 DNS 서버를 사용하여 가용성 그룹 리스너, 기본 복제본, 보조 복제본을 확인할 수 있습니다. 온프레미스 DNS 서버를 사용하려면 Amazon Route 53 Resolver를 구성하세요. 자세한 설명은 자체 온프레미스 이름 서버 사용 섹션을 참조하세요.
4. 다음 추가 연결 속성을 소스 엔드포인트에 추가합니다.

추가 연결 속성	값	설명
applicationIntent	ReadOnly	이 ODBC 설정이 없으면 복제 작업이 기본 가용성 그룹 복제본으로 라우팅됩니다. 자세한 내용은 SQL Server 설명서의 SQL Server Native Client Support for High Availability, Disaster Recovery을 참조하세요.
multiSubnetFailover	yes	자세한 내용은 SQL Server 설명서의 SQL Server Native Client Support for High Availability, Disaster Recovery을 참조하세요.
alwaysOnSharedSynchedBackupIsEnabled	false	자세한 설명은 SQL Server를 원본으로 사용할 때의 엔드포인트 설정 AWS DMS 섹션을 참조하세요.
activateSafeguard	false	자세한 내용은 제한 사항 단원을 참조하십시오.
setUpMsCdcForTables	false	자세한 내용은 제한 사항 단원을 참조하십시오.

5. 가용성 그룹의 모든 복제본에서 배포 옵션을 활성화합니다. 배포자 목록에 모든 노드를 추가합니다. 자세한 설명은 배포를 설정하려면 섹션을 참조하세요.
6. 기본 읽기-쓰기 복제본에서 다음 쿼리를 실행하여 데이터베이스를 게시할 수 있도록 합니다. 이 쿼리는 데이터베이스에 대해 한 번만 실행합니다.

```
sp_replicationdboption @dbname = N'<source DB name>', @optname = N'publish', @value
 = N'true';
```

제한 사항

다음은 보조 가용성 그룹 복제본 작업의 제한 사항입니다.

- AWS DMS 읽기 전용 가용성 그룹 복제본을 원본으로 사용하는 경우 Safeguard를 지원하지 않습니다. 자세한 설명은 SQL Server를 원본으로 사용할 때의 엔드포인트 설정 AWS DMS 섹션을 참조하세요.
- AWS DMS 읽기 전용 가용성 그룹 복제본을 원본으로 사용하는 경우 `setUpMsCdcForTables` 추가 연결 속성을 지원하지 않습니다. 자세한 설명은 SQL Server를 원본으로 사용할 때의 엔드포인트 설정 AWS DMS 섹션을 참조하세요.
- AWS DMS 자체 관리되는 보조 가용성 그룹 복제본을 버전 3.4.7부터 지속적인 복제 (변경 데이터 캡처 또는 CDC) 를 위한 원본 데이터베이스로 사용할 수 있습니다. Cloud SQL Server 다중 AZ 읽기 전용 복제본은 지원되지 않습니다. 이전 버전의 AWS DMS 을 사용하는 경우 주 가용성 그룹 복제본을 CDC의 원본 데이터베이스로 사용해야 합니다.

다른 노드로의 장애 조치

엔드포인트의 `ApplicationIntent` 추가 연결 속성을 로 설정하면 AWS DMS 작업은 읽기 전용 `ReadOnly` 라우팅 우선 순위가 가장 높은 읽기 전용 노드에 연결됩니다. 우선 순위가 가장 높은 읽기 전용 노드를 사용할 수 없는 경우, 가용성 그룹에 있는 다른 읽기 전용 노드로 장애 조치됩니다. 설정하지 않으면 AWS DMS 작업은 `ApplicationIntent` 가용성 그룹의 기본 (읽기/쓰기) 노드에만 연결됩니다.

SQL Server를 원본으로 사용할 때의 보안 요구 사항 AWS Database Migration Service

AWS DMS 사용자 계정에는 연결하려는 원본 SQL Server 데이터베이스의 `db_owner` 사용자 역할이 최소한 있어야 합니다.

SQL Server를 원본으로 사용할 때의 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 SQL Server 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 의 `create-endpoint` 명령을 `--microsoft-sql-server-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 원본 엔드포인트를 생성할 때 설정을 지정합니다. AWS CLI

SQL Server를 소스로 할 때 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

명칭	설명
`ActivateSafeguard`	이 속성은 Safeguard를 켜거나 끕니다. Safeguard에 대한 자세한 내용은 다음 `SafeguardPolicy` 를 참조하세요. 기본 값: `true`

명칭	설명
	유효값: {false, true} 예제: '{"ActivateSafeguard": true}'
AlwaysOnSharedSynchedBackupIsEnabled	이 속성은 Always On 가용성 그룹 클러스터의 일부로 호스팅되는 SQL Server 원본 데이터베이스에서 마이그레이션할 AWS DMS 때의 동작을 조정합니다. AWS DMS Always On 클러스터에서 실행되도록 구성된 SQL Server 원본 데이터베이스에 대한 지원이 향상되었습니다. 이 경우 AWS DMS 는 소스 데이터베이스 인스턴스가 호스팅되는 노드가 아닌 Always On 클러스터의 노드에서 트랜잭션 백업이 실행되고 있는지 추적하려고 시도합니다. 마이그레이션 작업을 시작할 때 클러스터의 각 노드에 연결을 AWS DMS 시도하지만 노드 중 하나에 연결할 수 없으면 연결이 실패합니다. 트랜잭션 백업을 AWS DMS 위해 Always On 클러스터의 모든 노드를 폴링해야 하는 경우 이 속성을 로 false 설정하십시오. 기본 값: true 유효값: true 또는 false 예제: '{"AlwaysOnSharedSynchedBackupIsEnabled": false}'
"ApplicationIntent": "readonly"	SQL Server는 이 ODBC 드라이버 속성 설정을 통해 우선순위가 가장 높은 읽기 전용 노드로 복제 작업을 라우팅합니다. 이 설정이 없으면 SQL Server는 복제 작업을 기본 읽기-쓰기 노드로 라우팅합니다.

명칭	설명
EnableNonSysadminWrapper	sysadmin 사용자 없이 독립 실행형 SQL 서버에서 지속적 복제를 설정하는 경우, 이 엔드포인트 설정을 사용하세요. 이 매개변수는 AWS DMS 버전 3.4.7 이상에서 지원됩니다. 독립 실행형 SQL Server에서 지속적 복제를 설정하는 방법에 대한 자세한 내용은 독립 실행형 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음 섹션을 참조하세요. 기본 값: false 유효값: true, false 예제: '{"EnableNonSysadminWrapper": true}'
ExecuteTimeout	이 추가 연결 속성(ECA)을 사용하여 SQL Server 인스턴스의 클라이언트 문 제한 시간을 초 단위로 설정합니다. 기본값은 60초입니다. 예제: '{"ExecuteTimeout": 100}'
FatalOnSimpleModel	이 설정을 true로 설정하면 SQL Server 데이터베이스 복구 모델이 simple로 설정된 경우 치명적 오류가 발생합니다. 기본 값: false 유효값: true 또는 false 예제: '{"FatalOnSimpleModel": true}'
ForceLobLookup	인라인 LOB에서 LOB 조회를 강제 실행합니다. 기본 값: false 유효값: true, false 예제: '{"ForceLobLookup": false}'

명칭	설명
`"MultiSubnetFailover": "Yes"`	이 ODBC 드라이버 속성은 가용성 그룹 장애 조치 시 DMS가 새 기본 가용 영역에 연결하는 데 도움이 됩니다. 이 속성은 연결이 끊어지거나 리스너 IP 주소가 잘못된 상황을 위해 설계되었습니다. 이러한 상황에서는 가용성 그룹 수신기와 연결된 모든 IP 주소에 연결을 AWS DMS 시도합니다.
`ReadBackupOnly`	이 속성을 사용하려면 sysadmin 권한이 필요합니다. 이 속성을 로 Y 설정하면 복제를 진행하는 동안 트랜잭션 로그 백업에서만 변경 내용을 AWS DMS 읽고 활성 트랜잭션 로그 파일에서는 읽지 않습니다. 이 파라미터를 Y로 설정하면 전체 로드 및 지속적 복제 작업 중에 활성 트랜잭션 로그 파일 증가를 제어할 수 있습니다. 하지만 약간의 소스 지연 시간이 지속적 복제에 추가될 수 있습니다 유효한 값: N 또는 Y 기본값은 N입니다. 예제: `'{"ReadBackupOnly": Y}'` 참고: RDS가 백업을 수행하는 방식 때문에 Amazon RDS SQL Server 소스 인스턴스에서는 이 파라미터가 작동하지 않습니다.

명칭	설명
SafeguardPolicy	최적의 성능을 위해 활성 트랜잭션 로그 (TLOG) 에서 읽지 않은 모든 변경 내용을 AWS DMS 캡처하려고 시도합니다. 하지만 경우에 따라 잘라내기로 인해 활성 TLOG에 읽지 않은 변경 사항 전체가 포함되지 않을 수도 있습니다. 이 경우 로그 백업에 AWS DMS 액세스하여 누락된 변경 내용을 캡처합니다. 로그 백업에 액세스할 필요성을 최소화하려면 다음 방법 중 하나를 사용하여 AWS DMS 잘림을 방지하십시오. 1. RELY_ON_SQL_SERVER_REPLICATION_AGENT (데이터베이스에서 트랜잭션 시작): 이 값이 기본값입니다. AWS DMS 이 설정을 사용하는 경우, AWS DMS 은 SQL Server Log Reader Agent가 실행 중이어야 AWS DMS 가 복제하도록 표시된 트랜잭션을 활성 TLOG에서 이동할 수 있습니다. Log Reader Agent가 실행되고 있지 않으면 활성 TLOG가 가득 차게 되어 문제를 해결할 수 있을 때까지 소스 데이터베이스가 읽기 전용 모드로 전환될 수 있습니다. 이외의 AWS DMS 목적으로 데이터베이스에서 Microsoft Replication을 사용하도록 설정해야 하는 경우 이 설정을 선택해야 합니다. 이 설정을 사용하면 라는 awsdms_truncation_safeguard 테이블을 만들어 로그 백업 읽기를 AWS DMS 최소화하고 데이터베이스에서 열린 트랜잭션을 모방하여 TLOG 잘림을 방지합니다. 이렇게 하면 데이터베이스에서 이벤트가 잘리고 5분(기본값) 동안 백업 로그로 이동되는 것을 방지할 수 있습니다. 유지 관리 작업 실패를 초래할 수 있으므로 유지 관리 계획에 테이블이 포함되지 않도록 하세요. Start Transactions 데이터베이스 옵션으로 구성된 작업이 없는 경우, 테이블을 안전하게 삭제할 수 있습니다. 2. EXCLUSIVE_AUTOMATIC_TRUNCATION (단일 작업에만 사용): 이 설정을 **sp_repldone** 사용하면 로

명칭	설명
	그 항목을 사용함으로 AWS DMS 표시하는 복제 에이전트 프로세스를 완전히 제어할 수 있습니다. `ready for truncation sp_repldone` 이 설정에서는 `RELY_ON_SQL_SERVER_REPLICATION_AGENT` (기본값) 설정과 달리 더미 트랜잭션을 사용하지 AWS DMS 않습니다. MS Replication을 원본 데이터베이스 이외의 AWS DMS 다른 용도로 사용하지 않는 경우에만 이 설정을 사용할 수 있습니다. 또한 이 설정을 사용하면 하나의 AWS DMS 작업만 데이터베이스에 액세스할 수 있습니다. 동일한 데이터베이스에 대해 병렬 AWS DMS 작업을 실행해야 하는 경우 를 사용하십시오`RELY_ON_SQL_SERVER_REPLICATION_AGENT` . • 이 설정을 사용하려면 데이터베이스에서 Log Reader Agent를 중지해야 합니다. 작업이 시작될 때 Log Reader 에이전트가 실행 중이면 AWS DMS 작업은 강제로 중지됩니다. 또는 작업을 시작하기 전에 수동으로 Log Reader Agent를 중지할 수도 있습니다. • MS-CDC에 이 방법을 사용하는 경우, MS-CDC 캡처 및 MS-CDC 정리 작업을 중지하고 비활성화해야 합니다. • 원격 배포자 컴퓨터에서 Microsoft SQL Server 마이그레이션 작업을 실행할 때는 원격 컴퓨터에 액세스할 수 AWS DMS 없으므로 이 설정을 사용할 수 없습니다. • Amazon RDS 사용자에게는 `sp_repldone` 저장 프로시저를 실행할 수 있는 액세스 권한이 없기 때문에 `EXCLUSIVE_AUTOMATIC_TRUNCATION` 는 Amazon RDS for SQL Server 소스 인스턴스에서는 작동하지 않습니다. • sysadmin 역할을 사용하지 않고 `Safeguard Policy` 를 `EXCLUSIVE_AUTOMATIC_TRUNCATION` 으로 설정하는 경우, `dmsuser` 사용자에게

명칭	설명
	dbo.syscategories 및 dbo.sysjobs 객체에 대한 권한을 부여해야 합니다. 기본 값: RELY_ON_SQL_SERVER_REPLICATION_AGENT 유효값: {EXCLUSIVE_AUTOMATIC_TRUNCATION , RELY_ON_SQL_SERVER_REPLICATION_AGENT } 예제: '{"SafeguardPolicy": "EXCLUSIVE_AUTOMATIC_TRUNCATION"}'
SetUpMsCdcForTables	이 속성은 MS-Replication이 활성화되지 않은 작업 매핑의 테이블과 소스 데이터베이스에서 MS-CDC를 켭니다. 이 값을 true로 설정하면 소스 데이터베이스에서 sp_cdc_enable_db 저장 프로시저가 실행되고, 소스 데이터베이스에서 MS-Replication이 활성화되지 않은 작업의 각 테이블에서 sp_cdc_enable_table 저장 프로시저가 실행됩니다. 배포를 켜는 방법에 대한 자세한 내용은 자체 관리형 SQL Server에서 지속적 복제 설정 섹션을 참조하세요. 유효값: {true, false} 예제: '{"SetUpMsCdcForTables": true}'
TlogAccessMode	CDC 데이터를 가져오는 데 사용되는 모드를 나타냅니다. 기본 값: PreferTlog 유효값: BackupOnly , PreferBackup , PreferTlog , TlogOnly 예제: '{"TlogAccessMode": "PreferTlog"}'
Use3rdPartyBackupDevice	이 속성이 Y로 설정된 경우, AWS DMS 는 네이티브 형식으로 생성된 타사 트랜잭션 로그 백업을 처리합니다.

SQL Server용 소스 데이터 형식

SQL Server를 원본으로 사용하는 데이터 마이그레이션은 대부분의 SQL Server 데이터 유형을 AWS DMS 지원합니다. 다음 표에는 사용 시 지원되는 SQL Server 원본 데이터 AWS DMS 형식과 데이터 유형에서의 AWS DMS 기본 매핑이 나와 있습니다.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하십시오.

AWS DMS 데이터 유형에 대한 자세한 내용은 을 참조하십시오AWS Database Migration Service에서 사용되는 데이터 형식.

SQL Server 데이터 형식	AWS DMS 데이터 유형
BIGINT	INT8
BIT	BOOLEAN
DECIMAL	NUMERIC
INT	INT4
MONEY	NUMERIC
NUMERIC(p,s)	NUMERIC
SMALLINT	INT2
SMALLMONEY	NUMERIC
TINYINT	UINT1
REAL	REAL4
FLOAT	REAL8
DATETIME	DATETIME
DATETIME2(SQL Server 2008 이상)	DATETIME
SMALLDATETIME	DATETIME

SQL Server 데이터 형식	AWS DMS 데이터 유형
날짜	날짜
TIME	TIME
DATETIMEOFFSET	WSTRING
CHAR	STRING
VARCHAR	STRING
VARCHAR(최대)	CLOB TEXT 에서 이 데이터 유형을 사용하려면 특정 작업에 CLOB 데이터 유형을 사용할 수 있도록 설정해야 합니다. AWS DMS SQL Server 테이블의 경우 SQL Server의 LOB 열 값을 변경하지 않는 UPDATE 문에 대해서도 대상의 LOB 열을 업데이트합니다. AWS DMS CDC 중에는 기본 키가 포함된 테이블에서만 CLOB 데이터 유형을 AWS DMS 지원합니다.
NCHAR	WSTRING
NVARCHAR(길이)	WSTRING

SQL Server 데이터 형식	AWS DMS 데이터 유형
NVARCHAR(최대)	NCLOB NTEXT 에서 이 데이터 유형을 사용하려면 특정 작업에 사용할 수 있도록 설정해야 합니다. AWS DMS SupportLobs Lob 지원 활성화에 대한 자세한 내용은 태스크의 소스 데이터베이스에 대한 LOB 지원 설정 AWS DMS 섹션을 참조하세요. SQL Server 테이블의 경우 SQL Server의 LOB 열 값을 변경하지 않는 UPDATE 문에 대해서도 대상의 LOB 열을 업데이트합니다. AWS DMS CDC 중에는 기본 키가 포함된 테이블에서만 CLOB 데이터 유형을 AWS DMS 지원합니다.
BINARY	BYTES
VARBINARY	BYTES
VARBINARY(최대)	BLOB IMAGE SQL Server 테이블의 경우 SQL Server의 LOB 열 값을 변경하지 않는 UPDATE 문에 대해서도 대상의 LOB 열을 업데이트합니다. AWS DMS 에서 이 데이터 형식을 사용하려면 특정 AWS DMS 작업에 BLOB 데이터 형식을 사용할 수 있도록 설정해야 합니다. AWS DMS 기본 키가 포함된 테이블에서만 BLOB 데이터 유형을 지원합니다.
TIMESTAMP	BYTES

SQL Server 데이터 형식	AWS DMS 데이터 유형
UNIQUEIDENTIFIER	STRING
HIERARCHYID	SQL Server 대상 엔드포인트에 복제할 때 HIERARCHYID를 사용합니다. 다른 모든 대상 엔드포인트에 복제할 때에는 WSTRING(250)을 사용합니다.
XML	NCLOB SQL Server 테이블의 경우 SQL Server의 LOB 열 값을 변경하지 않는 UPDATE 문에 대해서도 대상의 LOB 열을 업데이트합니다. AWS DMS 에서 이 데이터 형식을 사용하려면 특정 AWS DMS 작업에 NCLOB 데이터 형식을 사용할 수 있도록 설정해야 합니다. CDC 중에는 기본 키가 포함된 테이블에서만 NCLOB 데이터 유형을 AWS DMS 지원합니다.
GEOMETRY	이 데이터 형식을 지원하는 대상 엔드포인트에 복제할 때에는 GEOMETRY를 사용합니다. 이 데이터 형식을 지원하지 않는 대상 엔드포인트에 복제할 때에는 CLOB를 사용합니다.
GEOGRAPHY	이 데이터 형식을 지원하는 대상 엔드포인트에 복제할 때에는 GEOGRAPHY를 사용합니다. 이 데이터 형식을 지원하지 않는 대상 엔드포인트에 복제할 때에는 CLOB를 사용합니다.

AWS DMS 다음 데이터 유형의 필드를 포함하는 테이블은 지원하지 않습니다.

- CURSOR
- SQL_VARIANT

- TABLE

> **Note**
>
> 사용자 정의 데이터 형식은 그 기반 유형에 따라 지원됩니다. 예를 들어, DATETIME을 기반으로 한 사용자 정의 데이터는 DATETIME 데이터 형식으로 처리됩니다.

AWS DMS용 소스로 Microsoft Azure SQL 데이터베이스 사용

AWS DMS를 활용하면 SQL Server와 거의 동일한 방식으로 Microsoft Azure SQL 데이터베이스를 소스로 사용할 수 있습니다. AWS DMS는 온프레미스 또는 Amazon EC2 인스턴스에서 실행 중인 SQL Server에 대해 지원되는 동일한 데이터베이스 버전 목록을 소스로 지원합니다.

자세한 내용은 Microsoft SQL Server 데이터베이스를 원본으로 사용 AWS DMS 섹션을 참조하세요.

> **Note**
>
> AWS DMS에서는 Azure SQL 데이터베이스를 이용한 변경 데이터 캡처 작업(CDC)을 지원하지 않습니다.

Microsoft Azure SQL 관리형 인스턴스를 AWS DMS용 소스로 사용

AWS DMS를 활용하면 SQL Server와 거의 동일한 방식으로 Microsoft Azure SQL 관리형 인스턴스를 소스로 사용할 수 있습니다. AWS DMS는 온프레미스 또는 Amazon EC2 인스턴스에서 실행 중인 SQL Server에 대해 지원되는 동일한 데이터베이스 버전 목록을 소스로 지원합니다.

자세한 내용은 Microsoft SQL Server 데이터베이스를 원본으로 사용 AWS DMS 섹션을 참조하세요.

Microsoft Azure Database for PostgreSQL 유연한 서버를 AWS DMS용 소스로 사용

AWS DMS를 활용하면 PostgreSQL을 사용하는 것과 거의 같은 방식으로 Microsoft Azure Database for PostgreSQL 유연한 서버를 소스로 사용할 수 있습니다.

AWS DMS가 소스로 지원하는 Microsoft Azure Database for PostgreSQL 유연한 서버에 대한 내용은 출처: AWS DMS 섹션을 참조하세요.

논리적 복제 및 디코딩을 위한 Microsoft Azure Database for PostgreSQL 유연한 서버 설정

데이터베이스 마이그레이션 중에 Microsoft Azure Database for PostgreSQL 유연한 서버에서 논리적 복제 및 디코딩 기능을 사용할 수 있습니다.

논리적 디코딩을 위해 DMS는 `test_decoding` 또는 `pglogical` 플러그인 중 하나를 사용합니다. 소스 PostgreSQL 데이터베이스에서 `pglogical` 플러그인을 사용할 수 있는 경우, DMS는 `pglogical`을 사용하여 복제 슬롯을 생성합니다. 그렇지 않을 경우에는 `test_decoding` 플러그인을 사용합니다.

Microsoft Azure Database for PostgreSQL 유연한 서버를 DMS의 소스 엔드포인트로 구성하려면 다음 단계를 수행합니다.

1. 포털에서 서버 파라미터 페이지를 엽니다.
2. `wal_level` 서버 파라미터를 `LOGICAL`로 설정합니다.
3. `pglogical` 확장을 사용하려면 `shared_preload_libraries` 및 `azure.extensions` 파라미터를 `pglogical`로 설정합니다.
4. `max_replication_slots` 파라미터를 동시에 실행하려는 최대 DMS 태스크 수로 설정합니다. Microsoft Azure의 경우, 이 파라미터의 기본값은 10입니다. 이 파라미터의 최대값은 PostgreSQL 인스턴스의 사용 가능한 메모리에 따라 달라지며, 메모리의 GB당 2~8개의 복제 슬롯이 허용됩니다.
5. `max_wal_senders` 파라미터를 1보다 큰 값으로 설정합니다. `max_wal_senders` 파라미터는 실행 가능한 동시 작업 개수를 설정합니다. 기본값은 10입니다.
6. `max_worker_processes` 파라미터 값을 최소 16으로 설정합니다. 그렇지 않을 경우, 다음과 같은 오류가 발생할 수 있습니다.

   ```
   WARNING: out of background worker slots.
   ```

7. 변경 사항을 저장합니다. 서버를 재시작하여 변경 사항을 적용합니다.
8. PostgreSQL 인스턴스가 연결 리소스의 네트워크 트래픽을 허용하는지 확인합니다.
9. 아래의 명령을 사용하여 기존 사용자에게 복제 권한을 부여하거나, 복제 권한이 있는 새 사용자를 생성합니다.

   - 아래의 명령을 사용하여 기존 사용자에게 복제 권한을 부여합니다.

     ```
     ALTER USER <existing_user> WITH REPLICATION;
     ```

- 아래의 명령을 사용하여 복제 권한이 있는 새 사용자를 생성합니다.

```
CREATE USER aws_dms_user PASSWORD 'aws_dms_user_password';
GRANT azure_pg_admin to aws_dms_user;
ALTER ROLE aws_dms_user REPLICATION LOGIN;
```

PostgreSQL을 사용한 논리적 복제에 대한 자세한 내용은 아래 주제를 참조하세요.

- 논리적 복제를 사용하여 변경 데이터 캡처(CDC) 활성화
- 네이티브 CDC 시작 지점을 사용하여 PostgreSQL 소스 엔드포인트의 CDC 로드 설정
- Azure Database for PostgreSQL 설명서의 Logical replication and logical decoding in Azure Database for PostgreSQL - Flexible Server.

Microsoft Azure Database for MySQL 유연한 서버를 AWS DMS용 소스로 사용

AWS DMS를 활용하면 MySQL을 사용하는 것과 거의 같은 방식으로 Microsoft Azure Database for MySQL 유연한 서버를 소스로 사용할 수 있습니다.

AWS DMS가 소스로 지원하는 Microsoft Azure Database for MySQL 유연한 서버에 대한 내용은 출처: AWS DMS 섹션을 참조하세요.

AWS DMS에서 고객 관리형 MySQL 호환 데이터베이스를 사용하는 방법에 대한 자세한 내용은 AWS DMS용 소스로 자체 관리형 MySQL 호환 데이터베이스 사용 섹션을 참조하세요.

Azure MySQL를 AWS Database Migration Service용 소스로 사용 시 제한 사항

- Azure MySQL 유연한 서버 시스템 변수의 기본값 `sql_generate_invisible_primary_key`는 `ON`이며, 서버는 보이지 않게 생성된 프라이머리 키(GIPK)를 명시적인 프라이머리 키 없이 생성되는 모든 테이블에 자동으로 추가합니다. AWS DMS는 GIPK 제약 조건이 있는 MySQL 테이블에 대해서는 지속적인 복제를 지원하지 않습니다.

OCI MySQL Heatwave를 AWS DMS용 소스로 사용

AWS DMS를 활용하면 MySQL을 사용하는 것과 거의 같은 방식으로 OCI MySQL Heatwave를 소스로 사용할 수 있습니다. OCI MySQL Heatwave를 소스로 사용하려면 몇 가지 추가 구성을 변경해야 합니다.

AWS DMS가 소스로 지원하는 OCI MySQL Heatwave 버전에 대한 자세한 내용은 출처: AWS DMS 섹션을 참조하세요.

논리적 복제를 위한 OCI MySQL Heatwave 설정

OCI MySQL Heatwave 인스턴스를 DMS의 소스 엔드포인트로 구성하려면 다음을 수행합니다.

1. OCI 콘솔에 로그인하고 왼쪽 상단의 기본 햄버거 메뉴(≡)를 엽니다.
2. 데이터베이스, DB 시스템을 선택합니다.
3. 구성 메뉴를 엽니다.
4. Create configuration(구성 생성)을 선택합니다.
5. 구성 이름(예: **`dms_configuration`**)을 입력합니다.
6. 현재 OCI MySQL Heatwave 인스턴스의 형태를 선택합니다. DB 시스템 구성:형태 섹션 아래에 있는 인스턴스의 DB 시스템 구성 속성 탭에서 형태를 찾을 수 있습니다.
7. 시스템 변수 섹션에서 `binlog_row_value_options` 시스템 변수를 선택합니다. 기본값은 `PARTIAL_JSON`입니다. 값을 지웁니다.
8. 생성 버튼을 선택합니다.
9. OCI MySQLHeatwave 인스턴스를 열고 편집 버튼을 선택합니다.
10. 구성 섹션에서 구성 변경 버튼을 선택하고, 4단계에서 생성한 형태 구성을 선택합니다.
11. 변경 사항이 적용되면 인스턴스는 논리적 복제를 수행할 준비가 완료됩니다.

Google Cloud for MySQL을 AWS DMS용 소스로 사용

AWS DMS를 활용하면 MySQL을 사용하는 것과 거의 같은 방식으로 Google Cloud for MySQL을 소스로 사용할 수 있습니다.

AWS DMS가 소스로 지원하는 GCP MySQL 버전에 대한 자세한 내용은 출처: AWS DMS 섹션을 참조하세요.

자세한 내용은 AWS DMS에서 MySQL 호환 데이터베이스를 소스로 사용 섹션을 참조하세요.

> **Note**
>
> GCP MySQL 8.0을 소스로 지원하는 기능은 AWS DMS 버전 3.4.6에서 제공됩니다.
> AWS DMS는 GCP for MySQL에 SSL 모드 `verify-full`을 지원하지 않습니다.

GCP MySQL 보안 설정 `Allow only SSL connections`를 사용하려면 서버 및 클라이언트 인증서 확인이 모두 필요하므로, 해당 설정은 지원되지 않습니다. AWS DMS는 서버 인증서 확인만 지원합니다.

AWS DMS는 `binlog_checksum` 데이터베이스 플래그에 대해 GCP CloudSQL for MySQL의 기본값 `CRC32`를 지원합니다.

Google Cloud for PostgreSQL을 AWS DMS용 소스로 사용

AWS DMS를 활용하면 자체 관리형 PostgreSQL 데이터베이스를 사용하는 것과 거의 같은 방식으로 Google Cloud for PostgreSQL을 소스로 사용할 수 있습니다.

AWS DMS가 소스로 지원하는 GCP PostgreSQL 버전에 관한 자세한 내용은 출처: AWS DMS 섹션을 참조하세요.

자세한 내용은 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용 섹션을 참조하세요.

논리적 복제 및 디코딩을 위해 Google Cloud for PostgreSQL 설정

데이터베이스 마이그레이션 중에 Google Cloud SQL for PostgreSQL에서 논리적 복제 및 디코딩 기능을 사용할 수 있습니다.

논리적 디코딩을 위해 DMS는 다음 플러그인 중 하나를 사용합니다.

- `test_decoding`
- `pglogical`

소스 PostgreSQL 데이터베이스에서 `pglogical` 플러그인을 사용할 수 있는 경우, DMS는 `pglogical`을 사용하여 복제 슬롯을 생성합니다. 그렇지 않을 경우에는 `test_decoding` 플러그인을 사용합니다.

AWS DMS로 논리적 디코딩을 사용하는 방법은 다음 내용을 참조하세요.

1. Google Cloud SQL for PostgreSQL을 사용할 경우, `cloudsql.logical_decoding` 플래그를 on으로 설정하여 논리적 디코딩을 활성화합니다.
2. `pglogical`을 활성화하려면 `cloudsql.enable_pglogical` 플래그를 on으로 설정하고, 데이터베이스를 다시 시작합니다.

3. 논리적 디코딩 기능을 사용하려면 REPLICATION 속성을 보유한 PostgreSQL 사용자를 생성합니다. pglogical 확장을 사용할 경우, 사용자에게 cloudsqlsuperuser 역할이 있어야 합니다. cloudsqlsuperuser 역할이 있는 사용자를 생성하려면 다음을 수행합니다.

   ```
   CREATE USER new_aws_dms_user WITH REPLICATION
   IN ROLE cloudsqlsuperuser LOGIN PASSWORD 'new_aws_dms_user_password';
   ```

   기존 사용자에 대해 이 속성을 설정하려면 다음 작업을 수행합니다.

   ```
   ALTER USER existing_user WITH REPLICATION;
   ```

4. max_replication_slots 파라미터를 동시에 실행하려는 최대 DMS 태스크 수로 설정합니다. Google Cloud SQL에서 이 파라미터의 기본값은 10입니다. 이 파라미터의 최대값은 PostgreSQL 인스턴스의 사용 가능한 메모리에 따라 달라지며, 메모리의 GB당 2~8개의 복제 슬롯이 허용됩니다.

PostgreSQL을 사용한 논리적 복제에 대한 자세한 내용은 아래 주제를 참조하세요.

- 논리적 복제를 사용하여 변경 데이터 캡처(CDC) 활성화
- 네이티브 CDC 시작 지점을 사용하여 PostgreSQL 소스 엔드포인트의 CDC 로드 설정
- Cloud SQL for PostgreSQL 설명서의 논리 복제 및 디코딩 설정.

PostgreSQL 데이터베이스를 AWS DMS 소스로 사용

AWS DMS를 사용하여 하나 이상의 PostgreSQL 데이터베이스에서 데이터를 마이그레이션할 수 있습니다. PostgreSQL 데이터베이스를 소스로 사용하여 다른 PostgreSQL 데이터베이스나 지원되는 다른 데이터베이스 중 하나로 데이터를 마이그레이션할 수 있습니다.

AWS DMS가 소스로 지원하는 PostgreSQL 버전에 관한 자세한 내용은 출처: AWS DMS 섹션을 참조하세요.

AWS DMS는 다음과 같은 유형의 데이터베이스에서 PostgreSQL을 지원합니다.

- 온프레미스 데이터베이스
- Amazon EC2 인스턴스의 데이터베이스
- Amazon RDS DB 인스턴스의 데이터베이스
- Amazon Aurora PostgreSQL 호환 버전 기반 DB 인스턴스의 데이터베이스

- Amazon Aurora PostgreSQL 호환 서버리스 버전 기반 DB 인스턴스의 데이터베이스

> **Note**
>
> DMS는 전체 로드에 한해 Amazon Aurora PostgreSQL Serverless V1을 소스로 지원합니다. 하지만 Amazon Aurora PostgreSQL Serverless V2는 전체 로드, 전체 로드 및 CDC, CDC 전용 작업의 소스로 사용할 수 있습니다.

사용할 AWS DMS 버전
사용 가능한 모든 AWS DMS 버전을 사용합니다.
AWS DMS 3.4.3 이상 버전을 사용합니다.
AWS DMS 3.4.7 이상 버전을 사용합니다.
AWS DMS 3.5.1 이상 버전을 사용합니다.

Secure Sockets Layer(SSL)을 사용하여 PostgreSQL 엔드포인트와 복제 인스턴스 간 연결을 암호화할 수 있습니다. PostgreSQL 엔드포인트에서 SSL을 사용하는 방법에 대한 자세한 내용은 AWS Database Migration Service에서 SSL 사용 섹션을 참조하십시오.

PostgreSQL을 소스로 사용 시 추가 보안 요구 사항으로, 지정된 사용자 계정은 PostgreSQL 데이터베이스에서 등록된 사용자여야 합니다.

PostgreSQL 데이터베이스를 AWS DMS 소스 엔드포인트로 구성하려면 다음을 수행합니다.

- PostgreSQL 소스 데이터베이스에 대한 AWS DMS 액세스를 제공할 수 있는 적절한 권한을 가진 PostgreSQL 사용자를 생성합니다.

  > **Note**
  >
  > - PostgreSQL 소스 데이터베이스가 자체 관리형인 경우, 자세한 내용은 자체 관리형 PostgreSQL 데이터베이스를 AWS DMS에서 소스로 사용 섹션을 참조하세요.
  > - PostgreSQL 소스 데이터베이스를 Amazon RDS가 관리하는 경우, 자세한 내용은 AWS 관리형 PostgreSQL 데이터베이스를 DMS 소스로 사용 섹션을 참조하세요.

- 선택한 PostgreSQL 데이터베이스 구성을 준수하는 PostgreSQL 소스 엔드포인트를 생성합니다.
- 작업 또는 작업 세트를 생성하여 테이블을 마이그레이션합니다.

  full-load-only 태스크를 생성하기 위해 추가 엔드포인트 구성은 필요하지 않습니다.

  변경 데이터 캡처 작업(CDC 전용 또는 전체 로드 및 CDC 작업)을 생성하기 전에 자체 관리형 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용하여 CDC 활성화 또는 AWS DMS를 사용하여 AWS 관리형 PostgreSQL DB 인스턴스에서 CDC 활성화 섹션을 참조하세요.

주제

- 자체 관리형 PostgreSQL 데이터베이스를 AWS DMS에서 소스로 사용
- AWS 관리형 PostgreSQL 데이터베이스를 DMS 소스로 사용
- 논리적 복제를 사용하여 변경 데이터 캡처(CDC) 활성화
- 네이티브 CDC 시작 지점을 사용하여 PostgreSQL 소스 엔드포인트의 CDC 로드 설정
- AWS DMS를 사용하여 PostgreSQL에서 PostgreSQL로 마이그레이션,
- PostgreSQL 소스 데이터베이스에서 AWS DMS 아티팩트 제거
- PostgreSQL 데이터베이스를 DMS 소스로 사용 시 추가 구성 설정
- MapBooleanAsBoolean PostgreSQL 엔드포인트 설정 사용
- PostgreSQL을 DMS 소스로 사용하는 경우의 엔드포인트 설정 및 추가 연결 속성 (ECA)
- PostgreSQL 데이터베이스를 DMS 소스로 사용 시 제한 사항
- PostgreSQL용 소스 데이터 형식

자체 관리형 PostgreSQL 데이터베이스를 AWS DMS에서 소스로 사용

자체 관리형 PostgreSQL 데이터베이스를 소스로 사용하여 다른 PostgreSQL 데이터베이스나 AWS DMS이 지원하는 다른 대상 데이터베이스 중 하나로 데이터를 마이그레이션할 수 있습니다. 온프레미스 데이터베이스나 Amazon EC2 인스턴스에서 실행되는 자체 관리형 엔진이 데이터베이스 소스가 될 수 있습니다. 전체 로드 작업 및 변경 데이터 캡처(CDC) 작업 모두에 DB 인스턴스를 사용할 수 있습니다.

PostgreSQL 데이터베이스를 AWS DMS 소스로 사용하기 위한 사전 요구 사항

자체 관리형 PostgreSQL 소스 데이터베이스에서 데이터를 마이그레이션하기 전에 다음을 수행합니다.

- 버전 9.4.x 이상의 PostgreSQL 데이터베이스를 사용해야 합니다.
- 전체 로드 및 변경 데이터 캡처(CDC) 작업 또는 CDC 전용 작업의 경우, PostgreSQL 소스 데이터베이스에 지정된 사용자 계정에 슈퍼유저 권한을 부여합니다. 사용자 계정이 소스의 복제 관련 함수에 액세스하려면 슈퍼유저 권한이 필요합니다. 전체 로드 전용 작업의 경우, 테이블을 마이그레이션하려면 사용자 계정에 SELECT 권한이 필요합니다.
- pg_hba.conf 구성 파일에 AWS DMS 복제 서버의 IP 주소를 추가하고 복제 및 소켓 연결을 활성화합니다. 예를 들면 다음과 같습니다.

```
# Replication Instance
host all all 12.3.4.56/00 md5
# Allow replication connections from localhost, by a user with the
# replication privilege.
host replication dms 12.3.4.56/00 md5
```

  PostgreSQL의 pg_hba.conf 구성 파일은 클라이언트 인증을 제어합니다. (HBA는 호스트 기반 인증을 의미합니다.) 이 파일은 일반적으로 데이터베이스 클러스터의 데이터 디렉터리에 저장됩니다.
- AWS DMS를 사용하여 데이터베이스를 논리적 복제를 위한 소스로 구성하는 경우, 자체 관리형 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용하여 CDC 활성화 섹션을 참조하세요.

> **Note**
>
> 일부 AWS DMS 트랜잭션은 DMS 엔진에서 다시 사용하기 전까지 얼마 동안 유휴 상태입니다. PostgreSQL 버전 9.6 이상에서 idle_in_transaction_session_timeout 파라미터를 사

용하면 유휴 트랜잭션이 시간 초과되어 실패하도록 할 수 있습니다. AWS DMS를 사용할 때는 유휴 트랜잭션을 종료하지 마십시오.

자체 관리형 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용하여 CDC 활성화

AWS DMS는 논리적 복제를 사용한 CDC(변경 데이터 캡처)를 지원합니다. 자체 관리형 PostgreSQL 소스 데이터베이스의 논리적 복제를 활성화하려면 `postgresql.conf` 구성 파일에서 다음 파라미터와 값을 설정합니다.

- `wal_level = logical`을 설정합니다.
- `max_replication_slots`를 1보다 큰 값으로 설정합니다.

  실행할 작업 개수에 따라 `max_replication_slots` 값을 설정합니다. 예를 들어, 5개 작업을 실행하려면 최소 5개의 슬롯을 설정해야 합니다. 작업이 시작된 즉시 슬롯이 자동으로 열리고 작업이 더 이상 실행되지 않더라도 계속 열려 있습니다. 열린 슬롯은 반드시 수동으로 삭제하세요. 참고로 DMS에서 슬롯을 생성한 경우 작업이 삭제되면 DMS가 복제 슬롯을 자동으로 삭제합니다.
- `max_wal_senders`를 1보다 큰 값으로 설정합니다.

  `max_wal_senders` 파라미터는 실행 가능한 동시 작업 개수를 설정합니다.
- `wal_sender_timeout` 파라미터는 지정된 밀리초보다 오래 사용되지 않는 복제 연결을 종료합니다. 온프레미스 PostgreSQL 데이터베이스의 기본값은 60,000밀리초(60초)입니다. 값을 0으로 설정하면 시간 제한 메커니즘이 비활성화되며, 이는 유효한 DMS 설정입니다.

  `wal_sender_timeout`을 0이 아닌 값으로 설정하는 경우, CDC가 있는 DMS 작업에는 최소 10,000밀리초(10초)가 필요하며, 값이 10,000보다 작으면 작업이 실패합니다. DMS 복제 인스턴스의 다중 AZ 장애 조치 중에 지연이 발생하지 않도록 하려면 값을 5분 미만으로 유지하세요.

일부 파라미터는 정적이며 서버 시작 시에만 설정할 수 있습니다. 구성 파일(자체 관리형 데이터베이스의 경우) 또는 DB 파라미터 그룹(RDS for PostgreSQL 데이터베이스의 경우)의 항목에 대한 모든 변경 사항은 서버를 다시 시작할 때까지 무시됩니다. 자세한 내용은 PostgreSQL 설명서를 참조하세요.

CDC 활성화에 대한 자세한 내용은 논리적 복제를 사용하여 변경 데이터 캡처(CDC) 활성화 섹션을 참조하세요.

AWS 관리형 PostgreSQL 데이터베이스를 DMS 소스로 사용

AWS 관리형 PostgreSQL DB 인스턴스를 AWS DMS 소스로 사용할 수 있습니다. AWS 관리형 PostgreSQL 소스를 사용하여 전체 로드 작업과 변경 데이터 캡처(CDC) 작업을 모두 수행할 수 있습니다.

AWS 관리형 PostgreSQL 데이터베이스를 DMS 소스로 사용하기 위한 사전 요구 사항

AWS 관리형 PostgreSQL 소스 데이터베이스에서 데이터를 마이그레이션하기 전에 다음을 수행합니다.

- PostgreSQL DB 인스턴스에 필요한 최소 권한이 있는 AWS 사용자 계정을 AWS DMS의 PostgreSQL 소스 엔드포인트 사용자 계정으로 사용하는 것이 좋습니다. 마스터 계정을 사용하는 것은 권장되지 않습니다. 계정에는 `rds_superuser` 역할과 `rds_replication` 역할이 있어야 합니다. `rds_replication` 역할은 논리적 슬롯을 관리하고 논리적 슬롯을 사용하여 데이터를 스트리밍할 수 있는 권한을 부여합니다.

  사용하는 계정의 마스터 사용자 계정에서 여러 객체를 생성해야 합니다. 객체 생성에 대한 자세한 내용은 마스터 사용자 계정을 사용하지 않고 Amazon RDS for PostgreSQL 데이터베이스 마이그레이션 섹션을 참조하세요.
- 소스 데이터베이스가 Virtual Private Cloud(VPC)에 있는 경우, 데이터베이스가 상주하는 DB 인스턴스에 대한 액세스 권한을 부여하는 VPC 보안 그룹을 선택합니다. 이는 DMS 복제 인스턴스를 소스 DB 인스턴스에 성공적으로 연결하는 데 필요합니다. 데이터베이스와 DMS 복제 인스턴스가 동일한 VPC에 있는 경우, 적절한 보안 그룹을 자체 인바운드 규칙에 추가합니다.

> **Note**
>
> 일부 AWS DMS 트랜잭션은 DMS 엔진에서 다시 사용하기 전까지 얼마 동안 유휴 상태입니다. PostgreSQL 버전 9.6 이상에서 `idle_in_transaction_session_timeout` 파라미터를 사용하면 유휴 트랜잭션이 시간 초과되어 실패하도록 할 수 있습니다. AWS DMS를 사용할 때는 유휴 트랜잭션을 종료하지 마십시오.

AWS DMS를 사용하여 AWS 관리형 PostgreSQL DB 인스턴스에서 CDC 활성화

DB 인스턴스가 논리적 복제를 사용하도록 구성된 경우, AWS DMS는 Amazon RDS PostgreSQL 데이터베이스에서 CDC를 지원합니다. 다음 표에는 각 AWS 관리형 PostgreSQL 버전의 논리적 복제 호환성이 요약되어 있습니다.

RDS for PostgreSQL 읽기 전용 복제본은 CDC(지속적 복제)에 사용할 수 없습니다.

PostgreSQL 버전	AWS DMS 전체 로드 지원	AWS DMS CDC 지원
PostgreSQL 10.5(또는 이하)와 호환되는 Aurora PostgreSQL 버전 2.1	예	아니요
PostgreSQL 10.6(또는 이상)과 호환되는 Aurora PostgreSQL 버전 2.2	예	예
PostgreSQL 10.21(또는 이상)과 호환되는 RDS for PostgreSQL	예	예

RDS PostgreSQL DB 인스턴스에 대한 논리적 복제를 활성화하려면

1. PostgreSQL DB 인스턴스의 AWS 마스터 사용자 계정을 PostgreSQL 소스 엔드포인트의 사용자 계정으로 사용합니다. 마스터 사용자 계정에는 CDC 설정을 허용하는 데 필요한 역할이 있습니다.

   마스터 사용자 계정 이외의 계정을 사용하는 경우, 사용하는 계정의 마스터 사용자 계정에서 여러 객체를 생성해야 합니다. 자세한 설명은 마스터 사용자 계정을 사용하지 않고 Amazon RDS for PostgreSQL 데이터베이스 마이그레이션 섹션을 참조하세요.
2. DB 클러스터 파라미터 그룹의 `rds.logical_replication` 파라미터를 1로 설정합니다. 이 정적 파라미터를 적용하려면 DB 인스턴스를 재부팅해야 합니다. 이 파라미터를 적용하는 중에 AWS DMS에서는 `wal_level`, `max_wal_senders`, `max_replication_slots`, `max_connections` 파라미터를 설정합니다. 이러한 파라미터 변경은 WAL(Write Ahead Log) 생성을 강화하므로 논리적 복제 슬롯을 사용할 때 `rds.logical_replication`만 설정하면 됩니다.
3. `wal_sender_timeout` 파라미터는 지정된 밀리초보다 오래 사용되지 않는 복제 연결을 종료합니다. AWS 관리형 PostgreSQL 데이터베이스의 기본값은 30,000밀리초(30초)입니다. 값을 0으로 설정하면 시간 제한 메커니즘이 비활성화되며, 이는 유효한 DMS 설정입니다.

   `wal_sender_timeout`을 0이 아닌 값으로 설정하는 경우, CDC가 있는 DMS 작업에는 최소 10,000밀리초(10초)가 필요하며, 값이 0에서 10,000 사이이면 작업이 실패합니다. DMS 복제 인스턴스의 다중 AZ 장애 조치 중에 지연이 발생하지 않도록 하려면 값을 5분 미만으로 유지하세요.

4. DB 클러스터 파라미터 그룹의 max_worker_processes 파라미터 값이 max_logical_replication_workers, autovacuum_max_workers, max_parallel_workers의 총 합계 값과 같거나 더 큰지 확인합니다. 백그라운드 작업자 프로세스 수가 많으면 소규모 인스턴스의 애플리케이션 워크로드에 영향을 미칠 수 있습니다. 따라서 max_worker_processes를 기본값보다 높게 설정한 경우, 데이터베이스의 성능을 모니터링하세요.

마스터 사용자 계정을 사용하지 않고 Amazon RDS for PostgreSQL 데이터베이스 마이그레이션

경우에 따라 소스로 사용 중인 Amazon RDS for PostgreSQL DB 인스턴스에 마스터 사용자 계정을 사용하지 못할 수 있습니다. 이러한 경우, 데이터 정의 언어(DDL) 이벤트를 캡처하기 위해 몇 개의 객체를 생성합니다. 마스터 계정 이외의 계정에서 이 객체를 생성하고 나서 마스터 사용자 계정에서 트리거를 생성합니다.

> **Note**
>
> 소스 엔드포인트에서 captureDDLs 엔드포인트 설정을 false로 설정하는 경우, 소스 데이터베이스에 다음 테이블 및 트리거를 생성할 필요가 없습니다.

다음 절차에 따라 이 객체를 생성합니다.

객체를 생성하려면

1. 객체가 생성되는 위치에 있는 스키마를 선택합니다. 기본 스키마는 public입니다. 스키마가 있는지와 이 스키마를 *OtherThanMaster* 계정으로 액세스할 수 있는지 확인합니다.
2. 마스터 계정이 아닌 사용자 계정(여기서는 *OtherThanMaster* 계정)을 사용하여 PostgreSQL DB 인스턴스에 로그인합니다.
3. 다음 명령을 실행하여 테이블 awsdms_ddl_audit를 생성하고, 다음 코드의 *objects_schema*를 사용할 스키마 이름으로 대체합니다.

```
CREATE TABLE objects_schema.awsdms_ddl_audit
(
  c_key    bigserial primary key,
  c_time   timestamp,    -- Informational
  c_user   varchar(64),  -- Informational: current_user
  c_txn    varchar(16),  -- Informational: current transaction
```

```
  c_tag    varchar(24),  -- Either 'CREATE TABLE' or 'ALTER TABLE' or 'DROP TABLE'
  c_oid    integer,      -- For future use - TG_OBJECTID
  c_name   varchar(64),  -- For future use - TG_OBJECTNAME
  c_schema varchar(64),  -- For future use - TG_SCHEMANAME. For now - holds
 current_schema
  c_ddlqry  text         -- The DDL query associated with the current DDL event
);
```

4. 다음 코드의 awsdms_intercept_ddl를 사용할 스키마의 이름으로 바꿔서 다음 명령을 실행하여 *objects_schema* 함수를 생성합니다.

```
CREATE OR REPLACE FUNCTION objects_schema.awsdms_intercept_ddl()
  RETURNS event_trigger
LANGUAGE plpgsql
SECURITY DEFINER
  AS $$
  declare _qry text;
BEGIN
  if (tg_tag='CREATE TABLE' or tg_tag='ALTER TABLE' or tg_tag='DROP TABLE' or
 tg_tag = 'CREATE TABLE AS') then
         SELECT current_query() into _qry;
         insert into objects_schema.awsdms_ddl_audit
         values
         (
         default,current_timestamp,current_user,cast(TXID_CURRENT()as
 varchar(16)),tg_tag,0,'',current_schema,_qry
         );
         delete from objects_schema.awsdms_ddl_audit;
end if;
END;
$$;
```

5. *OtherThanMaster* 계정에서 로그아웃하고 rds_superuser 역할이 할당된 계정으로 로그인합니다.
6. 다음 명령을 실행하여 이벤트 트리거 awsdms_intercept_ddl을 생성합니다.

```
CREATE EVENT TRIGGER awsdms_intercept_ddl ON ddl_command_end
EXECUTE PROCEDURE objects_schema.awsdms_intercept_ddl();
```

7. 이러한 이벤트에 액세스하는 모든 사용자와 역할에 필요한 DDL 권한이 있는지 확인합니다. 예:

```
grant all on public.awsdms_ddl_audit to public;
grant all on public.awsdms_ddl_audit_c_key_seq to public;
```

이전 절차를 완료하면, *OtherThanMaster* 계정을 사용하여 AWS DMS 소스 엔드포인트를 생성할 수 있습니다.

> **Note**
>
> 이러한 이벤트는 CREATE TABLE, ALTER TABLE, DROP TABLE 문에 의해 트리거됩니다.

논리적 복제를 사용하여 변경 데이터 캡처(CDC) 활성화

PostgreSQL의 네이티브 논리적 복제 기능을 사용하여 PostgreSQL 소스의 데이터베이스 마이그레이션 중에 변경 데이터 캡처(CDC)를 활성화할 수 있습니다. 이 기능은 자체 관리형 PostgreSQL 및 Amazon RDS for PostgreSQL SQL DB 인스턴스에 사용할 수 있습니다. 이 접근 방식은 가동 중지 시간을 줄이고 대상 데이터베이스를 소스 PostgreSQL 데이터베이스와 동기화하는 데 도움이 됩니다.

AWS DMS는 프라이머리 키가 있는 PostgreSQL 테이블의 변경 데이터 캡처(CDC)를 지원합니다. 테이블에 프라이머리 키가 없는 경우, 미리 쓰기 로그(WAL)에 데이터베이스 행의 이전 이미지가 포함되지 않습니다. 이 경우 DMS는 테이블을 업데이트할 수 없습니다. 여기서는 추가 구성 설정과 테이블 복제본 ID를 해결 방법으로 사용할 수 있습니다. 하지만 이 접근 방식을 사용하면 추가 로그가 생성될 수 있습니다. 신중하게 테스트한 후에만 테이블 복제본 ID를 해결 방법으로 사용하는 것이 좋습니다. 자세한 설명은 PostgreSQL 데이터베이스를 DMS 소스로 사용 시 추가 구성 설정 섹션을 참조하세요.

> **Note**
>
> REPLICA IDENTITY FULL은 논리적 디코딩 플러그인에서는 지원되지만 pglogical 플러그인에서는 지원되지 않습니다. 자세한 내용은 pglogical 설명서를 참조하세요.

전체 로드 및 CDC 작업과 CDC 전용 작업의 경우, AWS DMS는 논리적 복제 슬롯을 사용하여 로그가 디코딩될 때까지 복제용 WAL 로그를 보존합니다. 전체 로드 및 CDC 작업 또는 CDC 작업을 다시 시작(재개가 아님)하면 복제 슬롯이 다시 생성됩니다.

> **Note**
>
> DMS는 논리적 디코딩에 test_decoding 또는 pglogical 플러그인을 사용합니다. 소스 PostgreSQL 데이터베이스에서 pglogical 플러그인을 사용할 수 있는 경우, DMS는 pglogical을 사용하여 복제 슬롯을 생성하고, 그렇지 않으면 test_decoding 플러그인을 사용합니다. test-decoding 플러그인에 대한 자세한 내용은 PostgreSQL 설명서를 참조하세요.
> 데이터베이스 파라미터 `max_slot_wal_keep_size`가 기본값이 아닌 값으로 설정되고, 복제 슬롯의 `restart_lsn`이 현재 LSN보다 이 크기 이상 모자라는 경우, 필요한 WAL 파일이 제거되기 때문에 DMS 작업이 실패합니다.

pglogical 플러그인 구성

PostgreSQL 확장으로 구현된 pglogical 플러그인은 선택적 데이터 복제를 위한 논리적 복제 시스템 및 모델입니다. 다음 표에는 pglogical 플러그인을 지원하는 소스 PostgreSQL 데이터베이스 버전이 나와 있습니다.

PostgreSQL 소스	pglogical 지원
자체 관리형 PostgreSQL 9.4 이상	예
Amazon RDS PostgreSQL 9.5 이하	아니요
Amazon RDS PostgreSQL 9.6 이상	예
Aurora PostgreSQL 1.x~2.5.x	아니요
Aurora PostgreSQL 2.6.x 이상	예
Aurora PostgreSQL 3.3.x 이상	예

AWS DMS에 사용할 pglogical을 구성하기 전에 먼저 PostgreSQL 소스 데이터베이스에서 변경 데이터 캡처(CDC)를 위한 논리적 복제를 활성화합니다.

- 자체 관리형 PostgreSQL 소스 데이터베이스에서 CDC의 논리적 복제를 활성화하는 방법에 대한 자세한 내용은 자체 관리형 PostgreSQL 데이터베이스를 AWS DMS 소스로 사용하여 CDC 활성화 섹션을 참조하세요.

- AWS 관리형 PostgreSQL 소스 데이터베이스에서 CDC의 논리적 복제를 활성화하는 방법에 대한 자세한 내용은 AWS DMS를 사용하여 AWS 관리형 PostgreSQL DB 인스턴스에서 CDC 활성화 섹션을 참조하세요.

PostgreSQL 소스 데이터베이스에서 논리적 복제를 활성화한 후 다음 단계를 사용하여 DMS에 사용할 pglogical을 구성합니다.

AWS DMS에서 PostgreSQL 소스 데이터베이스에서의 논리적 복제에 pglogical 플러그인을 사용하려면

1. 소스 PostgreSQL 데이터베이스에서 pglogical 확장을 생성합니다.

   a. 올바른 파라미터를 설정하세요.

      - 자체 관리형 PostgreSQL 데이터베이스의 경우, 데이터베이스 파라미터 `shared_preload_libraries= 'pglogical'`을 설정합니다.
      - Amazon RDS 기반 PostgreSQL과 Amazon Aurora PostgreSQL 호환 버전 데이터베이스의 경우, 동일한 RDS 파라미터 그룹에서 `shared_preload_libraries` 파라미터를 `pglogical`로 설정합니다.

   b. PostgreSQL 소스 데이터베이스를 다시 시작합니다.

   c. PostgreSQL 데이터베이스에서 `create extension pglogical;` 명령을 실행합니다.

2. 다음 명령을 실행하여 pglogical이 성공적으로 설치되었는지 확인합니다.

   ```
   select * FROM pg_catalog.pg_extension
   ```

이제 PostgreSQL 소스 데이터베이스 엔드포인트의 변경 데이터 캡처를 수행하는 AWS DMS 작업을 생성할 수 있습니다.

> **Note**
>
> PostgreSQL 소스 데이터베이스에서 pglogical을 활성화하지 않으면 AWS DMS는 기본적으로 `test_decoding` 플러그인을 사용합니다. 논리적 디코딩을 위해 pglogical이 활성화되면 AWS DMS는 기본적으로 pglogical을 사용합니다. 하지만 `test_decoding` 플러그인을 대신 사용하도록 추가 연결 속성 `PluginName`을 설정할 수 있습니다.

네이티브 CDC 시작 지점을 사용하여 PostgreSQL 소스 엔드포인트의 CDC 로드 설정

PostgreSQL을 소스로 사용하여 네이티브 CDC 시작 지점을 활성화하려면 엔드포인트를 생성할 때 `slotName` 추가 연결 속성을 기존 논리적 복제 슬롯의 이름으로 설정합니다. 이 논리적 복제 슬롯에는 엔드포인트 생성 시점부터 지속적인 변경 사항이 포함되므로 이전 시점의 복제를 지원합니다.

PostgreSQL은 AWS DMS가 논리적 복제 슬롯에서 변경 사항을 성공적으로 읽은 후에만 삭제되는 WAL 파일에 데이터베이스 변경 사항을 기록합니다. 논리적 복제 슬롯을 사용하면 기록된 변경 사항이 복제 엔진에서 사용되기 전에 삭제되지 않도록 보호할 수 있습니다.

그러나 변경 비율과 사용 비율에 따라 논리적 복제 슬롯에 변경 사항이 포함되어 디스크 사용량이 증가할 수 있습니다. 논리적 복제 슬롯을 사용할 때 소스 PostgreSQL 인스턴스에서 공간 사용 경보를 설정하는 것이 좋습니다. `slotName` 추가 연결 속성 설정에 대한 자세한 내용은 PostgreSQL을 DMS 소스로 사용하는 경우의 엔드포인트 설정 및 추가 연결 속성 (ECA) 단원을 참조하십시오.

다음 절차에서는 이 접근 방식에 대해 자세히 설명합니다.

고유 CDC 시작 지점을 사용하여 PostgreSQL 소스 엔드포인트의 CDC 로드를 설정하려면

1. 시작 지점으로 사용할 이전 복제 작업(상위 작업)에서 사용하는 논리적 복제 슬롯을 식별합니다. 그런 다음 원본 데이터베이스에서 `pg_replication_slots` 뷰를 쿼리하여 이 슬롯에 활성 연결이 없는지 확인합니다. 활성 연결이 있다면 계속하기 전에 연결을 확인하고 닫습니다.

   다음 단계에서는 논리적 복제 슬롯이 `abc1d2efghijk_34567890_z0yx98w7_6v54_32ut_1srq_1a2b34c5d67ef`라고 가정합니다.
2. 다음 추가 연결 속성 설정을 포함하는 새 소스 엔드포인트를 만듭니다.

   ```
   slotName=abc1d2efghijk_34567890_z0yx98w7_6v54_32ut_1srq_1a2b34c5d67ef;
   ```
3. 콘솔, AWS CLI 또는 AWS DMS API를 사용하여 새 CDC 전용 작업을 만듭니다. 예를 들어 CLI를 사용하여 다음 `create-replication-task` 명령을 실행할 수 있습니다.

   ```
   aws dms create-replication-task --replication-task-identifier postgresql-slot-name-
   test
   --source-endpoint-arn arn:aws:dms:us-
   west-2:012345678901:endpoint:ABCD1EFGHIJK2LMNOPQRST3UV4
   --target-endpoint-arn arn:aws:dms:us-
   west-2:012345678901:endpoint:ZYX9WVUTSRQONM8LKJIHGF7ED6
   --replication-instance-arn arn:aws:dms:us-
   west-2:012345678901:rep:AAAAAAAAAAA5BB4CCC3DDDD2EE
   ```

```
--migration-type cdc --table-mappings "file://mappings.json" --cdc-start-position
 "4AF/B00000D0"
--replication-task-settings "file://task-pg.json"
```

위의 명령에서는 다음 옵션이 설정됩니다.

- source-endpoint-arn 옵션은 2단계에서 만든 새 값으로 설정됩니다.
- replication-instance-arn 옵션은 1단계의 상위 작업과 동일한 값으로 설정됩니다.
- table-mappings 및 replication-task-settings 옵션은 1단계의 상위 작업과 동일한 값으로 설정됩니다.
- cdc-start-position 옵션은 시작 위치 값으로 설정됩니다. 이 시작 위치를 찾으려면 원본 데이터베이스의 pg_replication_slots 뷰를 쿼리하거나 1단계에서 상위 작업에 대한 콘솔 세부 정보를 확인합니다. 자세한 설명은 CDC 기본 시작점 결정 섹션을 참조하세요.

AWS DMS 콘솔을 사용하여 새 CDC 전용 작업을 생성할 때 사용자 지정 CDC 시작 모드를 활성화하려면 다음을 수행합니다.

- 작업 설정 섹션의 소스 트랜잭션의 CDC 시작 모드에서 사용자 지정 CDC 시작 모드 활성화를 선택합니다.
- 소스 트랜잭션의 사용자 지정 CDC 시작 지점에서 로그 시퀀스 번호 지정을 선택합니다. 시스템 변경 번호를 지정하거나 복구 체크포인트 지정을 선택하고 복구 체크포인트를 제공합니다.

이 CDC 작업이 실행될 때 지정된 논리적 복제 슬롯이 없으면 AWS DMS에서 오류가 발생합니다. 또한 작업이 cdc-start-position에 올바른 설정으로 생성되지 않은 경우에도 오류가 발생합니다.

pglogical 플러그인과 함께 네이티브 CDC 시작 지점을 사용하고 새 복제 슬롯을 사용하려는 경우, CDC 작업을 생성하기 전에 다음 설정 단계를 완료하세요.

이전에 다른 DMS 작업의 일부로 생성하지 않은 새 복제 슬롯을 사용하려면

1. 다음과 같이 복제 슬롯을 생성합니다.

```
SELECT * FROM pg_create_logical_replication_slot('replication_slot_name',
 'pglogical');
```

2. 데이터베이스에서 복제 슬롯이 생성된 후 해당 슬롯의 restart_lsn 및 confirmed_flush_lsn 값을 가져와 적어 둡니다.

```
select * from pg_replication_slots where slot_name like 'replication_slot_name';
```

복제 슬롯 이후에 생성된 CDC 작업의 네이티브 CDC 시작 위치는 confirmed_flush_lsn 값보다 이전일 수 없습니다.

restart_lsn 및 confirmed_flush_lsn 값에 대한 자세한 내용은 pg_replication_slots를 참조하세요.

3. pglogical 노드를 생성합니다.

```
SELECT pglogical.create_node(node_name := 'node_name', dsn := 'your_dsn_name');
```

4. `pglogical.create_replication_set` 함수를 사용하여 두 개의 복제 세트를 생성합니다. 첫 번째 복제 세트는 프라이머리 키가 있는 테이블의 업데이트 및 삭제를 추적합니다. 두 번째 복제 세트는 삽입만 추적하며, 첫 번째 복제 세트와 같은 이름에 접두사 'i'가 추가됩니다.

```
SELECT pglogical.create_replication_set('replication_slot_name', false, true, true,
 false);
SELECT pglogical.create_replication_set('ireplication_slot_name', true, false,
 false, true);
```

5. 복제본 세트에 테이블을 추가합니다.

```
SELECT pglogical.replication_set_add_table('replication_slot_name',
 'schemaname.tablename', true);
SELECT pglogical.replication_set_add_table('ireplication_slot_name',
 'schemaname.tablename', true);
```

6. 소스 엔드포인트를 생성할 때 추가 연결 속성(ECA)을 다음과 같이 설정합니다.

```
PluginName=PGLOGICAL;slotName=slot_name;
```

이제 새 복제 슬롯을 사용하여 PostgreSQL 네이티브 시작 지점이 있는 CDC 전용 작업을 생성할 수 있습니다. pglogical 플러그인에 대한 자세한 내용은 pglogical 3.7 설명서를 참조하세요.

AWS DMS를 사용하여 PostgreSQL에서 PostgreSQL로 마이그레이션,

PostgreSQL이 아닌 데이터베이스 엔진에서 PostgreSQL 데이터베이스로 마이그레이션하는 경우, 사용할 수 있는 최고의 마이그레이션 도구는 거의 언제나 AWS DMS입니다. 하지만 PostgreSQL 데이터베이스에서 PostgreSQL 데이터베이스로 마이그레이션하는 경우, PostgreSQL 도구가 더 효과적입니다.

PostgreSQL 네이티브 도구를 사용하여 데이터 마이그레이션

다음과 같은 경우에는 pg_dump와 같은 PostgreSQL 데이터베이스 마이그레이션 도구를 사용하는 것이 좋습니다.

- 같은 유형의 마이그레이션을 수행합니다. 이 경우 PostgreSQL 데이터베이스에서 대상 PostgreSQL 데이터베이스로 마이그레이션합니다.
- 전체 데이터베이스를 마이그레이션합니다.
- 기본 도구를 사용하여 최소한의 가동 중지로 데이터를 마이그레이션할 수 있습니다.

pg_dump 유틸리티는 COPY 명령을 사용하여 PostgreSQL 데이터베이스의 스키마와 데이터 덤프를 만듭니다. pg-dump에 의해 생성되는 덤프 스크립트는 같은 이름을 가진 데이터베이스로 데이터를 로드하고 테이블, 인덱스, 외래 키를 다시 만듭니다. 데이터를 다른 이름의 데이터베이스로 복구하려면 `pg_restore` 명령과 `-d` 파라미터를 사용하세요.

EC2에서 실행되는 PostgreSQL 소스 데이터베이스에서 Amazon RDS for PostgreSQL 대상으로 데이터를 마이그레이션하는 경우, pglogical 플러그인을 사용할 수 있습니다.

PostgreSQL 데이터베이스를 Amazon RDS for PostgreSQL 또는 Amazon Aurora PostgreSQL 호환 버전으로 가져오기에 대한 자세한 내용은 https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/PostgreSQL.Procedural.Importing.html 섹션을 참조하세요.

DMS를 사용하여 PostgreSQL에서 PostgreSQL로 데이터 마이그레이션

AWS DMS는 예를 들어 온프레미스의 소스 PostgreSQL 데이터베이스에서 대상 Amazon RDS for PostgreSQL 또는 Amazon Aurora PostgreSQL 인스턴스로 데이터를 마이그레이션할 수 있습니다. 코어 또는 베이직 PostgreSQL 데이터 유형은 대부분은 성공적으로 마이그레이션됩니다.

> **Note**
>
> 파티션된 테이블을 PostgreSQL 소스에서 PostgreSQL 대상으로 복제할 때는 DMS 작업에서 선택 기준의 일부로 상위 테이블을 언급하지 않아도 됩니다. 상위 테이블을 언급하면 대상의

하위 테이블에서 데이터가 중복되어 PK 위반이 발생할 수 있습니다. 테이블 매핑 선택 기준에서 하위 테이블만 선택하면 상위 테이블이 자동으로 채워집니다.

원본 데이터베이스에서는 지원되지만 대상에서는 지원되지 않는 데이터 형식은 성공적으로 마이그레이션되지 않을 수 있습니다. AWS DMS는 데이터 형식이 알려지지 않은 일부 데이터를 문자열로 스트리밍합니다. XML 또는 JSON과 같은 어떤 데이터 유형은 작업 파일을 성공적으로 마이그레이션할 수 있지만 대형 문서의 경우 실패할 수 있습니다.

데이터 형식 마이그레이션을 수행할 때는 다음 사항에 유의하세요.

- PostgreSQL NUMERIC(p,s) 데이터 형식이 정밀도와 스케일을 지정하지 않는 경우도 있습니다. DMS 버전 3.4.2 이하 버전에서 DMS는 기본적으로 정밀도 28과 스케일 6인 NUMERIC(28,6)을 사용합니다. 예를 들어 소스의 0.611111104488373 값은 PostgreSQL 대상에서 0.611111로 변환됩니다.
- ARRAY 데이터 형식이 있는 테이블에는 프라이머리 키가 있어야 합니다. 프라이머리 키가 없는 ARRAY 데이터 형식이 있는 테이블은 전체 로드 중에 일시 중단됩니다.

다음 표는 소스 PostgreSQL 데이터 유형과 이 유형들이 성공적으로 마이그레이션되었는지 나타냅니다.

데이터 유형	마이그레이션 성공	부분적으로 마이그레이션	마이그레이션 하지 않음	설명
INTEGER	X			
SMALLINT	X			
BIGINT	X			
NUMERIC/DECIMAL(p,s)		X		여기서, 0<p<39 및 0<s
NUMERIC/DECIMAL		X		여기서, p>38 또는 p=s=0
REAL	X			

데이터 유형	마이그레이션 성공	부분적으로 마이그레이션	마이그레이션 하지 않음	설명
DOUBLE	X			
SMALLSERIAL	X			
SERIAL	X			
BIGSERIAL	X			
MONEY	X			
CHAR		X		지정한 정밀도 없이
CHAR(n)	X			
VARCHAR		X		지정한 정밀도 없이
VARCHAR(n)	X			
TEXT	X			
BYTEA	X			
TIMESTAMP	X			양과 음의 무한대 값은 각각 '9999-12-31 23:59:59'와 '4713-01-01 00:00:00 BC'로 잘립니다.
TIMESTAMP(시간대 사용)		X		
날짜	X			

데이터 유형	마이그레이션 성공	부분적으로 마이그레이션	마이그레이션 하지 않음	설명
TIME	X			
TIME WITH TIME ZONE		X		
INTERVAL		X		
BOOLEAN	X			
ENUM			X	
CIDR	X			
INET			X	
MACADDR			X	
TSVECTOR			X	
TSQUERY			X	
XML		X		
POINT	X			PostGIS 공간 데이터 형식
LINE			X	
LSEG			X	
BOX			X	
PATH			X	
POLYGON	X			PostGIS 공간 데이터 형식
CIRCLE			X	
JSON		X		

데이터 유형	마이그레이션 성공	부분적으로 마이그레이션	마이그레이션 하지 않음	설명
ARRAY	X			프라이머리 키 필요
COMPOSITE			X	
RANGE			X	
LINESTRING	X			PostGIS 공간 데이터 형식
MULTIPOINT	X			PostGIS 공간 데이터 형식
MULTILINESTRING	X			PostGIS 공간 데이터 형식
MULTIPOLYGON	X			PostGIS 공간 데이터 형식
GEOMETRYCOLLECTION	X			PostGIS 공간 데이터 형식

PostGIS 공간 데이터 형식 마이그레이션

공간 데이터는 공간에서 객체 또는 위치의 지오메트리 정보를 식별합니다. PostgreSQL 객체 관계형 데이터베이스는 PostGIS 공간 데이터 형식을 지원합니다.

PostgreSQL 공간 데이터 객체를 마이그레이션하기 전에 PostGIS 플러그인이 전역 수준에서 활성화되어 있는지 확인하세요. 그러면 AWS DMS가 PostgreSQL 대상 DB 인스턴스의 정확한 소스 공간 데이터 열을 생성합니다.

PostgreSQL 간 동종 마이그레이션의 경우 AWS DMS는 다음과 같은 PostGIS 기하학적 및 지리적(측지 좌표) 데이터 객체 유형 및 하위 유형의 마이그레이션을 지원합니다.

- POINT
- LINESTRING

- POLYGON
- MULTIPOINT
- MULTILINESTRING
- MULTIPOLYGON
- GEOMETRYCOLLECTION

PostgreSQL 소스 데이터베이스에서 AWS DMS 아티팩트 제거

DDL 이벤트를 캡처하기 위해 마이그레이션 작업이 시작되면 AWS DMS는 PostgreSQL 데이터베이스에서 다양한 아티팩트를 생성합니다. 작업이 완료되면 이 아티팩트를 제거해야 할 수도 있습니다.

아티팩트를 제거하려면 다음 문을 (표시되는 순서대로) 발행합니다. 여기서 `{AmazonRDSMigration}`은 아티팩트가 생성되는 스키마입니다. 스키마 삭제를 수행할 때는 매우 주의해야 합니다. 절대로 연산 스키마(특히 퍼블릭 스키마)를 삭제해서는 안 됩니다.

```
drop event trigger awsdms_intercept_ddl;
```

이벤트 트리거는 특정 스키마에 속하지 않습니다.

```
drop function {AmazonRDSMigration}.awsdms_intercept_ddl()
drop table {AmazonRDSMigration}.awsdms_ddl_audit
drop schema {AmazonRDSMigration}
```

PostgreSQL 데이터베이스를 DMS 소스로 사용 시 추가 구성 설정

다음 두 가지 방법으로 PostgreSQL 데이터베이스에서 데이터를 마이그레이션할 때 구성 설정을 추가할 수 있습니다.

- 연결 속성을 추가하여 DDL 이벤트를 캡처하고 연산 DDL 데이터베이스 아티팩트가 생성된 스키마를 지정할 수 있습니다. 자세한 설명은 PostgreSQL을 DMS 소스로 사용하는 경우의 엔드포인트 설정 및 추가 연결 속성 (ECA) 섹션을 참조하세요.
- 연결 문자열 파라미터를 재정의할 수 있습니다. 다음 중 하나를 수행하려면 이 옵션을 선택합니다.
  - 내부 AWS DMS 파라미터를 지정합니다. 이러한 파라미터는 거의 필요하지 않으므로 사용자 인터페이스에 표시되지 않습니다.
  - 특정 데이터베이스 클라이언트에서 통과(패스스루) 값을 지정합니다. AWS DMS는 데이터베이스 클라이언트에 전달되는 연결 문자열에 통과 파라미터를 포함합니다.

- PostgreSQL 버전 9.4 이상에서 테이블 수준 파라미터 REPLICA IDENTITY를 사용하면 미리 쓰기 로그(WAL)에 기록되는 정보를 제어할 수 있습니다. 특히 업데이트되거나 삭제된 행을 식별하는 WAL의 경우 더욱 그렇습니다. REPLICA IDENTITY FULL은 행에 있는 모든 열의 이전 값을 기록합니다. FULL은 필요하지 않을 수도 있는 WAL을 추가로 생성하므로 각 테이블에 REPLICA IDENTITY FULL을 신중하게 사용하세요. 자세한 내용은 ALTER TABLE-REPLICA IDENTITY를 참조하세요.

MapBooleanAsBoolean PostgreSQL 엔드포인트 설정 사용

PostgreSQL 엔드포인트 설정을 사용하여 PostgreSQL 소스의 불을 Amazon Redshift 대상에 매핑할 수 있습니다. 기본적으로 불 형식은 varchar(5)로 마이그레이션됩니다. 다음 예제에 나온 것처럼 PostgreSQL이 불 형식을 불로 마이그레이션하도록 MapBooleanAsBoolean을 지정할 수 있습니다.

```
--postgre-sql-settings '{"MapBooleanAsBoolean": true}'
```

단, 이 설정이 적용되려면 소스 엔드포인트와 대상 엔드포인트 모두에서 이 설정을 지정해야 합니다.

MySQL에는 불 형식이 없으므로 불 데이터를 MySQL로 마이그레이션할 때는 이 설정 대신 변환 규칙을 사용하세요.

PostgreSQL을 DMS 소스로 사용하는 경우의 엔드포인트 설정 및 추가 연결 속성 (ECA)

엔드포인트 설정 및 추가 연결 속성 (ECA) 을 사용하여 PostgreSQL 소스 데이터베이스를 구성할 수 있습니다. AWS DMS콘솔을 사용하거나 에서 JSON 구문을 사용하여 create-endpoint 명령을 사용하여 소스 엔드포인트를 생성할 때 엔드포인트 설정을 지정합니다. AWS CLI--postgre-sql-settings '{"*EndpointSetting*": *"value"*, *...*}'

다음 표에는 PostgreSQL을 소스로 사용할 수 있는 엔드포인트 설정 및 ECA가 나와 있습니다.

속성 이름	설명
CaptureDDLs	DDL 이벤트를 캡처하기 위해 AWS DMS는 작업을 시작할 때 PostgreSQL 데이터베이스에 다양한 아티팩트를 생성합니다. PostgreSQL 소스 데이터베이스에서 AWS DMS 아티팩트 제거의 설명에 따라 이 아티팩트를 제거할 수 있습니다. 이 값을 false로 설정하면 소스 데이터베이스에 테이블이나 트리거를 생성할 필요가 없습니다.

속성 이름	설명
	스트리밍된 DDL 이벤트가 캡처됩니다. 기본 값: `true` 유효값: true/false 예제: `--postgre-sql-settings '{"CaptureDDLs": true}'`
`ConsumeMonotonicEvents`	로그 시퀀스 번호(LSN)가 중복된 모놀리식 트랜잭션이 복제되는 방식을 제어하는 데 사용됩니다. 이 파라미터가 `false`인 경우, LSN이 중복된 이벤트가 대상에서 소비되고 복제됩니다. 이 파라미터가 `true`인 경우, 첫 번째 이벤트만 복제되고 LSN이 중복된 이벤트는 대상에서 소비되거나 복제되지 않습니다. 기본 값: `false` 유효한 값: false/true 예제: `--postgre-sql-settings '{"ConsumeMonotonicEvents": true}'`
`DdlArtifactsSchema`	연산 DDL 데이터베이스 아티팩트가 생성되는 스키마를 설정합니다. 기본값: 퍼블릭 유효값: 문자열 예제: `--postgre-sql-settings '{"DdlArtifactsSchema": "` *`xyzddlschema`* `"}'`
`ExecuteTimeout`	PostgreSQL 인스턴스의 클라이언트 문 제한 시간을 초 단위로 설정합니다. 기본값은 60초입니다. 예제: `--postgre-sql-settings '{"ExecuteTimeout": 100}'`

속성 이름	설명
FailTasksOnLobTruncation	이 값을 true로 설정하면 LOB 열의 실제 크기가 지정된 LobMaxSize 보다 클 경우 작업이 실패합니다. 작업이 제한된 LOB 모드로 설정되어 있고 이 옵션이 true로 설정된 경우, LOB 데이터가 잘리지 않는 대신에 작업이 실패합니다. 기본값: false 유효 값: 불 예제: --postgre-sql-settings '{"FailTasksOnLobTruncation": true}'
fetchCacheSize	이 추가 연결 속성(ECA)은 전체 로드 작업 중에 커서가 가져올 행 수를 설정합니다. 복제 인스턴스에서 사용할 수 있는 리소스에 따라 값을 높이거나 낮출 수 있습니다. 기본 값: 10000 유효한 값: 숫자 ECA 예제: fetchCacheSize=10000;
HeartbeatFrequency	WAL 하트비트 빈도(분)를 설정합니다. 기본 값: 5 유효한 값: 숫자 예제: --postgre-sql-settings '{"HeartbeatFrequency": 1}'

속성 이름	설명
HeartbeatSchema	하트비트 아티팩트가 생성되는 스키마를 설정합니다. 기본 값: public 유효값: 문자열 예제: --postgre-sql-settings '{"Heartb eatSchema": "xyzheartbeatschema"}'
MapJsonbAsClob	기본적으로 AWS DMS는 JSONB를 NCLOB에 매핑합니다. PostgreSQL이 JSONB 형식을 CLOB로 마이그레이션하도록 MapJsonbAsClob 을 지정할 수 있습니다. 예제: --postgre-sql-settings='{"M apJsonbAsClob": "true"}'
MapLongVarcharAs	기본적으로 AWS DMS는 VARCHAR를 WSTRING에 매핑합니다. PostgreSQL이 VARCHAR(N) 형식(여기서 N은 16387보다 큼)을 다음 형식으로 마이그레이션하도록 MapLongVarcharAs 를 지정할 수 있습니다. • WSTRING • CLOB • NCLOB 예제: --postgre-sql-settings='{"M apLongVarcharAs": "CLOB"}'

속성 이름	설명
MapUnboundedNumericAsString	이 파라미터는 숫자 값의 정밀도를 유지하면서 성공적으로 마이그레이션할 수 있도록 제한이 없는 숫자 데이터 형식의 열을 문자열로 취급합니다. 이 파라미터는 PostgreSQL 소스에서 PostgreSQL 대상 또는 PostgreSQL과 호환되는 데이터베이스로 복제하는 경우에만 사용하세요. 기본 값: false 유효값: false/true 예제: --postgre-sql-settings '{"MapUnboundedNumericAsString": true}' 이 파라미터를 사용하면 숫자에서 문자열로, 다시 숫자로 변환되므로 복제 성능이 일부 저하될 수 있습니다. 이 파라미터는 DMS 버전 3.4.4 이상에서 사용할 수 있습니다. **Note** 단지 MapUnboundedNumericAsString 을 PostgreSQL 소스 및 대상 엔드포인트에서 함께 사용해야 합니다. 소스 PostgreSQL 엔드포인트에서 MapUnboundedNumericAsString 을 사용하면 CDC 중에 정밀도가 28로 제한됩니다. 대상 엔드포인트에서 MapUnboundedNumericAsString 을 사용하면 정밀도 28, 스케일 6으로 데이터를 마이그레이션할 수 있습니다. PostgreSQL이 아닌 대상에 MapUnboundedNumericAsString 을 사용하지 마세요.

속성 이름	설명
PluginName	복제 슬롯을 생성하는 데 사용할 플러그인을 지정합니다. 유효값: pglogical , test_decoding 예제: --postgre-sql-settings '{"Plugin Name": "test_decoding"}'
SlotName	PostgreSQL 원본 인스턴스의 CDC 로드에 대해 이전에 만든 논리적 복제 슬롯의 이름을 설정합니다. AWS DMS API CdcStartPosition 요청 파라미터와 함께 사용할 경우 이 속성은 CDC 네이티브 시작 지점을 사용할 수도 있습니다. DMS는 CDC 로드 작업을 시작하기 전에 지정된 논리적 복제 슬롯이 존재하는지 확인합니다. 또한 작업이 CdcStartPosition 에 대한 유효한 설정으로 생성되었는지 확인합니다. 지정된 슬롯이 존재하지 않거나 유효한 CdcStartPosition 설정이 작업에 없으면 DMS에서 오류가 발생합니다. CdcStartPosition 요청 파라미터 설정에 대한 자세한 내용은 CDC 기본 시작점 결정 단원을 참조하십시오. CdcStartPosition 사용에 대한 자세한 내용은 AWS Database Migration Service API 참조의 CreateReplicationTask , StartReplicationTask 및 ModifyReplicationTask API 작업 설명서를 참조하세요. 유효값: 문자열 예제: --postgre-sql-settings '{"SlotName": "abc1d2efghijk_34567890_z0yx98w7_6v54_32ut_1srq_1a2b34c5d67ef"}'
unboundedVarcharMaxSize	이 추가 연결 속성 (ECA) 은 최대 길이 지정자 VarChar 없이 유형으로 정의된 데이터 열의 최대 크기를 정의합니다. 기본값은 8000바이트입니다. 최대값은 10485760바이트입니다.

PostgreSQL 데이터베이스를 DMS 소스로 사용 시 제한 사항

PostgreSQL을 AWS DMS에서 원본으로 사용하는 경우 다음 제한 사항이 적용됩니다.

- AWS DMS는 Amazon RDS for PostgreSQL 10.4 또는 Amazon Aurora PostgreSQL 10.4를 소스 또는 대상으로 사용할 수 없습니다.
- 캡처된 테이블에는 프라이머리 키가 있어야 합니다. 테이블에 프라이머리 키가 없는 경우, AWS DMS는 해당 테이블의 DELETE 및 UPDATE 레코드 작업을 무시합니다. 해결 방법은 논리적 복제를 사용한 변경 데이터 캡처(CDC) 활성화를 참조하세요.

  참고: 프라이머리 키/고유 인덱스 없이 마이그레이션하는 것은 권장하지 않습니다. 마이그레이션할 경우, “NO” Batch 적용 기능, 전체 LOB 기능, 데이터 검증 및 Redshift 대상으로 효율적으로 복제할 수 없는 등의 추가 제한이 적용됩니다.
- AWS DMS는 프라이머리 키 세그먼트 업데이트 시도를 무시합니다. 이 경우, 대상에서 아무 행도 업데이트하지 않는 것으로 업데이트를 식별합니다. 그러나, PostgreSQL의 프라이머리 키 업데이트 결과는 예측할 수 없기 때문에 예외 테이블에 레코드가 기록되지 않습니다.
- AWS DMS는 타임스탬프에서 프로세스 변경 시작 실행 옵션을 지원하지 않습니다.
- AWS DMS는 파티션 또는 하위 파티션 작업(ADD, DROP 또는 TRUNCATE)으로 인해 발생하는 변경 사항을 복제하지 않습니다.
- 이름이 같지만 각 이름의 대소문자가 다른 여러 테이블(예: table1, TABLE1, Table1)을 복제하면 예측할 수 없는 동작이 발생할 수 있습니다. 이 문제 때문에 AWS DMS는 이런 유형의 복제를 지원하지 않습니다.
- 대부분의 경우, AWS DMS는 테이블의 CREATE, ALTER, DROP DDL 문 변경 처리를 지원하지만 테이블이 내부 함수 또는 프로시저 본문 블록이나 중첩된 구성체에 있는 경우에는 AWS DMS가 이런 변경 처리를 지원하지 않습니다.

  예를 들어 다음 변경은 캡처되지 않습니다.

```
CREATE OR REPLACE FUNCTION attu.create_distributors1() RETURNS void
LANGUAGE plpgsql
AS $$
BEGIN
create table attu.distributors1(did serial PRIMARY KEY,name
varchar(40) NOT NULL);
END;
$$;
```

- 현재 PostgreSQL 소스의 `boolean` 데이터 형식은 값이 일치하지 않는 `bit` 데이터 형식으로 SQL Server 대상에 마이그레이션됩니다. 이 문제를 해결하려면 `VARCHAR(1)` 데이터 형식으로 열의 테이블을 미리 생성하거나 AWS DMS가 테이블을 생성하도록 합니다. 그런 다음, 다운스트림 처리에서 'F'를 False로 처리하고 'T'를 True로 처리합니다.
- AWS DMS는 TRUNCATE 작업의 변경 처리를 지원하지 않습니다.
- OID LOB 데이터 형식은 대상으로 마이그레이션되지 않습니다.
- AWS DMS는 동종 마이그레이션에서만 PostGIS 데이터 형식을 지원합니다.
- 소스가 온프레미스 또는 Amazon EC2 인스턴스에 있는 PostgreSQL 데이터베이스인 경우, 소스 엔드포인트에 test_decoding 출력 플러그인이 설치되어 있는지 확인합니다. 이 플러그인은 PostgreSQL contrib 패키지에서 찾을 수 있습니다. test-decoding 플러그인에 대한 자세한 내용은 PostgreSQL 설명서를 참조하십시오.
- AWS DMS는 (ALTER TABLE 문에서 ALTER COLUMN SET DEFAULT 절을 사용하여) 열 기본값을 설정 및 설정 해제하기 위한 변경 처리를 지원하지 않습니다.
- AWS DMS는 (ALTER TABLE 문에서 ALTER COLUMN [SET|DROP] NOT NULL 절을 사용하여) 열 null 가능 여부를 설정하기 위한 변경 처리를 지원하지 않습니다.
- 논리적 복제가 활성화된 경우, 트랜잭션당 메모리에 보관되는 최대 변경 사항 수는 4MB입니다. 그 후에는 변경 사항이 디스크로 유출됩니다. 따라서 트랜잭션이 완료 또는 중지되고 롤백이 완료될 때까지 `ReplicationSlotDiskUsage`가 증가하며 `restart_lsn`이 진행되지 않습니다. 트랜잭션이 길기 때문에 롤백하는 데 시간이 오래 걸릴 수 있습니다. 따라서 논리적 복제가 활성화된 경우 장기 실행 트랜잭션이나 많은 하위 트랜잭션을 피하십시오. 대신 트랜잭션을 여러 개의 작은 트랜잭션으로 나누세요.

  Aurora PostgreSQL 버전 13 이상에서는 `logical_decoding_work_mem` 파라미터를 조정하여 DMS 유출로 인해 데이터가 디스크로 변경되는 시기를 제어할 수 있습니다. 자세한 설명은 Aurora PostgreSQL에서 파일을 유출하세요 섹션을 참조하세요.
- ARRAY 데이터 형식이 있는 테이블에는 프라이머리 키가 있어야 합니다. 프라이머리 키가 없는 ARRAY 데이터 형식이 있는 테이블은 전체 로드 중에 일시 중단됩니다.
- AWS DMS는 분할된 테이블 복제를 지원하지 않습니다. 분할된 테이블이 감지되면, 다음 작업이 진행됩니다.
  - 엔드포인트는 상위 및 하위 테이블 목록을 보고합니다.
  - AWS DMS는 대상에서 선택한 테이블과 동일한 속성을 지닌 일반 테이블로서 테이블을 생성합니다.
  - 원본 데이터베이스의 상위 테이블에 하위 테이블과 동일한 프라이머리 키 값이 있으면, "중복 키" 오류가 발생합니다.

- 파티션된 테이블을 PostgreSQL 소스에서 PostgreSQL 대상으로 복제하려면, 먼저 대상에 상위 및 하위 테이블을 수동으로 생성해야 합니다. 그런 다음 이 테이블에 복제할 별도의 작업을 정의합니다. 이 경우에 작업 구성을 로딩 전 자르기로 설정합니다.
- PostgreSQL NUMERIC 데이터 형식의 크기는 고정되어 있지 않습니다. NUMERIC 데이터 형식이지만 정밀도 및 배율이 없는 데이터를 전송하는 경우, DMS는 기본적으로 NUMERIC(28,6)(정밀도 28, 배율 6)을 사용합니다. 예를 들어 소스의 0.611111104488373 값은 PostgreSQL 대상에서 0.611111로 변환됩니다.
- AWS DMS는 Aurora PostgreSQL Serverless V1을 전체 로드 작업의 소스로만 지원합니다. AWS DMS는 Aurora PostgreSQL Serverless V2를 전체 로드, 전체 로드 및 CDC, CDC 전용 작업의 소스로 지원합니다.
- AWS DMS는 coalesce 함수로 만든 고유 인덱스가 있는 테이블의 복제를 지원하지 않습니다.
- LOB 모드를 사용하는 경우, 소스 테이블과 해당 대상 테이블 모두에 동일한 프라이머리 키가 있어야 합니다. 테이블 중 하나에 프라이머리 키가 없는 경우, DELETE 및 UPDATE 레코드 작업의 결과를 예측할 수 없습니다.
- 병렬 로드 기능을 사용하는 경우, 파티션 또는 하위 파티션에 따른 테이블 분할은 지원되지 않습니다. 병렬 로드에 관한 자세한 내용은 선택한 테이블, 뷰 및 컬렉션에 병렬 로드 사용 섹션을 참조하세요.
- AWS DMS는 지연된 제약 조건을 지원하지 않습니다.
- AWS DMS 버전 3.4.7은 PostgreSQL 14.x를 소스로 지원하지만 다음과 같은 제한이 있습니다.
  - AWS DMS는 2단계 커밋의 변경 처리를 지원하지 않습니다.
  - AWS DMS는 오랫동안 진행 중인 트랜잭션을 스트리밍하기 위한 논리적 복제를 지원하지 않습니다.
- AWS DMS는 소스로 사용되는 PostgreSQL용 Amazon RDS 프록시에서 CDC를 지원하지 않습니다.
- 프라이머리 키 열을 포함하지 않는 소스 필터를 사용하면 DELETE 작업이 캡처되지 않습니다.
- 소스 데이터베이스가 다른 타사 복제 시스템의 대상이기도 한 경우, CDC 중에는 DDL 변경 내용이 마이그레이션되지 않을 수 있습니다. 이러한 상황으로 인해 awsdms_intercept_ddl 이벤트 트리거가 실행되지 않을 수 있기 때문입니다. 이 상황을 해결하려면 소스 데이터베이스에서 해당 트리거를 다음과 같이 수정하세요.

```
alter event trigger awsdms_intercept_ddl enable always;
```

- RDS for PostgreSQL 다중 AZ 데이터베이스 클러스터가 논리적 복제를 지원하지 않으므로 AWS DMS는 PostgreSQL이 소스로 사용될 때 Amazon RDS for PostgreSQL에서 CDC를 지원하지 않습니다.

PostgreSQL용 소스 데이터 형식

다음 표에는 AWS DMS를 사용하는 경우 지원되는 PostgreSQL 원본 데이터 형식과 AWS DMS 데이터 형식으로부터의 기본 매핑이 나와 있습니다.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하십시오.

AWS DMS 데이터 형식에 관한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

PostgreSQL 데이터 형식	DMS 데이터 형식
INTEGER	INT4
SMALLINT	INT2
BIGINT	INT8
NUMERIC(p,s)	정밀도 범위가 0~38인 경우, NUMERIC을 사용합니다. 정밀도가 39 이상인 경우, STRING을 사용합니다.
DECIMAL(P,S)	정밀도 범위가 0~38인 경우, NUMERIC을 사용합니다. 정밀도가 39 이상인 경우, STRING을 사용합니다.
REAL	REAL4
DOUBLE	REAL8
SMALLSERIAL	INT2
SERIAL	INT4
BIGSERIAL	INT8
MONEY	NUMERIC(38,4)

PostgreSQL 데이터 형식	DMS 데이터 형식
	MONEY 데이터 형식은 SQL Server에서 FLOAT에 매핑됩니다.
CHAR	WSTRING (1)
CHAR(N)	WSTRING (n)
VARCHAR(N)	WSTRING (n)
TEXT	NCLOB
BYTEA	BLOB
TIMESTAMP	DATETIME
TIMESTAMP(시간대 사용)	DATETIME
날짜	날짜
TIME	TIME
TIME WITH TIME ZONE	TIME
INTERVAL	STRING (128) - 1 YEAR, 2 MONTHS, 3 DAYS, 4 HOURS, 5 MINUTES, 6 SECONDS
BOOLEAN	CHAR (5) 거짓 혹은 참
ENUM	STRING (64)
CIDR	STRING (50)
INET	STRING (50)
MACADDR	STRING (18)
BIT (n)	STRING (n)
BIT VARYING (n)	STRING (n)

PostgreSQL 데이터 형식	DMS 데이터 형식
UUID	STRING
TSVECTOR	CLOB
TSQUERY	CLOB
XML	CLOB
POINT	STRING (255) "(x,y)"
LINE	STRING (255) "(x,y,z)"
LSEG	STRING (255) "((x1,y1),(x2,y2))"
BOX	STRING (255) "((x1,y1),(x2,y2))"
PATH	CLOB "((x1,y1),(xn,yn))"
POLYGON	CLOB "((x1,y1),(xn,yn))"
CIRCLE	STRING (255) "(x,y),r"
JSON	NCLOB
JSONB	NCLOB
ARRAY	NCLOB
COMPOSITE	NCLOB
HSTORE	NCLOB
INT4RANGE	STRING (255)
INT8RANGE	STRING (255)
NUMRANGE	STRING (255)
STRRANGE	STRING (255)

PostgreSQL에 LOB 소스 데이터 형식 사용

PostgreSQL 열 크기는 PostgreSQL LOB 데이터 형식의 AWS DMS 데이터 형식으로의 변환에 영향을 미칩니다. 이를 위해 다음 AWS DMS 데이터 형식의 경우, 다음 단계에 따르십시오.

- BLOB - 작업 생성 시 LOB 크기를 다음으로 제한을 최대 LOB 크기(KB) 값으로 설정합니다.
- CLOB - 복제는 각 문자를 UTF8 문자로 취급합니다. 따라서 열에서 가장 긴 문자 텍스트의 길이를 찾습니다(여기에서는 `max_num_chars_text`로 표시). 이 길이를 사용하여 LOB 크기를 다음으로 제한 값을 지정합니다. 데이터에 4바이트 문자가 포함된 경우, 2를 곱해 Limit LOB size to(LOB 크기를 다음으로 제한) 값(바이트 단위)을 지정합니다. 이 경우, Limit LOB size to(LOB 크기를 다음으로 제한)는 `max_num_chars_text` x 2와 같습니다.
- NCLOB - 복제는 각 문자를 2바이트 문자로 취급합니다. 따라서 열에서 가장 긴 문자 텍스트의 길이(`max_num_chars_text`)를 찾고 2를 곱합니다. 이렇게 하여 LOB 크기를 다음으로 제한 값을 지정합니다. 이 경우, Limit LOB size to(LOB 크기를 다음으로 제한)는 `max_num_chars_text` x 2와 같습니다. 데이터에 4바이트 문자가 포함된 경우에는 다시 2를 곱합니다. 이 경우, Limit LOB size to(LOB 크기를 다음으로 제한)는 `max_num_chars_text` x 4와 같습니다.

AWS DMS에서 MySQL 호환 데이터베이스를 소스로 사용

AWS Database Migration Service를 사용하여 MySQL 호환 데이터베이스(MySQL, MariaDB 또는 Amazon Aurora MySQL)의 데이터를 마이그레이션할 수 있습니다.

AWS DMS가 소스로 지원하는 MySQL 버전에 대한 자세한 내용은 출처: AWS DMS 섹션을 참조하세요.

SSL을 사용하여 MySQL 호환 엔드포인트와 복제 인스턴스 간의 연결을 암호화할 수 있습니다. MySQL 호환 엔드포인트에서 SSL 사용에 관한 자세한 내용은 AWS Database Migration Service에서 SSL 사용 섹션을 참조하십시오.

다음 섹션에서 '자체 관리형'이라는 용어는 온프레미스나 Amazon EC2에 설치되어 있는 데이터베이스에 적용됩니다. 'AWS 관리형'이라는 용어는 Amazon RDS, Amazon Aurora 또는 Amazon S3에 있는 모든 데이터베이스에 적용됩니다.

PostgreSQL 호환 데이터베이스 및 AWS DMS 사용에 대한 세부 정보는 다음 섹션을 참조하십시오.

주제

- AWS DMS를 사용하여 MySQL에서 MySQL로 마이그레이션.
- AWS DMS용 소스로 MySQL 호환 데이터베이스 사용

- AWS DMS용 소스로 자체 관리형 MySQL 호환 데이터베이스 사용
- AWS DMS 소스로 AWS 관리형 MySQL 호환 데이터베이스 사용
- AWS DMS에서 MySQL 데이터베이스를 소스로 사용 시 적용되는 제한 사항
- XA 트랜잭션 지원
- MySQL을 AWS DMS 소스로 사용할 경우 엔드포인트 설정
- MySQL용 소스 데이터 형식

AWS DMS를 사용하여 MySQL에서 MySQL로 마이그레이션.

MySQL이 아닌 데이터베이스 엔진에서 MySQL 데이터베이스로 마이그레이션하는 이종 마이그레이션의 경우, 사용할 수 있는 최고의 마이그레이션 도구는 거의 언제나 AWS DMS입니다. 하지만 MySQL 데이터베이스에서 MySQL 데이터베이스로 마이그레이션하는 동종 마이그레이션의 경우, 동종 데이터 마이그레이션 프로젝트를 사용하는 것이 좋습니다. 동종 데이터 마이그레이션은 네이티브 데이터베이스 도구를 사용하여 AWS DMS와 비교할 때 향상된 데이터 마이그레이션 성능과 정확성을 제공합니다.

AWS DMS용 소스로 MySQL 호환 데이터베이스 사용

AWS DMS용 원본으로 MySQL 데이터베이스를 사용하기 전에 다음 사전 요구 사항이 있는지 확인해야 합니다. 이러한 사전 요구 사항은 자체 관리형 또는 AWS 관리형 소스에 적용됩니다.

Replication Admin 역할이 할당된 AWS DMS용 계정이 있어야 합니다. 역할에는 다음 권한이 있어야 합니다.

- REPLICATION CLIENT - 이 권한은 변경 데이터 캡처(CDC) 작업에만 필요합니다. 즉, full-load-only 작업에는 이 권한이 필요하지 않습니다.
- REPLICATION SLAVE - 이 권한은 변경 데이터 캡처(CDC) 작업에만 필요합니다. 즉, full-load-only 작업에는 이 권한이 필요하지 않습니다.
- SUPER - 이 권한은 MySQL 5.6.6 이전 버전에서만 필요합니다.

또한, AWS DMS 사용자에게는 복제를 위해 지정된 원본 테이블에 대해 SELECT 권한이 있어야 합니다.

AWS DMS용 소스로 자체 관리형 MySQL 호환 데이터베이스 사용

AWS DMS용 원본으로 다음 자체 관리형 MySQL 호환 데이터베이스를 사용할 수 있습니다.

- MySQL Community Edition
- MySQL Standard Edition
- MySQL Enterprise Edition
- MySQL Cluster Carrier Grade Edition
- MariaDB Community Edition
- MariaDB Enterprise Edition
- MariaDB Column Store

CDC를 사용하려면 이진 로깅을 활성화해야 합니다. 이진 로깅을 활성화하려면, MySQL의 `my.ini`(Windows) 또는 `my.cnf`(UNIX) 파일에서 다음 파라미터를 구성해야 합니다.

파라미터	값
`server_id`	이 파라미터 값을 1 이상의 값으로 설정합니다.
`log-bin`	`log-bin=E:\MySql_Logs\BinLog` 과 같은 이진 로그 파일로 이 경로를 설정합니다. 파일 확장자를 포함해서는 안 됩니다.
`binlog_format`	이 파라미터를 `ROW`으로 설정합니다. `binlog_format` 을 `STATEMENT` 로 설정하면 대상에 데이터를 복제할 때 불일치가 발생할 수 있으므로 복제 중에 이 설정을 사용하는 것이 좋습니다. `binlog_format` 이 `MIXED`로 설정되면 데이터베이스 엔진도 비슷한 일관되지 않은 데이터를 대상에 기록합니다. 대상 데이터베이스에 일관되지 않은 데이터가 기록되도록 할 수 있는 `STATEMENT` 기반 로깅으로 데이터베이스 엔진이 자동으로 전환되기 때문입니다.
`expire_logs_days`	이 파라미터 값을 1 이상의 값으로 설정합니다. 디스크 공간의 과사용을 방지하려면 기본값 0을 사용하지 않는 것이 좋습니다.
`binlog_checksum`	이 파라미터를 DMS 버전 3.4.7 또는 이전 버전으로 설정하십시오. `NONE`
`binlog_row_image`	이 파라미터를 `FULL`으로 설정합니다.
`log_slave_updates`	원본으로 MySQL 또는 MariaDB 읽기 전용 복제본을 사용하고 있는 경우 이 파라미터를 `TRUE`로 설정합니다.

원본에서 NDB(클러스터 방식) 데이터베이스 엔진을 사용하는 경우, 해당 스토리지 엔진을 사용하는 테이블에 대해 CDC를 활성화하도록 다음 파라미터를 구성해야 합니다. 이러한 변경 사항을 MySQL의 `my.ini`(Windows) 또는 `my.cnf`(UNIX) 파일에 추가합니다.

파라미터	값
`ndb_log_bin`	이 파라미터를 `ON`으로 설정합니다. 이 값은 클러스터링된 테이블의 변경 사항이 이진 로그에 로깅되도록 보장해 줍니다.
`ndb_log_update_as_write`	이 파라미터를 `OFF`으로 설정합니다. 이 값은 이진 로그에서 UPDATE 문을 INSERT 문으로 쓰지 않도록 해줍니다.
`ndb_log_updated_only`	이 파라미터를 `OFF`으로 설정합니다. 이 값은 이진 로그가 변경된 열뿐만 아니라 전체 행을 포함하도록 해줍니다.

AWS DMS 소스로 AWS 관리형 MySQL 호환 데이터베이스 사용

AWS DMS 소스로 사용할 수 있는 AWS 관리형 MySQL 호환 데이터베이스는 다음과 같습니다.

- MySQL Community Edition
- MariaDB Community Edition
- Amazon Aurora MySQL 호환 버전

AWS 관리형 MySQL 호환 데이터베이스를 AWS DMS 소스로 사용하는 경우, 다음과 같은 CDC 사전 요구 사항을 충족해야 합니다.

- RDS for MySQL 및 RDS for MariaDB에 이진 로그를 활성화하려면 인스턴스 수준에서 자동 백업을 활성화하세요. Aurora MySQL 클러스터의 이진 로그를 활성화하려면 파라미터 그룹에서 `binlog_format` 변수를 변경합니다.

  자동 백업 설정에 관한 자세한 내용은 Amazon RDS 사용 설명서의 백업 작업 섹션을 참조하세요.

  Amazon RDS for MySQL 데이터베이스에 이진 로깅을 설정하는 방법에 대한 자세한 내용은 Amazon RDS 사용 설명서의 MySQL 이진 로깅 구성을 참조하세요.

  Aurora MySQL 클러스터의 이진 로깅을 설정하는 방법에 대한 자세한 내용은 Amazon Aurora MySQL 클러스터에 대한 바이너리 로깅을 활성화하려면 어떻게 해야 하나요?를 참조하세요.

- CDC를 사용하려면 이진 로깅을 켭니다. Amazon RDS for MySQL 데이터베이스에 이진 로깅을 설정하는 방법에 대한 자세한 내용은 Amazon RDS 사용 설명서의 MySQL 이진 로깅 형식 구성을 참조하세요.
- AWS DMS에 바이너리 로깅을 사용할 수 있어야 합니다. AWS 관리형 MySQL 호환 데이터베이스는 가능한 빨리 이진 로그를 삭제하므로, 로그가 사용 가능한 상태로 유지되는 시간을 늘려야 합니다. 예를 들어, 로그 보존 시간을 24시간으로 늘리려면 다음 명령을 실행합니다.

```
call mysql.rds_set_configuration('binlog retention hours', 24);
```

- `binlog_format` 파라미터를 `"ROW"`로 설정합니다.

> **Note**
>
> MySQL 또는 MariaDB에서는 `binlog_format`이 동적 파라미터이므로 새 값을 적용하기 위해 재부팅할 필요가 없습니다. 하지만 새 값은 새 세션에만 적용됩니다. 복제를 위해 `binlog_format`을 `ROW`로 전환해도 값을 변경하기 전에 해당 세션이 시작된 경우, 데이터베이스는 `MIXED` 형식을 사용하여 후속 이진 로그를 계속 생성할 수 있습니다. 이로 인해 AWS DMS가 소스 데이터베이스의 모든 변경 내용을 제대로 캡처하지 못할 수 있습니다. MariaDB 또는 MySQL 데이터베이스에서 `binlog_format` 설정을 변경하는 경우, 데이터베이스를 다시 시작하여 기존 세션을 모두 닫거나 DML(데이터 조작 언어) 작업을 수행하는 애플리케이션을 다시 시작해야 합니다. `binlog_format` 파라미터를 `ROW`로 변경한 후 데이터베이스가 강제로 모든 세션을 다시 시작하도록 하면 데이터베이스가 모든 후속 소스 데이터베이스 변경 사항을 올바른 형식으로 기록하므로 AWS DMS가 해당 변경 내용을 제대로 캡처할 수 있습니다.

- `binlog_row_image` 파라미터를 `"Full"`로 설정합니다.
- `binlog_checksum`파라미터를 DMS 버전 `"NONE"` 3.4.7 또는 이전 버전으로 설정합니다. Amazon RDS MySQL의 파라미터 설정에 대한 자세한 내용은 Amazon RDS 사용 설명서의 백업 작업을 참조하세요.
- Amazon RDS MySQL 또는 Amazon RDS MariaDB 읽기 전용 복제본을 소스로 사용하고 있는 경우, 읽기 전용 복제본에서 백업을 활성화하고 `log_slave_updates` 파라미터를 `TRUE`로 설정해야 합니다.

AWS DMS에서 MySQL 데이터베이스를 소스로 사용 시 적용되는 제한 사항

MySQL 데이터베이스를 원본으로 사용하는 경우 다음 사항을 고려하십시오.

- Amazon RDS MySQL 5.5 이하의 경우, 변경 데이터 캡처(CDC)가 지원되지 않습니다. Amazon RDS MySQL의 경우, CDC를 활성화하려면 버전 5.6, 5.7 또는 8.0을 사용해야 합니다. CDC는 자체 관리형 MySQL 5.5 소스에서 지원됩니다.
- CDC의 경우, `CREATE TABLE`, `ADD COLUMN`, `DROP COLUMN` 열 데이터 형식 변경과 `renaming a column`이 지원됩니다. 하지만 `DROP TABLE`, `RENAME TABLE` 및 열 기본값, 열 Null 허용 여부, 문자 집합 등과 같은 다른 속성에 대한 업데이트는 지원되지 않습니다.
- 소스에서 분할된 테이블의 경우, 대상 테이블 준비 모드를 대상에서 테이블 삭제로 지정하면 AWS DMS가 파티션 없이 MySQL 대상에 심플 테이블을 생성합니다. 파티션된 테이블을 대상의 파티션된 테이블로 마이그레이션하려면, 대상 MySQL 데이터베이스에서 파티션된 테이블을 미리 생성합니다.
- `ALTER TABLE table_name ADD COLUMN column_name` 문을 사용하여 테이블의 시작(FIRST) 또는 중간(AFTER)에 열을 추가하는 것은 지원되지 않습니다. 열은 항상 테이블 끝에 추가됩니다.
- 테이블 이름에 대/소문자가 포함되어 있고, 대/소문자를 구분하지 않는 파일 이름이 있는 운영 체제에서 소스 엔진이 호스팅되는 경우, CDC가 지원되지 않습니다. 예로는 HFS+를 사용하는 Microsoft Windows 또는 OS X가 있습니다.
- Aurora MySQL 호환 버전 서버리스는 전체 로드에 사용할 수 있지만 CDC에는 사용할 수 없습니다. MySQL 사전 요구 사항을 활성화할 수 없기 때문입니다. 자세한 내용은 파라미터 그룹 및 Aurora Serverless v1을 참조하세요.
- 열의 AUTO_INCREMENT 속성은 대상 데이터베이스 열로 마이그레이션되지 않습니다.
- 표준 블록 스토리지에 이진 로그가 저장되지 않으면 변경 사항 캡처가 지원되지 않습니다. 예를 들어 이진 로그가 Amazon S3에 저장되면 CDC가 작동하지 않습니다.
- 기본적으로 AWS DMS에서는 InnoDB 스토리지 엔진이 있는 대상 테이블을 생성합니다. InnoDB가 아닌 스토리지 엔진을 사용해야 하는 경우, 테이블을 수동으로 생성하고 이를 아무 작업 안 함 모드를 사용하여 마이그레이션해야 합니다.
- DMS 마이그레이션 작업 모드가 기존 데이터 마이그레이션(전체 로드만 해당)이 아니라면 Aurora MySQL 복제본을 AWS DMS 소스로 사용할 수 없습니다.
- MySQL 호환 소스가 전체 로드 도중 중지되는 경우, AWS DMS 작업은 오류로 중지되지 않습니다. 작업은 성공적으로 종료되지만 대상과 원본이 동기화되지 않았을 수 있습니다. 그러한 경우, 작업을 다시 시작하거나 영향을 받은 테이블을 다시 로드하십시오.
- 열의 일부에 생성된 색인 값은 마이그레이션되지 않습니다. 예를 들어 CREATE INDEX first_ten_chars ON 색인 고객(이름(10))은 대상에서 생성되지 않습니다.

- 작업이 LOB를 복제하지 않도록 구성된 경우도 있습니다 (작업 설정의 SupportLobs "" 은 false 이거나 작업 콘솔에서 LOB 열 포함 안 함을 선택함). 이 경우, AWS DMS가 MEDIUMBLOB, LONGBLOB, MEDIUMTEXT 및 LONGTEXT 열을 대상으로 마이그레이션하지 않습니다.

  BLOB, TINYBLOB, TEXT 및 TINYTEXT 열은 영향을 받지 않으며 대상으로 마이그레이션됩니다.
- 임시 데이터 테이블 또는 시스템 버전이 지정된 테이블은 MariaDB 소스 및 대상 데이터베이스에서 지원되지 않습니다.
- 두 Amazon RDS Aurora MySQL 클러스터 간에 마이그레이션하는 경우, RDS Aurora MySQL 소스 엔드포인트는 복제본 인스턴스가 아니라 읽기/쓰기 인스턴스여야 합니다.
- AWS DMS는 현재 MariaDB의 뷰 마이그레이션을 지원하지 않습니다.
- AWS DMS는 MySQL의 파티션된 테이블에 대한 DDL 변경을 지원하지 않습니다. CDC 도중 파티션 DDL 변경으로 인한 테이블 일시 중단을 건너뛰려면 `skipTableSuspensionForPartitionDdl`을 `true`로 설정합니다.
- AWS DMS는 버전 3.5.0 이상에서만 XA 트랜잭션을 지원합니다. 이전 버전은 XA 트랜잭션을 지원하지 않습니다. AWS DMS는 MariaDB 버전 10.6에서는 XA 트랜잭션을 지원하지 않습니다. 자세한 내용은 the section called "XA 트랜잭션 지원" 단원을 참조하십시오.
- AWS DMS는 소스 데이터에 GTID가 포함되어 있더라도 복제에 GTID를 사용하지 않습니다.
- AWS DMS는 이진 로그 트랜잭션 압축을 지원하지 않습니다.
- AWS DMS는 InnoDB 스토리지 엔진을 사용하여 MySQL 데이터베이스의 ON DELETE CASCADE 및 ON UPDATE CASCADE 이벤트를 전파하지 않습니다. 이러한 이벤트의 경우, MySQL은 하위 테이블의 계단식 작업을 반영하기 위해 binlog 이벤트를 생성하지 않습니다. 따라서 AWS DMS는 해당 변경 내용을 하위 테이블에 복제할 수 없습니다. 자세한 설명은 인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음 섹션을 참조하세요.
- AWS DMS는 계산된 (`VIRTUAL` 및 `GENERATED ALWAYS`) 열의 변경 내용을 캡처하지 않습니다. 이러한 제한을 해결하려면 다음을 수행하세요.
  - 대상 데이터베이스에서 대상 테이블을 미리 생성하고, `DO_NOTHING` 또는 `TRUNCATE_BEFORE_LOAD` 전체 로드 작업 설정을 사용하여 AWS DMS 작업을 생성합니다.
  - 계산된 열을 작업 범위에서 제거하는 변환 규칙을 추가합니다. 변환에 대한 자세한 내용은 변환 규칙 및 작업 섹션을 참조하세요.

XA 트랜잭션 지원

확장 아키텍처(XA) 트랜잭션은 여러 트랜잭션 리소스의 일련의 작업을 신뢰할 수 있는 단일 전역 트랜잭션으로 그룹화하는 데 사용할 수 있는 트랜잭션입니다. XA 트랜잭션은 2단계 커밋 프로토콜을 사용합니다. 일반적으로 열린 XA 트랜잭션이 있는 동안 변경 내용을 캡처하면 데이터가 손실될 수 있

습니다. 데이터베이스에서 XA 트랜잭션을 사용하지 않는 경우, 기본값 TRUE를 사용하여 이 권한과 IgnoreOpenXaTransactionsCheck 구성을 무시할 수 있습니다. XA 트랜잭션이 있는 소스에서 복제를 시작하려면 다음을 수행합니다.

- AWS DMS 엔드포인트 사용자에게 다음 권한이 있는지 확인합니다.

```
grant XA_RECOVER_ADMIN on *.* to 'userName'@'%';
```

- 엔드포인트 설정 IgnoreOpenXaTransactionsCheck를 false로 설정합니다.

> **Note**
>
> AWS DMS는 MariaDB 소스 DB 버전 10.6에서 XA 트랜잭션을 지원하지 않습니다.

MySQL을 AWS DMS 소스로 사용할 경우 엔드포인트 설정

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 MySQL 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI에서 create-endpoint 명령을 --my-sql-settings '{"*EndpointSetting*": *"value"*, *...*}' JSON 구문과 함께 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다.

MySQL을 소스로 하여 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

명칭	설명
EventsPollInterval	데이터베이스가 유휴 상태일 때 이진 로그에서 새 변경 사항/이벤트를 확인하는 빈도를 지정합니다. 기본값: 5 유효한 값: 1~60 예제: --my-sql-settings '{"EventsPollInterval": 5}' 이 예에서 AWS DMS는 이전 로그에 변경 사항이 있는지 5초마다 확인합니다.

명칭	설명
ExecuteTimeout	AWS DMS 버전 3.4.7 이상의 경우, MySQL 소스 엔드포인트의 클라이언트 문 제한 시간을 초 단위로 설정합니다. 기본값: 60 예제: `--my-sql-settings '{"ExecuteTimeout" : 1500}'`
ServerTimezone	원본 MySQL 데이터베이스의 시간대를 지정합니다. 예제: `--my-sql-settings '{"ServerTimezone" : " US/Pacific "}'`
AfterConnectScript	AWS DMS가 엔드포인트에 연결된 즉시 실행할 스크립트를 지정합니다. SQL 문의 성공 여부에 상관없이 마이그레이션 작업이 계속 실행됩니다. 유효한 값: 하나 이상의 유효한 SQL 문(세미콜론으로 끝남). 예제: `--my-sql-settings '{"AfterConnectScript": "ALTER SESSION SET CURRENT_SCHEMA=system"}'`
CleanSrcMetadataOnMismatch	불일치 발생 시 복제 인스턴스에서 테이블 메타데이터 정보를 지운 후 다시 생성합니다. 예컨대 테이블에서 alter DDL을 실행하면 복제 인스턴스에 캐시된 테이블에 관해 다른 정보가 생길 수 있는 경우를 들 수 있습니다. 불. 기본 값: `false` 예제: `--my-sql-settings '{"CleanSrcMetadataOnMismatch": false}'`

명칭	설명
skipTableSuspensionForPartitionDdl	AWS DMS는 MySQL의 파티션된 테이블에 대한 DDL 변경을 지원하지 않습니다. AWS DMS버전 3.4.6 이상에서는 CDC 중 파티션 DDL 변경으로 인한 테이블 일시 중지를 true 건너뛰도록 설정할 수 있습니다. AWS DMS partitioned-table-related DDL을 무시하고 추가 바이너리 로그 변경을 계속 처리합니다. 기본 값: false 예제: --my-sql-settings '{"skipTableSuspensionForPartitionDdl": true}'
IgnoreOpenXaTransactionsCheck	AWS DMS 버전 3.5.0 이상의 경우, 시작할 때 열린 XA 트랜잭션을 작업이 무시할지 여부를 지정합니다. 소스에 XA 트랜잭션이 있는 경우, 이 값을 false로 설정하세요. 기본 값: true 예제: --my-sql-settings '{"IgnoreOpenXaTransactionsCheck": false}'

MySQL용 소스 데이터 형식

다음 표에는 AWS DMS를 사용하는 경우 지원되는 MySQL 데이터베이스 원본 데이터 형식과 AWS DMS 데이터 형식으로부터의 기본 매핑이 나와 있습니다.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하십시오.

AWS DMS 데이터 형식에 관한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

MySQL 데이터 형식	AWS DMS 데이터 유형
INT	INT4
BIGINT	INT8

MySQL 데이터 형식	AWS DMS 데이터 유형
MEDIUMINT	INT4
TINYINT	INT1
SMALLINT	INT2
UNSIGNED TINYINT	UINT1
UNSIGNED SMALLINT	UINT2
UNSIGNED MEDIUMINT	UINT4
UNSIGNED INT	UINT4
UNSIGNED BIGINT	UINT8
DECIMAL(10)	NUMERIC (10,0)
BINARY	BYTES(1)
BIT	BOOLEAN
BIT(64)	BYTES(8)
BLOB	BYTES(65535)
LONGBLOB	BLOB
MEDIUMBLOB	BLOB
TINYBLOB	BYTES(255)
날짜	날짜

MySQL 데이터 형식	AWS DMS 데이터 유형
DATETIME	DATETIME 괄호 안의 값이 없는 DATETIME은 밀리초 없이 복제됩니다. 괄호 안의 값이 1~5인 DATETIME(예: `DATETIME(5)`)은 밀리초를 사용하여 복제됩니다. DATETIME 열을 복제할 때 대상에서 시간이 동일하게 유지됩니다. UTC로 변환되지 않습니다.
TIME	STRING
TIMESTAMP	DATETIME TIMESTAMP 열을 복제할 때는 대상에서 시간이 UTC로 변환됩니다.
YEAR	INT2
DOUBLE	REAL8
FLOAT	REAL(DOUBLE) FLOAT 값이 다음 범위에 속하지 않는 경우, 변환을 사용하여 FLOAT를 STRING에 매핑하세요. 변환에 대한 자세한 내용은 변환 규칙 및 작업 섹션을 참조하세요. 지원되는 FLOAT 범위는 -1.79E+308~-2.23E-308, 0, 그리고 2.23E-308~1.79E+308입니다.
VARCHAR (45)	WSTRING (45)
VARCHAR (2000)	WSTRING (2000)
VARCHAR (4000)	WSTRING (4000)
VARBINARY (4000)	BYTES (4000)

MySQL 데이터 형식	AWS DMS 데이터 유형
VARBINARY (2000)	BYTES (2000)
CHAR	WSTRING
TEXT	WSTRING
LONGTEXT	NCLOB
MEDIUMTEXT	NCLOB
TINYTEXT	WSTRING(255)
GEOMETRY	BLOB
POINT	BLOB
LINESTRING	BLOB
POLYGON	BLOB
MULTIPOINT	BLOB
MULTILINESTRING	BLOB
MULTIPOLYGON	BLOB
GEOMETRYCOLLECTION	BLOB
ENUM	WSTRING (*length*) 여기서 *length*는 ENUM에서 가장 긴 값의 길이입니다.
SET	WSTRING (*length*) 여기서 *length*는 쉼표를 포함하여 SET에 있는 모든 값의 총 길이입니다.
JSON	CLOB

> **Note**
>
> 경우에 따라 '0' 값으로 DATETIME 및 TIMESTAMP 데이터 형식(즉, 0000-00-00)을 지정할 수 있습니다. 이 경우, 복제 작업의 대상 데이터베이스가 DATETIME 및 TIMESTAMP 데이터 형식으로 '0' 값을 지원하는지 확인하세요. 그렇지 않으면, 이 값이 대상에서 null로 기록됩니다.

AWS DMS에서 SAP ASE 데이터베이스를 소스로 사용

AWS DMS를 사용하여 SAP Adaptive Server Enterprise(ASE) 데이터베이스(구 Sybase)의 데이터를 마이그레이션할 수 있습니다. SAP ASE 데이터베이스를 소스로 사용하여 지원되는 다른 AWS DMS 대상 데이터베이스로 데이터를 마이그레이션할 수 있습니다.

AWS DMS가 소스로 지원하는 SAP ASE 버전에 대한 자세한 내용은 출처: AWS DMS 섹션을 참조하세요.

SAP ASE 데이터베이스 및 AWS DMS 사용에 대한 자세한 내용은 다음 섹션을 참조하십시오.

주제

- SAP ASE 데이터베이스를 AWS DMS에서 소스로 사용하기 위한 사전 요구 사항
- SAP ASE를 AWS DMS에서 소스로 사용 시 적용되는 제한 사항
- AWS DMS에서 SAP ASE를 소스로 사용하는 데 필요한 권한
- 잘림 지점 제거
- SAP ASE를 AWS DMS 소스로 사용 시 엔드포인트 설정
- SAP ASE용 소스 데이터 형식

SAP ASE 데이터베이스를 AWS DMS에서 소스로 사용하기 위한 사전 요구 사항

SAP ASE 데이터베이스가 AWS DMS에서 원본으로 사용되려면 다음 작업을 수행해야 합니다.

- `sp_setreptable` 명령을 사용하여 테이블에서 SAP ASE 복제를 활성화합니다. 자세한 내용은 Sybase Infocenter Archive를 참조하세요.
- SAP ASE 데이터베이스에서 `RepAgent`를 비활성화합니다. 자세한 내용은 Stop and disable the RepAgent thread in the primary database를 참조하세요.
- 비라틴 문자(예: 중국어)에 대해 구성된 Windows EC2 인스턴스에 SAP ASE 버전 15.7을 복제하려면 대상 컴퓨터에 SAP ASE 15.7 SP121을 설치해야 합니다.

> **Note**
>
> 지속적 변경 데이터 캡처(CDC) 복제의 경우, DMS는 `dbcc logtransfer`와 `dbcc log`를 실행하여 트랜잭션 로그에서 데이터를 읽습니다.

SAP ASE를 AWS DMS에서 소스로 사용 시 적용되는 제한 사항

SAP ASE 데이터베이스를 AWS DMS에서 원본으로 사용 시 다음 제한 사항이 적용됩니다.

- 각 SAP ASE 데이터베이스마다 지속적 복제 또는 CDC를 사용하는 AWS DMS 작업은 하나만 실행할 수 있습니다. 여러 개의 전체 로드 전용 작업을 병렬로 실행할 수 있습니다.
- 테이블의 이름을 바꿀 수 없습니다. 예를 들어 다음 명령은 실패합니다.

```
sp_rename 'Sales.SalesRegion', 'SalesReg;
```

- 열의 이름을 바꿀 수 없습니다. 예를 들어 다음 명령은 실패합니다.

```
sp_rename 'Sales.Sales.Region', 'RegID', 'COLUMN';
```

- 이진 데이터 형식 문자열 끝에 있는 0 값은 대상 데이터베이스에 복제될 때 잘립니다. 예를 들어, 원본 테이블의 `0x0000000000000000000000001000001000000000`은 대상 테이블에서 `0x00000000000000000000000001000001`이 됩니다.
- 데이터베이스 기본값이 NULL 값을 허용하지 않도록 설정된 경우 AWS DMS에서는 NULL 값을 허용하지 않는 열이 포함된 대상 테이블을 생성합니다. 결과적으로 전체 로드 또는 CDC 복제 작업에 빈 값이 포함되어 있으면 AWS DMS에서 오류가 발생합니다. 다음 명령을 사용하여 원본 데이터베이스에서 NULL 값을 허용함으로써 이러한 오류를 방지할 수 있습니다.

```
sp_dboption database_name, 'allow nulls by default', 'true'
go
use database_name
CHECKPOINT
go
```

- `reorg rebuild` 인덱스 명령은 지원되지 않습니다.
- AWS DMS는 클러스터 또는 MSA(Multi-Site Availability)/Warm Standby를 소스로 사용하는 것을 지원하지 않습니다.
- 매핑 규칙에 `AR_H_TIMESTAMP` 변환 헤더 표현식을 사용하는 경우, 추가된 열에서는 밀리초가 캡처되지 않습니다.

- CDC 중에 병합 작업을 실행하면 복구할 수 없는 오류가 발생합니다. 대상을 다시 동기화하려면 전체 로드를 실행하세요.
- 데이터 행 잠금 체계를 사용하는 테이블에는 롤백 트리거 이벤트가 지원되지 않습니다.
- AWS DMS는 작업 범위 내의 테이블을 소스 SAP 데이터베이스에서 삭제한 후에 복제 작업을 재개할 수 없습니다. DMS 복제 작업을 중지하고 DML 작업(INSERT, UPDATE, DELETE)을 수행한 후 테이블을 삭제한 경우, 복제 작업을 다시 시작해야 합니다.

AWS DMS에서 SAP ASE를 소스로 사용하는 데 필요한 권한

SAP ASE 데이터베이스를 AWS DMS 작업에서 소스로 사용하려면 권한을 부여해야 합니다. AWS DMS 데이터베이스 정의에 지정된 사용자 계정에 SAP ASE 데이터베이스의 다음 권한을 부여합니다.

- sa_role
- replication_role
- sybase_ts_role
- 기본적으로 `sp_setreptable` 저장 프로시저를 실행할 권한이 필요한 경우, AWS DMS는 SAP ASE 복제 옵션을 활성화합니다. AWS DMS 자체를 통하지 않고 데이터베이스 엔드포인트에서 직접 `sp_setreptable`을 테이블에서 실행하려는 경우, `enableReplication` 추가 연결 속성을 사용할 수 있습니다. 자세한 내용은 SAP ASE를 AWS DMS 소스로 사용 시 엔드포인트 설정 섹션을 참조하세요.

잘림 지점 제거

작업이 시작되면, AWS DMS는 `syslogshold` 시스템 보기에 `$replication_truncation_point` 항목을 설정하여 복제가 진행되고 있음을 나타냅니다. AWS DMS가 작동하는 동안 대상에 이미 복사된 데이터 양에 따라 정기적으로 복제 잘림 지점을 진행합니다.

`$replication_truncation_point` 항목이 설정되고 나면 AWS DMS 작업을 항상 실행 상태로 유지하여 데이터베이스 로그가 과도하게 커지지 않게 해야 합니다. AWS DMS 작업을 영구적으로 중지하고 싶을 경우 다음 명령을 발행하여 복제 잘림 지점을 제거하십시오.

```
dbcc settrunc('ltm','ignore')
```

잘림 지점이 제거되고 나면 AWS DMS 작업을 다시 시작할 수 있습니다. 로그는 체크포인트에서 계속 자동으로 잘립니다(자동 잘림이 설정된 경우).

SAP ASE를 AWS DMS 소스로 사용 시 엔드포인트 설정

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 SAP ASE 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI에서 `create-endpoint` 명령을 `--sybase-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다.

SAP ASE를 소스로 하여 사용할 수 있는 엔드포인트 설정은 다음 표에 나와 있습니다.

이름	설명
`Charset`	이 속성을 국제 문자 집합에 해당하는 SAP ASE 이름으로 설정합니다. 기본값: `iso_1` 예: `--sybase-settings '{"Charset": "utf8"}'` 유효 값: • `acsii_8` • `big5hk` • `cp437` • `cp850` • `cp852` • `cp852` • `cp855` • `cp857` • `cp858` • `cp860` • `cp864` • `cp866` • `cp869` • `cp874` • `cp932`

이름	설명
	• cp936 • cp950 • cp1250 • cp1251 • cp1252 • cp1253 • cp1254 • cp1255 • cp1256 • cp1257 • cp1258 • deckanji • euccns • eucgb • eucjis • eucksc • gb18030 • greek8 • iso_1 • iso88592 • iso88595 • iso88596 • iso88597 • iso88598 • iso88599 • iso15 • kz1048 • koi8 • roman8

이름	설명
	• iso88599 • sjis • tis620 • turkish8 • utf8 SAP ASE 데이터베이스에서 지원되는 문자 집합에 대한 추가 질문은 Adaptive Server Enterprise: Supported character sets를 참조하세요.
EnableReplication	AWS DMS를 통하지 않고 데이터베이스 끝의 테이블에서 sp_setreptable 을 활성화하려면 이 속성을 설정합니다. 기본값: true 유효한 값: true 또는 false 예: --sybase-settings '{"EnableReplication": false}'
EncryptPassword	소스 데이터베이스에서 "net password encryption reqd"를 활성화한 경우, 이 속성을 설정합니다. 기본값: 0 유효한 값: 0, 1 또는 2 예: --sybase-settings '{"EncryptPassword": 1}' 이러한 파라미터 값에 대한 자세한 내용은 Adaptive Server Enterprise: Using the EncryptPassword Connection string property를 참조하세요.

이름	설명
`Provider`	ASE 15.7 이상 버전에 전송 계층 보안(TLS) 1.2를 사용하려면 이 속성을 설정하세요. AWS에는 TLS 버전 1.2 이상이 필요하며, 버전 1.3이 권장됩니다. 기본값: `Adaptive Server Enterprise` 유효한 값: `Adaptive Server Enterprise 16.03.06` 예: `--sybase-settings '{"Provider": "Adaptive Server Enterprise 16.03.06"}'`

SAP ASE용 소스 데이터 형식

AWS DMS를 사용하는 경우 지원되는 SAP ASE 원본 데이터 형식과 AWS DMS 데이터 형식으로부터의 기본 매핑의 목록은 다음 표를 참조하십시오. AWS DMS는 사용자 정의 형식(UDT) 데이터 형식 열이 있는 SAP ASE 원본 테이블은 지원하지 않습니다. 이 데이터 형식을 사용하는 복제된 열은 NULL로 생성됩니다.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 대상 엔드포인트의 마이그레이션에 적합한 대상 섹션을 참조하십시오.

AWS DMS 데이터 형식에 대한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

SAP ASE 데이터 형식	AWS DMS 데이터 형식
BIGINT	INT8
UNSIGNED BIGINT	UINT8
INT	INT4
UNSIGNED INT	UINT4
SMALLINT	INT2

SAP ASE 데이터 형식	AWS DMS 데이터 형식
UNSIGNED SMALLINT	UINT2
TINYINT	UINT1
DECIMAL	NUMERIC
NUMERIC	NUMERIC
FLOAT	REAL8
DOUBLE	REAL8
REAL	REAL4
MONEY	NUMERIC
SMALLMONEY	NUMERIC
DATETIME	DATETIME
BIGDATETIME	DATETIME(6)
SMALLDATETIME	DATETIME
DATE	DATE
TIME	TIME
BIGTIME	TIME
CHAR	STRING
UNICHAR	WSTRING
NCHAR	WSTRING
VARCHAR	STRING
UNIVARCHAR	WSTRING

SAP ASE 데이터 형식	AWS DMS 데이터 형식
NVARCHAR	WSTRING
BINARY	BYTES
VARBINARY	BYTES
BIT	BOOLEAN
TEXT	CLOB
UNITEXT	NCLOB
IMAGE	BLOB

MongoDB를 AWS DMS에서 소스로 사용

AWS DMS가 소스로 지원하는 MongoDB 버전에 대한 자세한 내용은 출처: AWS DMS 섹션을 참조하세요.

MongoDB 버전 지원에 대한 다음 사항에 유의하세요.

- AWS DMS 3.4.5 이상 버전은 MongoDB 버전 4.2와 4.4를 지원합니다.
- AWS DMS 3.4.5 이상 버전과 MongoDB 4.2 이상 버전은 분산 트랜잭션을 지원합니다. MongoDB 분산 트랜잭션에 대한 자세한 내용은 MongoDB 설명서의 Transactions를 참조하세요.
- AWS DMS 3.5.0 이상 버전은 MongoDB 3.6 이전 버전을 지원하지 않습니다.
- AWS DMS 3.5.1 이상 버전은 MongoDB 버전 5.0을 지원합니다.
- AWS DMS 3.5.2 이상 버전은 MongoDB 버전 6.0을 지원합니다.

MongoDB가 생소하다면 다음과 같은 MongoDB 데이터베이스의 중요 개념을 잘 이해하시기 바랍니다.

- MongoDB의 레코드는 필드와 값으로 구성된 데이터 구조인 문서입니다. 필드의 값에는 다른 문서, 배열 및 문서의 배열이 포함될 수 있습니다. 문서는 대체로 관계형 데이터베이스 테이블의 행에 상응합니다.
- MongoDB의 모음은 문서 그룹이며 대략 관계형 데이터베이스 테이블에 상응합니다.
- MongoDB의 데이터베이스는 모음 세트이며, 대략 관계형 데이터베이스의 스키마에 해당합니다.

- 내부적으로 MongoDB 문서는 이진 JSON(BSON) 파일에 문서의 각 필드별로 형식을 포함하는 압축 형식으로 저장됩니다. 각 문서에는 고유 ID가 있습니다.

AWS DMS는 MongoDB를 소스로 사용 시 두 가지 마이그레이션 모드, 즉 문서 모드 또는 테이블 모드를 지원합니다. MongoDB 엔드포인트를 생성할 때 또는 AWS DMS 콘솔에서 메타데이터 모드 파라미터를 설정하여 사용할 마이그레이션 모드를 지정합니다. 선택에 따라 엔드포인트 구성 패널에서 _id를 별도의 열로 확인 표시 버튼을 선택하여 프라이머리 키 역할을 하는 `_id`라는 두 번째 열을 생성할 수 있습니다.

마이그레이션 모드 선택은 아래 설명된 대로 결과로 얻는 대상 데이터의 형식에 영향을 미칩니다.

문서 모드

문서 모드에서 MongoDB 문서는 있는 그대로 마이그레이션됩니다. 즉 해당 문서 데이터는 대상 테이블에서 `_doc`라는 단일 열로 통합됩니다. 문서 모드는 MongoDB를 소스 엔드포인트로 사용할 때의 기본 설정입니다.

예를 들어, myCollection이라는 MongoDB 모음의 문서를 가정해 보십시오.

```
> db.myCollection.find()
{ "_id" : ObjectId("5a94815f40bd44d1b02bdfe0"), "a" : 1, "b" : 2, "c" : 3 }
{ "_id" : ObjectId("5a94815f40bd44d1b02bdfe1"), "a" : 4, "b" : 5, "c" : 6 }
```

데이터를 문서 모드를 사용하는 관계형 데이터베이스 테이블로 마이그레이션 하면, 데이터의 구조는 다음과 같습니다. MongoDB 문서의 데이터 필드는 `_doc` 열로 통합됩니다.

oid_id	_doc
5a94815f40bd44d1b02bdfe0	`{ "a" : 1, "b" : 2, "c" : 3 }`
5a94815f40bd44d1b02bdfe1	`{ "a" : 4, "b" : 5, "c" : 6 }`

선택적으로 추가 연결 속성 `extractDocID`를 true로 설정하여 프라이머리 키 역할을 하는 `"_id"`라는 두 번째 열을 생성할 수 있습니다. CDC를 사용할 경우, 이 파라미터를 true로 설정합니다.

문서 모드에서 AWS DMS는 다음과 같이 모음 생성 및 이름 바꾸기를 관리합니다.

- 소스 데이터베이스에 새 모음을 추가하면 AWS DMS가 모음의 새 대상 테이블을 생성하고 모든 문서를 복제합니다.
- 소스 데이터베이스에서 기존 모음의 이름을 바꾸면 AWS DMS는 대상 테이블의 이름을 바꾸지 않습니다.

대상 엔드포인트가 Amazon DocumentDB인 경우, 문서 모드에서 마이그레이션을 실행하세요.

테이블 모드

테이블 모드에서 AWS DMS는 MongoDB 문서의 각 상위 수준 필드를 대상 테이블의 열로 변환합니다. 필드가 중첩된 경우, AWS DMS는 중첩된 값을 단일 열로 평면화합니다. 그런 다음 AWS DMS는 대상 테이블의 열 집합에 키 필드와 데이터 형식을 추가합니다.

각 MongoDB 문서별로 AWS DMS는 각 키와 유형을 대상 테이블의 열 집합에 추가합니다. 예를 들어 AWS DMS는 테이블 모드를 사용해 앞의 예제를 다음 테이블로 마이그레이션합니다.

oid_id	a	b	c
5a94815f4 0bd44d1b02bdfe0	1	2	3
5a94815f4 0bd44d1b02bdfe1	4	5	6

중첩된 값은 점으로 분리된 키 이름을 포함하는 열에 평면화됩니다. 열은 마침표로 각각 분리된 연속된 평면화된 필드 이름으로 이름이 지정됩니다. 예를 들어 AWS DMS는 `a.b.c.`라는 이름이 지정된 열로 `{"a" : {"b" : {"c": 1}}}` 같은 중첩된 값 필드를 가진 JSON 문서를 마이그레이션합니다.

AWS DMS는 대상 열을 생성하기 위해 지정된 수의 MongoDB 문서를 스캔하고 모든 필드와 유형의 집합을 생성합니다. 그런 다음 AWS DMS는 이 집합을 사용해 대상 테이블의 열을 생성합니다. 콘솔을 사용해 MongoDB 소스 엔드포인트를 생성하거나 수정한다면, 스캔(검사)할 문서의 수를 지정할 수 있습니다. 기본 값은 1000 개 문서입니다. AWS CLI를 사용하는 경우, 추가 연결 속성 `docsToInvestigate`를 사용할 수 있습니다.

테이블 모드에서 AWS DMS는 문서와 모음을 다음과 같이 관리합니다.

- 기존 모음에 문서를 추가할 때 문서가 복제됩니다. 대상에 없는 필드가 있으면 이 필드는 복제되지 않습니다.

- 문서를 업데이트하면 업데이트된 문서가 복제됩니다. 대상에 없는 필드가 있으면 이 필드는 복제되지 않습니다.
- 문서 삭제는 완벽하게 지원됩니다.
- 새 모음을 CDC 작업 중에 추가해도 대상에 테이블이 새로 생성되지 않습니다.
- AWS DMS는 변경 데이터 캡처(CDC) 단계에서 모음 이름 변경을 지원하지 않습니다.

주제

- MongoDB를 AWS DMS 소스로 사용할 때 필요한 권한
- CDC를 위한 MongoDB 복제본 세트 구성
- AWS DMS에서 MongoDB를 소스로 사용하기 위한 보안 요구 사항
- MongoDB 모음을 분할하고 병렬로 마이그레이션
- MongoDB를 AWS DMS 소스로 사용 시 여러 데이터베이스 마이그레이션
- AWS DMS에서 MongoDB를 소스로 사용 시 적용되는 제한 사항
- MongoDB를 AWS DMS 소스로 사용 시 엔드포인트 구성 설정
- MongoDB용 소스 데이터 형식

MongoDB를 AWS DMS 소스로 사용할 때 필요한 권한

MongoDB 소스를 이용한 AWS DMS 마이그레이션의 경우에는 루트 권한을 가진 사용자 계정을 생성하거나 마이그레이션할 데이터베이스에서만 권한이 있는 사용자를 생성할 수 있습니다.

다음 코드는 루트 계정이 되도록 사용자를 생성합니다.

```
use admin
db.createUser(
  {
    user: "root",
    pwd: "password",
    roles: [ { role: "root", db: "admin" } ]
  }
)
```

MongoDB 3.x 소스의 경우, 다음 코드는 마이그레이션할 데이터베이스에 대한 최소 권한이 있는 사용자를 생성합니다.

```
use database_to_migrate
```

```
db.createUser(
{
    user: "dms-user",
    pwd: "password",
    roles: [ { role: "read", db: "local" }, "read"]
})
```

MongoDB 4.x 소스의 경우, 다음 코드는 최소 권한을 가진 사용자를 생성합니다.

```
{ resource: { db: "", collection: "" }, actions: [ "find", "changeStream" ] }
```

예를 들어 “admin” 데이터베이스에 다음 역할을 생성합니다.

```
use admin
db.createRole(
{
role: "changestreamrole",
privileges: [
{ resource: { db: "", collection: "" }, actions: [ "find","changeStream" ] }
],
roles: []
}
)
```

그리고 역할이 생성되면 마이그레이션할 데이터베이스에 사용자를 생성합니다.

```
> use test
> db.createUser(
{
user: "dms-user12345",
pwd: "password",
roles: [ { role: "changestreamrole", db: "admin" }, "read"]
})
```

CDC를 위한 MongoDB 복제본 세트 구성

MongoDB에서 지속적 복제 또는 CDC를 사용하려면 AWS DMS가 MongoDB 작업 로그(oplog)에 액세스할 수 있어야 합니다. oplog를 생성하려면 복제 세트를 배포하여 합니다(복제 세트가 없는 경우). 자세한 내용은 MongoDB 설명서를 참조하십시오.

CDC는 소스 엔드포인트로 설정된 MongoDB 복제본 세트의 기본 또는 보조 노드와 함께 사용할 수 있습니다.

독립형 인스턴스를 복제본 집합으로 변환하려면

1. 명령줄을 사용하여 mongo에 연결합니다.

```
mongo localhost
```

2. mongod 서비스를 중단합니다.

```
service mongod stop
```

3. 다음 명령을 사용하여 mongod를 다시 시작합니다.

```
mongod --replSet "rs0" --auth -port port_number
```

4. 다음 명령을 사용하여 복제본 집합에 대한 연결을 테스트합니다.

```
mongo -u root -p password --host rs0/localhost:port_number
  --authenticationDatabase "admin"
```

문서 모드 마이그레이션을 계획하고 있다면 MongoDB 엔드포인트 생성 시 `_id as a separate column` 옵션을 선택하십시오. 이 옵션을 선택하면 프라이머리 키로 작동하는 두 번째 열인 `_id` 가 생성됩니다. 이 두 번째 열은 AWS DMS가 데이터 조작 언어(DML) 작업을 지원하는 데 필요합니다.

> **Note**
>
> AWS DMS는 작업 로그(oplog)를 사용하여 지속적 복제 중의 변경 사항을 캡처합니다. AWS DMS가 레코드를 읽기 전에 MongoDB가 oplog에서 레코드를 삭제하면 작업이 실패합니다. 24시간 이상 변경 사항이 유지되도록 oplog의 크기를 조정하는 것이 좋습니다.

AWS DMS에서 MongoDB를 소스로 사용하기 위한 보안 요구 사항

AWS DMS는 MongoDB의 두 가지 인증 메서드를 지원합니다. 이 두 가지 인증 메서드는 암호를 암호화하는 데 사용되므로 `authType` 파라미터를 PASSWORD로 설정할 때만 사용됩니다.

MongoDB 인증 메서드는 다음과 같습니다.

- MONGODB-CR - 이전 버전과의 호환성을 위해
- SCRAM-SHA-1 - MongoDB 버전 3.x 및 4.0 사용 시 기본값

인증 메서드가 지정되어 있지 않으면 AWS DMS는 MongoDB 소스 버전의 기본 메서드를 사용합니다.

MongoDB 모음을 분할하고 병렬로 마이그레이션

마이그레이션 작업의 성능을 개선하기 위해 MongoDB 소스 엔드포인트는 테이블 매핑에서 병렬 전체 로드 옵션 두 가지를 지원합니다.

다시 말해 JSON 설정에서 병렬 전체 로드를 위해 테이블 매핑에 자동 분할 또는 범위 분할을 사용하여 모음을 병렬로 마이그레이션할 수 있습니다. 자동 분할을 사용하면 AWS DMS가 각 스레드에서 마이그레이션 소스를 자동으로 분할하는 기준을 지정할 수 있습니다. 범위 분할을 사용하면 DMS가 각 스레드에서 마이그레이션할 각 세그먼트의 특정 범위를 AWS DMS에 지시할 수 있습니다. 이러한 설정에 대한 자세한 내용은 테이블 및 컬렉션 설정 규칙과 작업 섹션을 참조하세요.

자동 분할 범위를 사용하여 MongoDB 데이터베이스를 병렬로 마이그레이션

AWS DMS가 각 스레드의 데이터를 자동으로 파티셔닝(분할)하는 기준을 지정하여 문서를 병렬로 마이그레이션할 수 있습니다. 특히 스레드당 마이그레이션할 문서 수를 지정합니다. AWS DMS는 이 방식을 사용하여 스레드당 성능을 극대화하는 세그먼트 경계 최적화를 시도합니다.

테이블 매핑에서 다음 테이블 설정 옵션을 사용하여 분할 기준을 지정할 수 있습니다.

테이블 설정 옵션	설명
`"type"`	(필수) MongoDB를 소스로 사용하는 경우 `"partitions-auto"` 로 설정합니다.
`"number-of-partitions"`	(선택 사항) 마이그레이션에 사용되는 총 파티션(세그먼트) 수. 기본값은 16입니다.
`"collection-count-from-metadata"`	(선택 사항) 이 옵션을 `true`로 설정하면 AWS DMS는 추정 모음 개수를 사용하여 파티션 수를 결정합니다. 이 옵션이 `false`로 설정된 경우, AWS DMS는 실제 모음 수를 사용합니다. 기본값은 `true`입니다.
`"max-records-skip-per-page"`	(선택 사항) 각 파티션의 경계를 결정할 때 한 번에 건너뛰는 레코드 수입니다. AWS DMS는 페이지 매김 건너뛰기 방식을 사용하여 파티션의 최소 경계를 결정합니다. 기본값은 10,000입니다.

테이블 설정 옵션	설명
	상대적으로 큰 값을 설정하면 커서 시간이 초과되고 작업이 실패할 수 있습니다. 상대적으로 작은 값을 설정하면 페이지당 작업 수가 늘어나고 전체 로드 속도가 느려집니다.
"batch-size"	(선택 사항) 배치 하나에서 반환되는 문서 수를 제한합니다. 배치마다 서버를 왕복해야 합니다. 배치 크기가 0인 경우, 커서는 서버에서 정의한 최대 배치 크기를 사용합니다. 기본값은 0입니다.

다음 예제는 자동 분할을 위한 테이블 매핑을 보여 줍니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "admin",
                "table-name": "departments"
            },
            "rule-action": "include",
            "filters": []
        },
        {
            "rule-type": "table-settings",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "admin",
                "table-name": "departments"
            },
            "parallel-load": {
                "type": "partitions-auto",
                "number-of-partitions": 5,
                "collection-count-from-metadata": "true",
                "max-records-skip-per-page": 1000000,
                "batch-size": 50000
```

```
            }
        }
    ]
}
```

자동 분할에는 다음과 같은 제한이 있습니다. 각 세그먼트의 마이그레이션은 모음 개수와 모음의 최소 `_id`를 개별적으로 가져옵니다. 그런 다음 페이지 매김 건너뛰기를 사용하여 해당 세그먼트의 최소 경계를 계산합니다.

따라서 모음의 모든 세그먼트 경계를 계산할 때까지 각 모음의 최소 `_id` 값이 일정하게 유지되어야 합니다. 세그먼트 경계를 계산하는 동안 모음의 최소 `_id` 값을 변경하면 데이터가 손실되거나 행 중복 오류가 발생할 수 있습니다.

범위 분할을 사용하여 MongoDB 데이터베이스를 병렬로 마이그레이션

스레드의 각 세그먼트마다 범위를 지정하여 문서를 병렬로 마이그레이션할 수 있습니다. 이 방식을 사용하면 스레드당 선택한 문서 범위에 따라 각 스레드에서 마이그레이션할 특정 문서를 AWS DMS에 알려 줄 수 있습니다.

다음 예제에 나온 MongoDB 모음에는 항목 7개와 프라이머리 키 `_id`가 있습니다.

Key	Value	Type
(1) ObjectId("5f805c74873173399a278d78")	{ 3 fields }	Object
_id	ObjectId("5f805c74873173399a278d78")	ObjectId
num	1	Int32
name	a	String
(2) ObjectId("5f805c97873173399a278d79")	{ 3 fields }	Object
_id	ObjectId("5f805c97873173399a278d79")	ObjectId
num	2	Int32
name	b	String
(3) ObjectId("5f805cb0873173399a278d7a")	{ 3 fields }	Object
_id	ObjectId("5f805cb0873173399a278d7a")	ObjectId
num	3	Int32
name	c	String
(4) ObjectId("5f805cbb873173399a278d7b")	{ 3 fields }	Object
_id	ObjectId("5f805cbb873173399a278d7b")	ObjectId
num	4	Int32
name	d	String
(5) ObjectId("5f805cc5873173399a278d7c")	{ 3 fields }	Object
_id	ObjectId("5f805cc5873173399a278d7c")	ObjectId
num	5	Int32
name	e	String
(6) ObjectId("5f805cd0873173399a278d7d")	{ 3 fields }	Object
_id	ObjectId("5f805cd0873173399a278d7d")	ObjectId
num	6	Int32
name	f	String
(7) ObjectId("5f805cdd873173399a278d7e")	{ 3 fields }	Object
_id	ObjectId("5f805cdd873173399a278d7e")	ObjectId
num	7	Int32
name	g	String

AWS DMS이 병렬로 마이그레이션할 3개의 특정 세그먼트로 모음을 분할하려면 마이그레이션 작업에 테이블 매핑 규칙을 추가하면 됩니다. 이 방법은 다음 JSON 예제에 나와 있습니다.

```
{ // Task table mappings:
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "testdatabase",
        "table-name": "testtable"
      },
      "rule-action": "include"
```

```
    }, // "selection" :"rule-type"
    {
      "rule-type": "table-settings",
      "rule-id": "2",
      "rule-name": "2",
      "object-locator": {
        "schema-name": "testdatabase",
        "table-name": "testtable"
      },
      "parallel-load": {
        "type": "ranges",
        "columns": [
           "_id",
           "num"
        ],
        "boundaries": [
          // First segment selects documents with _id less-than-or-equal-to
 5f805c97873173399a278d79
          // and num less-than-or-equal-to 2.
          [
             "5f805c97873173399a278d79",
             "2"
          ],
          // Second segment selects documents with _id > 5f805c97873173399a278d79 and
          // _id less-than-or-equal-to 5f805cc5873173399a278d7c and
          // num > 2 and num less-than-or-equal-to 5.
          [
             "5f805cc5873173399a278d7c",
             "5"
          ]
          // Third segment is implied and selects documents with _id >
 5f805cc5873173399a278d7c.
       ] // :"boundaries"
      } // :"parallel-load"
    } // "table-settings" :"rule-type"
  ] // :"rules"
} // :Task table mappings
```

이 테이블 매핑 정의는 소스 모음을 세 개의 세그먼트로 분할하고 병렬로 마이그레이션합니다. 분할 경계는 다음과 같습니다.

```
Data with _id less-than-or-equal-to "5f805c97873173399a278d79" and num less-than-or-
equal-to 2 (2 records)
Data with _id > "5f805c97873173399a278d79" and num > 2 and _id  less-than-or-equal-to
 "5f805cc5873173399a278d7c" and num less-than-or-equal-to 5 (3 records)
Data with _id > "5f805cc5873173399a278d7c" and num > 5 (2 records)
```

마이그레이션 작업이 완료되면 다음 예제와 같이 작업 로그에서 테이블이 병렬로 로드된 것을 확인할 수 있습니다. 소스 테이블에서 각 세그먼트를 언로드하는 데 사용된 MongoDB `find` 절도 확인할 수 있습니다.

```
[TASK_MANAGER    ] I:  Start loading segment #1 of 3 of table
 'testdatabase'.'testtable' (Id = 1) by subtask 1. Start load timestamp
 0005B191D638FE86  (replicationtask_util.c:752)

[SOURCE_UNLOAD   ] I:  Range Segmentation filter for Segment #0 is initialized.
 (mongodb_unload.c:157)

[SOURCE_UNLOAD   ] I:  Range Segmentation filter for Segment #0 is: { "_id" :
 { "$lte" : { "$oid" : "5f805c97873173399a278d79" } }, "num" : { "$lte" :
 { "$numberInt" : "2" } } }  (mongodb_unload.c:328)

[SOURCE_UNLOAD   ] I: Unload finished for segment #1 of segmented table
 'testdatabase'.'testtable' (Id = 1). 2 rows sent.

[TASK_MANAGER    ] I: Start loading segment #1 of 3 of table
 'testdatabase'.'testtable' (Id = 1) by subtask 1. Start load timestamp
 0005B191D638FE86 (replicationtask_util.c:752)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is initialized.
 (mongodb_unload.c:157)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is: { "_id" : { "$lte" :
 { "$oid" : "5f805c97873173399a278d79" } }, "num" : { "$lte" : { "$numberInt" :
 "2" } } } (mongodb_unload.c:328)

[SOURCE_UNLOAD   ] I: Unload finished for segment #1 of segmented table
 'testdatabase'.'testtable' (Id = 1). 2 rows sent.

[TARGET_LOAD     ] I: Load finished for segment #1 of segmented table
 'testdatabase'.'testtable' (Id = 1). 1 rows received. 0 rows skipped. Volume
 transfered 480.
```

```
[TASK_MANAGER    ] I: Load finished for segment #1 of table
 'testdatabase'.'testtable' (Id = 1) by subtask 1. 2 records transferred.
```

현재, AWS DMS는 다음과 같은 MongoDB 데이터 형식을 세그먼트 키 열로 지원합니다.

- Double
- String
- ObjectId
- 32비트 정수
- 64비트 정수

MongoDB를 AWS DMS 소스로 사용 시 여러 데이터베이스 마이그레이션

AWS DMS 버전 3.4.5 이상은 지원되는 모든 MongoDB 버전에서 단일 작업으로 여러 데이터베이스 마이그레이션을 지원합니다. 여러 데이터베이스를 마이그레이션하려면 다음 단계에 따릅니다.

1. MongoDB 소스 엔드포인트를 생성할 때 다음 중 하나를 수행합니다.
   - DMS 콘솔의 엔드포인트 생성 페이지에서 엔드포인트 구성 아래 데이터베이스 이름을 비워 둡니다.
   - AWS CLI `CreateEndpoint` 명령을 사용하여 `MongoDBSettings`의 `DatabaseName` 파라미터에 빈 문자열 값을 할당합니다.
2. MongoDB 소스에서 마이그레이션하려는 각 데이터베이스의 데이터베이스 이름을 작업 테이블 매핑의 스키마 이름으로 지정합니다. 콘솔의 안내 입력을 사용하거나 JSON에서 직접 이 작업을 수행할 수 있습니다. 안내 입력에 대한 자세한 내용은 콘솔에서 테이블 선택 및 변환 규칙 지정 섹션을 참조하세요. JSON에 대한 자세한 내용은 선택 규칙 및 작업 섹션을 참조하세요.

예를 들면 다음과 같이 JSON을 지정하여 MongoDB 데이터베이스 3개를 마이그레이션할 수 있습니다.

Example 스키마에서 모든 테이블 마이그레이션

다음 JSON은 소스 엔드포인트에 있는 `Customers`, `Orders`, `Suppliers` 데이터베이스의 모든 테이블을 대상 엔드포인트로 마이그레이션합니다.

```
{
    "rules": [
        {
```

```
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "Customers",
                "table-name": "%"
            },
            "rule-action": "include",
            "filters": []
        },
        {
            "rule-type": "selection",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "Orders",
                "table-name": "%"
            },
            "rule-action": "include",
            "filters": []
        },
        {
            "rule-type": "selection",
            "rule-id": "3",
            "rule-name": "3",
            "object-locator": {
                "schema-name": "Inventory",
                "table-name": "%"
            },
            "rule-action": "include",
            "filters": []
        }
    ]
}
```

AWS DMS에서 MongoDB를 소스로 사용 시 적용되는 제한 사항

다음은 MongoDB를 AWS DMS에서 원본으로 사용 시 제한 사항입니다.

- 테이블 모드에서 모음에 있는 문서의 데이터 형식은 동일한 필드의 값에 사용하는 데이터 형식과 일치해야 합니다. 예를 들어 모음의 문서에 `'{ a:{ b:value ... }'`가 포함된 경우, 해당 `a.b` 필드의 *value*를 참조하는 모음의 모든 문서는 모음의 어디에 표시되든 *value*에 동일한 데이터 형식을 사용해야 합니다.

- `_id` 옵션이 별도의 열로 설정된 경우 ID 문자열이 200자를 초과할 수 없습니다.
- 객체 ID와 배열 형식 키는 테이블 모드에서 `oid` 및 `array` 접두사가 붙은 열로 변환됩니다.

  내부적으로 이러한 열은 접두사 이름으로 참조됩니다. AWS DMS에서 이러한 열을 참조하는 변환 규칙을 사용하는 경우, 접두사가 붙은 열을 지정해야 합니다. 예를 들면 `${oid__id}`를 지정하고 `${_id}`를 지정하지 않거나 `${array__addresses}`를 지정하고 `${_addresses}`는 지정하지 않습니다.
- 모음 이름과 키 이름에는 달러 기호($)를 포함할 수 없습니다.
- AWS DMS는 RDBMS 대상이 있는 테이블 모드에서 대소문자가 다른 동일한 필드를 포함하는 모음은 지원하지 않습니다. 예를 들어 AWS DMS는 `Field1`과 `field1`이라는 두 개의 모음을 사용할 수 없습니다.
- 테이블 모드와 문서 모드에는 앞서 설명한 제한이 있습니다.
- 자동 분할을 사용한 병렬 마이그레이션에는 앞에서 설명한 제한 사항이 있습니다.
- MongoDB에는 소스 필터가 지원되지 않습니다.
- AWS DMS는 중첩 수준이 97보다 큰 문서는 지원하지 않습니다.
- AWS DMS는 MongoDB 버전 5.0의 다음 기능을 지원하지 않습니다.
  - 라이브 리샤딩
  - 클라이언트 측 필드 수준 암호화(CSFLE)
  - 시계열 모음 마이그레이션

> **Note**
>
> Amazon DocumentDB는 시계열 모음을 지원하지 않기 때문에 전체 로드 단계에서 마이그레이션된 시계열 모음은 Amazon DocumentDB에서 일반 모음으로 변환됩니다.

MongoDB를 AWS DMS 소스로 사용 시 엔드포인트 구성 설정

MongoDB 소스 엔드포인트를 설정할 때 AWS DMS 콘솔을 사용하여 여러 엔드포인트 구성 설정을 지정할 수 있습니다.

다음 표에서는 MongoDB 데이터베이스를 AWS DMS 소스로 사용할 때 제공되는 구성 설정을 설명합니다.

설정(속성)	유효값	기본값과 설명
인증 모드	"none" "password"	값 "password" 는 사용자 이름과 암호를 입력하라는 메시지를 표시합니다. "none"을 지정하면 사용자 이름 및 암호 파라미터가 사용되지 않습니다.
인증 소스	유효한 MongoDB 데이터베이스 이름입니다.	인증을 위한 보안 인증 정보를 검증하는 데 사용할 MongoDB 데이터베이스의 이름입니다. 기본 값은 "admin"입니다.
인증 메커니즘	"default" "mongodb_cr" "scram_sha_1"	인증 메커니즘입니다. "default" 값은 "scram_sha_1" 입니다. authType이 "no"로 설정된 경우에는 이 설정이 사용되지 않습니다.
메타데이터 모드	문서 및 테이블	문서 모드 또는 테이블 모드를 선택합니다.
스캔할 문서 수(docsToInvestigate)	0보다 큰 양의 정수입니다.	이 옵션은 테이블 모드에서 대상 테이블 정의를 정의할 때만 사용하세요.
_id를 별도의 열로	상자의 확인 표시	프라이머리 키 역할을 하는 _id라는 두 번째 열을 생성하는 선택적 확인 표시 상자입니다.
socketTimeoutMS	NUMBER 추가 연결 속성(ECA) 전용입니다.	이 설정은 밀리초 단위이며 MongoDB 클라이언트의 연결 제한 시간을 구성합니다. 값이 0보다 작거나 같으면 MongoDB 클라이언트 기본값이 사용됩니다.
UseUpdateLookUp	불 true false	true인 경우, CDC 업데이트 이벤트 중에 AWS DMS는 업데이트된 전체 문서를 대상으로 복사합니다. false로 설정하면 AWS DMS는 MongoDB update 명령을 사용하여 대상에서 문서의 수정된 필드만 업데이트합니다.
ReplicateShardCollections	불 true	true인 경우, AWS DMS는 데이터를 샤드 모음에 복제합니다. AWS DMS는 대상 엔드포인트가 DocumentDB Elastic 클러스터인 경우에만 이 설정을 사용합니다.

설정(속성)	유효값	기본값과 설명
	`false`	이 설정이 true인 경우 다음 사항에 유의하세요. • `TargetTablePrepMode` 를 `nothing`으로 설정해야 합니다. • AWS DMS는 자동으로 `useUpdateLookup` 을 `false`로 설정합니다.

문서를 메타데이터 모드로 선택하면 다른 옵션을 사용할 수 있습니다.

대상 엔드포인트가 DocumentDB인 경우, 문서 모드에서 마이그레이션을 실행해야 합니다. 또한 소스 엔드포인트를 수정하고 _id를 별도의 열로 옵션을 선택합니다. 이는 소스 MongoDB 워크로드에 트랜잭션이 포함된 경우에 필수 사전 요구 사항입니다.

MongoDB용 소스 데이터 형식

AWS DMS용 원본으로 MongoDB를 사용하는 데이터 마이그레이션은 대다수 MongoDB 데이터 형식을 지원합니다. 다음 표에서 AWS DMS를 사용하는 경우 지원되는 MongoDB 원본 데이터 형식과 AWS DMS 데이터 형식으로부터의 기본 매핑을 확인할 수 있습니다. MongoDB 데이터 형식에 대한 자세한 내용은 MongoDB 설명서의 BSON Types(BSON 형식)를 참조하십시오.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하십시오.

AWS DMS 데이터 형식에 관한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

MongoDB 데이터 형식	AWS DMS 데이터 유형
불	부울
바이너리	BLOB
날짜	날짜
타임스탬프	날짜
정수	INT4

MongoDB 데이터 형식	AWS DMS 데이터 유형
Long	INT8
Double	REAL8
String(UTF-8)	CLOB
배열	CLOB
OID	String
REGEX	CLOB
코드	CLOB

Amazon DocumentDB(MongoDB 호환)를 AWS DMS 소스로 사용

AWS DMS가 소스로 지원하는 Amazon DocumentDB(MongoDB 호환) 버전에 대한 자세한 내용은 출처: AWS DMS 섹션을 참조하세요.

Amazon DocumentDB를 소스로 사용하여 한 Amazon DocumentDB 클러스터에서 다른 Amazon DocumentDB 클러스터로 데이터를 마이그레이션할 수 있습니다. Amazon DocumentDB 클러스터의 데이터를 AWS DMS가 지원하는 다른 대상 엔드포인트 중 하나로 마이그레이션할 수도 있습니다.

Amazon DocumentDB를 처음 사용하는 경우, Amazon DocumentDB 데이터베이스의 다음 중요 개념을 알고 있어야 합니다.

- Amazon DocumentDB의 레코드는 필드와 값의 쌍으로 구성된 데이터 구조인 문서입니다. 필드의 값에는 다른 문서, 배열 및 문서의 배열이 포함될 수 있습니다. 문서는 대체로 관계형 데이터베이스 테이블의 행에 상응합니다.
- Amazon DocumentDB의 모음은 문서 그룹이며, 대략 관계형 데이터베이스 테이블에 해당합니다.
- Amazon DocumentDB의 데이터베이스는 모음 세트이며, 대략 관계형 데이터베이스의 스키마에 해당합니다.

AWS DMS는 Amazon DocumentDB를 소스로 사용할 때 두 가지 마이그레이션 모드, 즉 문서 모드와 테이블 모드를 지원합니다. AWS DMS 콘솔에서 Amazon DocumentDB 소스 엔드포인트를 생성할 때 메타데이터 모드 옵션 또는 `nestingLevel` 추가 연결 속성을 사용하여 마이그레이션 모드를 지정합

니다. 마이그레이션 모드 선택이 대상 데이터의 형식에 결과적으로 어떤 영향을 미치는지 아래에서 설명합니다.

문서 모드

문서 모드에서는 JSON 문서가 그대로 마이그레이션됩니다. 즉, 문서 데이터가 두 항목 중 하나로 통합됩니다. 관계형 데이터베이스를 대상으로 사용하는 경우, 데이터는 대상 테이블에서 이름이 _doc인 단일 열입니다. 비관계형 데이터베이스를 대상으로 사용하는 경우, 데이터는 단일 JSON 문서입니다. 문서 모드가 기본 모드이며, Amazon DocumentDB 대상으로 마이그레이션할 때 사용하는 것이 좋습니다.

예를 들어 myCollection이라는 Amazon DocumentDB 모음에 있는 다음 문서를 생각해 보세요.

```
> db.myCollection.find()
{ "_id" : ObjectId("5a94815f40bd44d1b02bdfe0"), "a" : 1, "b" : 2, "c" : 3 }
{ "_id" : ObjectId("5a94815f40bd44d1b02bdfe1"), "a" : 4, "b" : 5, "c" : 6 }
```

데이터를 문서 모드를 사용하는 관계형 데이터베이스 테이블로 마이그레이션 하면, 데이터의 구조는 다음과 같습니다. 문서의 데이터 필드는 _doc 열로 통합됩니다.

oid_id	_doc
5a94815f40bd44d1b02bdfe0	{ "a" : 1, "b" : 2, "c" : 3 }
5a94815f40bd44d1b02bdfe1	{ "a" : 4, "b" : 5, "c" : 6 }

선택적으로 추가 연결 속성 extractDocID를 true로 설정하여 프라이머리 키 역할을 하는 "_id"라는 두 번째 열을 생성할 수 있습니다. 변경 데이터 캡처(CDC)를 사용하려는 경우, Amazon DocumentDB를 대상으로 사용하는 경우를 제외하고 이 파라미터를 true로 설정합니다.

> **Note**
>
> 소스 데이터베이스에 새 모음을 추가하면 AWS DMS가 모음의 새 대상 테이블을 생성하고 모든 문서를 복제합니다.

테이블 모드

테이블 모드에서 AWS DMS는 Amazon DocumentDB 문서의 각 최상위 수준 필드를 대상 테이블의 열로 변환합니다. 필드가 중첩된 경우, AWS DMS는 중첩된 값을 단일 열로 평면화합니다. 그런 다음 AWS DMS는 대상 테이블의 열 집합에 키 필드와 데이터 형식을 추가합니다.

AWS DMS는 각 Amazon DocumentDB 문서별로 각 키와 유형을 대상 테이블의 열 집합에 추가합니다. 예를 들어 AWS DMS는 테이블 모드를 사용해 앞의 예제를 다음 테이블로 마이그레이션합니다.

oid_id	a	b	c
5a94815f4 0bd44d1b02bdfe0	1	2	3
5a94815f4 0bd44d1b02bdfe1	4	5	6

중첩된 값은 점으로 분리된 키 이름을 포함하는 열에 평면화됩니다. 열의 이름은 마침표로 구분되고 평면화된 필드 이름의 연결을 사용하여 지정됩니다. 예를 들어 AWS DMS는 `a.b.c.`라는 이름이 지정된 열로 `{"a" : {"b" : {"c": 1}}}` 같은 중첩된 값 필드를 가진 JSON 문서를 마이그레이션합니다.

AWS DMS는 대상 열을 생성하기 위해 지정된 수의 Amazon DocumentDB 문서를 스캔하고 모든 필드와 유형의 집합을 생성합니다. 그런 다음 AWS DMS는 이 집합을 사용해 대상 테이블의 열을 생성합니다. 콘솔을 사용하여 Amazon DocumentDB 소스 엔드포인트를 생성하거나 수정하는 경우, 스캔할 문서 수를 지정할 수 있습니다. 기본값은 1,000 개의 문서입니다. AWS CLI를 사용하는 경우, 추가 연결 속성 `docsToInvestigate`를 사용할 수 있습니다.

테이블 모드에서 AWS DMS는 문서와 모음을 다음과 같이 관리합니다.

- 기존 모음에 문서를 추가할 때 문서가 복제됩니다. 대상에 없는 필드가 있으면 이 필드는 복제되지 않습니다.
- 문서를 업데이트하면 업데이트된 문서가 복제됩니다. 대상에 없는 필드가 있으면 이 필드는 복제되지 않습니다.
- 문서 삭제는 완벽하게 지원됩니다.
- 새 모음을 CDC 작업 중에 추가해도 대상에 테이블이 새로 생성되지 않습니다.

- AWS DMS는 변경 데이터 캡처(CDC) 단계에서 모음 이름 변경을 지원하지 않습니다.

주제

- Amazon DocumentDB를 소스로 사용하기 위한 권한 설정
- Amazon DocumentDB 클러스터를 위한 CDC 구성
- TLS를 사용하여 Amazon DocumentDB에 연결
- Amazon DocumentDB 소스 엔드포인트 생성
- Amazon DocumentDB 모음을 분할하고 병렬로 마이그레이션
- Amazon DocumentDB를 AWS DMS 소스로 사용 시 여러 데이터베이스 마이그레이션
- Amazon DocumentDB를 AWS DMS 소스로 사용 시 제한 사항
- Amazon DocumentDB를 소스로 하는 엔드포인트 설정 사용
- Amazon DocumentDB의 소스 데이터 형식

Amazon DocumentDB를 소스로 사용하기 위한 권한 설정

Amazon DocumentDB를 AWS DMS 마이그레이션의 소스로 사용하는 경우, 루트 권한이 있는 사용자 계정을 생성할 수 있습니다. 또는 마이그레이션할 데이터베이스에 대한 권한만 있는 사용자를 생성할 수도 있습니다.

다음 코드는 루트 계정으로 사용자를 생성합니다.

```
use admin
db.createUser(
  {
    user: "root",
    pwd: "password",
    roles: [ { role: "root", db: "admin" } ]
  })
```

Amazon DocumentDB 3.6의 경우, 다음 코드는 마이그레이션할 데이터베이스에 대한 최소 권한이 있는 사용자를 생성합니다.

```
use database_to_migrate
db.createUser(
```

```
{
    user: "dms-user",
    pwd: "password",
    roles: [ { role: "read", db: "db_name" }, "read"]
})
```

Amazon DocumentDB 4.0 이상의 경우, AWS DMS는 배포 전체의 변경 스트림을 사용합니다. 여기서 다음 코드는 최소 권한을 가진 사용자를 생성합니다.

```
db.createUser(
{
    user: "dms-user",
    pwd: "password",
    roles: [ { role: "readAnyDatabase", db: "admin" }]
})
```

Amazon DocumentDB 클러스터를 위한 CDC 구성

Amazon DocumentDB에서 지속적 복제 또는 CDC를 사용하려면 AWS DMS가 Amazon DocumentDB 클러스터의 변경 스트림에 액세스할 수 있어야 합니다. 클러스터 모음 및 데이터베이스의 시간순 업데이트 이벤트 시퀀스에 대한 설명은 Amazon DocumentDB 개발자 안내서의 변경 스트림 사용을 참조하세요.

MongoDB 쉘을 사용하여 Amazon DocumentDB 클러스터에 인증합니다. 그러고 나서 다음 명령을 실행하여 변경 스트림을 활성화합니다.

```
db.adminCommand({modifyChangeStreams: 1,
    database: "DB_NAME",
    collection: "",
    enable: true});
```

이 방식을 사용하면 데이터베이스의 모든 모음에 변경 스트림을 활성화할 수 있습니다. 변경 스트림을 활성화한 후 기존 데이터를 마이그레이션하는 동시에 진행 중인 변경 사항을 복제하는 마이그레이션 작업을 생성할 수 있습니다. AWS DMS는 대량 데이터가 로드된 후에도 변경 사항을 계속 캡처하고 적용합니다. 결국 소스 데이터베이스와 대상 데이터베이스가 동기화되어 마이그레이션의 가동 중지 시간이 최소화됩니다.

> **Note**
>
> AWS DMS는 작업 로그(oplog)를 사용하여 지속적 복제 중의 변경 사항을 캡처합니다. AWS DMS가 레코드를 읽기 전에 Amazon DocumentDB가 oplog에서 레코드를 삭제하면 작업이 실패합니다. 24시간 이상 변경 사항이 유지되도록 oplog의 크기를 조정하는 것이 좋습니다.

TLS를 사용하여 Amazon DocumentDB에 연결

기본적으로 새로 생성한 Amazon DocumentDB 클러스터는 전송 계층 보안(TLS)을 사용한 보안 연결만 허용합니다. TLS가 활성화되면 Amazon DocumentDB에 대한 모든 연결에 퍼블릭 키가 필요합니다.

AWS에서 호스팅된 Amazon S3 버킷에서 `rds-combined-ca-bundle.pem` 파일을 다운로드하여 Amazon DocumentDB의 퍼블릭 키를 검색할 수 있습니다. 이 파일을 다운로드하는 방법에 관한 자세한 내용은 Amazon DocumentDB 개발자 안내서의 TLS를 사용한 연결 암호화를 참조하세요.

`rds-combined-ca-bundle.pem` 파일을 다운로드한 후 파일에 포함된 퍼블릭 키를 AWS DMS에 가져올 수 있습니다. 다음 단계에서 그 방법을 설명합니다.

AWS DMS 콘솔을 사용하여 퍼블릭 키를 가져오려면

1. AWS Management Console에 로그인하고 AWS DMS를 선택합니다.
2. 탐색 창에서 [Certificates]를 선택합니다.
3. 인증서 가져오기를 선택합니다. 새 CA 인증서 가져오기 페이지가 나타납니다.
4. 인증서 구성 섹션에서 다음 중 하나를 수행합니다.
   - 인증서 식별자에 해당 인증서의 고유 이름(예: `docdb-cert`)을 입력합니다.
   - 파일 선택을 선택하고 `rds-combined-ca-bundle.pem` 파일을 저장한 위치로 이동한 다음 파일을 선택합니다.
5. 새 CA 인증서 추가를 선택합니다.

다음 AWS CLI 예제에서는 AWS DMS `import-certificate` 명령을 사용하여 퍼블릭 키 `rds-combined-ca-bundle.pem` 파일을 가져옵니다.

```
aws dms import-certificate \
    --certificate-identifier docdb-cert \
```

```
    --certificate-pem file://./rds-combined-ca-bundle.pem
```

Amazon DocumentDB 소스 엔드포인트 생성

콘솔 또는 AWS CLI를 사용하여 Amazon DocumentDB 소스 엔드포인트를 생성할 수 있습니다. 콘솔에서는 다음 절차를 사용하세요.

AWS DMS 콘솔을 사용하여 Amazon DocumentDB 소스 엔드포인트를 구성하려면

1. AWS Management Console에 로그인하고 AWS DMS를 선택합니다.
2. 탐색 창에서 엔드포인트를 선택하고 엔드포인트 생성을 선택합니다.
3. 엔드포인트 식별자에 쉽게 식별할 수 있는 이름(예: `docdb-source`)을 제공합니다.
4. 소스 엔진에서 Amazon DocumentDB(MongoDB 호환)를 선택합니다.
5. 서버 이름에 Amazon DocumentDB 데이터베이스 엔드포인트가 있는 서버의 이름을 입력합니다. 예를 들어 Amazon EC2 인스턴스의 퍼블릭 DNS 이름(예: `democluster.cluster-cjf6q8nxfefi.us-east-2.docdb.amazonaws.com`)을 입력할 수 있습니다.
6. 포트에 27017을 입력합니다.
7. SSL 모드에서 verify-full을 선택합니다. Amazon DocumentDB 클러스터에서 SSL을 비활성화한 경우, 이 단계를 건너뛰어도 됩니다.
8. CA 인증서에서 Amazon DocumentDB 인증서인 `rds-combined-ca-bundle.pem`을 선택합니다. 이 인증서를 추가하는 방법에 대한 지침은 TLS를 사용하여 Amazon DocumentDB에 연결 섹션을 참조하세요.
9. 데이터베이스 이름에 마이그레이션할 데이터베이스의 이름을 입력합니다.

CLI에서는 다음 절차를 사용하세요.

AWS CLI를 사용하여 Amazon DocumentDB 소스 엔드포인트를 구성하려면

- 다음 AWS DMS `create-endpoint` 명령을 실행하고 자리 표시자를 자체 값으로 바꿔 Amazon DocumentDB 소스 엔드포인트를 구성합니다.

```
aws dms create-endpoint \
           --endpoint-identifier a_memorable_name \
           --endpoint-type source \
           --engine-name docdb \
           --username value \
```

```
        --password value \
        --server-name servername_where_database_endpoint_resides \
        --port 27017 \
        --database-name name_of_endpoint_database
```

Amazon DocumentDB 모음을 분할하고 병렬로 마이그레이션

마이그레이션 작업의 성능을 개선하기 위해 Amazon DocumentDB 소스 엔드포인트는 테이블 매핑에서 병렬 전체 로드 기능 옵션 두 가지를 지원합니다. 다시 말해 JSON 설정에서 병렬 전체 로드를 위해 테이블 매핑에서 자동 분할 옵션이나 범위 분할 옵션을 사용하여 모음을 병렬로 마이그레이션할 수 있습니다. 자동 분할 옵션을 사용하면 AWS DMS가 각 스레드에서 마이그레이션할 소스를 자동으로 분할하는 기준을 지정할 수 있습니다. 범위 분할 옵션을 사용하면 DMS가 각 스레드에서 마이그레이션할 각 세그먼트의 특정 범위를 AWS DMS에 지시할 수 있습니다. 이러한 설정에 대한 자세한 내용은 테이블 및 컬렉션 설정 규칙과 작업 섹션을 참조하세요.

자동 분할 범위를 사용하여 Amazon DocumentDB 데이터베이스를 병렬로 마이그레이션

AWS DMS가 각 스레드의 데이터를 자동으로 파티셔닝(분할)하는 기준, 특히 스레드당 마이그레이션할 문서 수를 지정하여 문서를 병렬로 마이그레이션할 수 있습니다. AWS DMS는 이 방식을 사용하여 스레드당 성능을 극대화하는 세그먼트 경계 최적화를 시도합니다.

테이블 매핑에서 다음 테이블 설정 옵션을 사용하여 분할 기준을 지정할 수 있습니다.

테이블 설정 옵션	설명
`"type"`	(필수) 소스로 사용하는 Amazon DocumentDB의 경우, `"partitions-auto"` 로 설정합니다.
`"number-of-partitions"`	(선택 사항) 마이그레이션에 사용되는 총 파티션(세그먼트) 수. 기본값은 16입니다.
`"collection-count-from-metadata"`	(선택 사항) `true`로 설정하면 AWS DMS는 추정 모음 개수를 사용하여 파티션 수를 결정합니다. `false`로 설정하면 AWS DMS는 실제 모음 수를 사용합니다. 기본값은 `true`입니다.
`"max-records-skip-per-page"`	(선택 사항) 각 파티션의 경계를 결정할 때 한 번에 건너뛰는 레코드 수입니다. AWS DMS는 페이

테이블 설정 옵션	설명
	지 매김 건너뛰기 방식을 사용하여 파티션의 최소 경계를 결정합니다. 기본값은 10,000입니다. 상대적으로 큰 값을 설정하면 커서 시간이 초과되고 작업이 실패할 수 있습니다. 상대적으로 작은 값을 설정하면 페이지당 작업 수가 늘어나고 전체 로드 속도가 느려집니다.
"batch-size"	(선택 사항) 배치 하나에서 반환되는 문서 수를 제한합니다. 배치마다 서버를 왕복해야 합니다. 배치 크기가 0인 경우, 커서는 서버에서 정의한 최대 배치 크기를 사용합니다. 기본값은 0입니다.

다음 예제는 자동 분할을 위한 테이블 매핑을 보여 줍니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "admin",
                "table-name": "departments"
            },
            "rule-action": "include",
            "filters": []
        },
        {
            "rule-type": "table-settings",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "admin",
                "table-name": "departments"
            },
            "parallel-load": {
                "type": "partitions-auto",
                "number-of-partitions": 5,
                "collection-count-from-metadata": "true",
```

```
                "max-records-skip-per-page": 1000000,
                "batch-size": 50000
            }
        }
    ]
}
```

자동 분할에는 다음과 같은 제한이 있습니다. 각 세그먼트의 마이그레이션은 모음 개수와 모음의 최소 _id를 개별적으로 가져옵니다. 그런 다음 페이지 매김 건너뛰기를 사용하여 해당 세그먼트의 최소 경계를 계산합니다. 따라서 모음의 모든 세그먼트 경계를 계산할 때까지 각 모음의 최소 _id 값이 일정하게 유지되어야 합니다. 세그먼트 경계를 계산하는 동안 모음의 최소 _id 값을 변경하면 데이터가 손실되거나 행 중복 오류가 발생할 수 있습니다.

특정 세그먼트 범위를 사용하여 Amazon DocumentDB 데이터베이스를 병렬로 마이그레이션

다음 예제에 나온 Amazon DocumentDB 모음에는 항목 7개와 프라이머리 키 _id가 있습니다.

Key	Value	Type
(1) ObjectId("5f805c74873173399a278d78")	{ 3 fields }	Object
_id	ObjectId("5f805c74873173399a278d78")	ObjectId
num	1	Int32
name	a	String
(2) ObjectId("5f805c97873173399a278d79")	{ 3 fields }	Object
_id	ObjectId("5f805c97873173399a278d79")	ObjectId
num	2	Int32
name	b	String
(3) ObjectId("5f805cb0873173399a278d7a")	{ 3 fields }	Object
_id	ObjectId("5f805cb0873173399a278d7a")	ObjectId
num	3	Int32
name	c	String
(4) ObjectId("5f805cbb873173399a278d7b")	{ 3 fields }	Object
_id	ObjectId("5f805cbb873173399a278d7b")	ObjectId
num	4	Int32
name	d	String
(5) ObjectId("5f805cc5873173399a278d7c")	{ 3 fields }	Object
_id	ObjectId("5f805cc5873173399a278d7c")	ObjectId
num	5	Int32
name	e	String
(6) ObjectId("5f805cd0873173399a278d7d")	{ 3 fields }	Object
_id	ObjectId("5f805cd0873173399a278d7d")	ObjectId
num	6	Int32
name	f	String
(7) ObjectId("5f805cdd873173399a278d7e")	{ 3 fields }	Object
_id	ObjectId("5f805cdd873173399a278d7e")	ObjectId
num	7	Int32
name	g	String

모음을 세 개의 세그먼트로 분할하고 병렬로 마이그레이션하려면 다음 JSON 예제와 같이 마이그레이션 작업에 테이블 매핑 규칙을 추가하면 됩니다.

```
{ // Task table mappings:
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "testdatabase",
        "table-name": "testtable"
      },
      "rule-action": "include"
```

```
    }, // "selection" :"rule-type"
    {
      "rule-type": "table-settings",
      "rule-id": "2",
      "rule-name": "2",
      "object-locator": {
        "schema-name": "testdatabase",
        "table-name": "testtable"
      },
      "parallel-load": {
        "type": "ranges",
        "columns": [
           "_id",
           "num"
        ],
        "boundaries": [
          // First segment selects documents with _id less-than-or-equal-to
 5f805c97873173399a278d79
          // and num less-than-or-equal-to 2.
          [
             "5f805c97873173399a278d79",
             "2"
          ],
          // Second segment selects documents with _id > 5f805c97873173399a278d79 and
          // _id less-than-or-equal-to 5f805cc5873173399a278d7c and
          // num > 2 and num less-than-or-equal-to 5.
          [
             "5f805cc5873173399a278d7c",
             "5"
          ]
          // Third segment is implied and selects documents with _id >
 5f805cc5873173399a278d7c.
       ] // :"boundaries"
      } // :"parallel-load"
    } // "table-settings" :"rule-type"
  ] // :"rules"
} // :Task table mappings
```

이 테이블 매핑 정의는 소스 모음을 세 개의 세그먼트로 분할하고 병렬로 마이그레이션합니다. 분할 경계는 다음과 같습니다.

```
Data with _id less-than-or-equal-to "5f805c97873173399a278d79" and num less-than-or-
equal-to 2 (2 records)
Data with _id less-than-or-equal-to "5f805cc5873173399a278d7c" and num less-than-or-
equal-to 5 and not in (_id less-than-or-equal-to  "5f805c97873173399a278d79" and num
 less-than-or-equal-to 2) (3 records)
Data not in (_id less-than-or-equal-to "5f805cc5873173399a278d7c" and num less-than-or-
equal-to 5) (2 records)
```

마이그레이션 작업이 완료되면 다음 예제와 같이 작업 로그에서 테이블이 병렬로 로드된 것을 확인할 수 있습니다. 소스 테이블에서 각 세그먼트를 언로드하는 데 사용된 Amazon DocumentDB `find` 절도 확인할 수 있습니다.

```
[TASK_MANAGER    ] I:  Start loading segment #1 of 3 of table
 'testdatabase'.'testtable' (Id = 1) by subtask 1. Start load timestamp
 0005B191D638FE86  (replicationtask_util.c:752)

[SOURCE_UNLOAD   ] I:  Range Segmentation filter for Segment #0 is initialized.
 (mongodb_unload.c:157)

[SOURCE_UNLOAD   ] I:  Range Segmentation filter for Segment #0 is: { "_id" :
 { "$lte" : { "$oid" : "5f805c97873173399a278d79" } }, "num" : { "$lte" :
 { "$numberInt" : "2" } } }  (mongodb_unload.c:328)

[SOURCE_UNLOAD   ] I: Unload finished for segment #1 of segmented table
 'testdatabase'.'testtable' (Id = 1). 2 rows sent.

[TASK_MANAGER    ] I: Start loading segment #1 of 3 of table
 'testdatabase'.'testtable' (Id = 1) by subtask 1. Start load timestamp
 0005B191D638FE86 (replicationtask_util.c:752)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is initialized.
 (mongodb_unload.c:157)

[SOURCE_UNLOAD   ] I: Range Segmentation filter for Segment #0 is: { "_id" : { "$lte" :
 { "$oid" : "5f805c97873173399a278d79" } }, "num" : { "$lte" : { "$numberInt" :
 "2" } } } (mongodb_unload.c:328)

[SOURCE_UNLOAD   ] I: Unload finished for segment #1 of segmented table
 'testdatabase'.'testtable' (Id = 1). 2 rows sent.
```

```
[TARGET_LOAD     ] I: Load finished for segment #1 of segmented table
 'testdatabase'.'testtable' (Id = 1). 1 rows received. 0 rows skipped. Volume
 transfered 480.

[TASK_MANAGER    ] I: Load finished for segment #1 of table
 'testdatabase'.'testtable' (Id = 1) by subtask 1. 2 records transferred.
```

현재, AWS DMS는 다음과 같은 Amazon DocumentDB 데이터 형식을 세그먼트 키 열로 지원합니다.

- Double
- String
- ObjectId
- 32비트 정수
- 64비트 정수

Amazon DocumentDB를 AWS DMS 소스로 사용 시 여러 데이터베이스 마이그레이션

AWS DMS 버전 3.4.5 이상에서는 Amazon DocumentDB 버전 4.0 이상만 단일 작업으로 여러 데이터베이스 마이그레이션을 지원합니다. 여러 데이터베이스를 마이그레이션하려면 다음을 수행하세요.

1. Amazon DocumentDB 소스 엔드포인트를 생성할 때:
   - AWS DMS의 AWS Management Console에서 엔드포인트 생성 페이지의 엔드포인트 구성 아래 데이터베이스 이름을 비워 둡니다.
   - AWS Command Line Interface(AWS CLI) 에서 작업에 대해 지정하는 DocumentDbSettingts의 DatabaseName매개 변수에 빈 문자열 값을 지정합니다. CreateEndpoint
2. 이 Amazon DocumentDB 소스 엔드포인트에서 마이그레이션하려는 각 데이터베이스마다 콘솔의 안내 입력을 사용하거나 JSON에서 직접 각 데이터베이스의 이름을 작업 테이블 매핑의 스키마 이름으로 지정합니다. 안내 입력에 대한 자세한 내용은 콘솔에서 테이블 선택 및 변환 규칙 지정의 설명을 참조하세요. JSON에 대한 자세한 내용은 선택 규칙 및 작업 섹션을 참조하세요.

예를 들면 다음과 같이 JSON을 지정하여 Amazon DocumentDB 데이터베이스 3개를 마이그레이션할 수 있습니다.

Example 스키마에서 모든 테이블 마이그레이션

다음 JSON은 소스 엔드포인트에 있는 Customers, Orders, Suppliers 데이터베이스의 모든 테이블을 대상 엔드포인트로 마이그레이션합니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "Customers",
                "table-name": "%"
            },
            "object-locator": {
                "schema-name": "Orders",
                "table-name": "%"
            },
            "object-locator": {
                "schema-name": "Inventory",
                "table-name": "%"
            },
            "rule-action": "include"
        }
    ]
}
```

Amazon DocumentDB를 AWS DMS 소스로 사용 시 제한 사항

Amazon DocumentDB를 AWS DMS 소스로 사용 시 제한 사항은 다음과 같습니다.

- _id 옵션이 별도의 열로 설정된 경우 ID 문자열이 200자를 초과할 수 없습니다.
- 객체 ID와 배열 형식 키는 테이블 모드에서 oid 및 array 접두사가 붙은 열로 변환됩니다.

  내부적으로 이러한 열은 접두사 이름으로 참조됩니다. AWS DMS에서 이러한 열을 참조하는 변환 규칙을 사용하는 경우, 접두사가 붙은 열을 지정해야 합니다. 예를 들면 ${_id}가 아니라 ${oid__id}, 또는 ${_addresses}가 아니라 ${array__addresses}를 지정합니다.
- 모음 이름과 키 이름에는 달러 기호($)를 포함할 수 없습니다.
- 테이블 모드와 문서 모드에는 앞서 설명한 제한이 있습니다.
- 자동 분할을 사용한 병렬 마이그레이션에는 앞에서 설명한 제한 사항이 있습니다.

- Amazon DocumentDB(MongoDB 호환) 소스는 변경 데이터 캡처(CDC) 시작 위치로 특정 타임스탬프를 사용하는 것을 지원하지 않습니다. 지속적 복제 작업은 타임스탬프에 상관없이 변경 내용을 캡처하기 시작합니다.
- DocumentDB(MongoDB 호환)를 소스로 사용하는 경우, DMS는 초당 최대 250개의 레코드를 처리할 수 있습니다.
- AWS DMS는 중첩 수준이 97보다 큰 문서는 지원하지 않습니다.
- DocumentDB에는 소스 필터가 지원되지 않습니다.
- AWS DMS탄력적 클러스터 모드에서는 DocumentDB에 대한 CDC (변경 데이터 캡처) 복제를 소스로 지원하지 않습니다.

Amazon DocumentDB를 소스로 하는 엔드포인트 설정 사용

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Amazon DocumentDB 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI에서 `create-endpoint` 명령을 `--doc-db-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다.

Amazon DocumentDB를 소스로 하여 사용할 수 있는 엔드포인트 설정이 다음 표에 나와 있습니다.

속성 이름	유효값	기본값과 설명

`NestingLevel`	`"none"`
	
	`"one"`
		`"none"` - 문서 모드를 사용하려면 `"none"`을 지정합니다. 테이블 모드를 사용하려면 `"one"`을 지정합니다.

`ExtractDocID`	불
	
	`true`
	
	`false`
		`false` - 이 속성은 `NestingLevel` 이 `"none"`으로 설정된 경우에 사용합니다.
		
		대상 데이터베이스가 Amazon DocumentDB인 경우, `'{"ExtractDocID": true}'` 를 설정합니다.

`DocsToInvestigate`	`0`보다 큰 양의 정수입니다.	`1000` - 이 속성은 `NestingLevel` 이 `"one"`으로 설정된 경우에 사용합니다.
`ReplicateShardCollections`	불 `true`	true인 경우, AWS DMS는 데이터를 샤드 모음에 복제합니다. AWS DMS는 대상 엔드포인트가 DocumentDB Elastic 클러스터인 경우에만 이 설정을 사용합니다.

속성 이름	유효값	기본값과 설명
	`false`	이 설정이 true인 경우 다음 사항에 유의하세요. • `TargetTablePrepMode` 를 `nothing`으로 설정해야 합니다. • AWS DMS는 자동으로 `useUpdateLookup` 을 `false`로 설정합니다.

Amazon DocumentDB의 소스 데이터 형식

다음 표에서 AWS DMS 사용 시 지원되는 Amazon DocumentDB 소스 데이터 형식을 찾을 수 있습니다. 이 표의 AWS DMS 데이터 형식에서 기본 매핑도 찾을 수 있습니다. 데이터 형식에 대한 자세한 내용은 MongoDB 설명서의 BSON Types를 참조하세요.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하십시오.

AWS DMS 데이터 형식에 관한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

Amazon DocumentDB 데이터 형식	AWS DMS 데이터 유형
불	부울
바이너리	BLOB
날짜	날짜
타임스탬프	날짜
정수	INT4
Long	INT8
Double	REAL8
String(UTF-8)	CLOB
배열	CLOB

Amazon DocumentDB 데이터 형식	AWS DMS 데이터 유형
OID	String

Amazon S3를 AWS DMS 소스로 사용

AWS DMS를 사용하여 Amazon S3 버킷에서 데이터를 마이그레이션할 수 있습니다. 이렇게 하려면 하나 이상의 데이터 파일을 포함하는 Amazon S3 버킷에 액세스합니다. 해당 S3 버킷에서, 데이터와 데이터의 데이터베이스 테이블 간 매핑을 설명하는 JSON 파일을 이러한 파일 안에 포함합니다.

전체 로드가 시작되기 전에 Amazon S3 버킷에 소스 데이터 파일이 있어야 합니다. `bucketName` 파라미터를 사용하여 버킷 이름을 지정합니다.

소스 데이터 파일은 쉼표로 구분된 값(.csv) 형식이어야 합니다. 다음과 같은 명명 규칙을 사용하여 이름을 지정합니다. 이 규칙에서 *`schemaName`*은 원본 스키마이고 *`tableName`*은 해당 스키마 내 테이블 이름입니다.

```
/schemaName/tableName/LOAD001.csv
/schemaName/tableName/LOAD002.csv
/schemaName/tableName/LOAD003.csv
...
```

예를 들면 다음 Amazon S3 경로에서 `mybucket`에 데이터 파일이 있다고 가정합니다.

```
s3://mybucket/hr/employee
```

로드 시, AWS DMS는 원본 스키마 이름이 `hr`이고 원본 테이블 이름이 `employee`라고 가정합니다.

`bucketName`(필수) 외에도, 원할 경우 `bucketFolder` 파라미터를 제공하여 AWS DMS가 Amazon S3 버킷에서 데이터 파일을 찾을 위치를 지정할 수 있습니다. 이전 예제에 이어 `bucketFolder`를 `sourcedata`로 설정하는 경우, AWS DMS는 다음 경로에서 데이터 파일을 읽습니다.

```
s3://mybucket/sourcedata/hr/employee
```

추가 연결 속성을 사용하여 열 구분 기호, 행 구분 기호, null 값 지표 등 여러 파라미터를 지정할 수 있습니다. 자세한 내용은 Amazon S3를 AWS DMS 소스로 사용 시 엔드포인트 설정 섹션을 참조하세요.

다음과 같이 ExpectedBucketOwner Amazon S3 엔드포인트 설정을 사용하여 버킷 소유자를 지정하고 스나이핑을 방지할 수 있습니다. 그런 다음, 연결을 테스트하거나 마이그레이션을 수행하도록 요청하면 S3는 지정된 파라미터와 비교하여 버킷 소유자의 계정 ID를 확인합니다.

```
--s3-settings='{"ExpectedBucketOwner": "AWS_Account_ID"}'
```

주제

- AWS DMS 소스로 사용되는 Amazon S3 외부 테이블 정의
- AWS DMS 소스로 사용되는 Amazon S3에서 CDC 사용
- Amazon S3를 AWS DMS 소스로 사용 시 사전 요구 사항
- Amazon S3를 AWS DMS 소스로 사용 시 제한 사항
- Amazon S3를 AWS DMS 소스로 사용 시 엔드포인트 설정
- Amazon S3의 소스 데이터 형식

AWS DMS 소스로 사용되는 Amazon S3 외부 테이블 정의

데이터 파일 외에, 외부 테이블 정의도 제공해야 합니다. 외부 테이블 정의는 AWS DMS가 Amazon S3에서 데이터를 해석하는 방식을 설명하는 JSON 문서입니다. 이 문서의 최대 크기는 2 MB입니다. AWS DMS 관리 콘솔을 사용하여 소스 엔드포인트를 생성할 경우, JSON을 테이블 매핑 상자에 직접 입력할 수 있습니다. AWS Command Line Interface(AWS CLI) 또는 AWS DMS API를 사용하여 마이그레이션을 수행할 경우 JSON 파일을 생성하여 외부 테이블 정의할 수 있습니다.

다음을 포함하는 데이터 파일이 있다고 가정해 보겠습니다.

```
101,Smith,Bob,2014-06-04,New York
102,Smith,Bob,2015-10-08,Los Angeles
103,Smith,Bob,2017-03-13,Dallas
104,Smith,Bob,2017-03-13,Dallas
```

다음은 이 데이터에 대한 외부 테이블 정의 예제입니다.

```
{
    "TableCount": "1",
    "Tables": [
        {
            "TableName": "employee",
            "TablePath": "hr/employee/",
            "TableOwner": "hr",
```

```
            "TableColumns": [
                {
                    "ColumnName": "Id",
                    "ColumnType": "INT8",
                    "ColumnNullable": "false",
                    "ColumnIsPk": "true"
                },
                {
                    "ColumnName": "LastName",
                    "ColumnType": "STRING",
                    "ColumnLength": "20"
                },
                {
                    "ColumnName": "FirstName",
                    "ColumnType": "STRING",
                    "ColumnLength": "30"
                },
                {
                    "ColumnName": "HireDate",
                    "ColumnType": "DATETIME"
                },
                {
                    "ColumnName": "OfficeLocation",
                    "ColumnType": "STRING",
                    "ColumnLength": "20"
                }
            ],
            "TableColumnsTotal": "5"
        }
    ]
}
```

이 JSON 문서의 요소는 다음과 같습니다.

`TableCount` - 소스 테이블 수. 이 예제에는 테이블이 하나뿐입니다.

`Tables` - 소스 테이블당 하나의 JSON 맵으로 구성된 배열. 이 예제에는 맵이 하나뿐입니다. 각 맵은 다음 요소로 구성됩니다.

- `TableName` - 소스 테이블의 이름.
- `TablePath` - AWS DMS가 전체 데이터 로드 파일을 찾을 수 있는 Amazon S3 버킷의 경로. `bucketFolder` 값이 지정된 경우, 해당 값이 경로에 추가됩니다.

- `TableOwner` - 이 테이블의 스키마 이름.
- `TableColumns` - 하나 이상의 맵으로 이루어진 배열. 각 맵은 소스 테이블의 열을 설명합니다.
  - `ColumnName` – 소스 테이블의 열 이름.
  - `ColumnType` - 열의 데이터 형식. 유효한 데이터 형식은 Amazon S3의 소스 데이터 형식 섹션을 참조하십시오.
  - `ColumnLength` - 이 열의 바이트 수. S3 소스가 전체 LOB 모드를 지원하지 않으므로 최대 열 길이는 2,147,483,647바이트(2,047메가바이트)로 제한됩니다. `ColumnLength`가 유효한 데이터 형식은 다음과 같습니다.
    - BYTE
    - STRING
  - `ColumnNullable` - 이 열에 NULL 값을 포함할 수 있는 경우 `true`인 불 값(기본값=`false`).
  - `ColumnIsPk` - 이 열이 프라이머리 키의 일부인 경우 `true`인 불 값(기본값=`false`).
  - `ColumnDateFormat` - DATE, TIME, DATETIME 형식이 있는 열의 입력 날짜 형식으로, 데이터 문자열을 날짜 객체로 파싱하는 데 사용됩니다. 가능한 값은 다음과 같습니다.

```
- YYYY-MM-dd HH:mm:ss
- YYYY-MM-dd HH:mm:ss.F
- YYYY/MM/dd HH:mm:ss
- YYYY/MM/dd HH:mm:ss.F
- MM/dd/YYYY HH:mm:ss
- MM/dd/YYYY HH:mm:ss.F
- YYYYMMdd HH:mm:ss
- YYYYMMdd HH:mm:ss.F
```

- `TableColumnsTotal` - 총 열 수. 이 숫자는 `TableColumns` 배열의 요소 수와 일치해야 합니다.

별도로 지정하지 않는 경우 AWS DMS에서는 `ColumnLength`가 0이라고 가정합니다.

> **Note**
>
> 지원되는 AWS DMS 버전에서는 S3 소스 데이터에 선택적 작업 열이 `TableName` 열 값 앞에 첫 번째 열로 포함될 수 있습니다. 이 작업 열은 전체 로드 중에 데이터를 S3 대상 엔드포인트로 마이그레이션하는 데 사용되는 작업(`INSERT`)을 식별합니다.
> 있는 경우 이 열의 값은 `INSERT` 작업 키워드(`I`)의 초기 문자열입니다. 지정된 경우 일반적으로 이 열은 이전 마이그레이션 중에 DMS에서 S3 대상으로 생성한 S3 소스입니다.

3.4.2 이전 DMS 버전에서 이 열은 이전 DMS 전체 로드에서 생성한 S3 소스 데이터에 없었습니다. S3 대상 데이터에 열을 추가하면 모든 행이 데이터의 전체 로드 중에 기록되는지 또는 CDC 로드 중에 기록되는지와 상관없이 S3 대상에 기록된 모든 행의 형식을 일관적으로 유지할 수 있습니다 S3 대상 데이터 서식 지정에 대한 자세한 내용은 마이그레이션된 S3 데이터에 소스 DB 작업 표시 단원을 참조하십시오

NUMERIC 유형 열의 경우, 정밀도와 배율을 지정합니다. 정밀도는 숫자의 총 자릿수이며, 배율은 소수점 오른쪽의 자릿수입니다. 다음과 같이, 이를 위해 `ColumnPrecision` 및 `ColumnScale` 요소를 사용할 수 있습니다.

```
...
    {
        "ColumnName": "HourlyRate",
        "ColumnType": "NUMERIC",
        "ColumnPrecision": "5"
        "ColumnScale": "2"
    }
...
```

소수점 이하 초가 포함된 데이터가 있는 DATETIME 유형 열의 경우, 스케일을 지정합니다. 스케일은 소수점 이하 초의 자릿수이며, 범위는 0에서 9까지입니다. 다음과 같이 이를 위해 `ColumnScale` 요소를 사용할 수 있습니다.

```
...
{
      "ColumnName": "HireDate",
      "ColumnType": "DATETIME",
      "ColumnScale": "3"
}
...
```

달리 지정하지 않으면 AWS DMS는 `ColumnScale`이 0이라고 가정하고 소수점 이하 초를 잘라냅니다.

AWS DMS 소스로 사용되는 Amazon S3에서 CDC 사용

AWS DMS에서 전체 데이터 로드를 수행한 후에는 선택적으로 데이터 변경 내용을 대상 엔드포인트로 복제할 수 있습니다. 이 작업을 수행하려면 변경 데이터 캡처 파일(CDC 파일)을 Amazon S3 버킷에 업로드할 수 있습니다. CDC 파일을 업로드할 경우, AWS DMS는 이러한 CDC 파일을 읽고 나서 대상 엔드포인트에서 변경 내용을 적용합니다.

CDC 파일은 다음과 같이 지정됩니다.

```
CDC00001.csv
CDC00002.csv
CDC00003.csv
...
```

> **Note**
>
> 변경 데이터 폴더에서 CDC 파일을 복제하려면 어휘(순차적) 순으로 CDC 파일을 성공적으로 업로드합니다. 예를 들어 CDC00002.csv 파일을 CDC00003.csv 파일 앞에 업로드합니다. 그렇지 않으면 CDC00002.csv를 건너뛰고 CDC00003.csv 이후에 로드해도 CDC00002.csv가 복제되지 않습니다. CDC00003.csv 이후에 CDC00004.csv를 로드하면 CDC00004.csv가 성공적으로 복제됩니다.

AWS DMS가 파일을 찾을 수 있는 위치를 나타내려면 `cdcPath` 파라미터를 지정하세요. 이전 예제를 계속하여, `cdcPath`를 *`changedata`*로 설정하는 경우 AWS DMS는 다음 경로에서 CDC 파일을 읽습니다.

```
s3://mybucket/changedata
```

`cdcPath`를 *`changedata`*로, `bucketFolder`를 *`myFolder`*로 설정하면 AWS DMS는 다음 경로에서 CDC 파일을 읽습니다.

```
s3://mybucket/myFolder/changedata
```

다음과 같이 CDC 파일의 레코드에는 형식 지정되어 있습니다.

- 작업 - `INSERT` 또는 `I`, `UPDATE` 또는 `U`, `DELETE` 또는 `D`와 같은 수행할 변경 작업. 이러한 키워드 및 문자 값은 대소문자를 구분합니다.

  > **Note**
  >
  > 지원되는 AWS DMS 버전에서 AWS DMS는 각 로드 레코드에 수행할 작업을 두 가지 방법으로 식별할 수 있습니다. AWS DMS는 레코드의 키워드 값(예: `INSERT`) 또는 키워드 첫 문자(예: `I`)로 수행할 작업을 식별할 수 있습니다. 이전 버전에서는 AWS DMS가 전체 키워드 값에서만 로드 작업을 인식했습니다.

이전 버전의 AWS DMS에서는 CDC 데이터를 로깅하기 위해 전체 키워드 값이 기록되었습니다. 또한 이전 버전에서는 키워드 첫 문자만 사용하여 작업 값을 S3 대상에 기록했습니다. 두 가지 형식을 모두 인식하면 S3 소스 데이터를 생성하기 위해 작업 열이 기록되는 방식과 상관없이 AWS DMS가 작업을 처리할 수 있습니다. 이 접근 방식은 이후 마이그레이션에 대한 소스로 S3 대상 데이터 사용을 지원합니다. 이 접근 방식을 사용하면 이후 S3 소스의 작업 열에 나타나는 키워드 초기 값의 형식을 변경할 필요가 없습니다.

- 테이블 이름 - 소스 테이블의 이름.
- 스키마 이름 - 소스 스키마의 이름.
- 데이터 - 변경할 데이터를 나타내는 하나 이상의 열.

다음은 `employee` 이름의 테이블에 대한 예제 CDC 파일입니다.

```
INSERT,employee,hr,101,Smith,Bob,2014-06-04,New York
UPDATE,employee,hr,101,Smith,Bob,2015-10-08,Los Angeles
UPDATE,employee,hr,101,Smith,Bob,2017-03-13,Dallas
DELETE,employee,hr,101,Smith,Bob,2017-03-13,Dallas
```

Amazon S3를 AWS DMS 소스로 사용 시 사전 요구 사항

Amazon S3를 AWS DMS 소스로 사용하려면 데이터를 마이그레이션하는 DMS 복제 인스턴스와 동일한 AWS 리전에 소스 S3 버킷이 있어야 합니다. 또한, 마이그레이션에 사용하는 AWS 계정에 소스 버킷에 대한 읽기 액세스 권한이 있어야 합니다.

마이그레이션 작업을 생성하는 데 사용되는 사용자 계정에 할당된 AWS Identity and Access Management(IAM) 역할에는 다음 권한 집합이 있어야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::mybucket*/*"
            ]
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::mybucket*"
            ]
        }
    ]
}
```

Amazon S3 버킷에 버전 관리가 활성화된 경우, 마이그레이션 작업을 생성하는 데 사용되는 사용자 계정에 할당된 AWS Identity and Access Management(IAM) 역할에는 다음 권한 집합이 있어야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "S3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::mybucket*/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::mybucket*"
            ]
        }
    ]
}
```

Amazon S3를 AWS DMS 소스로 사용 시 제한 사항

Amazon S3를 소스로 사용할 때 적용되는 제한 사항은 다음과 같습니다.

- S3의 버전 관리를 활성화하지 마세요. S3 버전 관리가 필요한 경우, 수명 주기 정책을 사용하여 이전 버전을 적극적으로 삭제하세요. 그렇지 않으면 S3 `list-object` 직접 호출 시간 초과로 인해 엔드포인트 테스트 연결 실패가 발생할 수 있습니다. S3 버킷의 수명 주기 정책을 생성하려면 스토리지 수명 주기 관리를 참조하세요. S3 객체의 버전을 삭제하려면 버전 관리 활성화 버킷에서 객체 버전 삭제를 참조하세요.
- VPC 지원(게이트웨이 VPC) S3 버킷은 버전 3.4.7 이상 버전에서 지원됩니다.

Amazon S3를 AWS DMS 소스로 사용 시 엔드포인트 설정

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Amazon S3 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI에서 `create-endpoint` 명령을 `--s3-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다.

Amazon S3를 소스로 하여 사용할 수 있는 엔드포인트 설정이 다음 표에 나와 있습니다.

옵션	설명
`BucketFolder`	(선택 사항) S3 버킷의 폴더 이름. 이 속성이 지정된 경우, 소스 데이터 파일과 CDC 파일을 경로 `s3://myBucket/bucketFolder /schemaName /tableName /`과 `s3://myBucket/bucketFolder /`에서 각각 읽습니다. 이 속성이 지정되지 않은 경우, 사용되는 경로는 `schemaName /tableName /`입니다. `'{"BucketFolder": " sourceData "}'`
`BucketName`	S3 버킷의 이름. `'{"BucketName": " myBucket"}'`
`CdcPath`	CDC 파일의 위치. 작업에서 변경 데이터를 캡처하는 경우에는 이 속성이 필수이며, 그렇지 않은 경우에는 선택 사항입니다. `CdcPath`가 있는 경우, AWS DMS는 이 경로에서 CDC 파일을 읽고 데이터 변경 사항을 대상 엔드포인트로 복제합니다. 자세한 내용은 AWS DMS 소스로 사용되는 Amazon S3에서 CDC 사용 섹션을 참조하세요. `'{"CdcPath": " changeData "}'`

옵션	설명
CsvDelimiter	원본 파일에서 열을 구분하는 데 사용되는 구분 기호입니다. 기본값은 쉼표입니다. 예를 들면 다음과 같습니다. `'{"CsvDelimiter": ","}'`
CsvNullValue	소스에서 읽을 때 AWS DMS가 null로 취급하는 사용자 정의 문자열입니다. 기본값은 빈 문자열입니다. 이 파라미터를 설정하지 않으면 AWS DMS는 빈 문자열을 null 값으로 취급합니다. 이 파라미터를 “\N”과 같은 문자열로 설정하면 AWS DMS는 이 문자열을 null 값으로 취급하고, 빈 문자열은 빈 문자열 값으로 취급합니다.
CsvRowDelimiter	원본 파일에서 행을 구분하는 데 사용되는 구분 기호입니다. 기본값은 줄 바꿈(\n)입니다. `'{"CsvRowDelimiter": "\n"}'`
IgnoreHeaderRows	이 값을 1로 설정하면 AWS DMS는 .csv 파일의 첫 행 헤더를 무시합니다. 1 값은 이 기능을 활성화하고, 0 값은 비활성화합니다. 기본값은 0입니다. `'{"IgnoreHeaderRows": 1}'`
Rfc4180	이 값을 true 또는 y로 설정하면 여는 큰따옴표 뒤에 닫는 큰따옴표가 와야 합니다. 이 형식은 RFC 4180을 준수합니다. 이 값을 false 또는 n으로 설정하면 문자열 리터럴이 그대로 대상에 복사됩니다. 이 경우 구분 기호(행 또는 열)는 필드의 끝을 알립니다. 이렇게 구분 기호는 값의 끝을 알리는 것이므로 문자열의 일부로 사용할 수 없습니다. 기본값은 true입니다. 유효한 값: true, false, y, n `'{"Rfc4180": false}'`

Amazon S3의 소스 데이터 형식

Amazon S3를 AWS DMS 소스로 사용하는 데이터 마이그레이션은 Amazon S3의 데이터를 AWS DMS 데이터 형식에 매핑해야 합니다. 자세한 내용은 AWS DMS 소스로 사용되는 Amazon S3 외부 테이블 정의 섹션을 참조하세요.

대상에서 매핑된 데이터 형식을 확인하는 방법에 대한 정보는 사용 중인 대상 엔드포인트에 대한 섹션을 참조하십시오.

AWS DMS 데이터 형식에 대한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

다음 AWS DMS 데이터 형식은 Amazon S3에서 소스로 사용됩니다.

- BYTE - `ColumnLength`가 필요합니다. 자세한 내용은 AWS DMS 소스로 사용되는 Amazon S3 외부 테이블 정의 섹션을 참조하세요.
- DATE
- TIME
- DATETIME - 자세한 내용과 예제는 AWS DMS 소스로 사용되는 Amazon S3 외부 테이블 정의에서 DATETIME 형식 예제를 참조하세요.
- INT1
- INT2
- INT4
- INT8
- NUMERIC - `ColumnPrecision` 및 `ColumnScale`이 필요합니다. AWS DMS가 지원하는 최대값은 다음과 같습니다.
  - ColumnPrecision: 38
  - ColumnScale: 31

  자세한 내용과 예제는 AWS DMS 소스로 사용되는 Amazon S3 외부 테이블 정의에서 NUMERIC 형식 예제를 참조하세요.
- REAL4
- REAL8
- STRING - `ColumnLength`가 필요합니다. 자세한 내용은 AWS DMS 소스로 사용되는 Amazon S3 외부 테이블 정의 섹션을 참조하세요.
- UINT1

- UINT2
- UINT4
- UINT8
- BLOB
- CLOB
- BOOLEAN

리눅스, 유닉스, 윈도우용 IBM Db2 및 아마존 RDS 데이터베이스 (Db2 LUW) 를 소스로 사용 AWS DMS

() 를 사용하여 리눅스, 유닉스, 윈도우 및 Amazon RDS (Db2 LUW) 용 IBM Db2 데이터베이스에서 지원되는 모든 대상 데이터베이스로 데이터를 마이그레이션할 수 있습니다. AWS Database Migration Service AWS DMS

소스로 지원하는 Linux, Unix, Windows 및 RDS의 Db2 버전에 대한 자세한 내용은 을 참조하십시오. AWS DMS 출처: AWS DMS

Secure Sockets Layer(SSL)를 사용하여 Db2 LUW 엔드포인트와 복제 인스턴스 간 연결을 암호화할 수 있습니다. Db2 LUW 엔드포인트에서 SSL을 사용하는 방법에 대한 자세한 내용은 AWS Database Migration Service에서 SSL 사용 섹션을 참조하십시오.

Db2 LUW를 원본으로 사용할 때의 사전 요구 사항 AWS DMS

Db2 LUW 데이터베이스를 소스로 사용하기 전에 다음 사전 조건이 요구됩니다.

변경 데이터 캡처(CDC)라고도 하는 지속적 복제를 활성화하기 위해서는 다음을 수행하십시오.

- 데이터베이스를 복구 가능으로 설정하여 변경 사항을 캡처해야 합니다. AWS DMS 데이터베이스 구성 파라미터인 `LOGARCHMETH1`과 `LOGARCHMETH2` 중 하나, 또는 둘 모두가 `ON`으로 설정되어 있어야 데이터베이스를 복구할 수 있습니다.

  데이터베이스를 복구할 수 있는 경우 필요한 경우 Db2에 액세스할 AWS DMS 수 있습니다. `ARCHIVE LOG`
- 처리할 수 있는 충분한 보존 기간을 두고 DB2 트랜잭션 로그를 사용할 수 있는지 확인하십시오. AWS DMS
- DB2에는 트랜잭션 로그 레코드를 추출하기 위한 `SYSADM` 또는 `DBADM` 권한 부여가 필요합니다. 사용자 계정에 다음 권한을 부여합니다.

- SYSADM 또는 DBADM
- DATAACCESS

> **Note**
>
> 전체 로드 전용 작업의 경우, DMS 사용자 계정에 DATAACCESS 권한이 필요합니다.

- IBM DB2 for LUW 버전 9.7을 소스로 사용하는 경우, 추가 연결 속성(ECA) CurrentLSN을 다음과 같이 설정합니다.

  CurrentLSN=*LSN*, 여기서 *LSN*은 복제가 시작될 로그 시퀀스 번호(LSN)를 지정합니다. 또는 CurrentLSN=*scan*.

Db2 LUW를 원본으로 사용할 때의 제한 사항 AWS DMS

AWS DMS 클러스터링된 데이터베이스를 지원하지 않습니다. 하지만 클러스터의 각 엔드포인트에 별도의 Db2 LUW를 정의할 수 있습니다. 예를 들어 클러스터의 노드 중 하나를 사용하여 전체 로드 마이그레이션 작업을 생성한 다음 각 노드에서 별도의 작업을 생성할 수 있습니다.

AWS DMS 소스 Db2 LUW 데이터베이스의 BOOLEAN 데이터 유형을 지원하지 않습니다.

지속적 복제(CDC)를 사용할 때 다음 제한 사항이 적용됩니다.

- 파티션이 여러 개 있는 테이블이 잘린 경우 AWS DMS 콘솔에 표시되는 DDL 이벤트 수는 파티션 수와 같습니다. Db2 LUW가 각 파티션에 별개의 DDL을 기록하기 때문입니다.
- 분할된 테이블에서는 다음 DDL 작업이 지원되지 않습니다.
  - ALTER TABLE ADD PARTITION
  - ALTER TABLE DETACH PARTITION
  - ALTER TABLE ATTACH PARTITION
- AWS DMS DB2 고가용성 재해 복구 (HADR) 대기 인스턴스에서의 지속적인 복제 마이그레이션은 지원하지 않습니다. 대기 인스턴스에 액세스할 수 없습니다.
- DECFLOAT 데이터 형식은 지원되지 않습니다. 따라서 지속적 복제 진행 중 DECFLOAT 열들의 변경 내용이 무시됩니다.
- RENAME COLUMN 문은 지원되지 않습니다.
- 다차원 클러스터링 (MDC) 테이블을 업데이트할 때 각 업데이트는 AWS DMS 콘솔에 INSERT + DELETE로 표시됩니다.

- 작업 설정 복제에 LOB 열 포함이 활성화되지 않은 경우, LOB 열이 있는 테이블은 지속적 복제 도중 일시 중지됩니다.
- Db2 LUW 버전 10.5 이상의 경우 저장된 데이터가 있는 가변 길이 문자열 열은 무시됩니다. out-of-row 이 제한은 VARCHAR 및 VARGRAPHIC과 같은 데이터 형식을 가진 열의 확장된 행 크기로 생성된 테이블에만 적용됩니다. 이 제한을 해결하려면 페이지 크기가 더 큰 테이블 공간으로 테이블을 이동하세요. 자세한 내용은 What can I do if I want to change the pagesize of DB2 tablespaces를 참조하세요.
- 지속적 복제의 경우, DMS는 DB2 LOAD 유틸리티에 의해 페이지 수준에서 로드된 데이터의 마이그레이션을 지원하지 않습니다. 대신 SQL 삽입을 사용하는 IMPORT 유틸리티를 사용하세요. 자세한 내용은 differences between the import and load utilities를 참조하세요.
- 복제 작업이 실행되는 동안 DMS는 테이블이 DATA CAPTURE CHANGE 속성으로 생성된 경우에만 CREATE TABLE DDL을 캡처합니다.
- DMS는 Db2 데이터베이스 파티션 기능 (DPF) 을 사용할 때 다음과 같은 제한이 있습니다.
  - DMS는 DPF 환경에서 Db2 노드 간의 트랜잭션을 조정할 수 없습니다. 이는 IBM DB2READLOG API 인터페이스 내의 제약 때문입니다. DPF에서는 DB2가 데이터를 분할하는 방식에 따라 트랜잭션이 여러 Db2 노드에 걸쳐 있을 수 있습니다. 따라서 DMS 솔루션은 각 Db2 노드의 트랜잭션을 독립적으로 캡처해야 합니다.
  - DMS는 여러 DMS 소스 엔드포인트에서 로 설정하여 DPF 클러스터의 각 Db2 노드에서 로컬 트랜잭션을 `connectNode` 캡처할 수 1 있습니다. 이 구성은 DB2 서버 구성 파일에 정의된 논리적 노드 번호와 일치합니다. `db2nodes.cfg`
  - 개별 Db2 노드의 로컬 트랜잭션은 대규모 글로벌 트랜잭션의 일부일 수 있습니다. DMS는 다른 Db2 노드의 트랜잭션과 조정하지 않고 각 로컬 트랜잭션을 타겟에 독립적으로 적용합니다. 이러한 독립적 처리는 특히 파티션 간에 행을 이동할 때 복잡성을 초래할 수 있습니다.
  - DMS가 여러 Db2 노드에서 복제하는 경우 DMS가 각 Db2 노드에 대해 독립적으로 작업을 적용하기 때문에 타겟에서 올바른 작업 순서를 보장할 수 없습니다. 각 Db2 노드와 독립적으로 로컬 트랜잭션을 캡처하는 것이 특정 사용 사례에 적합한지 확인해야 합니다.
  - DPF 환경에서 마이그레이션할 때는 먼저 캐시된 이벤트 없이 Full Load 작업을 실행한 다음 CDC 전용 작업을 실행하는 것이 좋습니다. 엔드포인트 설정을 사용하여 설정한 Full Load 시작 타임스탬프 또는 LRI (로그 레코드 식별자) 부터 시작하여 Db2 노드당 하나의 작업을 실행하는 것이 좋습니다. `StartFromContext` 복제 시작점 결정에 대한 자세한 내용은 IBM Support 설명서에서 복제 시작을 위한 LSN 또는 LRI 값 찾기를 참조하십시오.
- 지속적 복제(CDC) 시 특정 타임스탬프에서 복제를 시작하려는 경우, `StartFromContext` 연결 속성을 필수 타임스탬프로 설정해야 합니다.

- 현재 DMS는 데이터베이스 솔루션을 확장하는 데 사용할 수 있는 DB2 LUW의 확장인 Db2 PureScale 기능을 지원하지 않습니다.
- AWS DMS Amazon RDS용 Db2를 소스로 사용할 때는 CDC를 지원하지 않습니다.

Db2 LUW를 소스로 사용할 때의 엔드포인트 설정 AWS DMS

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Db2 LUW 소스 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 의 `create-endpoint` 명령을 `--ibm-db2-settings '{"EndpointSetting": "value", ...}'` JSON 구문으로 사용하여 소스 엔드포인트를 생성할 때 설정을 지정합니다. AWS CLI

Db2 LUW를 소스로 하여 사용할 수 있는 엔드포인트 설정은 다음 표에 나와 있습니다.

명칭	설명
`CurrentLSN`	지속적 복제(CDC)의 경우, `CurrentLSN` 을 사용하여 복제가 시작될 로그 시퀀스 번호(LSN)를 지정합니다.
`MaxKBytesPerRead`	읽기 당 바이트의 최대 수(NUMBER 값) 기본값은 64KB입니다.
`SetDataCaptureChanges`	불 값으로 지속적 복제(CDC)를 활성화합니다. 기본값은 true입니다.
`StartFromContext`	지속적 복제(CDC)의 경우, `StartFromContext` 를 사용하여 복제가 시작될 로그 하한선을 지정합니다. `StartFromContext` 는 다양한 형식의 값을 허용합니다. 유효한 값으로는 다음이 포함됩니다. • `timestamp` (UTC). 예: `'{"StartFromContext": "timestamp:2021-09-21T13:00:00"}'` • `NOW`

명칭	설명

	IBM DB2 LUW 버전 10.5 이상의 경우, CurrentLSN: scan과 조합된 NOW는 최신 LSO에서 작업을 시작합니다. 예:
	`'{"CurrentLSN": "scan", "StartFromContext": "NOW"}'`
	• 특정 LRI. 예:
	`'{"StartFromContext": "0100000000000022C C000000000004FB13"}'`
	로그 파일의 LRI/LSN 범위를 확인하려면 다음 예제와 같이 db2flsn 명령을 실행합니다.
	`db2flsn -db SAMPLE -lrirange 2`
	예제의 출력은 다음과 비슷합니다.
	`S0000002.LOG: has LRI range 000000000000000100 00000000002254000000000004F9A6 to 00000000000000010000000000022CC000000000004 FB13`
	이 출력의 로그 파일은 S0000002 .LOG이고 StartFrom ContextLRI 값은 범위 끝에 있는 34바이트입니다.
	`0100000000000022CC000000000004FB13`

IBM Db2 LUW용 소스 데이터 형식

Db2 LUW를 원본으로 사용하는 데이터 마이그레이션은 대부분의 Db2 LUW 데이터 유형을 AWS DMS 지원합니다. 다음 표에는 사용 시 지원되는 Db2 LUW 소스 데이터 유형과 데이터 유형별 기본 매핑이 AWS DMS 나와 있습니다. AWS DMS Db2 LUW 데이터 형식에 대한 자세한 내용은 Db2 LUW 설명서를 참조하십시오.

대상에서 매핑된 데이터 형식을 보는 방법에 대한 자세한 내용은 사용 중인 대상 엔드포인트 단원을 참조하십시오.

AWS DMS 데이터 유형에 대한 자세한 내용은 을 참조하십시오. AWS Database Migration Service에서 사용되는 데이터 형식

Db2 LUW 데이터 형식	AWS DMS 데이터 유형
INTEGER	INT4
SMALLINT	INT2
BIGINT	INT8
DECIMAL (p,s)	NUMERIC(p,s)
FLOAT	REAL8
DOUBLE	REAL8
REAL	REAL4
DECFLOAT (p)	정밀도가 16이면 REAL8, 정밀도가 34이면 STRING
GRAPHIC (n)	길이가 0보다 크고 127 이하인 2바이트 문자 고정 길이 그래픽 문자열의 경우, WSTRING
VARGRAPHIC (n)	길이가 0보다 크고 16,352 이하인 2바이트 문자 가변 길이 그래픽 문자열의 경우, WSTRING
LONG VARGRAPHIC (n)	길이가 0보다 크고 16,352 이하인 2바이트 문자 가변 길이 그래픽 문자열의 경우, CLOB

Db2 LUW 데이터 형식	AWS DMS 데이터 유형
CHARACTER (n)	길이가 0보다 크고 255 이하인 2바이트 문자 고정 길이 문자열의 경우, STRING
VARCHAR (n)	길이가 0보다 크고 32,704 이하인 2바이트 문자 가변 길이 문자열의 경우, STRING
LONG VARCHAR (n)	길이가 0보다 크고 32,704 이하인 2바이트 문자 가변 길이 문자열의 경우, CLOB
CHAR (n) FOR BIT DATA	BYTES
VARCHAR (n) FOR BIT DATA	BYTES
LONG VARCHAR FOR BIT DATA	BYTES
날짜	날짜
TIME	TIME
TIMESTAMP	DATETIME
BLOB (n)	BLOB 최대 길이는 2,147,483,647바이트
CLOB (n)	CLOB 최대 길이는 2,147,483,647바이트
DBCLOB (n)	CLOB 최대 길이는 1,073,741,824 2바이트 문자
XML	CLOB

IBM Db2 for z/OS 데이터베이스를 AWS DMS 소스로 사용

AWS Database Migration Service(AWS DMS)를 사용하여 IBM for z/OS 데이터베이스에서 데이터를 마이그레이션할 수 있습니다.

AWS DMS가 소스로 지원하는 Db2 for z/OS 버전에 대한 자세한 내용은 출처: AWS DMS 섹션을 참조하세요.

Db2 for z/OS를 AWS DMS 소스로 사용 시 사전 요구 사항

AWS DMS에서 IBM Db2 for z/OS 데이터베이스를 소스로 사용하려면 소스 엔드포인트 연결 설정에 지정된 Db2 for z/OS 사용자에게 다음 권한을 부여합니다.

```
GRANT SELECT ON SYSIBM.SYSTABLES TO Db2USER;
GRANT SELECT ON SYSIBM.SYSTABLESPACE TO Db2USER;
GRANT SELECT ON SYSIBM.SYSTABLEPART TO Db2USER;
GRANT SELECT ON SYSIBM.SYSCOLUMNS TO Db2USER;
GRANT SELECT ON SYSIBM.SYSDATABASE TO Db2USER;
GRANT SELECT ON SYSIBM.SYSDUMMY1 TO Db2USER
```

user defined 소스 테이블에 대한 SELECT 권한도 부여하세요.

AWS DMS IBM Db2 for z/OS 소스 엔드포인트는 IBM Data Server Driver for ODBC를 사용하여 데이터에 액세스합니다. DMS가 이 엔드포인트에 연결하려면 데이터베이스 서버에 유효한 IBM ODBC Connect 라이선스가 있어야 합니다.

Db2 for z/OS를 AWS DMS 소스로 사용 시 제한 사항

IBM Db2 for z/OS를 AWS DMS 소스로 사용 시 다음 제한 사항이 적용됩니다.

- 전체 로드 복제 작업만 지원됩니다. 변경 데이터 캡처(CDC)는 지원되지 않습니다.
- 병렬 로드는 지원되지 않습니다.
- 뷰의 데이터 검증은 지원되지 않습니다.
- 열/테이블 수준 변환 및 행 수준 선택 필터를 위한 테이블 매핑에서 스키마, 테이블, 열 이름을 대문자로 지정해야 합니다.

IBM Db2 for z/OS 소스 데이터 형식

Db2 for z/OS를 AWS DMS 소스로 사용하는 데이터 마이그레이션은 대부분의 Db2 for z/OS 데이터 형식을 지원합니다. 다음 표에는 AWS DMS 사용 시 지원되는 Db2 for z/OS 소스 데이터 형식과 AWS DMS 데이터 형식으로부터의 기본 매핑이 나와 있습니다.

Db2 for z/OS 데이터 형식에 대한 자세한 내용은 IBM Db2 for z/OS 설명서를 참조하세요.

대상에서 매핑된 데이터 형식을 보는 방법에 대한 자세한 내용은 사용 중인 대상 엔드포인트 단원을 참조하십시오.

AWS DMS 데이터 형식에 대한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

Db2 for z/OS 데이터 형식	AWS DMS 데이터 형식
INTEGER	INT4
SMALLINT	INT2
BIGINT	INT8
DECIMAL (p,s)	NUMERIC (p,s) DB2 구성에서 소수점이 쉼표(,)로 설정된 경우, DB2 설정을 지원하도록 Replicate를 구성하세요.
FLOAT	REAL8
DOUBLE	REAL8
REAL	REAL4
DECFLOAT (p)	정밀도가 16이면 REAL8, 정밀도가 34이면 STRING
GRAPHIC (n)	n>=127이면 길이가 0보다 크고 127 이하인 2바이트 문자 고정 길이 그래픽 문자열의 경우, WSTRING
VARGRAPHIC (n)	길이가 0보다 크고 16,352 이하인 2바이트 문자 가변 길이 그래픽 문자열의 경우, WSTRING
LONG VARGRAPHIC (n)	길이가 0보다 크고 16,352 이하인 2바이트 문자 가변 길이 그래픽 문자열의 경우, CLOB

Db2 for z/OS 데이터 형식	AWS DMS 데이터 형식
CHARACTER (n)	길이가 0보다 크고 255 이하인 2바이트 문자 고정 길이 문자열의 경우, STRING
VARCHAR (n)	길이가 0보다 크고 32,704 이하인 2바이트 문자 가변 길이 문자열의 경우, STRING
LONG VARCHAR (n)	길이가 0보다 크고 32,704 이하인 2바이트 문자 가변 길이 문자열의 경우, CLOB
CHAR (n) FOR BIT DATA	BYTES
VARCHAR (n) FOR BIT DATA	BYTES
LONG VARCHAR FOR BIT DATA	BYTES
DATE	DATE
TIME	TIME
TIMESTAMP	DATETIME
BLOB (n)	BLOB 최대 길이는 2,147,483,647바이트
CLOB (n)	CLOB 최대 길이는 2,147,483,647바이트
DBCLOB (n)	CLOB 최대 길이는 1,073,741,824 2바이트 문자
XML	CLOB
BINARY	BYTES
VARBINARY	BYTES

Db2 for z/OS 데이터 형식	AWS DMS 데이터 형식
ROWID	BYTES ROWID 작업에 대한 자세한 내용은 다음을 참조하세요.
TIMESTAMP(시간대 사용)	지원하지 않음.

작업의 대상 테이블 준비 모드가 DROP_AND_CREATE(기본값)로 설정된 경우, ROWID 열이 기본적으로 마이그레이션됩니다. 특정 데이터베이스 및 테이블 외부에서는 행이 무의미하므로 데이터 검증은 이러한 열을 무시합니다. 이러한 열의 마이그레이션을 끄려면 다음 준비 단계 중 하나를 수행하면 됩니다.

- 이러한 열이 없는 대상 테이블을 미리 생성합니다. 그런 다음 작업의 대상 테이블 준비 모드를 DO_NOTHING 또는 TRUNCATE_BEFORE_LOAD로 설정합니다. AWS Schema Conversion Tool(AWS SCT)을 사용하여 열이 없는 대상 테이블을 미리 생성할 수 있습니다.
- 이러한 열을 필터링하여 무시되도록 하는 테이블 매핑 규칙을 작업에 추가합니다. 자세한 내용은 변환 규칙 및 작업 섹션을 참조하세요.

AWS Mainframe Modernization 서비스를 위한 PostgreSQL의 EBCDIC 데이터 정렬

AWS Mainframe Modernization 서비스는 메인프레임 애플리케이션을 AWS 관리형 런타임 환경으로 현대화하는 데 도움이 됩니다. 이 서비스는 마이그레이션 및 현대화 프로젝트를 계획하고 구현하는 데 도움이 되는 도구와 리소스를 제공합니다. 메인프레임 현대화 및 마이그레이션에 대한 자세한 내용은 AWS를 통한 메인프레임 현대화를 참조하세요.

일부 IBM Db2 for z/OS 데이터 세트는 EBCDIC(Extended Binary Coded Decimal Interchange) 문자 집합으로 인코딩됩니다. 이 문자 집합은 ASCII(American Standard Code for Information Interchange)가 일반적으로 사용되기 전에 개발되었습니다. 코드 페이지는 텍스트의 각 문자를 문자 집합의 문자에 매핑합니다. 기존 코드 페이지에는 코드 포인트와 문자 ID 간의 매핑 정보가 들어 있습니다. 문자 ID는 8바이트 문자 데이터 문자열입니다. 코드 포인트는 문자를 나타내는 8비트 2진수입니다. 코드 포인트는 일반적으로 해당 2진수 값의 16진수 표현으로 표시됩니다.

현재 Mainframe Modernization 서비스의 Micro Focus 또는 BluAge 구성 요소를 사용하고 있다면 특정 코드 포인트를 시프트(번역)하도록 AWS DMS에 지시해야 합니다. AWS DMS 작업 설정을 사용하여 시프트를 수행할 수 있습니다. 다음 예제는 AWS DMS `CharacterSetSettings` 작업을 사용하여 DMS 작업 설정에서 시프트를 매핑하는 방법을 보여 줍니다.

```
"CharacterSetSettings": {
        "CharacterSetSupport": null,
        "CharacterReplacements": [
{"SourceCharacterCodePoint": "0000","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "00B8","TargetCharacterCodePoint": "0160"}
,{"SourceCharacterCodePoint": "00BC","TargetCharacterCodePoint": "0161"}
,{"SourceCharacterCodePoint": "00BD","TargetCharacterCodePoint": "017D"}
,{"SourceCharacterCodePoint": "00BE","TargetCharacterCodePoint": "017E"}
,{"SourceCharacterCodePoint": "00A8","TargetCharacterCodePoint": "0152"}
,{"SourceCharacterCodePoint": "00B4","TargetCharacterCodePoint": "0153"}
,{"SourceCharacterCodePoint": "00A6","TargetCharacterCodePoint": "0178"}
            }
        ]
    }
```

필요한 시프팅을 이해하는 EBCDIC 데이터 정렬이 PostgreSQL에 이미 일부 있습니다. 여러 가지 코드 페이지가 지원됩니다. 다음 섹션에서는 지원되는 모든 코드 페이지에서 시프트해야 하는 항목의 JSON 샘플을 제공합니다. DMS 작업에 필요한 필수 JSON을 간단히 복사하여 붙여넣을 수 있습니다.

Micro Focus 관련 EBCDIC 데이터 정렬

Micro Focus의 경우, 다음 데이터 정렬에 필요한 문자 하위 집합을 시프트합니다.

```
da-DK-cp1142m-x-icu
de-DE-cp1141m-x-icu
en-GB-cp1146m-x-icu
en-US-cp1140m-x-icu
es-ES-cp1145m-x-icu
fi-FI-cp1143m-x-icu
fr-FR-cp1147m-x-icu
it-IT-cp1144m-x-icu
nl-BE-cp1148m-x-icu
```

Example 데이터 정렬에 따른 Micro Focus 데이터 시프트:

en_us_cp1140m

코드 시프트:

```
0000    0180
```

```
00A6    0160
00B8    0161
00BC    017D
00BD    017E
00BE    0152
00A8    0153
00B4    0178
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0000","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "00A6","TargetCharacterCodePoint": "0160"}
,{"SourceCharacterCodePoint": "00B8","TargetCharacterCodePoint": "0161"}
,{"SourceCharacterCodePoint": "00BC","TargetCharacterCodePoint": "017D"}
,{"SourceCharacterCodePoint": "00BD","TargetCharacterCodePoint": "017E"}
,{"SourceCharacterCodePoint": "00BE","TargetCharacterCodePoint": "0152"}
,{"SourceCharacterCodePoint": "00A8","TargetCharacterCodePoint": "0153"}
,{"SourceCharacterCodePoint": "00B4","TargetCharacterCodePoint": "0178"}
```

en_us_cp1141m

코드 시프트:

```
0000    0180
00B8    0160
00BC    0161
00BD    017D
00BE    017E
00A8    0152
00B4    0153
00A6    0178
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0000","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "00B8","TargetCharacterCodePoint": "0160"}
,{"SourceCharacterCodePoint": "00BC","TargetCharacterCodePoint": "0161"}
,{"SourceCharacterCodePoint": "00BD","TargetCharacterCodePoint": "017D"}
,{"SourceCharacterCodePoint": "00BE","TargetCharacterCodePoint": "017E"}
,{"SourceCharacterCodePoint": "00A8","TargetCharacterCodePoint": "0152"}
,{"SourceCharacterCodePoint": "00B4","TargetCharacterCodePoint": "0153"}
```

```
,{"SourceCharacterCodePoint": "00A6","TargetCharacterCodePoint": "0178"}
```

en_us_cp1142m

코드 시프트:

```
0000    0180
00A6    0160
00B8    0161
00BC    017D
00BD    017E
00BE    0152
00A8    0153
00B4    0178
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0000","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "00A6","TargetCharacterCodePoint": "0160"}
,{"SourceCharacterCodePoint": "00B8","TargetCharacterCodePoint": "0161"}
,{"SourceCharacterCodePoint": "00BC","TargetCharacterCodePoint": "017D"}
,{"SourceCharacterCodePoint": "00BD","TargetCharacterCodePoint": "017E"}
,{"SourceCharacterCodePoint": "00BE","TargetCharacterCodePoint": "0152"}
,{"SourceCharacterCodePoint": "00A8","TargetCharacterCodePoint": "0153"}
,{"SourceCharacterCodePoint": "00B4","TargetCharacterCodePoint": "0178"}
```

en_us_cp1143m

코드 시프트:

```
0000    0180
00B8    0160
00BC    0161
00BD    017D
00BE    017E
00A8    0152
00B4    0153
```

```
00A6    0178
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0000","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "00B8","TargetCharacterCodePoint": "0160"}
,{"SourceCharacterCodePoint": "00BC","TargetCharacterCodePoint": "0161"}
,{"SourceCharacterCodePoint": "00BD","TargetCharacterCodePoint": "017D"}
,{"SourceCharacterCodePoint": "00BE","TargetCharacterCodePoint": "017E"}
,{"SourceCharacterCodePoint": "00A8","TargetCharacterCodePoint": "0152"}
,{"SourceCharacterCodePoint": "00B4","TargetCharacterCodePoint": "0153"}
,{"SourceCharacterCodePoint": "00A6","TargetCharacterCodePoint": "0178"}
```

en_us_cp1144m

코드 시프트:

```
0000    0180
00B8    0160
00BC    0161
00BD    017D
00BE    017E
00A8    0152
00B4    0153
00A6    0178
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0000","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "00B8","TargetCharacterCodePoint": "0160"}
,{"SourceCharacterCodePoint": "00BC","TargetCharacterCodePoint": "0161"}
,{"SourceCharacterCodePoint": "00BD","TargetCharacterCodePoint": "017D"}
,{"SourceCharacterCodePoint": "00BE","TargetCharacterCodePoint": "017E"}
,{"SourceCharacterCodePoint": "00A8","TargetCharacterCodePoint": "0152"}
,{"SourceCharacterCodePoint": "00B4","TargetCharacterCodePoint": "0153"}
,{"SourceCharacterCodePoint": "00A6","TargetCharacterCodePoint": "0178"}
```

en_us_cp1145m

코드 시프트:

```
0000    0180
00A6    0160
00B8    0161
00A8    017D
00BC    017E
00BD    0152
00BE    0153
00B4    0178
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0000","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "00A6","TargetCharacterCodePoint": "0160"}
,{"SourceCharacterCodePoint": "00B8","TargetCharacterCodePoint": "0161"}
,{"SourceCharacterCodePoint": "00A8","TargetCharacterCodePoint": "017D"}
,{"SourceCharacterCodePoint": "00BC","TargetCharacterCodePoint": "017E"}
,{"SourceCharacterCodePoint": "00BD","TargetCharacterCodePoint": "0152"}
,{"SourceCharacterCodePoint": "00BE","TargetCharacterCodePoint": "0153"}
,{"SourceCharacterCodePoint": "00B4","TargetCharacterCodePoint": "0178"}
```

en_us_cp1146m

코드 시프트:

```
0000    0180
00A6    0160
00B8    0161
00BC    017D
00BD    017E
00BE    0152
00A8    0153
00B4    0178
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0000","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "00A6","TargetCharacterCodePoint": "0160"}
,{"SourceCharacterCodePoint": "00B8","TargetCharacterCodePoint": "0161"}
,{"SourceCharacterCodePoint": "00BC","TargetCharacterCodePoint": "017D"}
,{"SourceCharacterCodePoint": "00BD","TargetCharacterCodePoint": "017E"}
,{"SourceCharacterCodePoint": "00BE","TargetCharacterCodePoint": "0152"}
,{"SourceCharacterCodePoint": "00A8","TargetCharacterCodePoint": "0153"}
,{"SourceCharacterCodePoint": "00B4","TargetCharacterCodePoint": "0178"}
```

en_us_cp1147m

코드 시프트:

```
0000    0180
00B8    0160
00A8    0161
00BC    017D
00BD    017E
00BE    0152
00B4    0153
00A6    0178
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0000","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "00B8","TargetCharacterCodePoint": "0160"}
,{"SourceCharacterCodePoint": "00A8","TargetCharacterCodePoint": "0161"}
,{"SourceCharacterCodePoint": "00BC","TargetCharacterCodePoint": "017D"}
,{"SourceCharacterCodePoint": "00BD","TargetCharacterCodePoint": "017E"}
,{"SourceCharacterCodePoint": "00BE","TargetCharacterCodePoint": "0152"}
,{"SourceCharacterCodePoint": "00B4","TargetCharacterCodePoint": "0153"}
,{"SourceCharacterCodePoint": "00A6","TargetCharacterCodePoint": "0178"}
```

en_us_cp1148m

코드 시프트:

```
0000    0180
```

```
00A6    0160
00B8    0161
00BC    017D
00BD    017E
00BE    0152
00A8    0153
00B4    0178
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0000","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "00A6","TargetCharacterCodePoint": "0160"}
,{"SourceCharacterCodePoint": "00B8","TargetCharacterCodePoint": "0161"}
,{"SourceCharacterCodePoint": "00BC","TargetCharacterCodePoint": "017D"}
,{"SourceCharacterCodePoint": "00BD","TargetCharacterCodePoint": "017E"}
,{"SourceCharacterCodePoint": "00BE","TargetCharacterCodePoint": "0152"}
,{"SourceCharacterCodePoint": "00A8","TargetCharacterCodePoint": "0153"}
,{"SourceCharacterCodePoint": "00B4","TargetCharacterCodePoint": "0178"}
```

BluAge 관련 EBCDIC 데이터 정렬

BluAge의 경우, 필요에 따라 다음의 낮은 값과 높은 값을 모두 시프트합니다. 이러한 데이터 정렬은 Mainframe Migration BluAge 서비스를 지원하는 용도로만 사용해야 합니다.

```
da-DK-cp1142b-x-icu
 da-DK-cp277b-x-icu
 de-DE-cp1141b-x-icu
 de-DE-cp273b-x-icu
 en-GB-cp1146b-x-icu
 en-GB-cp285b-x-icu
 en-US-cp037b-x-icu
 en-US-cp1140b-x-icu
 es-ES-cp1145b-x-icu
 es-ES-cp284b-x-icu
 fi-FI-cp1143b-x-icu
 fi-FI-cp278b-x-icu
 fr-FR-cp1147b-x-icu
 fr-FR-cp297b-x-icu
 it-IT-cp1144b-x-icu
 it-IT-cp280b-x-icu
 nl-BE-cp1148b-x-icu
```

```
nl-BE-cp500b-x-icu
```

Example BluAge 데이터 시프트:

da-DK-cp277b 및 da-DK-cp1142b

코드 시프트:

```
0180    0180
0001    0181
0002    0182
0003    0183
009C    0184
0009    0185
0086    0186
007F    0187
0097    0188
008D    0189
008E    018A
000B    018B
000C    018C
000D    018D
000E    018E
000F    018F
0010    0190
0011    0191
0012    0192
0013    0193
009D    0194
0085    0195
0008    0196
0087    0197
0018    0198
0019    0199
0092    019A
008F    019B
001C    019C
001D    019D
001E    019E
001F    019F
0080    01A0
0081    01A1
0082    01A2
```

```
0083    01A3
0084    01A4
000A    01A5
0017    01A6
001B    01A7
0088    01A8
0089    01A9
008A    01AA
008B    01AB
008C    01AC
0005    01AD
0006    01AE
0007    01AF
0090    01B0
0091    01B1
0016    01B2
0093    01B3
0094    01B4
0095    01B5
0096    01B6
0004    01B7
0098    01B8
0099    01B9
009A    01BA
009B    01BB
0014    01BC
0015    01BD
009E    01BE
001A    01BF
009F    027F
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0180","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "0001","TargetCharacterCodePoint": "0181"}
,{"SourceCharacterCodePoint": "0002","TargetCharacterCodePoint": "0182"}
,{"SourceCharacterCodePoint": "0003","TargetCharacterCodePoint": "0183"}
,{"SourceCharacterCodePoint": "009C","TargetCharacterCodePoint": "0184"}
,{"SourceCharacterCodePoint": "0009","TargetCharacterCodePoint": "0185"}
,{"SourceCharacterCodePoint": "0086","TargetCharacterCodePoint": "0186"}
,{"SourceCharacterCodePoint": "007F","TargetCharacterCodePoint": "0187"}
,{"SourceCharacterCodePoint": "0097","TargetCharacterCodePoint": "0188"}
```

```
,{"SourceCharacterCodePoint": "008D","TargetCharacterCodePoint": "0189"}
,{"SourceCharacterCodePoint": "008E","TargetCharacterCodePoint": "018A"}
,{"SourceCharacterCodePoint": "000B","TargetCharacterCodePoint": "018B"}
,{"SourceCharacterCodePoint": "000C","TargetCharacterCodePoint": "018C"}
,{"SourceCharacterCodePoint": "000D","TargetCharacterCodePoint": "018D"}
,{"SourceCharacterCodePoint": "000E","TargetCharacterCodePoint": "018E"}
,{"SourceCharacterCodePoint": "000F","TargetCharacterCodePoint": "018F"}
,{"SourceCharacterCodePoint": "0010","TargetCharacterCodePoint": "0190"}
,{"SourceCharacterCodePoint": "0011","TargetCharacterCodePoint": "0191"}
,{"SourceCharacterCodePoint": "0012","TargetCharacterCodePoint": "0192"}
,{"SourceCharacterCodePoint": "0013","TargetCharacterCodePoint": "0193"}
,{"SourceCharacterCodePoint": "009D","TargetCharacterCodePoint": "0194"}
,{"SourceCharacterCodePoint": "0085","TargetCharacterCodePoint": "0195"}
,{"SourceCharacterCodePoint": "0008","TargetCharacterCodePoint": "0196"}
,{"SourceCharacterCodePoint": "0087","TargetCharacterCodePoint": "0197"}
,{"SourceCharacterCodePoint": "0018","TargetCharacterCodePoint": "0198"}
,{"SourceCharacterCodePoint": "0019","TargetCharacterCodePoint": "0199"}
,{"SourceCharacterCodePoint": "0092","TargetCharacterCodePoint": "019A"}
,{"SourceCharacterCodePoint": "008F","TargetCharacterCodePoint": "019B"}
,{"SourceCharacterCodePoint": "001C","TargetCharacterCodePoint": "019C"}
,{"SourceCharacterCodePoint": "001D","TargetCharacterCodePoint": "019D"}
,{"SourceCharacterCodePoint": "001E","TargetCharacterCodePoint": "019E"}
,{"SourceCharacterCodePoint": "001F","TargetCharacterCodePoint": "019F"}
,{"SourceCharacterCodePoint": "0080","TargetCharacterCodePoint": "01A0"}
,{"SourceCharacterCodePoint": "0081","TargetCharacterCodePoint": "01A1"}
,{"SourceCharacterCodePoint": "0082","TargetCharacterCodePoint": "01A2"}
,{"SourceCharacterCodePoint": "0083","TargetCharacterCodePoint": "01A3"}
,{"SourceCharacterCodePoint": "0084","TargetCharacterCodePoint": "01A4"}
,{"SourceCharacterCodePoint": "000A","TargetCharacterCodePoint": "01A5"}
,{"SourceCharacterCodePoint": "0017","TargetCharacterCodePoint": "01A6"}
,{"SourceCharacterCodePoint": "001B","TargetCharacterCodePoint": "01A7"}
,{"SourceCharacterCodePoint": "0088","TargetCharacterCodePoint": "01A8"}
,{"SourceCharacterCodePoint": "0089","TargetCharacterCodePoint": "01A9"}
,{"SourceCharacterCodePoint": "008A","TargetCharacterCodePoint": "01AA"}
,{"SourceCharacterCodePoint": "008B","TargetCharacterCodePoint": "01AB"}
,{"SourceCharacterCodePoint": "008C","TargetCharacterCodePoint": "01AC"}
,{"SourceCharacterCodePoint": "0005","TargetCharacterCodePoint": "01AD"}
,{"SourceCharacterCodePoint": "0006","TargetCharacterCodePoint": "01AE"}
,{"SourceCharacterCodePoint": "0007","TargetCharacterCodePoint": "01AF"}
,{"SourceCharacterCodePoint": "0090","TargetCharacterCodePoint": "01B0"}
,{"SourceCharacterCodePoint": "0091","TargetCharacterCodePoint": "01B1"}
,{"SourceCharacterCodePoint": "0016","TargetCharacterCodePoint": "01B2"}
,{"SourceCharacterCodePoint": "0093","TargetCharacterCodePoint": "01B3"}
,{"SourceCharacterCodePoint": "0094","TargetCharacterCodePoint": "01B4"}
```

```
,{"SourceCharacterCodePoint": "0095","TargetCharacterCodePoint": "01B5"}
,{"SourceCharacterCodePoint": "0096","TargetCharacterCodePoint": "01B6"}
,{"SourceCharacterCodePoint": "0004","TargetCharacterCodePoint": "01B7"}
,{"SourceCharacterCodePoint": "0098","TargetCharacterCodePoint": "01B8"}
,{"SourceCharacterCodePoint": "0099","TargetCharacterCodePoint": "01B9"}
,{"SourceCharacterCodePoint": "009A","TargetCharacterCodePoint": "01BA"}
,{"SourceCharacterCodePoint": "009B","TargetCharacterCodePoint": "01BB"}
,{"SourceCharacterCodePoint": "0014","TargetCharacterCodePoint": "01BC"}
,{"SourceCharacterCodePoint": "0015","TargetCharacterCodePoint": "01BD"}
,{"SourceCharacterCodePoint": "009E","TargetCharacterCodePoint": "01BE"}
,{"SourceCharacterCodePoint": "001A","TargetCharacterCodePoint": "01BF"}
,{"SourceCharacterCodePoint": "009F","TargetCharacterCodePoint": "027F"}
```

de-DE-273b 및 de-DE-1141b

코드 시프트:

```
0180    0180
0001    0181
0002    0182
0003    0183
009C    0184
0009    0185
0086    0186
007F    0187
0097    0188
008D    0189
008E    018A
000B    018B
000C    018C
000D    018D
000E    018E
000F    018F
0010    0190
0011    0191
0012    0192
0013    0193
009D    0194
0085    0195
0008    0196
0087    0197
0018    0198
```

```
0019    0199
0092    019A
008F    019B
001C    019C
001D    019D
001E    019E
001F    019F
0080    01A0
0081    01A1
0082    01A2
0083    01A3
0084    01A4
000A    01A5
0017    01A6
001B    01A7
0088    01A8
0089    01A9
008A    01AA
008B    01AB
008C    01AC
0005    01AD
0006    01AE
0007    01AF
0090    01B0
0091    01B1
0016    01B2
0093    01B3
0094    01B4
0095    01B5
0096    01B6
0004    01B7
0098    01B8
0099    01B9
009A    01BA
009B    01BB
0014    01BC
0015    01BD
009E    01BE
001A    01BF
009F    027F
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0180","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "0001","TargetCharacterCodePoint": "0181"}
,{"SourceCharacterCodePoint": "0002","TargetCharacterCodePoint": "0182"}
,{"SourceCharacterCodePoint": "0003","TargetCharacterCodePoint": "0183"}
,{"SourceCharacterCodePoint": "009C","TargetCharacterCodePoint": "0184"}
,{"SourceCharacterCodePoint": "0009","TargetCharacterCodePoint": "0185"}
,{"SourceCharacterCodePoint": "0086","TargetCharacterCodePoint": "0186"}
,{"SourceCharacterCodePoint": "007F","TargetCharacterCodePoint": "0187"}
,{"SourceCharacterCodePoint": "0097","TargetCharacterCodePoint": "0188"}
,{"SourceCharacterCodePoint": "008D","TargetCharacterCodePoint": "0189"}
,{"SourceCharacterCodePoint": "008E","TargetCharacterCodePoint": "018A"}
,{"SourceCharacterCodePoint": "000B","TargetCharacterCodePoint": "018B"}
,{"SourceCharacterCodePoint": "000C","TargetCharacterCodePoint": "018C"}
,{"SourceCharacterCodePoint": "000D","TargetCharacterCodePoint": "018D"}
,{"SourceCharacterCodePoint": "000E","TargetCharacterCodePoint": "018E"}
,{"SourceCharacterCodePoint": "000F","TargetCharacterCodePoint": "018F"}
,{"SourceCharacterCodePoint": "0010","TargetCharacterCodePoint": "0190"}
,{"SourceCharacterCodePoint": "0011","TargetCharacterCodePoint": "0191"}
,{"SourceCharacterCodePoint": "0012","TargetCharacterCodePoint": "0192"}
,{"SourceCharacterCodePoint": "0013","TargetCharacterCodePoint": "0193"}
,{"SourceCharacterCodePoint": "009D","TargetCharacterCodePoint": "0194"}
,{"SourceCharacterCodePoint": "0085","TargetCharacterCodePoint": "0195"}
,{"SourceCharacterCodePoint": "0008","TargetCharacterCodePoint": "0196"}
,{"SourceCharacterCodePoint": "0087","TargetCharacterCodePoint": "0197"}
,{"SourceCharacterCodePoint": "0018","TargetCharacterCodePoint": "0198"}
,{"SourceCharacterCodePoint": "0019","TargetCharacterCodePoint": "0199"}
,{"SourceCharacterCodePoint": "0092","TargetCharacterCodePoint": "019A"}
,{"SourceCharacterCodePoint": "008F","TargetCharacterCodePoint": "019B"}
,{"SourceCharacterCodePoint": "001C","TargetCharacterCodePoint": "019C"}
,{"SourceCharacterCodePoint": "001D","TargetCharacterCodePoint": "019D"}
,{"SourceCharacterCodePoint": "001E","TargetCharacterCodePoint": "019E"}
,{"SourceCharacterCodePoint": "001F","TargetCharacterCodePoint": "019F"}
,{"SourceCharacterCodePoint": "0080","TargetCharacterCodePoint": "01A0"}
,{"SourceCharacterCodePoint": "0081","TargetCharacterCodePoint": "01A1"}
,{"SourceCharacterCodePoint": "0082","TargetCharacterCodePoint": "01A2"}
,{"SourceCharacterCodePoint": "0083","TargetCharacterCodePoint": "01A3"}
,{"SourceCharacterCodePoint": "0084","TargetCharacterCodePoint": "01A4"}
,{"SourceCharacterCodePoint": "000A","TargetCharacterCodePoint": "01A5"}
,{"SourceCharacterCodePoint": "0017","TargetCharacterCodePoint": "01A6"}
,{"SourceCharacterCodePoint": "001B","TargetCharacterCodePoint": "01A7"}
,{"SourceCharacterCodePoint": "0088","TargetCharacterCodePoint": "01A8"}
,{"SourceCharacterCodePoint": "0089","TargetCharacterCodePoint": "01A9"}
,{"SourceCharacterCodePoint": "008A","TargetCharacterCodePoint": "01AA"}
```

```
,{"SourceCharacterCodePoint": "008B","TargetCharacterCodePoint": "01AB"}
,{"SourceCharacterCodePoint": "008C","TargetCharacterCodePoint": "01AC"}
,{"SourceCharacterCodePoint": "0005","TargetCharacterCodePoint": "01AD"}
,{"SourceCharacterCodePoint": "0006","TargetCharacterCodePoint": "01AE"}
,{"SourceCharacterCodePoint": "0007","TargetCharacterCodePoint": "01AF"}
,{"SourceCharacterCodePoint": "0090","TargetCharacterCodePoint": "01B0"}
,{"SourceCharacterCodePoint": "0091","TargetCharacterCodePoint": "01B1"}
,{"SourceCharacterCodePoint": "0016","TargetCharacterCodePoint": "01B2"}
,{"SourceCharacterCodePoint": "0093","TargetCharacterCodePoint": "01B3"}
,{"SourceCharacterCodePoint": "0094","TargetCharacterCodePoint": "01B4"}
,{"SourceCharacterCodePoint": "0095","TargetCharacterCodePoint": "01B5"}
,{"SourceCharacterCodePoint": "0096","TargetCharacterCodePoint": "01B6"}
,{"SourceCharacterCodePoint": "0004","TargetCharacterCodePoint": "01B7"}
,{"SourceCharacterCodePoint": "0098","TargetCharacterCodePoint": "01B8"}
,{"SourceCharacterCodePoint": "0099","TargetCharacterCodePoint": "01B9"}
,{"SourceCharacterCodePoint": "009A","TargetCharacterCodePoint": "01BA"}
,{"SourceCharacterCodePoint": "009B","TargetCharacterCodePoint": "01BB"}
,{"SourceCharacterCodePoint": "0014","TargetCharacterCodePoint": "01BC"}
,{"SourceCharacterCodePoint": "0015","TargetCharacterCodePoint": "01BD"}
,{"SourceCharacterCodePoint": "009E","TargetCharacterCodePoint": "01BE"}
,{"SourceCharacterCodePoint": "001A","TargetCharacterCodePoint": "01BF"}
,{"SourceCharacterCodePoint": "009F","TargetCharacterCodePoint": "027F"}
```

en-GB-285b 및 en-GB-1146b

코드 시프트:

```
0180    0180
0001    0181
0002    0182
0003    0183
009C    0184
0009    0185
0086    0186
007F    0187
0097    0188
008D    0189
008E    018A
000B    018B
000C    018C
000D    018D
000E    018E
```

```
000F    018F
0010    0190
0011    0191
0012    0192
0013    0193
009D    0194
0085    0195
0008    0196
0087    0197
0018    0198
0019    0199
0092    019A
008F    019B
001C    019C
001D    019D
001E    019E
001F    019F
0080    01A0
0081    01A1
0082    01A2
0083    01A3
0084    01A4
000A    01A5
0017    01A6
001B    01A7
0088    01A8
0089    01A9
008A    01AA
008B    01AB
008C    01AC
0005    01AD
0006    01AE
0007    01AF
0090    01B0
0091    01B1
0016    01B2
0093    01B3
0094    01B4
0095    01B5
0096    01B6
0004    01B7
0098    01B8
0099    01B9
009A    01BA
```

```
009B    01BB
0014    01BC
0015    01BD
009E    01BE
001A    01BF
009F    027F
```

AWS DMS 작업에 해당하는 입력 매핑:

```
{"SourceCharacterCodePoint": "0180","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "0001","TargetCharacterCodePoint": "0181"}
,{"SourceCharacterCodePoint": "0002","TargetCharacterCodePoint": "0182"}
,{"SourceCharacterCodePoint": "0003","TargetCharacterCodePoint": "0183"}
,{"SourceCharacterCodePoint": "009C","TargetCharacterCodePoint": "0184"}
,{"SourceCharacterCodePoint": "0009","TargetCharacterCodePoint": "0185"}
,{"SourceCharacterCodePoint": "0086","TargetCharacterCodePoint": "0186"}
,{"SourceCharacterCodePoint": "007F","TargetCharacterCodePoint": "0187"}
,{"SourceCharacterCodePoint": "0097","TargetCharacterCodePoint": "0188"}
,{"SourceCharacterCodePoint": "008D","TargetCharacterCodePoint": "0189"}
,{"SourceCharacterCodePoint": "008E","TargetCharacterCodePoint": "018A"}
,{"SourceCharacterCodePoint": "000B","TargetCharacterCodePoint": "018B"}
,{"SourceCharacterCodePoint": "000C","TargetCharacterCodePoint": "018C"}
,{"SourceCharacterCodePoint": "000D","TargetCharacterCodePoint": "018D"}
,{"SourceCharacterCodePoint": "000E","TargetCharacterCodePoint": "018E"}
,{"SourceCharacterCodePoint": "000F","TargetCharacterCodePoint": "018F"}
,{"SourceCharacterCodePoint": "0010","TargetCharacterCodePoint": "0190"}
,{"SourceCharacterCodePoint": "0011","TargetCharacterCodePoint": "0191"}
,{"SourceCharacterCodePoint": "0012","TargetCharacterCodePoint": "0192"}
,{"SourceCharacterCodePoint": "0013","TargetCharacterCodePoint": "0193"}
,{"SourceCharacterCodePoint": "009D","TargetCharacterCodePoint": "0194"}
,{"SourceCharacterCodePoint": "0085","TargetCharacterCodePoint": "0195"}
,{"SourceCharacterCodePoint": "0008","TargetCharacterCodePoint": "0196"}
,{"SourceCharacterCodePoint": "0087","TargetCharacterCodePoint": "0197"}
,{"SourceCharacterCodePoint": "0018","TargetCharacterCodePoint": "0198"}
,{"SourceCharacterCodePoint": "0019","TargetCharacterCodePoint": "0199"}
,{"SourceCharacterCodePoint": "0092","TargetCharacterCodePoint": "019A"}
,{"SourceCharacterCodePoint": "008F","TargetCharacterCodePoint": "019B"}
,{"SourceCharacterCodePoint": "001C","TargetCharacterCodePoint": "019C"}
,{"SourceCharacterCodePoint": "001D","TargetCharacterCodePoint": "019D"}
,{"SourceCharacterCodePoint": "001E","TargetCharacterCodePoint": "019E"}
,{"SourceCharacterCodePoint": "001F","TargetCharacterCodePoint": "019F"}
,{"SourceCharacterCodePoint": "0080","TargetCharacterCodePoint": "01A0"}
```

```
,{"SourceCharacterCodePoint": "0081","TargetCharacterCodePoint": "01A1"}
,{"SourceCharacterCodePoint": "0082","TargetCharacterCodePoint": "01A2"}
,{"SourceCharacterCodePoint": "0083","TargetCharacterCodePoint": "01A3"}
,{"SourceCharacterCodePoint": "0084","TargetCharacterCodePoint": "01A4"}
,{"SourceCharacterCodePoint": "000A","TargetCharacterCodePoint": "01A5"}
,{"SourceCharacterCodePoint": "0017","TargetCharacterCodePoint": "01A6"}
,{"SourceCharacterCodePoint": "001B","TargetCharacterCodePoint": "01A7"}
,{"SourceCharacterCodePoint": "0088","TargetCharacterCodePoint": "01A8"}
,{"SourceCharacterCodePoint": "0089","TargetCharacterCodePoint": "01A9"}
,{"SourceCharacterCodePoint": "008A","TargetCharacterCodePoint": "01AA"}
,{"SourceCharacterCodePoint": "008B","TargetCharacterCodePoint": "01AB"}
,{"SourceCharacterCodePoint": "008C","TargetCharacterCodePoint": "01AC"}
,{"SourceCharacterCodePoint": "0005","TargetCharacterCodePoint": "01AD"}
,{"SourceCharacterCodePoint": "0006","TargetCharacterCodePoint": "01AE"}
,{"SourceCharacterCodePoint": "0007","TargetCharacterCodePoint": "01AF"}
,{"SourceCharacterCodePoint": "0090","TargetCharacterCodePoint": "01B0"}
,{"SourceCharacterCodePoint": "0091","TargetCharacterCodePoint": "01B1"}
,{"SourceCharacterCodePoint": "0016","TargetCharacterCodePoint": "01B2"}
,{"SourceCharacterCodePoint": "0093","TargetCharacterCodePoint": "01B3"}
,{"SourceCharacterCodePoint": "0094","TargetCharacterCodePoint": "01B4"}
,{"SourceCharacterCodePoint": "0095","TargetCharacterCodePoint": "01B5"}
,{"SourceCharacterCodePoint": "0096","TargetCharacterCodePoint": "01B6"}
,{"SourceCharacterCodePoint": "0004","TargetCharacterCodePoint": "01B7"}
,{"SourceCharacterCodePoint": "0098","TargetCharacterCodePoint": "01B8"}
,{"SourceCharacterCodePoint": "0099","TargetCharacterCodePoint": "01B9"}
,{"SourceCharacterCodePoint": "009A","TargetCharacterCodePoint": "01BA"}
,{"SourceCharacterCodePoint": "009B","TargetCharacterCodePoint": "01BB"}
,{"SourceCharacterCodePoint": "0014","TargetCharacterCodePoint": "01BC"}
,{"SourceCharacterCodePoint": "0015","TargetCharacterCodePoint": "01BD"}
,{"SourceCharacterCodePoint": "009E","TargetCharacterCodePoint": "01BE"}
,{"SourceCharacterCodePoint": "001A","TargetCharacterCodePoint": "01BF"}
,{"SourceCharacterCodePoint": "009F","TargetCharacterCodePoint": "027F"}
```

en-us-037b 및 en-us-1140b

코드 시프트:

```
0180    0180
0001    0181
0002    0182
0003    0183
009C    0184
```

```
0009    0185
0086    0186
007F    0187
0097    0188
008D    0189
008E    018A
000B    018B
000C    018C
000D    018D
000E    018E
000F    018F
0010    0190
0011    0191
0012    0192
0013    0193
009D    0194
0085    0195
0008    0196
0087    0197
0018    0198
0019    0199
0092    019A
008F    019B
001C    019C
001D    019D
001E    019E
001F    019F
0080    01A0
0081    01A1
0082    01A2
0083    01A3
0084    01A4
000A    01A5
0017    01A6
001B    01A7
0088    01A8
0089    01A9
008A    01AA
008B    01AB
008C    01AC
0005    01AD
0006    01AE
0007    01AF
0090    01B0
```

```
0091    01B1
0016    01B2
0093    01B3
0094    01B4
0095    01B5
0096    01B6
0004    01B7
0098    01B8
0099    01B9
009A    01BA
009B    01BB
0014    01BC
0015    01BD
009E    01BE
001A    01BF
009F    027F
```

AWS DMS 작업에 해당하는 입력 매핑:

```
{"SourceCharacterCodePoint": "0180","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "0001","TargetCharacterCodePoint": "0181"}
,{"SourceCharacterCodePoint": "0002","TargetCharacterCodePoint": "0182"}
,{"SourceCharacterCodePoint": "0003","TargetCharacterCodePoint": "0183"}
,{"SourceCharacterCodePoint": "009C","TargetCharacterCodePoint": "0184"}
,{"SourceCharacterCodePoint": "0009","TargetCharacterCodePoint": "0185"}
,{"SourceCharacterCodePoint": "0086","TargetCharacterCodePoint": "0186"}
,{"SourceCharacterCodePoint": "007F","TargetCharacterCodePoint": "0187"}
,{"SourceCharacterCodePoint": "0097","TargetCharacterCodePoint": "0188"}
,{"SourceCharacterCodePoint": "008D","TargetCharacterCodePoint": "0189"}
,{"SourceCharacterCodePoint": "008E","TargetCharacterCodePoint": "018A"}
,{"SourceCharacterCodePoint": "000B","TargetCharacterCodePoint": "018B"}
,{"SourceCharacterCodePoint": "000C","TargetCharacterCodePoint": "018C"}
,{"SourceCharacterCodePoint": "000D","TargetCharacterCodePoint": "018D"}
,{"SourceCharacterCodePoint": "000E","TargetCharacterCodePoint": "018E"}
,{"SourceCharacterCodePoint": "000F","TargetCharacterCodePoint": "018F"}
,{"SourceCharacterCodePoint": "0010","TargetCharacterCodePoint": "0190"}
,{"SourceCharacterCodePoint": "0011","TargetCharacterCodePoint": "0191"}
,{"SourceCharacterCodePoint": "0012","TargetCharacterCodePoint": "0192"}
,{"SourceCharacterCodePoint": "0013","TargetCharacterCodePoint": "0193"}
,{"SourceCharacterCodePoint": "009D","TargetCharacterCodePoint": "0194"}
,{"SourceCharacterCodePoint": "0085","TargetCharacterCodePoint": "0195"}
,{"SourceCharacterCodePoint": "0008","TargetCharacterCodePoint": "0196"}
```

```
,{"SourceCharacterCodePoint": "0087","TargetCharacterCodePoint": "0197"}
,{"SourceCharacterCodePoint": "0018","TargetCharacterCodePoint": "0198"}
,{"SourceCharacterCodePoint": "0019","TargetCharacterCodePoint": "0199"}
,{"SourceCharacterCodePoint": "0092","TargetCharacterCodePoint": "019A"}
,{"SourceCharacterCodePoint": "008F","TargetCharacterCodePoint": "019B"}
,{"SourceCharacterCodePoint": "001C","TargetCharacterCodePoint": "019C"}
,{"SourceCharacterCodePoint": "001D","TargetCharacterCodePoint": "019D"}
,{"SourceCharacterCodePoint": "001E","TargetCharacterCodePoint": "019E"}
,{"SourceCharacterCodePoint": "001F","TargetCharacterCodePoint": "019F"}
,{"SourceCharacterCodePoint": "0080","TargetCharacterCodePoint": "01A0"}
,{"SourceCharacterCodePoint": "0081","TargetCharacterCodePoint": "01A1"}
,{"SourceCharacterCodePoint": "0082","TargetCharacterCodePoint": "01A2"}
,{"SourceCharacterCodePoint": "0083","TargetCharacterCodePoint": "01A3"}
,{"SourceCharacterCodePoint": "0084","TargetCharacterCodePoint": "01A4"}
,{"SourceCharacterCodePoint": "000A","TargetCharacterCodePoint": "01A5"}
,{"SourceCharacterCodePoint": "0017","TargetCharacterCodePoint": "01A6"}
,{"SourceCharacterCodePoint": "001B","TargetCharacterCodePoint": "01A7"}
,{"SourceCharacterCodePoint": "0088","TargetCharacterCodePoint": "01A8"}
,{"SourceCharacterCodePoint": "0089","TargetCharacterCodePoint": "01A9"}
,{"SourceCharacterCodePoint": "008A","TargetCharacterCodePoint": "01AA"}
,{"SourceCharacterCodePoint": "008B","TargetCharacterCodePoint": "01AB"}
,{"SourceCharacterCodePoint": "008C","TargetCharacterCodePoint": "01AC"}
,{"SourceCharacterCodePoint": "0005","TargetCharacterCodePoint": "01AD"}
,{"SourceCharacterCodePoint": "0006","TargetCharacterCodePoint": "01AE"}
,{"SourceCharacterCodePoint": "0007","TargetCharacterCodePoint": "01AF"}
,{"SourceCharacterCodePoint": "0090","TargetCharacterCodePoint": "01B0"}
,{"SourceCharacterCodePoint": "0091","TargetCharacterCodePoint": "01B1"}
,{"SourceCharacterCodePoint": "0016","TargetCharacterCodePoint": "01B2"}
,{"SourceCharacterCodePoint": "0093","TargetCharacterCodePoint": "01B3"}
,{"SourceCharacterCodePoint": "0094","TargetCharacterCodePoint": "01B4"}
,{"SourceCharacterCodePoint": "0095","TargetCharacterCodePoint": "01B5"}
,{"SourceCharacterCodePoint": "0096","TargetCharacterCodePoint": "01B6"}
,{"SourceCharacterCodePoint": "0004","TargetCharacterCodePoint": "01B7"}
,{"SourceCharacterCodePoint": "0098","TargetCharacterCodePoint": "01B8"}
,{"SourceCharacterCodePoint": "0099","TargetCharacterCodePoint": "01B9"}
,{"SourceCharacterCodePoint": "009A","TargetCharacterCodePoint": "01BA"}
,{"SourceCharacterCodePoint": "009B","TargetCharacterCodePoint": "01BB"}
,{"SourceCharacterCodePoint": "0014","TargetCharacterCodePoint": "01BC"}
,{"SourceCharacterCodePoint": "0015","TargetCharacterCodePoint": "01BD"}
,{"SourceCharacterCodePoint": "009E","TargetCharacterCodePoint": "01BE"}
,{"SourceCharacterCodePoint": "001A","TargetCharacterCodePoint": "01BF"}
,{"SourceCharacterCodePoint": "009F","TargetCharacterCodePoint": "027F"}
```

es-ES-284b 및 es-ES-1145b

코드 시프트:

```
0180    0180
0001    0181
0002    0182
0003    0183
009C    0184
0009    0185
0086    0186
007F    0187
0097    0188
008D    0189
008E    018A
000B    018B
000C    018C
000D    018D
000E    018E
000F    018F
0010    0190
0011    0191
0012    0192
0013    0193
009D    0194
0085    0195
0008    0196
0087    0197
0018    0198
0019    0199
0092    019A
008F    019B
001C    019C
001D    019D
001E    019E
001F    019F
0080    01A0
0081    01A1
0082    01A2
0083    01A3
0084    01A4
000A    01A5
```

```
0017    01A6
001B    01A7
0088    01A8
0089    01A9
008A    01AA
008B    01AB
008C    01AC
0005    01AD
0006    01AE
0007    01AF
0090    01B0
0091    01B1
0016    01B2
0093    01B3
0094    01B4
0095    01B5
0096    01B6
0004    01B7
0098    01B8
0099    01B9
009A    01BA
009B    01BB
0014    01BC
0015    01BD
009E    01BE
001A    01BF
009F    027F
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0180","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "0001","TargetCharacterCodePoint": "0181"}
,{"SourceCharacterCodePoint": "0002","TargetCharacterCodePoint": "0182"}
,{"SourceCharacterCodePoint": "0003","TargetCharacterCodePoint": "0183"}
,{"SourceCharacterCodePoint": "009C","TargetCharacterCodePoint": "0184"}
,{"SourceCharacterCodePoint": "0009","TargetCharacterCodePoint": "0185"}
,{"SourceCharacterCodePoint": "0086","TargetCharacterCodePoint": "0186"}
,{"SourceCharacterCodePoint": "007F","TargetCharacterCodePoint": "0187"}
,{"SourceCharacterCodePoint": "0097","TargetCharacterCodePoint": "0188"}
,{"SourceCharacterCodePoint": "008D","TargetCharacterCodePoint": "0189"}
,{"SourceCharacterCodePoint": "008E","TargetCharacterCodePoint": "018A"}
,{"SourceCharacterCodePoint": "000B","TargetCharacterCodePoint": "018B"}
```

```
,{"SourceCharacterCodePoint": "000C","TargetCharacterCodePoint": "018C"}
,{"SourceCharacterCodePoint": "000D","TargetCharacterCodePoint": "018D"}
,{"SourceCharacterCodePoint": "000E","TargetCharacterCodePoint": "018E"}
,{"SourceCharacterCodePoint": "000F","TargetCharacterCodePoint": "018F"}
,{"SourceCharacterCodePoint": "0010","TargetCharacterCodePoint": "0190"}
,{"SourceCharacterCodePoint": "0011","TargetCharacterCodePoint": "0191"}
,{"SourceCharacterCodePoint": "0012","TargetCharacterCodePoint": "0192"}
,{"SourceCharacterCodePoint": "0013","TargetCharacterCodePoint": "0193"}
,{"SourceCharacterCodePoint": "009D","TargetCharacterCodePoint": "0194"}
,{"SourceCharacterCodePoint": "0085","TargetCharacterCodePoint": "0195"}
,{"SourceCharacterCodePoint": "0008","TargetCharacterCodePoint": "0196"}
,{"SourceCharacterCodePoint": "0087","TargetCharacterCodePoint": "0197"}
,{"SourceCharacterCodePoint": "0018","TargetCharacterCodePoint": "0198"}
,{"SourceCharacterCodePoint": "0019","TargetCharacterCodePoint": "0199"}
,{"SourceCharacterCodePoint": "0092","TargetCharacterCodePoint": "019A"}
,{"SourceCharacterCodePoint": "008F","TargetCharacterCodePoint": "019B"}
,{"SourceCharacterCodePoint": "001C","TargetCharacterCodePoint": "019C"}
,{"SourceCharacterCodePoint": "001D","TargetCharacterCodePoint": "019D"}
,{"SourceCharacterCodePoint": "001E","TargetCharacterCodePoint": "019E"}
,{"SourceCharacterCodePoint": "001F","TargetCharacterCodePoint": "019F"}
,{"SourceCharacterCodePoint": "0080","TargetCharacterCodePoint": "01A0"}
,{"SourceCharacterCodePoint": "0081","TargetCharacterCodePoint": "01A1"}
,{"SourceCharacterCodePoint": "0082","TargetCharacterCodePoint": "01A2"}
,{"SourceCharacterCodePoint": "0083","TargetCharacterCodePoint": "01A3"}
,{"SourceCharacterCodePoint": "0084","TargetCharacterCodePoint": "01A4"}
,{"SourceCharacterCodePoint": "000A","TargetCharacterCodePoint": "01A5"}
,{"SourceCharacterCodePoint": "0017","TargetCharacterCodePoint": "01A6"}
,{"SourceCharacterCodePoint": "001B","TargetCharacterCodePoint": "01A7"}
,{"SourceCharacterCodePoint": "0088","TargetCharacterCodePoint": "01A8"}
,{"SourceCharacterCodePoint": "0089","TargetCharacterCodePoint": "01A9"}
,{"SourceCharacterCodePoint": "008A","TargetCharacterCodePoint": "01AA"}
,{"SourceCharacterCodePoint": "008B","TargetCharacterCodePoint": "01AB"}
,{"SourceCharacterCodePoint": "008C","TargetCharacterCodePoint": "01AC"}
,{"SourceCharacterCodePoint": "0005","TargetCharacterCodePoint": "01AD"}
,{"SourceCharacterCodePoint": "0006","TargetCharacterCodePoint": "01AE"}
,{"SourceCharacterCodePoint": "0007","TargetCharacterCodePoint": "01AF"}
,{"SourceCharacterCodePoint": "0090","TargetCharacterCodePoint": "01B0"}
,{"SourceCharacterCodePoint": "0091","TargetCharacterCodePoint": "01B1"}
,{"SourceCharacterCodePoint": "0016","TargetCharacterCodePoint": "01B2"}
,{"SourceCharacterCodePoint": "0093","TargetCharacterCodePoint": "01B3"}
,{"SourceCharacterCodePoint": "0094","TargetCharacterCodePoint": "01B4"}
,{"SourceCharacterCodePoint": "0095","TargetCharacterCodePoint": "01B5"}
,{"SourceCharacterCodePoint": "0096","TargetCharacterCodePoint": "01B6"}
,{"SourceCharacterCodePoint": "0004","TargetCharacterCodePoint": "01B7"}
```

```
,{"SourceCharacterCodePoint": "0098","TargetCharacterCodePoint": "01B8"}
,{"SourceCharacterCodePoint": "0099","TargetCharacterCodePoint": "01B9"}
,{"SourceCharacterCodePoint": "009A","TargetCharacterCodePoint": "01BA"}
,{"SourceCharacterCodePoint": "009B","TargetCharacterCodePoint": "01BB"}
,{"SourceCharacterCodePoint": "0014","TargetCharacterCodePoint": "01BC"}
,{"SourceCharacterCodePoint": "0015","TargetCharacterCodePoint": "01BD"}
,{"SourceCharacterCodePoint": "009E","TargetCharacterCodePoint": "01BE"}
,{"SourceCharacterCodePoint": "001A","TargetCharacterCodePoint": "01BF"}
,{"SourceCharacterCodePoint": "009F","TargetCharacterCodePoint": "027F"}
```

fi_FI-278b 및 fi-FI-1143b

코드 시프트:

```
0180    0180
0001    0181
0002    0182
0003    0183
009C    0184
0009    0185
0086    0186
007F    0187
0097    0188
008D    0189
008E    018A
000B    018B
000C    018C
000D    018D
000E    018E
000F    018F
0010    0190
0011    0191
0012    0192
0013    0193
009D    0194
0085    0195
0008    0196
0087    0197
0018    0198
0019    0199
0092    019A
008F    019B
```

```
001C    019C
001D    019D
001E    019E
001F    019F
0080    01A0
0081    01A1
0082    01A2
0083    01A3
0084    01A4
000A    01A5
0017    01A6
001B    01A7
0088    01A8
0089    01A9
008A    01AA
008B    01AB
008C    01AC
0005    01AD
0006    01AE
0007    01AF
0090    01B0
0091    01B1
0016    01B2
0093    01B3
0094    01B4
0095    01B5
0096    01B6
0004    01B7
0098    01B8
0099    01B9
009A    01BA
009B    01BB
0014    01BC
0015    01BD
009E    01BE
001A    01BF
009F    027F
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0180","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "0001","TargetCharacterCodePoint": "0181"}
```

```
,{"SourceCharacterCodePoint": "0002","TargetCharacterCodePoint": "0182"}
,{"SourceCharacterCodePoint": "0003","TargetCharacterCodePoint": "0183"}
,{"SourceCharacterCodePoint": "009C","TargetCharacterCodePoint": "0184"}
,{"SourceCharacterCodePoint": "0009","TargetCharacterCodePoint": "0185"}
,{"SourceCharacterCodePoint": "0086","TargetCharacterCodePoint": "0186"}
,{"SourceCharacterCodePoint": "007F","TargetCharacterCodePoint": "0187"}
,{"SourceCharacterCodePoint": "0097","TargetCharacterCodePoint": "0188"}
,{"SourceCharacterCodePoint": "008D","TargetCharacterCodePoint": "0189"}
,{"SourceCharacterCodePoint": "008E","TargetCharacterCodePoint": "018A"}
,{"SourceCharacterCodePoint": "000B","TargetCharacterCodePoint": "018B"}
,{"SourceCharacterCodePoint": "000C","TargetCharacterCodePoint": "018C"}
,{"SourceCharacterCodePoint": "000D","TargetCharacterCodePoint": "018D"}
,{"SourceCharacterCodePoint": "000E","TargetCharacterCodePoint": "018E"}
,{"SourceCharacterCodePoint": "000F","TargetCharacterCodePoint": "018F"}
,{"SourceCharacterCodePoint": "0010","TargetCharacterCodePoint": "0190"}
,{"SourceCharacterCodePoint": "0011","TargetCharacterCodePoint": "0191"}
,{"SourceCharacterCodePoint": "0012","TargetCharacterCodePoint": "0192"}
,{"SourceCharacterCodePoint": "0013","TargetCharacterCodePoint": "0193"}
,{"SourceCharacterCodePoint": "009D","TargetCharacterCodePoint": "0194"}
,{"SourceCharacterCodePoint": "0085","TargetCharacterCodePoint": "0195"}
,{"SourceCharacterCodePoint": "0008","TargetCharacterCodePoint": "0196"}
,{"SourceCharacterCodePoint": "0087","TargetCharacterCodePoint": "0197"}
,{"SourceCharacterCodePoint": "0018","TargetCharacterCodePoint": "0198"}
,{"SourceCharacterCodePoint": "0019","TargetCharacterCodePoint": "0199"}
,{"SourceCharacterCodePoint": "0092","TargetCharacterCodePoint": "019A"}
,{"SourceCharacterCodePoint": "008F","TargetCharacterCodePoint": "019B"}
,{"SourceCharacterCodePoint": "001C","TargetCharacterCodePoint": "019C"}
,{"SourceCharacterCodePoint": "001D","TargetCharacterCodePoint": "019D"}
,{"SourceCharacterCodePoint": "001E","TargetCharacterCodePoint": "019E"}
,{"SourceCharacterCodePoint": "001F","TargetCharacterCodePoint": "019F"}
,{"SourceCharacterCodePoint": "0080","TargetCharacterCodePoint": "01A0"}
,{"SourceCharacterCodePoint": "0081","TargetCharacterCodePoint": "01A1"}
,{"SourceCharacterCodePoint": "0082","TargetCharacterCodePoint": "01A2"}
,{"SourceCharacterCodePoint": "0083","TargetCharacterCodePoint": "01A3"}
,{"SourceCharacterCodePoint": "0084","TargetCharacterCodePoint": "01A4"}
,{"SourceCharacterCodePoint": "000A","TargetCharacterCodePoint": "01A5"}
,{"SourceCharacterCodePoint": "0017","TargetCharacterCodePoint": "01A6"}
,{"SourceCharacterCodePoint": "001B","TargetCharacterCodePoint": "01A7"}
,{"SourceCharacterCodePoint": "0088","TargetCharacterCodePoint": "01A8"}
,{"SourceCharacterCodePoint": "0089","TargetCharacterCodePoint": "01A9"}
,{"SourceCharacterCodePoint": "008A","TargetCharacterCodePoint": "01AA"}
,{"SourceCharacterCodePoint": "008B","TargetCharacterCodePoint": "01AB"}
,{"SourceCharacterCodePoint": "008C","TargetCharacterCodePoint": "01AC"}
,{"SourceCharacterCodePoint": "0005","TargetCharacterCodePoint": "01AD"}
```

```
,{"SourceCharacterCodePoint": "0006","TargetCharacterCodePoint": "01AE"}
,{"SourceCharacterCodePoint": "0007","TargetCharacterCodePoint": "01AF"}
,{"SourceCharacterCodePoint": "0090","TargetCharacterCodePoint": "01B0"}
,{"SourceCharacterCodePoint": "0091","TargetCharacterCodePoint": "01B1"}
,{"SourceCharacterCodePoint": "0016","TargetCharacterCodePoint": "01B2"}
,{"SourceCharacterCodePoint": "0093","TargetCharacterCodePoint": "01B3"}
,{"SourceCharacterCodePoint": "0094","TargetCharacterCodePoint": "01B4"}
,{"SourceCharacterCodePoint": "0095","TargetCharacterCodePoint": "01B5"}
,{"SourceCharacterCodePoint": "0096","TargetCharacterCodePoint": "01B6"}
,{"SourceCharacterCodePoint": "0004","TargetCharacterCodePoint": "01B7"}
,{"SourceCharacterCodePoint": "0098","TargetCharacterCodePoint": "01B8"}
,{"SourceCharacterCodePoint": "0099","TargetCharacterCodePoint": "01B9"}
,{"SourceCharacterCodePoint": "009A","TargetCharacterCodePoint": "01BA"}
,{"SourceCharacterCodePoint": "009B","TargetCharacterCodePoint": "01BB"}
,{"SourceCharacterCodePoint": "0014","TargetCharacterCodePoint": "01BC"}
,{"SourceCharacterCodePoint": "0015","TargetCharacterCodePoint": "01BD"}
,{"SourceCharacterCodePoint": "009E","TargetCharacterCodePoint": "01BE"}
,{"SourceCharacterCodePoint": "001A","TargetCharacterCodePoint": "01BF"}
,{"SourceCharacterCodePoint": "009F","TargetCharacterCodePoint": "027F"}
```

fr-FR-297b 및 fr-FR-1147b

코드 시프트:

```
0180    0180
0001    0181
0002    0182
0003    0183
009C    0184
0009    0185
0086    0186
007F    0187
0097    0188
008D    0189
008E    018A
000B    018B
000C    018C
000D    018D
000E    018E
000F    018F
0010    0190
0011    0191
0012    0192
```

```
0013    0193
009D    0194
0085    0195
0008    0196
0087    0197
0018    0198
0019    0199
0092    019A
008F    019B
001C    019C
001D    019D
001E    019E
001F    019F
0080    01A0
0081    01A1
0082    01A2
0083    01A3
0084    01A4
000A    01A5
0017    01A6
001B    01A7
0088    01A8
0089    01A9
008A    01AA
008B    01AB
008C    01AC
0005    01AD
0006    01AE
0007    01AF
0090    01B0
0091    01B1
0016    01B2
0093    01B3
0094    01B4
0095    01B5
0096    01B6
0004    01B7
0098    01B8
0099    01B9
009A    01BA
009B    01BB
0014    01BC
0015    01BD
009E    01BE
```

```
001A    01BF
009F    027F
```

AWS DMS 작업에 해당하는 입력 매핑:

```
{"SourceCharacterCodePoint": "0180","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "0001","TargetCharacterCodePoint": "0181"}
,{"SourceCharacterCodePoint": "0002","TargetCharacterCodePoint": "0182"}
,{"SourceCharacterCodePoint": "0003","TargetCharacterCodePoint": "0183"}
,{"SourceCharacterCodePoint": "009C","TargetCharacterCodePoint": "0184"}
,{"SourceCharacterCodePoint": "0009","TargetCharacterCodePoint": "0185"}
,{"SourceCharacterCodePoint": "0086","TargetCharacterCodePoint": "0186"}
,{"SourceCharacterCodePoint": "007F","TargetCharacterCodePoint": "0187"}
,{"SourceCharacterCodePoint": "0097","TargetCharacterCodePoint": "0188"}
,{"SourceCharacterCodePoint": "008D","TargetCharacterCodePoint": "0189"}
,{"SourceCharacterCodePoint": "008E","TargetCharacterCodePoint": "018A"}
,{"SourceCharacterCodePoint": "000B","TargetCharacterCodePoint": "018B"}
,{"SourceCharacterCodePoint": "000C","TargetCharacterCodePoint": "018C"}
,{"SourceCharacterCodePoint": "000D","TargetCharacterCodePoint": "018D"}
,{"SourceCharacterCodePoint": "000E","TargetCharacterCodePoint": "018E"}
,{"SourceCharacterCodePoint": "000F","TargetCharacterCodePoint": "018F"}
,{"SourceCharacterCodePoint": "0010","TargetCharacterCodePoint": "0190"}
,{"SourceCharacterCodePoint": "0011","TargetCharacterCodePoint": "0191"}
,{"SourceCharacterCodePoint": "0012","TargetCharacterCodePoint": "0192"}
,{"SourceCharacterCodePoint": "0013","TargetCharacterCodePoint": "0193"}
,{"SourceCharacterCodePoint": "009D","TargetCharacterCodePoint": "0194"}
,{"SourceCharacterCodePoint": "0085","TargetCharacterCodePoint": "0195"}
,{"SourceCharacterCodePoint": "0008","TargetCharacterCodePoint": "0196"}
,{"SourceCharacterCodePoint": "0087","TargetCharacterCodePoint": "0197"}
,{"SourceCharacterCodePoint": "0018","TargetCharacterCodePoint": "0198"}
,{"SourceCharacterCodePoint": "0019","TargetCharacterCodePoint": "0199"}
,{"SourceCharacterCodePoint": "0092","TargetCharacterCodePoint": "019A"}
,{"SourceCharacterCodePoint": "008F","TargetCharacterCodePoint": "019B"}
,{"SourceCharacterCodePoint": "001C","TargetCharacterCodePoint": "019C"}
,{"SourceCharacterCodePoint": "001D","TargetCharacterCodePoint": "019D"}
,{"SourceCharacterCodePoint": "001E","TargetCharacterCodePoint": "019E"}
,{"SourceCharacterCodePoint": "001F","TargetCharacterCodePoint": "019F"}
,{"SourceCharacterCodePoint": "0080","TargetCharacterCodePoint": "01A0"}
,{"SourceCharacterCodePoint": "0081","TargetCharacterCodePoint": "01A1"}
,{"SourceCharacterCodePoint": "0082","TargetCharacterCodePoint": "01A2"}
,{"SourceCharacterCodePoint": "0083","TargetCharacterCodePoint": "01A3"}
,{"SourceCharacterCodePoint": "0084","TargetCharacterCodePoint": "01A4"}
```

```
,{"SourceCharacterCodePoint": "000A","TargetCharacterCodePoint": "01A5"}
,{"SourceCharacterCodePoint": "0017","TargetCharacterCodePoint": "01A6"}
,{"SourceCharacterCodePoint": "001B","TargetCharacterCodePoint": "01A7"}
,{"SourceCharacterCodePoint": "0088","TargetCharacterCodePoint": "01A8"}
,{"SourceCharacterCodePoint": "0089","TargetCharacterCodePoint": "01A9"}
,{"SourceCharacterCodePoint": "008A","TargetCharacterCodePoint": "01AA"}
,{"SourceCharacterCodePoint": "008B","TargetCharacterCodePoint": "01AB"}
,{"SourceCharacterCodePoint": "008C","TargetCharacterCodePoint": "01AC"}
,{"SourceCharacterCodePoint": "0005","TargetCharacterCodePoint": "01AD"}
,{"SourceCharacterCodePoint": "0006","TargetCharacterCodePoint": "01AE"}
,{"SourceCharacterCodePoint": "0007","TargetCharacterCodePoint": "01AF"}
,{"SourceCharacterCodePoint": "0090","TargetCharacterCodePoint": "01B0"}
,{"SourceCharacterCodePoint": "0091","TargetCharacterCodePoint": "01B1"}
,{"SourceCharacterCodePoint": "0016","TargetCharacterCodePoint": "01B2"}
,{"SourceCharacterCodePoint": "0093","TargetCharacterCodePoint": "01B3"}
,{"SourceCharacterCodePoint": "0094","TargetCharacterCodePoint": "01B4"}
,{"SourceCharacterCodePoint": "0095","TargetCharacterCodePoint": "01B5"}
,{"SourceCharacterCodePoint": "0096","TargetCharacterCodePoint": "01B6"}
,{"SourceCharacterCodePoint": "0004","TargetCharacterCodePoint": "01B7"}
,{"SourceCharacterCodePoint": "0098","TargetCharacterCodePoint": "01B8"}
,{"SourceCharacterCodePoint": "0099","TargetCharacterCodePoint": "01B9"}
,{"SourceCharacterCodePoint": "009A","TargetCharacterCodePoint": "01BA"}
,{"SourceCharacterCodePoint": "009B","TargetCharacterCodePoint": "01BB"}
,{"SourceCharacterCodePoint": "0014","TargetCharacterCodePoint": "01BC"}
,{"SourceCharacterCodePoint": "0015","TargetCharacterCodePoint": "01BD"}
,{"SourceCharacterCodePoint": "009E","TargetCharacterCodePoint": "01BE"}
,{"SourceCharacterCodePoint": "001A","TargetCharacterCodePoint": "01BF"}
,{"SourceCharacterCodePoint": "009F","TargetCharacterCodePoint": "027F"}
```

it-IT-280b 및 it-IT-1144b

코드 시프트:

```
0180    0180
0001    0181
0002    0182
0003    0183
009C    0184
0009    0185
0086    0186
007F    0187
0097    0188
```

```
008D    0189
008E    018A
000B    018B
000C    018C
000D    018D
000E    018E
000F    018F
0010    0190
0011    0191
0012    0192
0013    0193
009D    0194
0085    0195
0008    0196
0087    0197
0018    0198
0019    0199
0092    019A
008F    019B
001C    019C
001D    019D
001E    019E
001F    019F
0080    01A0
0081    01A1
0082    01A2
0083    01A3
0084    01A4
000A    01A5
0017    01A6
001B    01A7
0088    01A8
0089    01A9
008A    01AA
008B    01AB
008C    01AC
0005    01AD
0006    01AE
0007    01AF
0090    01B0
0091    01B1
0016    01B2
0093    01B3
0094    01B4
```

```
0095    01B5
0096    01B6
0004    01B7
0098    01B8
0099    01B9
009A    01BA
009B    01BB
0014    01BC
0015    01BD
009E    01BE
001A    01BF
009F    027F
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0180","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "0001","TargetCharacterCodePoint": "0181"}
,{"SourceCharacterCodePoint": "0002","TargetCharacterCodePoint": "0182"}
,{"SourceCharacterCodePoint": "0003","TargetCharacterCodePoint": "0183"}
,{"SourceCharacterCodePoint": "009C","TargetCharacterCodePoint": "0184"}
,{"SourceCharacterCodePoint": "0009","TargetCharacterCodePoint": "0185"}
,{"SourceCharacterCodePoint": "0086","TargetCharacterCodePoint": "0186"}
,{"SourceCharacterCodePoint": "007F","TargetCharacterCodePoint": "0187"}
,{"SourceCharacterCodePoint": "0097","TargetCharacterCodePoint": "0188"}
,{"SourceCharacterCodePoint": "008D","TargetCharacterCodePoint": "0189"}
,{"SourceCharacterCodePoint": "008E","TargetCharacterCodePoint": "018A"}
,{"SourceCharacterCodePoint": "000B","TargetCharacterCodePoint": "018B"}
,{"SourceCharacterCodePoint": "000C","TargetCharacterCodePoint": "018C"}
,{"SourceCharacterCodePoint": "000D","TargetCharacterCodePoint": "018D"}
,{"SourceCharacterCodePoint": "000E","TargetCharacterCodePoint": "018E"}
,{"SourceCharacterCodePoint": "000F","TargetCharacterCodePoint": "018F"}
,{"SourceCharacterCodePoint": "0010","TargetCharacterCodePoint": "0190"}
,{"SourceCharacterCodePoint": "0011","TargetCharacterCodePoint": "0191"}
,{"SourceCharacterCodePoint": "0012","TargetCharacterCodePoint": "0192"}
,{"SourceCharacterCodePoint": "0013","TargetCharacterCodePoint": "0193"}
,{"SourceCharacterCodePoint": "009D","TargetCharacterCodePoint": "0194"}
,{"SourceCharacterCodePoint": "0085","TargetCharacterCodePoint": "0195"}
,{"SourceCharacterCodePoint": "0008","TargetCharacterCodePoint": "0196"}
,{"SourceCharacterCodePoint": "0087","TargetCharacterCodePoint": "0197"}
,{"SourceCharacterCodePoint": "0018","TargetCharacterCodePoint": "0198"}
,{"SourceCharacterCodePoint": "0019","TargetCharacterCodePoint": "0199"}
,{"SourceCharacterCodePoint": "0092","TargetCharacterCodePoint": "019A"}
```

```
,{"SourceCharacterCodePoint": "008F","TargetCharacterCodePoint": "019B"}
,{"SourceCharacterCodePoint": "001C","TargetCharacterCodePoint": "019C"}
,{"SourceCharacterCodePoint": "001D","TargetCharacterCodePoint": "019D"}
,{"SourceCharacterCodePoint": "001E","TargetCharacterCodePoint": "019E"}
,{"SourceCharacterCodePoint": "001F","TargetCharacterCodePoint": "019F"}
,{"SourceCharacterCodePoint": "0080","TargetCharacterCodePoint": "01A0"}
,{"SourceCharacterCodePoint": "0081","TargetCharacterCodePoint": "01A1"}
,{"SourceCharacterCodePoint": "0082","TargetCharacterCodePoint": "01A2"}
,{"SourceCharacterCodePoint": "0083","TargetCharacterCodePoint": "01A3"}
,{"SourceCharacterCodePoint": "0084","TargetCharacterCodePoint": "01A4"}
,{"SourceCharacterCodePoint": "000A","TargetCharacterCodePoint": "01A5"}
,{"SourceCharacterCodePoint": "0017","TargetCharacterCodePoint": "01A6"}
,{"SourceCharacterCodePoint": "001B","TargetCharacterCodePoint": "01A7"}
,{"SourceCharacterCodePoint": "0088","TargetCharacterCodePoint": "01A8"}
,{"SourceCharacterCodePoint": "0089","TargetCharacterCodePoint": "01A9"}
,{"SourceCharacterCodePoint": "008A","TargetCharacterCodePoint": "01AA"}
,{"SourceCharacterCodePoint": "008B","TargetCharacterCodePoint": "01AB"}
,{"SourceCharacterCodePoint": "008C","TargetCharacterCodePoint": "01AC"}
,{"SourceCharacterCodePoint": "0005","TargetCharacterCodePoint": "01AD"}
,{"SourceCharacterCodePoint": "0006","TargetCharacterCodePoint": "01AE"}
,{"SourceCharacterCodePoint": "0007","TargetCharacterCodePoint": "01AF"}
,{"SourceCharacterCodePoint": "0090","TargetCharacterCodePoint": "01B0"}
,{"SourceCharacterCodePoint": "0091","TargetCharacterCodePoint": "01B1"}
,{"SourceCharacterCodePoint": "0016","TargetCharacterCodePoint": "01B2"}
,{"SourceCharacterCodePoint": "0093","TargetCharacterCodePoint": "01B3"}
,{"SourceCharacterCodePoint": "0094","TargetCharacterCodePoint": "01B4"}
,{"SourceCharacterCodePoint": "0095","TargetCharacterCodePoint": "01B5"}
,{"SourceCharacterCodePoint": "0096","TargetCharacterCodePoint": "01B6"}
,{"SourceCharacterCodePoint": "0004","TargetCharacterCodePoint": "01B7"}
,{"SourceCharacterCodePoint": "0098","TargetCharacterCodePoint": "01B8"}
,{"SourceCharacterCodePoint": "0099","TargetCharacterCodePoint": "01B9"}
,{"SourceCharacterCodePoint": "009A","TargetCharacterCodePoint": "01BA"}
,{"SourceCharacterCodePoint": "009B","TargetCharacterCodePoint": "01BB"}
,{"SourceCharacterCodePoint": "0014","TargetCharacterCodePoint": "01BC"}
,{"SourceCharacterCodePoint": "0015","TargetCharacterCodePoint": "01BD"}
,{"SourceCharacterCodePoint": "009E","TargetCharacterCodePoint": "01BE"}
,{"SourceCharacterCodePoint": "001A","TargetCharacterCodePoint": "01BF"}
,{"SourceCharacterCodePoint": "009F","TargetCharacterCodePoint": "027F"}
```

nl-BE-500b 및 nl-BE-1148b

코드 시프트:

```
0180    0180
0001    0181
0002    0182
0003    0183
009C    0184
0009    0185
0086    0186
007F    0187
0097    0188
008D    0189
008E    018A
000B    018B
000C    018C
000D    018D
000E    018E
000F    018F
0010    0190
0011    0191
0012    0192
0013    0193
009D    0194
0085    0195
0008    0196
0087    0197
0018    0198
0019    0199
0092    019A
008F    019B
001C    019C
001D    019D
001E    019E
001F    019F
0080    01A0
0081    01A1
0082    01A2
0083    01A3
0084    01A4
000A    01A5
0017    01A6
001B    01A7
0088    01A8
0089    01A9
008A    01AA
```

```
008B    01AB
008C    01AC
0005    01AD
0006    01AE
0007    01AF
0090    01B0
0091    01B1
0016    01B2
0093    01B3
0094    01B4
0095    01B5
0096    01B6
0004    01B7
0098    01B8
0099    01B9
009A    01BA
009B    01BB
0014    01BC
0015    01BD
009E    01BE
001A    01BF
009F    027F
```

AWS DMS 작업에 해당하는 입력 매핑:

```
 {"SourceCharacterCodePoint": "0180","TargetCharacterCodePoint": "0180"}
,{"SourceCharacterCodePoint": "0001","TargetCharacterCodePoint": "0181"}
,{"SourceCharacterCodePoint": "0002","TargetCharacterCodePoint": "0182"}
,{"SourceCharacterCodePoint": "0003","TargetCharacterCodePoint": "0183"}
,{"SourceCharacterCodePoint": "009C","TargetCharacterCodePoint": "0184"}
,{"SourceCharacterCodePoint": "0009","TargetCharacterCodePoint": "0185"}
,{"SourceCharacterCodePoint": "0086","TargetCharacterCodePoint": "0186"}
,{"SourceCharacterCodePoint": "007F","TargetCharacterCodePoint": "0187"}
,{"SourceCharacterCodePoint": "0097","TargetCharacterCodePoint": "0188"}
,{"SourceCharacterCodePoint": "008D","TargetCharacterCodePoint": "0189"}
,{"SourceCharacterCodePoint": "008E","TargetCharacterCodePoint": "018A"}
,{"SourceCharacterCodePoint": "000B","TargetCharacterCodePoint": "018B"}
,{"SourceCharacterCodePoint": "000C","TargetCharacterCodePoint": "018C"}
,{"SourceCharacterCodePoint": "000D","TargetCharacterCodePoint": "018D"}
,{"SourceCharacterCodePoint": "000E","TargetCharacterCodePoint": "018E"}
,{"SourceCharacterCodePoint": "000F","TargetCharacterCodePoint": "018F"}
,{"SourceCharacterCodePoint": "0010","TargetCharacterCodePoint": "0190"}
```

```
,{"SourceCharacterCodePoint": "0011","TargetCharacterCodePoint": "0191"}
,{"SourceCharacterCodePoint": "0012","TargetCharacterCodePoint": "0192"}
,{"SourceCharacterCodePoint": "0013","TargetCharacterCodePoint": "0193"}
,{"SourceCharacterCodePoint": "009D","TargetCharacterCodePoint": "0194"}
,{"SourceCharacterCodePoint": "0085","TargetCharacterCodePoint": "0195"}
,{"SourceCharacterCodePoint": "0008","TargetCharacterCodePoint": "0196"}
,{"SourceCharacterCodePoint": "0087","TargetCharacterCodePoint": "0197"}
,{"SourceCharacterCodePoint": "0018","TargetCharacterCodePoint": "0198"}
,{"SourceCharacterCodePoint": "0019","TargetCharacterCodePoint": "0199"}
,{"SourceCharacterCodePoint": "0092","TargetCharacterCodePoint": "019A"}
,{"SourceCharacterCodePoint": "008F","TargetCharacterCodePoint": "019B"}
,{"SourceCharacterCodePoint": "001C","TargetCharacterCodePoint": "019C"}
,{"SourceCharacterCodePoint": "001D","TargetCharacterCodePoint": "019D"}
,{"SourceCharacterCodePoint": "001E","TargetCharacterCodePoint": "019E"}
,{"SourceCharacterCodePoint": "001F","TargetCharacterCodePoint": "019F"}
,{"SourceCharacterCodePoint": "0080","TargetCharacterCodePoint": "01A0"}
,{"SourceCharacterCodePoint": "0081","TargetCharacterCodePoint": "01A1"}
,{"SourceCharacterCodePoint": "0082","TargetCharacterCodePoint": "01A2"}
,{"SourceCharacterCodePoint": "0083","TargetCharacterCodePoint": "01A3"}
,{"SourceCharacterCodePoint": "0084","TargetCharacterCodePoint": "01A4"}
,{"SourceCharacterCodePoint": "000A","TargetCharacterCodePoint": "01A5"}
,{"SourceCharacterCodePoint": "0017","TargetCharacterCodePoint": "01A6"}
,{"SourceCharacterCodePoint": "001B","TargetCharacterCodePoint": "01A7"}
,{"SourceCharacterCodePoint": "0088","TargetCharacterCodePoint": "01A8"}
,{"SourceCharacterCodePoint": "0089","TargetCharacterCodePoint": "01A9"}
,{"SourceCharacterCodePoint": "008A","TargetCharacterCodePoint": "01AA"}
,{"SourceCharacterCodePoint": "008B","TargetCharacterCodePoint": "01AB"}
,{"SourceCharacterCodePoint": "008C","TargetCharacterCodePoint": "01AC"}
,{"SourceCharacterCodePoint": "0005","TargetCharacterCodePoint": "01AD"}
,{"SourceCharacterCodePoint": "0006","TargetCharacterCodePoint": "01AE"}
,{"SourceCharacterCodePoint": "0007","TargetCharacterCodePoint": "01AF"}
,{"SourceCharacterCodePoint": "0090","TargetCharacterCodePoint": "01B0"}
,{"SourceCharacterCodePoint": "0091","TargetCharacterCodePoint": "01B1"}
,{"SourceCharacterCodePoint": "0016","TargetCharacterCodePoint": "01B2"}
,{"SourceCharacterCodePoint": "0093","TargetCharacterCodePoint": "01B3"}
,{"SourceCharacterCodePoint": "0094","TargetCharacterCodePoint": "01B4"}
,{"SourceCharacterCodePoint": "0095","TargetCharacterCodePoint": "01B5"}
,{"SourceCharacterCodePoint": "0096","TargetCharacterCodePoint": "01B6"}
,{"SourceCharacterCodePoint": "0004","TargetCharacterCodePoint": "01B7"}
,{"SourceCharacterCodePoint": "0098","TargetCharacterCodePoint": "01B8"}
,{"SourceCharacterCodePoint": "0099","TargetCharacterCodePoint": "01B9"}
,{"SourceCharacterCodePoint": "009A","TargetCharacterCodePoint": "01BA"}
,{"SourceCharacterCodePoint": "009B","TargetCharacterCodePoint": "01BB"}
,{"SourceCharacterCodePoint": "0014","TargetCharacterCodePoint": "01BC"}
```

```
,{"SourceCharacterCodePoint": "0015","TargetCharacterCodePoint": "01BD"}
,{"SourceCharacterCodePoint": "009E","TargetCharacterCodePoint": "01BE"}
,{"SourceCharacterCodePoint": "001A","TargetCharacterCodePoint": "01BF"}
,{"SourceCharacterCodePoint": "009F","TargetCharacterCodePoint": "027F"}
```

마이그레이션에 적합한 대상

AWS Database Migration Service(AWS DMS)는 데이터 복제에서 대상으로서 가장 인기 있는 데이터베이스를 다수 사용할 수 있습니다. 대상은 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, Amazon Relational Database Service(RDS) 인스턴스 또는 온프레미스 데이터베이스에 있을 수 있습니다.

유효한 대상의 전체 목록은 AWS DMS 대상 섹션을 참조하세요.

> **Note**
>
> AWS DMS는 다음 대상 엔드포인트 유형에 대한 AWS 리전 간 마이그레이션을 지원하지 않습니다.
>
> - Amazon DynamoDB
> - Amazon OpenSearch Service
> - Amazon Kinesis Data Streams

주제

- AWS Database Migration Service에서 Oracle 데이터베이스를 대상으로 사용
- Microsoft SQL Server 데이터베이스를 AWS Database Migration Service의 대상으로 사용
- AWS Database Migration Service에서 PostgreSQL 데이터베이스를 대상으로 사용
- AWS Database Migration Service에서 MySQL 호환 데이터베이스를 대상으로 사용
- AWS Database Migration Service의 대상으로 Amazon Redshift 데이터베이스 사용
- SAP ASE 데이터베이스를 AWS Database Migration Service의 대상으로 사용
- Amazon S3를 AWS Database Migration Service의 대상으로 사용
- AWS Database Migration Service 대상으로 Amazon DynamoDB 데이터베이스 사용
- Amazon Kinesis Data Streams를 대상으로 사용 AWS Database Migration Service

- 아파치 카프카를 타겟으로 사용 AWS Database Migration Service
- Amazon OpenSearch Service 클러스터를 AWS Database Migration Service의 대상으로 사용
- Amazon DocumentDB를 AWS Database Migration Service의 대상으로 사용
- Amazon Neptune을 AWS Database Migration Service의 대상으로 사용
- Redis를 AWS Database Migration Service의 대상으로 사용
- Babelfish를 AWS Database Migration Service 대상으로 사용
- Amazon Timestream을 AWS Database Migration Service의 대상으로 사용
- Db2용 Amazon RDS와 IBM Db2 LUW를 AWS DMS의 대상으로 사용

AWS Database Migration Service에서 Oracle 데이터베이스를 대상으로 사용

AWS DMS를 사용하여 다른 Oracle 데이터베이스에서나 지원되는 다른 데이터베이스 중 하나에서 Oracle 데이터베이스로 데이터를 마이그레이션할 수 있습니다. Secure Sockets Layer(SSL)을 사용하여 Oracle 엔드포인트와 복제 인스턴스 연결을 암호화할 수 있습니다. Oracle 엔드포인트에서 SSL을 사용하는 방법에 관한 자세한 내용은 AWS Database Migration Service에서 SSL 사용 단원을 참조하십시오. AWS DMS는 Oracle TDE(Transparent Data Encryption)를 사용하여 대상 데이터베이스에서 저장 데이터를 암호화할 수도 있습니다. Oracle TDE는 데이터베이스에 기록하는 데 암호화 키나 암호를 요구하지 않습니다.

AWS DMS가 대상으로 지원하는 Oracle 버전에 관한 자세한 내용은 대상: AWS DMS를 참조하십시오.

Oracle을 대상으로 사용할 때는 데이터를 대상 연결에 사용되는 스키마 또는 사용자로 마이그레이션해야 한다고 가정합니다. 데이터를 다른 스키마로 마이그레이션하려면 스키마 변환을 사용해야 합니다. 예를 들어, 대상 엔드포인트가 사용자 `RDSMASTER`에 연결되고 사용자 `PERFDATA1`에서 `PERFDATA2`로 마이그레이션하려 한다고 가정합니다. 이 경우, 다음과 같이 변환을 생성합니다.

```
{
   "rule-type": "transformation",
   "rule-id": "2",
   "rule-name": "2",
   "rule-action": "rename",
   "rule-target": "schema",
   "object-locator": {
   "schema-name": "PERFDATA1"
},
```

```
"value": "PERFDATA2"
}
```

Oracle을 대상으로 사용하는 경우, AWS DMS는 모든 테이블 및 인덱스를 대상의 기본 테이블 및 인덱스 테이블스페이스로 마이그레이션합니다. 테이블과 인덱스를 다른 테이블 및 인덱스 테이블스페이스로 마이그레이션하려면 테이블스페이스 변환을 사용해야 합니다. 예를 들어, Oracle 소스의 일부 테이블스페이스에 할당된 INVENTORY 스키마에 테이블 집합이 있다고 가정해 보겠습니다. 마이그레이션을 위해 이러한 모든 테이블을 대상의 단일 INVENTORYSPACE 테이블스페이스에 할당하려고 합니다. 이 경우, 다음과 같이 변환을 생성합니다.

```
{
   "rule-type": "transformation",
   "rule-id": "3",
   "rule-name": "3",
   "rule-action": "rename",
   "rule-target": "table-tablespace",
   "object-locator": {
      "schema-name": "INVENTORY",
      "table-name": "%",
      "table-tablespace-name": "%"
   },
   "value": "INVENTORYSPACE"
}
```

변환에 관한 자세한 내용은 JSON을 사용하여 테이블 선택 및 변환 지정 섹션을 참조하십시오.

Oracle이 소스와 대상인 경우, Oracle 소스 추가 접속 속성인 enableHomogenousTablespace=true를 설정하여 기존 테이블 또는 인덱스 테이블스페이스 할당을 유지할 수 있습니다. 자세한 정보는 Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS 섹션을 참조하세요.

Oracle 데이터베이스를 AWS DMS에 대한 대상으로 사용하여 작업하는 방법에 관한 자세한 내용은 다음 단원을 참조하십시오.

주제

- AWS Database Migration Service 대상으로서 Oracle에 적용되는 제한 사항
- Oracle을 대상으로 사용할 때 필요한 사용자 계정 권한
- AWS Database Migration Service에서 Oracle 데이터베이스를 대상으로 구성
- Oracle을 AWS DMS의 대상으로 사용할 경우 엔드포인트 설정

- Oracle용 대상 데이터 형식

AWS Database Migration Service 대상으로서 Oracle에 적용되는 제한 사항

데이터 마이그레이션에서 Oracle 데이터베이스를 대상으로 사용할 때 적용되는 제한 사항에는 다음이 있습니다.

- AWS DMS는 대상 Oracle 데이터베이스에서 스키마를 생성하지 않습니다. 대상 Oracle 데이터베이스에서 필요한 스키마를 생성해야 합니다. Oracle 대상에서 스키마 이름이 이미 있어야 합니다. 원본 스키마의 테이블을 대상 인스턴스에 연결하기 위해 AWS DMS가 사용하는 사용자 또는 스키마로 가져옵니다. 여러 스키마를 마이그레이션하려는 경우, 다중 복제 작업을 생성하면 됩니다. 데이터를 대상의 다른 스키마로 마이그레이션할 수도 있습니다. 이렇게 하려면 AWS DMS 테이블 매핑에서 스키마 변환 규칙을 사용해야 합니다.
- AWS DMS는 INDEXTYPE CONTEXT를 사용한 테이블에서 `Use direct path full load` 옵션을 지원하지 않습니다. 차선책으로서 어레이 로드를 사용할 수 있습니다.
- 배치 작업에 최적화된 적용 옵션을 사용하면 넷 변경 테이블에 로드할 때 XML 유형을 지원하지 않는 직접 경로가 사용됩니다. 차선책으로서 트랜잭션 적용 모드를 사용할 수 있습니다.
- 소스 데이터베이스에서 마이그레이션된 빈 문자열은 Oracle 대상에서 다르게 처리될 수 있습니다(예: 하나의 공백 문자열로 변환). 이로 인해 AWS DMS 검증에서 불일치가 보고될 수 있습니다.
- 다음 공식을 사용하여 Batch 최적화 적용 모드에서 지원되는 테이블당 총 열 수를 표현할 수 있습니다.

  ```
  2 * columns_in_original_table + columns_in_primary_key <= 999
  ```

  예를 들어 원본 테이블에 25개의 열이 있고 기본 키가 5개의 열로 구성된 경우, 총 열 수는 55개입니다. 테이블이 지원되는 열 수를 초과하는 경우, 모든 변경 내용은 1대1 모드로 적용됩니다.
- AWS DMS는 Oracle Cloud Infrastructure(OCI)의 자율 DB를 지원하지 않습니다.

Oracle을 대상으로 사용할 때 필요한 사용자 계정 권한

AWS Database Migration Service 작업에서 Oracle 대상을 사용하려면 Oracle 데이터베이스에서 다음 권한을 부여합니다. 이러한 권한은 AWS DMS의 Oracle 데이터베이스 정의에서 지정된 사용자 계정에 부여합니다.

- SELECT ANY TRANSACTION
- SELECT on V$NLS_PARAMETERS

- SELECT on V$TIMEZONE_NAMES
- SELECT on ALL_INDEXES
- SELECT on ALL_OBJECTS
- SELECT on DBA_OBJECTS
- SELECT on ALL_TABLES
- SELECT on ALL_USERS
- SELECT on ALL_CATALOG
- SELECT on ALL_CONSTRAINTS
- SELECT on ALL_CONS_COLUMNS
- SELECT on ALL_TAB_COLS
- SELECT on ALL_IND_COLUMNS
- DROP ANY TABLE
- SELECT ANY TABLE
- INSERT ANY TABLE
- UPDATE ANY TABLE
- CREATE ANY VIEW
- DROP ANY VIEW
- CREATE ANY PROCEDURE
- ALTER ANY PROCEDURE
- DROP ANY PROCEDURE
- CREATE ANY SEQUENCE
- ALTER ANY SEQUENCE
- DROP ANY SEQUENCE
- DELETE ANY TABLE

다음과 같은 요구 사항에서 이러한 추가 권한을 부여합니다.

- 특정 테이블 목록을 사용하려면 SELECT를 복제된 테이블에 부여하고 ALTER도 복제된 테이블에 부여합니다.
- 사용자가 기본 테이블스페이스에서 테이블을 생성하도록 허용하려면 권한 GRANT UNLIMITED TABLESPACE를 부여합니다.

- 로그온의 경우, 권한 CREATE SESSION을 부여합니다.
- 직접 경로(전체 로드의 기본값)를 사용하는 경우 `GRANT LOCK ANY TABLE to dms_user;`을 실행합니다.
- “DROP and CREATE” 테이블 준비 모드를 사용할 때 스키마가 다른 경우, `GRANT CREATE ANY INDEX to dms_user;`를 실행합니다.
- 일부 전체 로드 시나리오의 경우, 대상 테이블 스키마가 DMS 사용자의 스키마와 다른 “테이블 DROP 및 CREATE” 또는 “로딩 전 TRUNCATE” 옵션을 선택할 수 있습니다. 이 경우, DROP ANY TABLE을 부여합니다.
- 대상 테이블 스키마가 DMS 사용자의 테이블 스키마와 다를 때 변경 테이블이나 감사 테이블에 변경 사항을 저장하려면 CREATE ANY TABLE 및 CREATE ANY INDEX를 부여합니다.

대상 데이터베이스에서 AWS Database Migration Service에 필요한 읽기 권한

다음 DBA 테이블의 읽기 권한을 AWS DMS 사용자 계정에 부여해야 합니다.

- SELECT on DBA_USERS
- SELECT on DBA_TAB_PRIVS
- SELECT on DBA_OBJECTS
- SELECT on DBA_SYNONYMS
- SELECT on DBA_SEQUENCES
- SELECT on DBA_TYPES
- SELECT on DBA_INDEXES
- SELECT on DBA_TABLES
- SELECT on DBA_TRIGGERS
- SELECT on SYS.DBA_REGISTRY

필요한 권한을 V$xxx에 부여할 수 없는 경우, V_$xxx에 부여합니다.

AWS Database Migration Service에서 Oracle 데이터베이스를 대상으로 구성

Oracle 데이터베이스를 마이그레이션 대상으로 사용하기 전에 Oracle 사용자 계정을 AWS DMS에 제공해야 합니다. 사용자 계정에는 Oracle을 대상으로 사용할 때 필요한 사용자 계정 권한에 지정된 바와 같이 Oracle 데이터베이스에서 읽기/쓰기 권한이 있어야 합니다.

Oracle을 AWS DMS의 대상으로 사용할 경우 엔드포인트 설정

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Oracle 대상을 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI의 `create-endpoint` 명령을 `--oracle-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Oracle을 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

이름	설명
`EscapeCharacter`	이 속성을 이스케이프 문자로 설정합니다. 이 이스케이프 문자를 사용하면 단일 와일드카드 문자가 테이블 매핑 표현식에서 일반 문자처럼 동작하게 할 수 있습니다. 자세한 내용은 테이블 매핑의 와일드카드 섹션을 참조하세요. 기본값: Null 유효값: 와일드카드 문자를 제외한 모든 문자 예: `--oracle-settings '{"EscapeCharacter": "#"}'`
`UseDirectPathFullLoad`	Y로 설정하면 AWS DMS는 직접 경로 전체 로드를 사용합니다. Oracle 호출 인터페이스(OCI)에서 직접 경로 프로토콜을 활성화하려면 이 값을 지정합니다. 이 OCI 프로토콜을 사용하면 전체 로드 중에 Oracle 대상 테이블을 대량 로드할 수 있습니다. 기본값: `true` 유효값: `true/false` 예: `--oracle-settings '{"UseDirectPathFullLoad": false}'`
`DirectPathParallelLoad`	`true`로 설정하면 `UseDirectPathFullLoad` 가 Y로 설정된 경우 이 속성이 병렬 로드를 지정합니다. 이 속성은 AWS DMS 병렬 로드 기능을 사용할 때만 적용됩니

이름	설명
	다. 자세한 내용은 테이블 및 컬렉션 설정 규칙과 작업의 `parallel-load` 작업에 관한 설명을 참조하십시오. 이 병렬 로드 설정을 지정할 때 제한 사항은 대상 테이블에 제약 조건이나 인덱스를 가질 수 없다는 것입니다. 이 제한 사항에 관한 자세한 내용은 병렬 직접 경로 로드 후 제약 조건 활성화를 참조하십시오. 제약 조건 또는 인덱스가 활성화된 경우, 이 속성을 `true`로 설정해도 아무 효과가 없습니다. 기본값: `false` 유효값: `true/false` 예: `--oracle-settings '{"DirectPathParallelLoad": true}'`
`DirectPathNoLog`	이 속성을 `true`로 설정하면 테이블에 직접 쓰고 데이터베이스 로그에 추적을 쓰지 않고 Oracle 대상 데이터베이스의 커밋 비율을 높일 수 있습니다. 자세한 내용은 Direct-Load INSERT를 참조하십시오. 이 속성은 `UseDirectPathFullLoad` 를 `Y`로 설정한 경우에만 적용됩니다. 기본값: `false` 유효값: `true/false` 예: `--oracle-settings '{"DirectPathNoLog": true}'`

이름	설명
CharLengthSemantics	문자 열의 길이가 바이트 단위인지 또는 문자 단위인지 지정합니다. 문자 열 길이가 문자 단위임을 나타내려면 이 속성을 CHAR로 설정합니다. 그렇지 않으면 문자 열 길이는 바이트 단위입니다. 기본값: CHAR로 설정되지 않음 유효값: CHAR 예: --oracle-settings '{"CharLengthSemantics": "CHAR"}'
AlwaysReplaceEmptyString	AWS DMS는 Oracle 대상으로 마이그레이션할 때 빈 문자열을 복제하기 위한 추가 공간을 추가합니다. 일반적으로 Oracle에는 빈 문자열에 대한 표기법이 없습니다. varchar2에 빈 문자열을 삽입하면 빈 문자열이 NULL로 로드됩니다. Oracle에서 데이터를 NULL로 삽입하려면 이 속성을 FALSE로 설정하십시오. 기본값: true 유효값: true/false 예: --oracle-settings '{"AlwaysReplaceEmptyString": false}'

Oracle용 대상 데이터 형식

AWS DMS에서 사용되는 대상 Oracle 데이터베이스는 대다수 Oracle 데이터 형식을 지원합니다. 다음 테이블에는 AWS DMS 사용 시 그리고 AWS DMS 데이터 형식의 기본 매핑을 사용할 때 지원하는 Oracle 대상 데이터 형식이 나와 있습니다. 소스에서 매핑된 데이터 형식을 확인하는 방법에 관한 자세한 내용은 사용 중인 소스에 대한 섹션을 참조하십시오.

AWS DMS 데이터 형식	Oracle 데이터 형식
BOOLEAN	NUMBER (1)

AWS DMS 데이터 형식	Oracle 데이터 형식
BYTES	RAW(길이)
DATE	DATETIME
TIME	TIMESTAMP (0)
DATETIME	TIMESTAMP(크기)
INT1	NUMBER (3)
INT2	NUMBER (5)
INT4	NUMBER (10)
INT8	NUMBER (19)
NUMERIC	NUMBER (p,s)
REAL4	FLOAT
REAL8	FLOAT
STRING	날짜 표시: DATE 포함 시간 표시: TIMESTAMP 포함 타임스탬프 표시: TIMESTAMP 포함 timestamp_with_timezone 표시: TIMESTAMP WITH TIMEZONE 포함 timestamp_with_local_timezone 표시: TIMESTAMP WITH LOCAL TIMEZONE 포함 interval_year_to_month 표시: INTERVAL YEAR TO MONTH 포함 interval_day_to_second 표시: INTERVAL DAY TO SECOND 포함 길이 > 4000인 경우: CLOB 다른 모든 경우: VARCHAR2(길이)

AWS DMS 데이터 형식	Oracle 데이터 형식
UINT1	NUMBER (3)
UINT2	NUMBER (5)
UINT4	NUMBER (10)
UINT8	NUMBER (19)
WSTRING	길이가 > 2000: NCLOB인 경우 다른 모든 경우: NVARCHAR2(길이)
BLOB	BLOB AWS DMS에서 이 데이터 형식을 사용하려면 특정 작업에서 BLOB 사용을 활성화해야 합니다. BLOB 데이터 형식은 기본 키를 포함하는 테이블에서만 지원됩니다.
CLOB	CLOB AWS DMS에서 이 데이터 형식을 사용하려면 특정 작업에서 CLOB 사용을 활성화해야 합니다. 변경 데이터 캡처(CDC) 중에 기본 키를 포함하는 테이블에서만 CLOB 데이터 형식이 지원됩니다. STRING 선언된 크기가 4000바이트보다 큰 소스의 Oracle VARCHAR2 데이터 형식이 AWS DMS CLOB를 통해 Oracle 대상의 STRING에 매핑됩니다.

AWS DMS 데이터 형식	Oracle 데이터 형식
NCLOB	NCLOB AWS DMS에서 이 데이터 형식을 사용하려면 특정 작업에서 NCLOB 사용을 활성화해야 합니다. CDC 중에 NCLOB 데이터 형식은 기본 키를 포함하는 테이블에서만 지원됩니다. WSTRING 선언된 크기가 4000바이트보다 큰 소스의 Oracle VARCHAR2 데이터 형식이 AWS DMS NCLOB를 통해 Oracle 대상의 WSTRING에 매핑됩니다.
XMLTYPE	XMLTYPE 대상 데이터 형식은 Oracle 간 복제 작업에서만 관련이 있습니다. 소스 데이터베이스가 Oracle이면 소스 데이터 형식은 Oracle 대상에 '그대로' 복제됩니다. 예를 들어, 원본의 XMLTYPE 데이터 형식은 대상에서 XMLTYPE 데이터 형식으로 생성됩니다.

Microsoft SQL Server 데이터베이스를 AWS Database Migration Service의 대상으로 사용

AWS DMS를 사용하여 Microsoft SQL Server 데이터베이스로 데이터를 마이그레이션할 수 있습니다. SQL Server 데이터베이스를 대상으로 사용할 때에는 다른 SQL Server 데이터베이스나 지원되는 다른 데이터베이스 중 하나에서 가져온 데이터베이스를 마이그레이션할 수 있습니다.

AWS DMS가 대상으로 지원하는 SQL Server 버전에 관한 자세한 내용은 대상: AWS DMS을 참조하십시오.

AWS DMS는 Enterprise, Standard, Workgroup 및 Developer의 온프레미스 및 Amazon RDS 에디션을 지원합니다.

AWS DMS 및 SQL Server 대상 데이터베이스 사용에 대한 추가 세부 정보는 다음을 참조하십시오.

주제

- AWS Database Migration Service에서 SQL Server를 대상으로 사용할 때 적용되는 제한 사항
- SQL Server를 AWS Database Migration Service의 대상으로 사용할 때의 보안 요구 사항

- SQL Server를 AWS DMS의 대상으로 사용할 경우 엔드포인트 설정
- Microsoft SQL Server용 대상 데이터 형식

AWS Database Migration Service에서 SQL Server를 대상으로 사용할 때 적용되는 제한 사항

다음 제한 사항은 SQL Server 데이터베이스를 AWS DMS의 대상으로 사용 시 적용됩니다.

- 계산된 열에서 SQL Server 대상 테이블을 수동으로 생성하면, BCP 대량 복사 유틸리티를 사용할 때 전체 로드 복제가 지원되지 않습니다. 전체 로드 복제를 사용하려면 엔드포인트에서 추가 연결 속성(ECA) `'useBCPFullLoad=false'`를 설정하여 BCP 로드를 비활성화하십시오. 엔드포인트에서 ECA 설정에 관한 자세한 내용은 소스 및 대상 엔드포인트 생성 단원을 참조하십시오. BCP 사용에 관한 자세한 내용은 Microsoft SQL Server 설명서를 참조하십시오.
- SQL Server 공간 데이터 형식(GEOMETRY 및 GEOGRAPHY)으로 테이블을 복제하면 AWS DMS는 삽입할 수 있었던 모든 공간 참조 식별자(SRID)를 기본 SRID로 대체합니다. GEOMETRY에서 기본 SRID는 0이고 GEOGRAPHY에서는 4326입니다.
- 임시 테이블은 지원되지 않습니다. 임시 테이블 마이그레이션은 이 테이블이 대상에서 수동으로 생성되지 않는 경우 Transactional Apply 모드에서 복제 전용 작업을 통해 수행할 수도 있습니다.
- 현재 PostgreSQL 소스의 `boolean` 데이터 형식은 값이 일치하지 않는 `bit` 데이터 형식으로 SQLServer 대상에 마이그레이션됩니다.

  차선책으로 다음을 수행합니다.

  - 열에 대한 `VARCHAR(1)` 데이터 형식으로 테이블을 미리 생성하거나 AWS DMS에서 테이블을 생성하도록 합니다. 그런 다음, 다운스트림 처리에서 'F'를 False로 처리하고 'T'를 True로 처리합니다.
  - 다운스트림 처리를 변경할 필요가 없도록 작업에 변환 규칙을 추가하여 'F' 값을 '0'으로 변경하고 'T' 값을 1로 변경한 다음 이를 SQL 서버 비트 데이터 형식으로 저장하십시오.
- AWS DMS는 열 Null 허용 여부를 설정하는 변경 처리를 지원하지 않습니다(`ALTER TABLE` 문이 있는 `ALTER COLUMN [SET|DROP] NOT NULL` 절 사용).
- Windows 인증은 지원되지 않습니다.

SQL Server를 AWS Database Migration Service의 대상으로 사용할 때의 보안 요구 사항

다음은 Microsoft SQL Server 대상과 함께 AWS DMS를 사용할 경우의 보안 요구 사항에 대해 설명합니다.

- AWS DMS 사용자 계정에는 연결 중인 SQL Server 데이터베이스에서 적어도 db_owner 사용자 역할이 있어야 합니다.
- SQL Server 시스템 관리자는 모든 AWS DMS 사용자 계정에 이 권한을 부여해야 합니다.

SQL Server를 AWS DMS의 대상으로 사용할 경우 엔드포인트 설정

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 SQL Server 대상 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI에서 `create-endpoint` 명령을 `--microsoft-sql-server-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

SQL Server를 대상으로 할 때 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

명칭	Description
`ControlTablesFileGroup`	AWS DMS 내부 테이블의 파일 그룹을 지정합니다. 복제 작업을 시작할 때에는 모든 내부 AWS DMS 제어 테이블(awsdms_ apply_exception, awsdms_apply, awsdms_changes)이 지정된 파일 그룹에 생성됩니다. 기본값: 해당 사항 없음 유효값: 문자열 예제: `--microsoft-sql-server-settings '{"ControlTablesFileGroup": "filegroup1"}'` 다음은 파일 그룹을 생성한 명령의 예입니다. `ALTER DATABASE replicate ADD FILEGROUP Test1FG1; GO ALTER DATABASE replicate`

명칭	Description

	<pre>ADD FILE (
	  NAME = test1dat5,
	  FILENAME = 'C:\temp\DATA\t1dat5.ndf',
	
	  SIZE = 5MB,
	  MAXSIZE = 100MB,
	  FILEGROWTH = 5MB
	)
	TO FILEGROUP Test1FG1;
	GO</pre>

`ExecuteTimeout`	이 추가 연결 속성(ECA)을 사용하여 SQL Server 인스턴스의 클라이언트 문 제한 시간을 초 단위로 설정합니다. 기본값은 60초입니다. 예제: `'{"ExecuteTimeout": 100}'`
`UseBCPFullLoad`	이 속성을 사용하여 BCP를 사용한 전체 로드 작업에서 데이터를 전송합니다. 소스 테이블에 없는 자격 증명 열이 대상 테이블에 포함되어 있다면 use BCP for loading table(테이블 로드 시 BCP 사용) 옵션을 비활성화해야 합니다. 기본값: true 유효값: true/false 예제: `--microsoft-sql-server-settings '{"UseBCPFullLoad": false}'`

Microsoft SQL Server용 대상 데이터 형식

다음 테이블에는 AWS DMS를 사용하고 기본적으로 AWS DMS 데이터 형식에 매핑할 때 지원되는 Microsoft SQL Server 대상 데이터 형식이 나와 있습니다. AWS DMS 데이터 형식에 관한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

AWS DMS 데이터 유형	SQL Server 데이터 형식
BOOLEAN	TINYINT
BYTES	VARBINARY(길이)
날짜	SQL Server 2008 및 이후 버전에서는 DATE를 사용합니다. 이전 버전의 경우에 크기가 3이하라면 DATETIME을 사용합니다. 그 외 모든 경우에서는 VARCHAR(37)을 사용합니다.
TIME	SQL Server 2008 및 이후 버전에서는 DATETIME2(%d)를 사용합니다. 이전 버전의 경우에 크기가 3이하라면 DATETIME을 사용합니다. 그 외 모든 경우에서는 VARCHAR(37)을 사용합니다.
DATETIME	SQL Server 2008 및 이후 버전에서는 DATETIME2(크기)를 사용합니다. 이전 버전의 경우에 크기가 3이하라면 DATETIME을 사용합니다. 그 외 모든 경우에서는 VARCHAR(37)을 사용합니다.
INT1	SMALLINT
INT2	SMALLINT
INT4	INT
INT8	BIGINT
NUMERIC	NUMERIC(p,s)
REAL4	REAL
REAL8	FLOAT
STRING	열이 날짜 또는 시간 열인 경우, 다음 작업을 수행합니다. • SQL Server 2008 이상에서는 DATETIME2를 사용합니다.

AWS DMS 데이터 유형	SQL Server 데이터 형식
	• 이전 버전의 경우에 크기가 3이하라면 DATETIME을 사용합니다. 그 외 모든 경우에서는 VARCHAR(37)을 사용합니다. 열이 날짜 또는 시간 열이 아닌 경우, VARCHAR(길이)를 사용합니다.
UINT1	TINYINT
UINT2	SMALLINT
UINT4	INT
UINT8	BIGINT
WSTRING	NVARCHAR(길이)
BLOB	VARBINARY(max) IMAGE AWS DMS에서 이 데이터 형식을 사용하려면 특정 작업에서 BLOB 사용을 활성화해야 합니다. AWS DMS는 기본 키를 포함하는 테이블에서만 BLOB 데이터 형식을 지원합니다.
CLOB	VARCHAR(최대) AWS DMS에서 이 데이터 형식을 사용하려면 특정 작업에서 CLOB 사용을 활성화해야 합니다. 변경 데이터 캡처(CDC) 중에 AWS DMS는 기본 키를 포함하는 데이터에서만 CLOB 데이터 형식을 지원합니다.
NCLOB	NVARCHAR(최대) AWS DMS에서 이 데이터 형식을 사용하려면 특정 작업에서 NCLOB 사용을 활성화해야 합니다. CDC 중에 AWS DMS는 기본 키를 포함하는 테이블에서만 NCLOB 데이터 형식을 지원합니다.

AWS Database Migration Service에서 PostgreSQL 데이터베이스를 대상으로 사용

다른 PostgreSQL 데이터베이스 또는 지원되는 다른 데이터베이스 중 하나를 사용하여 PostgreSQL 데이터베이스로 AWS DMS데이터를 마이그레이션할 수 있습니다.

대상으로 AWS DMS 지원하는 PostgreSQL 버전에 대한 자세한 내용은 을 참조하십시오. 대상: AWS DMS

AWS DMS 사용할 버전
사용 가능한 모든 AWS DMS 버전을 사용하세요.
AWS DMS 버전 3.4.3 이상을 사용하십시오.
AWS DMS 버전 3.4.7 이상을 사용하십시오.
AWS DMS 버전 3.5.1 이상을 사용하십시오.

> **Note**
>
> - Amazon Aurora Serverless는 PostgreSQL과 호환되는 Amazon Aurora의 대상으로 사용할 수 있습니다. Amazon Aurora 서버리스에 대한 자세한 내용은 Amazon Aurora 사용 설명서의 Amazon Aurora 서버리스 v2 사용을 참조하십시오.
> - Aurora Serverless DB 클러스터는 Amazon VPC에서만 액세스할 수 있으며 퍼블릭 IP 주소를 사용할 수 없습니다. 따라서 Aurora PostgreSQL Serverless가 아닌 다른 지역에 복제 인스턴스를 사용하려는 경우, vpc 피어링을 구성해야 합니다. 그렇지 않으면 Aurora PostgreSQL 서버리스 지역의 가용성을 확인하고 Aurora PostgreSQL 서버리스와 복제 인스턴스 모두에 해당 지역 중 하나를 사용하기로 결정하십시오.

- Babelfish 기능은 Amazon Aurora에 내장되어 있으며 추가 비용이 들지 않습니다. 자세한 내용은 AWS Database Migration Service의 대상으로 Babelfish for Aurora PostgreSQL 사용을 참조하십시오.

AWS DMS 전체 로드 단계에서 소스에서 타겟으로 데이터를 마이그레이션할 때 table-by-table 접근 방식을 취합니다. 전체 로드 단계 시 테이블 순서를 보장할 수 없습니다. 일반적으로 전체 로드 단계가 수행되는 동안과 개별 테이블의 캐시된 트랜잭션이 적용되는 동안 테이블은 동기화되지 않습니다. 결과적으로 활성 참조 무결성 제약으로 인해 전체 로드 단계 중에 작업 실패가 발생합니다.

PostgreSQL에서는 트리거를 사용하여 외래 키(참조 무결성 제약)가 구현됩니다. 전체 로드 단계에서는 각 테이블을 한 번에 하나씩 AWS DMS 로드합니다. 다음 방법 중 하나를 사용하여 전체 로드 동안 외래 키 제약을 비활성화하는 것이 좋습니다.

- 인스턴스에서 모든 트리거를 임시로 비활성화한 후 전체 로드를 완료합니다.
- PostgreSQL에서 `session_replication_role` 파라미터를 사용합니다.

해당 시점에 트리거는 `origin`, `replica`, `always` 또는 `disabled` 상태 중 하나일 수 있습니다. `session_replication_role` 파라미터가 `replica`로 설정되는 경우, `replica` 상태의 트리거만 활성 상태이며 호출될 때 트리거됩니다. 그렇지 않으면, 트리거가 비활성 상태로 유지됩니다.

PostgreSQL에는 `session_replication_role`이 설정되었더라도 테이블 잘림을 방지하기 위한 안전 메커니즘이 있습니다. 전체 로드 실행이 완료될 수 있도록 이 메커니즘을 대안으로 사용하여 트리거를 비활성화할 수 있습니다. 이렇게 하려면 대상 테이블 준비 모드를 `DO_NOTHING`으로 설정합니다. 그렇지 않으면 외래 키 제약이 있는 경우 DROP 및 TRUNCATE 작업이 실패합니다.

Amazon RDS에서는 파라미터 그룹을 사용하여 이 파라미터 설정을 제어할 수 있습니다. Amazon EC2에서 실행 중인 PostgreSQL 인스턴스의 경우, 파라미터를 직접 설정할 수 있습니다.

PostgreSQL 데이터베이스를 대상으로 사용하는 방법에 AWS DMS대한 자세한 내용은 다음 섹션을 참조하십시오.

주제

- PostgreSQL을 타겟으로 사용할 때의 제한 사항 AWS Database Migration Service
- PostgreSQL 데이터베이스를 대상으로 사용할 때의 보안 요구 사항 AWS Database Migration Service
- PostgreSQL을 대상으로 사용할 때의 엔드포인트 설정 및 추가 연결 속성 (ECA) AWS DMS

- PostgreSQL용 대상 데이터 형식
- Aurora용 Babelfish를 PostgreSQL의 타겟으로 사용 AWS Database Migration Service

PostgreSQL을 타겟으로 사용할 때의 제한 사항 AWS Database Migration Service

다음 제한 사항은 AWS DMS에서 PostgreSQL 데이터베이스를 대상으로서 사용할 때 적용됩니다.

- 이기종 마이그레이션의 경우 내부적으로 JSON 데이터 형식이 네이티브 CLOB 데이터 형식으로 변환됩니다.
- Oracle에서 PostgreSQL로 마이그레이션할 때 Oracle의 열에 NULL 문자 (16진수 값 U+0000) 가 포함된 경우 NULL 문자를 공백 (16진수 값 U+0020) AWS DMS 으로 변환합니다. 이는 PostgreSQL 제한 사항 때문입니다.
- AWS DMS coalesce 함수로 생성된 고유 인덱스가 있는 테이블로의 복제를 지원하지 않습니다.
- 테이블에서 시퀀스를 사용하는 경우 원본 데이터베이스에서 복제를 중지한 후 대상 데이터베이스의 각 시퀀스에 NEXTVAL 대한 값을 업데이트하십시오. AWS DMS 원본 데이터베이스의 데이터를 복사하지만 복제가 진행 중인 동안에는 시퀀스를 대상으로 마이그레이션하지 않습니다.

PostgreSQL 데이터베이스를 대상으로 사용할 때의 보안 요구 사항 AWS Database Migration Service

보안상의 목적으로 데이터 마이그레이션에 사용되는 사용자 계정은 대상으로 사용하는 모든 PostgreSQL 데이터베이스에서 등록된 사용자여야 합니다.

PostgreSQL 대상 엔드포인트에는 마이그레이션을 AWS DMS 실행하려면 최소 사용자 권한이 필요합니다. 다음 예를 참조하십시오.

```
CREATE USER newuser WITH PASSWORD 'your-password';
ALTER SCHEMA schema_name OWNER TO newuser;
```

또는

```
GRANT USAGE ON SCHEMA schema_name TO myuser;
GRANT CONNECT ON DATABASE postgres to myuser;
GRANT CREATE ON DATABASE postgres TO myuser;
GRANT CREATE ON SCHEMA schema_name TO myuser;
```

```
    GRANT UPDATE, INSERT, SELECT, DELETE, TRUNCATE ON ALL TABLES IN SCHEMA schema_name
TO myuser;
    GRANT TRUNCATE ON schema_name."BasicFeed" TO myuser;
```

PostgreSQL을 대상으로 사용할 때의 엔드포인트 설정 및 추가 연결 속성 (ECA) AWS DMS

엔드포인트 설정 및 추가 연결 속성 (ECA) 을 사용하여 PostgreSQL 대상 데이터베이스를 구성할 수 있습니다.

AWS DMS 콘솔을 사용하거나 에서 JSON 구문을 사용하여 `create-endpoint` 명령을 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다. AWS CLI`--postgre-sql-settings '{"EndpointSetting": "value", ...}'`

엔드포인트의 `ExtraConnectionAttributes` 파라미터를 사용하여 ECA를 지정합니다.

PostgreSQL을 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

명칭	설명
`MaxFileSize`	데이터를 PostgreSQL로 전송할 때 사용되는 .csv 파일의 최대 크기(단위: KB)를 지정합니다. 기본값: 32,768KB(32MB) 유효값: 1~1,048,576KB(최대 1.1GB) 예제: `--postgre-sql-settings '{"MaxFileSize": 512}'`
`ExecuteTimeout`	PostgreSQL 인스턴스의 클라이언트 문 제한 시간을 초 단위로 설정합니다. 기본값은 60초입니다. 예제: `--postgre-sql-settings '{"ExecuteTimeout": 100}'`
`AfterConnectScript= SET session_replication_role = replica`	이 속성에는 AWS DMS 데이터를 대량으로 로드하는 데 걸리는 시간을 줄이기 위한 우회 외부 키와 사용자 트리거가 있습니다.

명칭	설명
MapUnboundedNumericAsString	이 파라미터는 숫자 값의 정밀도를 유지하면서 성공적으로 마이그레이션할 수 있도록 제한이 없는 숫자 데이터 형식의 열을 문자열로 취급합니다. 이 파라미터는 PostgreSQL 소스에서 PostgreSQL 대상 또는 PostgreSQL과 호환되는 데이터베이스로 복제하는 경우에만 사용하십시오. 기본값: false 유효값: false/true 예제: `--postgre-sql-settings '{"MapUnboundedNumericAsString": "true"}` 이 파라미터를 사용하면 숫자에서 문자열로, 다시 숫자로 변환되므로 복제 성능이 일부 저하될 수 있습니다. 이 파라미터는 DMS 버전 3.4.4 이상에서 사용할 수 있습니다. **Note** 단지 MapUnboundedNumericAsString 을 PostgreSQL 소스 및 대상 엔드포인트에서 함께 사용해야 합니다. 소스 PostgreSQL 엔드포인트에서 MapUnboundedNumericAsString 을 사용하면 CDC 중에 정밀도가 28로 제한됩니다. 대상 엔드포인트에서 MapUnboundedNumericAsString 을 사용하면 정밀도 28, 스케일 6으로 데이터를 마이그레이션할 수 있습니다. PostgreSQL이 아닌 대상에 MapUnboundedNumericAsString 을 사용하지 마세요.

명칭	설명
loadUsingCSV	이 추가 연결 속성 (ECA) 을 사용하면\ COPY 명령을 사용하여 전체 로드 작업에 필요한 데이터를 전송할 수 있습니다. 기본 값: true 유효값: true/false ECA 예제: loadUsingCSV=true; 참고: 이 ECA를 false로 설정하면 INSERT가 직접 실행되기 때문에 복제 성능이 일부 저하될 수 있습니다.
DatabaseMode	이 속성을 사용하면 Babelfish 엔드포인트와 같이 일부 추가 구성이 필요한 PostgreSQL 호환 엔드포인트에 대한 복제 처리의 기본 동작을 변경할 수 있습니다. 기본 값: DEFAULT 유효값: DEFAULT, BABELFISH 예제: DatabaseMode=default;
BabelfishDatabaseName	이 속성을 사용하여 마이그레이션할 대상 Babelfish T-SQL 데이터베이스의 이름을 지정할 수 있습니다. DatabaseMode 가 Babelfish 로 설정된 경우 이 데이터베이스는 필수입니다. 이는 예약된 babelfish_db 데이터베이스가 아닙니다. 예제: BabelfishDatabaseName=TargetDb;

PostgreSQL용 대상 데이터 형식

에 대한 PostgreSQL 데이터베이스 엔드포인트는 대부분의 PostgreSQL 데이터베이스 데이터 AWS DMS 유형을 지원합니다. 다음 표에는 을 AWS DMS 사용할 때 지원되는 PostgreSQL 데이터베이스 대상 데이터 유형과 데이터 유형으로부터의 기본 매핑이 나와 있습니다. AWS DMS

AWS DMS 데이터 유형에 대한 자세한 내용은 을 참조하십시오. AWS Database Migration Service에서 사용되는 데이터 형식

AWS DMS 데이터 유형	PostgreSQL 데이터 형식
BOOLEAN	BOOLEAN
BLOB	BYTEA
BYTES	BYTEA
날짜	날짜
TIME	TIME
DATETIME	크기 범위가 0~6인 경우, TIMESTAMP를 사용합니다. 크기 범위가 7~9인 경우, VARCHAR(37)을 사용합니다.
INT1	SMALLINT
INT2	SMALLINT
INT4	INTEGER
INT8	BIGINT
NUMERIC	DECIMAL (P,S)
REAL4	FLOAT4
REAL8	FLOAT8
STRING	길이가 1~21,845인 경우, VARCHAR(바이트 단위의 길이)을 사용합니다. 길이가 21,846~2,147,483,647인 경우, VARCHAR(65535)을 사용합니다.
UINT1	SMALLINT
UINT2	INTEGER

AWS DMS 데이터 유형	PostgreSQL 데이터 형식
UINT4	BIGINT
UINT8	BIGINT
WSTRING	길이가 1~21,845인 경우, VARCHAR(바이트 단위의 길이)을 사용합니다. 길이가 21,846~2,147,483,647인 경우, VARCHAR(65535)을 사용합니다.
NCLOB	TEXT
CLOB	TEXT

> **Note**
>
> PostgreSQL AWS DMS 소스에서 복제하는 경우 사용자 정의 데이터 유형의 열을 제외하고 모든 열에 대해 동일한 데이터 유형을 사용하여 대상 테이블을 만듭니다. 그러한 경우에 데이터 형식은 대상에서 'character varying'으로 생성됩니다.

Aurora용 Babelfish를 PostgreSQL의 타겟으로 사용 AWS Database Migration Service

AWS Database Migration Service을 사용하여 SQL Server 소스 테이블을 Babelfish for Amazon Aurora PostgreSQL 대상으로 마이그레이션할 수 있습니다. Aurora PostgreSQL은 Babelfish를 사용하여 Microsoft SQL Server의 전용 SQL 방언인 T-SQL을 이해하며 동일한 통신 프로토콜을 지원합니다. 따라서 이제 SQL Server용으로 작성된 애플리케이션은 코드 변경 횟수를 줄이면서 Aurora와 연동할 수 있습니다. Babelfish 기능은 Amazon Aurora에 내장되어 있으며 추가 비용이 들지 않습니다. Amazon RDS 콘솔에서 Amazon Aurora 클러스터의 Babelfish를 활성화할 수 있습니다.

AWS DMS 콘솔, API 또는 CLI 명령을 사용하여 AWS DMS 대상 엔드포인트를 생성할 때는 대상 엔진을 Amazon Aurora PostgreSQL로 지정하고 데이터베이스 이름을 babelfish_db로 지정합니다. 엔드포인트 설정 섹션에서 `DatabaseMode` 설정을 `Babelfish`에 추가하고 대상 Babelfish T-SQL 데이터베이스의 이름에 `BabelfishDatabaseName`을 추가합니다.

마이그레이션 작업에 변환 규칙 추가

Babelfish 대상에 대한 마이그레이션 작업을 정의할 때는 DMS가 대상 데이터베이스의 사전 생성된 T-SQL Babelfish 테이블을 사용하도록 하는 변환 규칙을 포함해야 합니다.

먼저 모든 테이블 이름을 소문자로 만드는 변환 규칙을 마이그레이션 작업에 추가하십시오. Babelfish는 T-SQL을 사용하여 생성한 테이블 이름을 PostgreSQL `pg_class` 카탈로그에 소문자로 저장합니다. 하지만 SQL Server 테이블에 대소문자가 혼합된 이름이 있는 경우, DMS는 T-SQL 호환 데이터 형식 대신 PostgreSQL 네이티브 데이터 형식을 사용하여 테이블을 생성합니다. 따라서 모든 테이블 이름을 소문자로 만드는 변환 규칙을 추가해야 합니다. 단, 열 이름은 소문자로 변환해서는 안 됩니다.

다음으로 클러스터를 정의할 때 다중 데이터베이스 마이그레이션 모드를 사용한 경우, 원본 SQL Server 스키마의 이름을 바꾸는 변환 규칙을 추가하십시오. T-SQL 데이터베이스의 이름을 포함하도록 SQL Server 스키마의 이름을 바꿔야 합니다. 예를 들어 원래 SQL Server 스키마 이름이 dbo이고 T-SQL 데이터베이스 이름이 `mydb`인 경우, 변환 규칙을 사용하여 스키마 이름을 `mydb_dbo`로 바꾸십시오.

단일 데이터베이스 모드를 사용하는 경우, 스키마 이름을 바꾸는 변환 규칙은 필요하지 않습니다. 스키마 이름은 Babelfish의 대상 T-SQL 데이터베이스와 매핑됩니다 one-to-one .

다음 샘플 변환 규칙은 모든 테이블 이름을 소문자로 만들고 원래 SQL Server 스키마 이름을 dbo에서 `mydb_dbo`로 바꿉니다.

```
{
   "rules": [
   {
      "rule-type": "transformation",
      "rule-id": "566251737",
      "rule-name": "566251737",
      "rule-target": "schema",
      "object-locator": {
         "schema-name": "dbo"
      },
      "rule-action": "rename",
      "value": "mydb_dbo",
      "old-value": null
   },
   {
      "rule-type": "transformation",
      "rule-id": "566139410",
      "rule-name": "566139410",
```

```
      "rule-target": "table",
      "object-locator": {
         "schema-name": "%",
         "table-name": "%"
      },
      "rule-action": "convert-lowercase",
      "value": null,
      "old-value": null
   },
   {
      "rule-type": "selection",
      "rule-id": "566111704",
      "rule-name": "566111704",
      "object-locator": {
         "schema-name": "dbo",
         "table-name": "%"
      },
      "rule-action": "include",
      "filters": []
   }
]
}
```

PostgreSQL 대상 엔드포인트를 Babelfish 테이블과 함께 사용할 때 적용되는 제한 사항

PostgreSQL 대상 엔드포인트를 Babelfish 테이블과 함께 사용할 때 적용되는 제한 사항은 다음과 같습니다.

- 대상 테이블 준비 모드의 경우, 아무것도 안 함 또는 잘라내기 모드만 사용하십시오. 대상에서 테이블 삭제 모드를 사용하지 마세요. 이 모드에서 DMS는 T-SQL이 인식하지 못할 수도 있는 PostgreSQL 테이블로 테이블을 생성합니다.
- AWS DMS sql_variant 데이터 유형을 지원하지 않습니다.
- Babelfish는 `HEIRARCHYID`, `GEOMETRY` 및 `GEOGRAPHY` 데이터 형식을 지원하지 않습니다. 이러한 데이터 형식을 마이그레이션하려면 변환 규칙을 추가하여 데이터 형식을 `wstring(250)`으로 변환할 수 있습니다.
- Babelfish는 해당 `BYTEA` 데이터 형식을 사용하여 `BINARY`, `VARBINARY` 및 `IMAGE` 데이터 형식의 마이그레이션만 지원합니다. Aurora PostgreSQL의 이전 버전에서는 DMS를 사용하여 이러한 테이블을 Babelfish 대상 엔드포인트로 마이그레이션할 수 있습니다. 다음 예제와 같이 `BYTEA` 데이터 형식에 길이를 지정할 필요가 없습니다.

```
[Picture] [VARBINARY](max) NULL
```

이전 T-SQL 데이터 형식을 T-SQL 지원 BYTEA 데이터 형식으로 변경합니다.

```
[Picture] BYTEA NULL
```

- 이전 버전의 Aurora PostgreSQL Babelfish에서 PostgreSQL 대상 엔드포인트를 사용하여 SQL Server에서 Babelfish로 지속적인 복제를 위한 마이그레이션 작업을 생성하는 경우, IDENTITY 열을 사용하는 모든 테이블에 SERIAL 데이터 형식을 할당해야 합니다. Aurora PostgreSQL(버전 15.3/14.8 및 이후 버전)과 Babelfish(버전 3.2.0 및 이후 버전)부터 ID 열이 지원되므로 시리얼 데이터 형식을 더 이상 할당할 필요가 없습니다. 자세한 내용은 SQL Server에서 Aurora PostgreSQL로의 마이그레이션 플레이북의 시퀀스 및 ID 섹션에서 시리얼 사용을 참조하십시오. 그런 다음, Babelfish에서 테이블을 생성할 때 다음의 열 정의를 변경하십시오.

```
    [IDCol] [INT] IDENTITY(1,1) NOT NULL PRIMARY KEY
```

이전 정의를 다음과 같이 변경하세요.

```
    [IDCol] SERIAL PRIMARY KEY
```

Babelfish와 호환되는 Aurora PostgreSQL은 기본 구성을 사용하여 하나의 시퀀스를 생성하고 해당 열에 NOT NULL 제약 조건을 추가합니다. 새로 만든 시퀀스는 (1씩 증가하는) 일반 시퀀스처럼 동작하며 복합 SERIAL 옵션이 없습니다.

- IDENTITY 열 또는 SERIAL 데이터 형식을 사용하는 테이블로 데이터를 마이그레이션한 후, 해당 열의 최대값을 기준으로 PostgreSQL 기반 시퀀스 객체를 재설정합니다. 테이블을 완전히 로드한 후 다음 T-SQL 쿼리를 사용하여 연결된 시퀀스 객체를 시드하는 문을 생성합니다.

```
DECLARE @schema_prefix NVARCHAR(200) = ''

IF current_setting('babelfishpg_tsql.migration_mode') = 'multi-db'
        SET @schema_prefix = db_name() + '_'

SELECT 'SELECT setval(pg_get_serial_sequence(''' + @schema_prefix +
 schema_name(tables.schema_id) + '.' + tables.name + ''', ''' + columns.name + ''')
               ,(select max(' + columns.name + ') from ' +
 schema_name(tables.schema_id) + '.' + tables.name + '));'
FROM sys.tables tables
JOIN sys.columns columns ON tables.object_id = columns.object_id
```

```
WHERE columns.is_identity = 1

UNION ALL

SELECT 'SELECT setval(pg_get_serial_sequence(''' + @schema_prefix + table_schema +
 '.' + table_name + ''',
''' + column_name + '''),(select max(' + column_name + ') from ' + table_schema + '.'
 + table_name + '));'
FROM information_schema.columns
WHERE column_default LIKE 'nextval(%';
```

쿼리는 최대 IDENTITY 및 SERIAL 값을 업데이트하기 위해 실행하는 일련의 SELECT 문을 생성합니다.

- Babelfish 3.2 이전 버전의 경우 전체 LOB 모드에서 테이블 오류가 발생할 수 있습니다. 이 경우 로드에 실패한 테이블에 대해 별도의 작업을 생성하십시오. 그런 다음 제한된 LOB 모드를 사용하여 최대 LOB 크기(KB)에 적합한 값을 지정합니다. 또 다른 옵션은 SQL Server 엔드포인트 연결 속성 설정을 `ForceFullLob=True`로 설정하는 것입니다.
- Babelfish 3.2 이전 버전의 경우, 정수 기반 기본 키를 사용하지 않는 Babelfish 테이블로 데이터 유효성 검사를 수행하면 적절한 고유 키를 찾을 수 없다는 메시지가 생성됩니다. Aurora PostgreSQL(버전 15.3/14.8 및 이후 버전) 및 Babelfish(버전 3.2.0 및 이후 버전)를 시작으로 정수가 아닌 기본 키에 대한 데이터 검증이 지원됩니다.
- 초 단위 소수점 자릿수의 정밀도 차이 때문에 DMS는 `DATETIME` 데이터 형식을 사용하는 Babelfish 테이블에 대한 데이터 검증 실패를 보고합니다. 이러한 실패를 억제하기 위해 `DATETIME` 데이터 형식에 대해 다음과 같은 유효성 검사 규칙 유형을 추가할 수 있습니다.

```
{
         "rule-type": "validation",
         "rule-id": "3",
         "rule-name": "3",
         "rule-target": "column",
         "object-locator": {
             "schema-name": "dbo",
             "table-name": "%",
             "column-name": "%",
             "data-type": "datetime"
         },
         "rule-action": "override-validation-function",
         "source-function": "case when ${column-name} is NULL then NULL else 0 end",
         "target-function": "case when ${column-name} is NULL then NULL else 0 end"
```

```
    }
```

AWS Database Migration Service에서 MySQL 호환 데이터베이스를 대상으로 사용

AWS DMS를 사용하면 AWS DMS가 지원하는 소스 데이터 엔진 중 하나에서 MySQL 호환 데이터베이스로 데이터를 마이그레이션할 수 있습니다. 온프레미스 MySQL 호환 데이터베이스로 마이그레이션하는 경우, AWS DMS는 소스 엔진이 AWS 에코시스템 내에 있도록 요구합니다. 이 엔진은 Amazon RDS, Amazon Aurora 또는 Amazon S3와 같은 AWS 관리형 서비스에서 사용할 수 있습니다. 또는 Amazon EC2의 자체 관리형 데이터베이스에 있을 수도 있습니다.

SSL을 사용하여 MySQL 호환 엔드포인트와 복제 인스턴스 간의 연결을 암호화할 수 있습니다. MySQL 호환 엔드포인트에서 SSL 사용에 관한 자세한 내용은 AWS Database Migration Service에서 SSL 사용 섹션을 참조하십시오.

AWS DMS가 대상으로 지원하는 MySQL 버전에 관한 자세한 내용은 대상: AWS DMS를 참조하십시오.

AWS DMS의 대상으로는 다음 MySQL 호환 데이터베이스를 사용할 수 있습니다.

- MySQL Community Edition
- MySQL Standard Edition
- MySQL Enterprise Edition
- MySQL Cluster Carrier Grade Edition
- MariaDB Community Edition
- MariaDB Enterprise Edition
- MariaDB Column Store
- Amazon Aurora MySQL

> **Note**
>
> 소스 스토리지 엔진(MyISAM, MEMORY 등)에 상관없이 AWS DMS는 기본적으로 MySQL 호환 대상 테이블을 InnoDB 테이블로서 생성합니다.

InnoDB 이외에 스토리지 엔진에 테이블이 있어야 하는 경우, MySQL 호환 대상에서 테이블을 수동으로 생성하고 아무 작업 안 함 옵션을 사용하여 테이블을 마이그레이션할 수 있습니다. 자세한 내용은 전체 로드 작업 설정 섹션을 참조하세요.

MySQL 호환 데이터베이스를 AWS DMS의 대상으로 사용하여 작업하는 방법에 관한 자세한 내용은 다음 단원을 참조하십시오.

주제

- AWS Database Migration Service에서 어떤 MySQL 호환 데이터베이스든지 대상으로 사용
- AWS Database Migration Service에서 MySQL 호환 데이터베이스를 대상으로 사용할 때 적용되는 제한 사항
- AWS DMS에서 MySQL 호환 데이터베이스를 대상으로 사용할 때 엔드포인트 설정
- MySQLa용 대상 데이터 형식

AWS Database Migration Service에서 어떤 MySQL 호환 데이터베이스든지 대상으로 사용

AWS DMS에서 MySQL 호환 데이터베이스를 대상으로 사용하기 전에 다음 사전 조건이 완료되어 있는지 확인해야 합니다.

- MySQL 호환 데이터베이스에 대한 읽기/쓰기 권한이 있는 사용자 계정을 AWS DMS에 제공합니다. 필요한 권한을 생성하려면 다음 명령을 실행합니다.

```
CREATE USER '<user acct>'@'%' IDENTIFIED BY '<user password>';
GRANT ALTER, CREATE, DROP, INDEX, INSERT, UPDATE, DELETE, SELECT ON <schema>.* TO
'<user acct>'@'%';
GRANT ALL PRIVILEGES ON awsdms_control.* TO '<user acct>'@'%';
```

- 전체 로드 마이그레이션 단계 중에 대상 테이블에 대해 외래 키를 비활성화해야 합니다. 전체 로드 중에 MySQL 호환 데이터베이스에서 외래 키 검사를 비활성화하려면 대상 엔드포인트에 대한 AWS DMS 콘솔의 추가 연결 속성 섹션에 다음 명령을 추가해야 합니다.

```
Initstmt=SET FOREIGN_KEY_CHECKS=0;
```

- AWS DMS가 데이터를 대상 데이터베이스로 로드할 수 있도록 데이터베이스 파라미터 `local_infile = 1`을 설정합니다.

AWS Database Migration Service에서 MySQL 호환 데이터베이스를 대상으로 사용할 때 적용되는 제한 사항

MySQL 데이터베이스를 대상으로 사용 시 AWS DMS는 다음을 지원하지 않습니다.

- 데이터 정의 언어(DDL) 문은 TRUNCATE PARTITION, DROP TABLE 및 CREATE TABLE 명령을 수행합니다.
- `ALTER TABLE table_name ADD COLUMN column_name` 문을 사용하여 테이블의 시작이나 중간에 열을 추가합니다.
- 전체 로드 작업에서 MySQL 호환 대상으로 데이터를 로드하는 경우 AWS DMS는 작업 로그의 제약 조건으로 인해 발생한 오류를 보고하지 않으므로 중복 키 오류가 발생하거나 레코드 수와 일치하지 않을 수 있습니다. 이러한 문제는 MySQL이 `LOAD DATA` 명령으로 로컬 데이터를 처리하는 방식 때문에 발생합니다. 전체 로드 단계에서 다음을 꼭 수행해야 합니다.
  - d제약 조건 비활성화
  - AWS DMS 검증을 사용하여 데이터의 일관성을 확인합니다.
- 열의 값을 기존 값으로 업데이트하면 MySQL 호환 데이터베이스는 `0 rows affected` 경고를 반환합니다. 이 동작은 사실 오류는 아니지만 다른 데이터베이스 엔진이 이러한 상황을 처리하는 방식과는 다릅니다. 예를 들어, Oracle은 행 하나를 업데이트합니다. MySQL 호환 데이터베이스의 경우, AWS DMS가 awsdms_apply_exceptions 제어 테이블에 하나의 항목을 생성하고 다음 경고를 로깅합니다.

```
Some changes from the source database had no impact when applied to
the target database. See awsdms_apply_exceptions table for details.
```

- MySQL 버전 5.6과 호환되는 Amazon Aurora 버전 1의 대상으로 Aurora Serverless를 사용할 수 있습니다. MySQL 버전 5.7과 호환되는 Amazon Aurora 버전 2의 대상으로 Aurora Serverless를 사용할 수 있습니다. (MySQL 버전 5.7과 호환되는 Aurora Serverless를 사용할 수 있도록 Aurora MySQL 버전 2.07.1을 선택합니다.) Aurora Serverless에 관한 자세한 내용을 알아보려면 Amazon Aurora 사용 설명서의 Amazon Aurora Serverless 사용을 참조하세요.
- 인스턴스가 쓰기 가능 모드인 경우, 즉 `read_only` 및 `innodb_read_only` 파라미터가 `0` 또는 `OFF`로 설정된 경우를 제외하면 AWS DMS는 Aurora 또는 Amazon RDS용 리더 엔드포인트 사용을

지원하지 않습니다. Amazon RDS 및 Aurora를 대상으로 사용하는 방법에 관한 자세한 내용은 다음을 참조하십시오.

- 연결되어 있는 DB 인스턴스 확인
- MySQL을 사용한 읽기 전용 복제본 업데이트

AWS DMS에서 MySQL 호환 데이터베이스를 대상으로 사용할 때 엔드포인트 설정

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 MySQL 호환 대상 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI의 `create-endpoint` 명령을 `--my-sql-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

MySQL을 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

이름	설명
`TargetDbType`	대상에 있는 소스 테이블을 단일 데이터베이스 또는 다중 데이터베이스로 마이그레이션할 위치를 지정합니다. `SPECIFIC_DATABASE` 를 지정하는 경우 AWS CLI 또는 AWS Management Console을 사용할 때 데이터베이스 이름을 지정해야 합니다. 기본값: `MULTIPLE_DATABASES` 유효값: {`SPECIFIC_DATABASE` , `MULTIPLE_DATABASES` } 예: `--my-sql-settings '{"TargetDbType": "MULTIPLE_DATABASES"}'`
`ParallelLoadThreads`	데이터를 MySQL 호환 대상 데이터베이스로 로드할 때의 성능을 개선합니다. 데이터를 MySQL 호환 대상 데이터베이스에 로드할 때 사용할 스레드 개수를 지정합니다. 각 스레드마다 별도의 연결이 필요하기 때문에 대량의 스레드를 설정하면 데이터베이스 성능에 부정적인 영향을 줄 수 있습니다. 기본값: 1

이름	설명
	유효값: 1~5 예: `--my-sql-settings '{"ParallelLoadThreads": 1}'`
`AfterConnectScript`	AWS DMS가 엔드포인트에 연결된 즉시 실행할 스크립트를 지정합니다. 예를 들면, MySQL 호환 대상이 받은 명령문을 latin1 문자 세트로 변환해야 함을 지정할 수 있습니다. 이 문자 세트는 데이터베이스의 기본 컴파일 입력 문자 세트입니다. 이 파라미터는 UTF8 클라이언트로부터 변환될 때 일반적으로 성능을 개선합니다. 예: `--my-sql-settings '{"AfterConnectScript": "SET character_set_connection='latin1'"}'`
`MaxFileSize`	데이터를 MySQL 호환 데이터베이스로 전송할 때 사용되는 .csv 파일의 최대 크기(단위: KB)를 지정합니다. 기본값: 32,768KB(32MB) 유효값: 1–1,048,576 `--my-sql-settings '{"MaxFileSize": 512}'`
`CleanSrcMetadataOnMismatch`	불일치 발생 시 복제 인스턴스에서 테이블 메타데이터 정보를 지운 후 다시 생성합니다. 예를 들어 테이블에서 Alter DDL 문을 실행하면 복제 인스턴스에 캐시된 테이블에 관한 다른 정보가 생길 수 있는 경우가 있습니다. 부울. 기본값: `false` 예: `--my-sql-settings '{"CleanSrcMetadataOnMismatch": false}'`

또한 추가 연결 속성을 사용하여 MySQL 호환 대상을 구성할 수도 있습니다.

다음 표에는 MySQL을 대상으로 하여 사용할 수 있는 추가 연결 속성이 나와 있습니다.

이름	설명
`Initstmt=SET FOREIGN_KEY_CHECKS=0;`	외래 키 점검을 비활성화합니다. 예: `--extra-connection-attributes "Initstmt=SET FOREIGN_KEY_CHECKS=0;"`
`Initstmt=SET time_zone`	대상 MySQL 호환 데이터베이스의 시간대를 지정합니다. 기본값: UTC 유효값: 대상 MySQL 데이터베이스에서 사용할 수 있는 시간대 이름. 예: `--extra-connection-attributes "Initstmt=SET time_zone= US/Pacific ;"`

또는 `--my-sql-settings` 명령의 `AfterConnectScript` 파라미터를 사용하여 외래 키 점검을 비활성화하고 데이터베이스의 시간대를 지정할 j수 있습니다.

MySQLa용 대상 데이터 형식

다음 테이블에는 AWS DMS를 사용하고 기본적으로 AWS DMS 데이터 형식에 매핑할 때 지원되는 MySQL 데이터베이스 대상 데이터 형식이 나와 있습니다.

AWS DMS 데이터 형식에 관한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

AWS DMS 데이터 형식	MySQL 데이터 형식
BOOLEAN	BOOLEAN
BYTES	길이가 1~65,535인 경우, VARBINARY(길이)를 사용합니다. 길이가 65,536~2,147,483,647인 경우, LONGLOB를 사용합니다.

AWS DMS 데이터 형식	MySQL 데이터 형식
DATE	DATE
TIME	TIME
TIMESTAMP	"크기가 0 이상이고 6 이하인 경우: DATETIME(크기) 크기가 7 이상이고 9 이하인 경우: VARCHAR (37)"
INT1	TINYINT
INT2	SMALLINT
INT4	INTEGER
INT8	BIGINT
NUMERIC	DECIMAL (p,s)
REAL4	FLOAT
REAL8	DOUBLE PRECISION
STRING	길이가 1~21,845인 경우, VARCHAR(길이)를 사용합니다. 길이가 21,846~2,147,483,647인 경우, LONGTEXT를 사용합니다.
UINT1	UNSIGNED TINYINT
UINT2	UNSIGNED SMALLINT
UINT4	UNSIGNED INTEGER
UINT8	UNSIGNED BIGINT

AWS DMS 데이터 형식	MySQL 데이터 형식
WSTRING	길이가 1~32,767인 경우, VARCHAR(길이)를 사용합니다. 길이가 32,768~2,147,483,647인 경우, LONGTEXT를 사용합니다.
BLOB	길이가 1~65,535인 경우, BLOB를 사용합니다. 길이가 65,536~2,147,483,647인 경우, LONGBLOB를 사용합니다. 길이가 0인 경우, LONGBLOB(전체 LOB 지원)를 사용합니다.
NCLOB	길이가 1~65,535인 경우, TEXT를 사용합니다. 길이가 65,536~2,147,483,647인 경우, CHARACTER SET용 ucs2와 함께 LONGTEXT를 사용합니다. 길이가 0인 경우, CHARACTER SET용 ucs2와 함께 LONGTEXT(전체 LOB 지원)를 사용합니다.
CLOB	길이가 1~65,535인 경우, TEXT를 사용합니다. 길이가 65,536~2,147,483,647인 경우, LONGTEXT를 사용합니다. 길이가 0인 경우, LONGTEXT(전체 LOB 지원)를 사용합니다.

AWS Database Migration Service의 대상으로 Amazon Redshift 데이터베이스 사용

AWS Database Migration Service을 사용하여 Amazon Redshift 데이터베이스로 데이터를 마이그레이션할 수 있습니다. Amazon Redshift는 클라우드에서 완전히 관리되는 페타바이트급 데이터 웨어하우

스 서비스입니다. Amazon Redshift 데이터베이스를 대상으로 사용할 때에는 지원되는 다른 소스 데이터베이스에서 가져온 데이터를 마이그레이션할 수 있습니다.

Amazon Redshift Serverless를 AWS DMS의 대상으로 사용할 수 있습니다. 자세한 내용은 대상으로서 Amazon Redshift Serverless와 함께 AWS DMS를 사용 단원을 참조하십시오.

Amazon Redshift 클러스터는 복제 인스턴스와 동일한 AWS 계정 및 AWS 리전에 있어야 합니다.

Amazon Redshift로 데이터베이스를 마이그레이션하는 동안 AWS DMS는 먼저 데이터를 Amazon S3 버킷으로 이동합니다. 파일이 Amazon S3 버킷에 있으면 AWS DMS는 sAmazon Redshift 데이터 웨어하우스의 해당 테이블로 이 파일을 전송합니다. AWS DMS는 Amazon Redshift 데이터베이스와 동일한 AWS 리전에 S3 버킷을 생성합니다. AWS DMS 복제 인스턴스는 동일한 AWS 리전에 있어야 합니다.

AWS CLI 또는 DMS API를 사용하여 데이터를 Amazon Redshift로 마이그레이션하는 경우, S3 액세스를 허용하도록 AWS Identity and Access Management(IAM) 역할을 설정합니다. 이 IAM 역할 생성에 관한 자세한 내용은 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성 섹션을 참조하십시오.

Amazon Redshift 엔드포인트가 전체 자동화를 지원하는 경우는 다음과 같습니다.

- 스키마 생성 및 데이터 형식 매핑
- 소스 데이터베이스 테이블 전체 로드
- 원본 테이블에 적용된 변경 사항을 증분식으로 로드
- 원본 테이블에 적용된 데이터 정의 언어(DDL)에서 스키마 변경 사항 적용
- 전체 로드와 변경 데이터 캡처(CDC) 프로세스 동기화.

AWS Database Migration Service은 전체 로드와 변경 처리 작업을 모두 지원합니다. AWS DMS는 소스 데이터베이스에서 데이터를 읽고 일련의 쉼표로 구분된 값(.csv) 파일을 생성합니다. 전체 로드 작업에서 AWS DMS는 테이블별로 파일을 생성합니다. 그런 다음 AWS DMS는 각 테이블의 테이블 파일을 Amazon S3에 있는 별도의 폴더에 복사합니다. 파일이 Amazon S3에 업로드되면, AWS DMS에서 COPY 명령을 전송하고 이 파일의 데이터가 Amazon Redshift에 복사됩니다. 변경 처리 작업에서 AWS DMS는 최종 변경 사항을 .csv 파일에 복사합니다. 그런 다음 AWS DMS는 최종 변경 파일을 Amazon S3에 업로드하고 이 데이터를 Amazon Redshift에 복사합니다.

Amazon Redshift를 AWS DMS의 대상으로 사용하여 작업하는 방법에 관한 자세한 내용은 다음 단원을 참조하십시오.

주제

- Amazon Redshift 데이터베이스를 AWS Database Migration Service의 대상으로 사용할 때의 사전 조건
- Redshift를 대상으로 사용하는 데 필요한 권한
- AWS Database Migration Service의 대상으로서 Amazon Redshift 사용 시 제한 사항
- AWS Database Migration Service에서 Amazon Redshift 데이터베이스를 대상으로 구성
- AWS Database Migration Service에서 대상으로서 Amazon Redshift와 함께 Enhanced VPC Routing 사용
- Amazon Redshift 대상 데이터 암호화를 위한 AWS KMS 키 생성 및 사용
- Amazon Redshift를 AWS DMS의 대상으로 사용 시 엔드포인트 설정
- 데이터 암호화 키와 Amazon S3 버킷을 중간 스토리지로 사용
- Amazon Redshift에 대한 멀티스레드 작업 설정
- Amazon Redshift에 대한 대상 데이터 형식
- 대상으로서 Amazon Redshift Serverless와 함께 AWS DMS를 사용

Amazon Redshift 데이터베이스를 AWS Database Migration Service의 대상으로 사용할 때의 사전 조건

다음 목록에는 데이터 마이그레이션에서 Amazon Redshift를 대상으로 사용할 때 필요한 사전 조건에 대한 설명이 나와 있습니다.

- AWS Management Console을 사용하여 Amazon Redshift 클러스터를 시작합니다. 암호, 사용자 이름, 데이터베이스 이름과 같은 AWS 계정과 사용자의 Amazon Redshift 클러스터에 대한 기본 정보를 기록합니다. Amazon Redshift 대상 엔드포인트를 생성할 때 이 값이 필요합니다.
- Amazon Redshift 클러스터는 복제 인스턴스와 동일한 AWS 계정과 동일한 AWS 리전에 있어야 합니다.
- AWS DMS 복제 인스턴스는 클러스터가 사용하는 Amazon Redshift 엔드포인트와의 네트워크 연결(호스트 이름과 포트)이 필요합니다.
- AWS DMS Amazon S3 버킷을 사용하여 Amazon Redshift 데이터베이스로 데이터를 전송합니다. AWS DMS에서 버킷을 생성하는 경우, 콘솔은 IAM 역할인 `dms-access-for-endpoint`를 사용합니다. AWS CLI 또는 DMS API를 사용하여 대상 데이터베이스인 Amazon Redshift를 통해 데이터베이스 마이그레이션을 생성하는 경우 이 IAM 역할을 생성해야 합니다. 이 역할 생성에 관한 자세한 내용은 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성 섹션을 참조하십시오.

- AWS DMS는 대상 Amazon Redshift 인스턴스에서 BLOB, CLOB 및 NCLOB를 IVARCHAR로 변환합니다. Amazon Redshift는 64KB보다 큰 VARCHAR 데이터 형식을 지원하지 않으므로 Amazon Redshift에는 기존 LOB를 저장할 수 없습니다.
- CDC 중에 Amazon Redshift 대상 테이블에 대한 변경 사항을 처리하려면 AWS DMS에 대한 대상 메타데이터 작업 설정 BatchApplyEnabled를 `true`로 설정합니다. 소스와 대상 테이블에 모두 기본 키가 필요합니다. 기본 키가 없으면 변경 사항이 문별로 적용됩니다. 또한 대상 지연 시간이 발생하고 클러스터 커밋 대기열에 영향을 주므로 CDC 중에 작업 성능이 저하될 수 있습니다.

Redshift를 대상으로 사용하는 데 필요한 권한

GRANT 명령을 사용하여 사용자 또는 사용자 그룹의 액세스 권한을 정의합니다. 권한은 테이블과 뷰에서 데이터를 읽거나 쓰고 테이블을 만들 수 있는 것과 같은 액세스 옵션을 포함합니다. Amazon Redshift에서 GRANT를 사용하는 방법에 관한 자세한 내용은 Amazon Redshift 데이터베이스 개발자 안내서의 GRANT를 참조하십시오.

다음은 테이블, 데이터베이스, 스키마, 함수, 프로시저 또는 Amazon Redshift 테이블 및 뷰의 언어 수준 권한에 대한 특정 권한을 부여하는 구문입니다.

```
GRANT { { SELECT | INSERT | UPDATE | DELETE | REFERENCES } [,...] | ALL
 [ PRIVILEGES ] }
    ON { [ TABLE ] table_name [, ...] | ALL TABLES IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]

GRANT { { CREATE | TEMPORARY | TEMP } [,...] | ALL [ PRIVILEGES ] }
    ON DATABASE db_name [, ...]
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]

GRANT { { CREATE | USAGE } [,...] | ALL [ PRIVILEGES ] }
    ON SCHEMA schema_name [, ...]
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON { FUNCTION function_name ( [ [ argname ] argtype [, ...] ] ) [, ...] | ALL
 FUNCTIONS IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON { PROCEDURE procedure_name ( [ [ argname ] argtype [, ...] ] ) [, ...] | ALL
 PROCEDURES IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]
```

```
GRANT USAGE
    ON LANGUAGE language_name [, ...]
    TO { username [ WITH GRANT OPTION ] | GROUP group_name | PUBLIC } [, ...]
```

다음은 Amazon Redshift 테이블 및 뷰에서 열 수준 권한에 대한 구문입니다.

```
GRANT { { SELECT | UPDATE } ( column_name [, ...] ) [, ...] | ALL [ PRIVILEGES ]
 ( column_name [,...] ) }
    ON { [ TABLE ] table_name [, ...] }
    TO { username | GROUP group_name | PUBLIC } [, ...]
```

다음은 지정된 역할을 가진 사용자 및 그룹에 부여된 ASSUMEROLE 권한에 대한 구문입니다.

```
GRANT ASSUMEROLE
    ON { 'iam_role' [, ...] | ALL }
    TO { username | GROUP group_name | PUBLIC } [, ...]
    FOR { ALL | COPY | UNLOAD } [, ...]
```

AWS Database Migration Service의 대상으로서 Amazon Redshift 사용 시 제한 사항

다음 제한 사항은 Amazon Redshift 데이터베이스를 대상으로 사용 시 적용됩니다.

- Amazon Redshift 대상의 중간 스토리지로 사용하는 S3 버킷의 버전 관리를 활성화하지 마십시오. S3 버전 관리가 필요한 경우, 수명 주기 정책을 사용하여 이전 버전을 적극적으로 삭제하십시오. 그렇지 않으면 S3 `list-object` 직접 호출 시간 초과로 인해 엔드포인트 테스트 연결 실패가 발생할 수 있습니다. S3 버킷의 수명 주기 정책을 생성하려면 스토리지 수명 주기 관리를 참조하세요. S3 객체의 버전을 삭제하려면 버전 관리 활성화 버킷에서 객체 버전 삭제를 참조하십시오.
- 다음 DDL은 지원되지 않습니다.

  ```
  ALTER TABLE table name MODIFY COLUMN column name data type;
  ```

- AWS DMS는 이름이 밑줄(_)로 시작하는 스키마로 변경 사항을 마이그레이션하거나 복제할 수 없습니다. 밑줄로 시작하는 이름의 스키마가 있다면 매핑 변환을 사용하여 대상에 있는 스키마의 이름을 바꿉니다.
- Amazon Redshift는 64KB를 초과하는 VARCHAR를 지원하지 않습니다. 기존 데이터베이스의 LOB는 Amazon Redshift에 저장할 수 없습니다.

- 기본 키 열 이름 중 하나에서 예약어를 사용하는 경우 다중 열 기본 키가 있는 테이블에 DELETE 문을 적용하는 것은 지원되지 않습니다. Amazon Redshift 예약어 목록을 보려면 여기로 이동합니다.
- 소스 시스템이 소스 테이블의 기본 키에 대해 UPDATE 작업을 수행하는 경우 성능 문제가 발생할 수 있습니다. 이러한 성능 문제는 대상에 변경 내용을 적용할 때 발생합니다. 이는 UPDATE(및 DELETE) 작업이 대상 행을 식별하는 기본 키 값에 의존하기 때문입니다. 소스 테이블의 기본 키를 업데이트하면 작업 로그에 다음과 같은 메시지가 포함됩니다.

```
Update on table 1 changes PK to a PK that was previously updated in the same bulk
 update.
```

- DMS는 Redshift 클러스터의 엔드포인트를 구성할 때 사용자 지정 DNS 이름을 지원하지 않으므로 Amazon에서 제공한 DNS 이름을 사용해야 합니다. Amazon Redshift 클러스터는 복제 인스턴스와 동일한 AWS 계정 및 리전에 있어야 하므로 사용자 지정 DNS 엔드포인트를 사용하면 검증이 실패합니다.
- Amazon Redshift에는 기본적으로 4시간의 유휴 세션 제한 시간이 있습니다. DMS 복제 작업 내에 활동이 없는 경우, Redshift는 4시간 후에 세션 연결을 끊습니다. DMS를 연결할 수 없고 다시 시작해야 할 수도 있기 때문에 오류가 발생할 수 있습니다. 해결 방법으로 DMS 복제 사용자의 세션 제한 시간을 4시간 이상으로 설정하십시오. 또는 Amazon Redshift 데이터베이스 개발자 안내서의 ALTER USER에 관한 설명을 참조하십시오.
- AWS DMS가 기본 키 또는 고유 키 없이 소스 테이블 데이터를 복제하는 경우, CDC 지연 시간이 길어져 허용 불가능한 수준의 성능이 발생할 수 있습니다.

AWS Database Migration Service에서 Amazon Redshift 데이터베이스를 대상으로 구성

AWS Database Migration Service은 Amazon Redshift 인스턴스를 사용할 수 있도록 구성해야 합니다. 다음 표에는 Amazon Redshift 엔드포인트에서 사용할 수 있는 구성 속성에 관한 설명이 나와 있습니다.

속성	Description
서버	사용 중인 Amazon Redshift 클러스터의 이름입니다.
포트	Amazon Redshift에 대한 포트 번호입니다. 기본값은 5439입니다.
사용자 이름	등록된 사용자의 Amazon Redshift 사용자 이름입니다.
비밀번호	사용자 이름 속성에 명명된 사용자의 암호입니다.

속성	Description
데이터베이스	사용 중인 Amazon Redshift 데이터 웨어하우스(서비스)의 이름입니다.

Amazon Redshift 엔드포인트에 추가 연결 문자열 속성을 추가하려면 `maxFileSize` 및 `fileTransferUploadStreams` 속성을 지정합니다. 이 속성에 관한 자세한 내용은 Amazon Redshift를 AWS DMS의 대상으로 사용 시 엔드포인트 설정 섹션을 참조하십시오.

AWS Database Migration Service에서 대상으로서 Amazon Redshift와 함께 Enhanced VPC Routing 사용

Amazon Redshift 대상에 Enhanced VPC Routing을 사용하는 경우, Amazon Redshift 클러스터와 데이터 리포지토리 간 모든 COPY 트래픽은 VPC를 통과합니다. Enhanced VPC Routing은 Amazon Redshift가 다른 리소스에 액세스하는 방식에 영향을 주기 때문에 VPC를 올바르게 구성하지 않았다면 COPY 명령이 실패할 수 있습니다.

AWS DMS는 COPY 명령을 사용하여 S3의 데이터를 Amazon Redshift 클러스터로 이동하기 때문에 이 동작의 영향을 받을 수 있습니다.

다음은 AWS DMS에서 Amazon Redshift 대상으로 데이터를 로드하는 데 필요한 단계입니다.

1. AWS DMS는 소스에서 복제 서버에 있는 .csv 파일로 데이터를 복사합니다.
2. AWS DMS는 AWS SDK를 사용하여 .csv 파일을 계정에 있는 S3 버킷으로 복사합니다.
3. 그런 다음 AWS DMS는 Amazon Redshift에서 COPY 명령을 사용하여 S3의 .csv 파일에서 Amazon Redshift의 해당 테이블로 데이터를 복사합니다.

Enhanced VPC Routing을 사용 설정하지 않은 경우, Amazon Redshift는 AWS 네트워크 내의 다른 서비스로 전송되는 트래픽을 포함하여 인터넷을 통해 트래픽을 라우팅합니다. 이 기능이 활성화되지 않을 경우, 네트워크 경로를 구성하지 않아도 됩니다. 이 기능이 활성화되면 특히 클러스터의 VPC와 사용자의 데이터 리소스 사이의 네트워크 경로를 생성해야 합니다. 필요한 구성에 관한 자세한 내용은 Amazon Redshift 설명서의 Enhanced VPC Routing을 참조하십시오.

Amazon Redshift 대상 데이터 암호화를 위한 AWS KMS 키 생성 및 사용

Amazon Redshift에 복사되기 전에 Amazon S3로 푸시되는 대상 데이터를 암호화할 수 있습니다. 이를 위해 사용자 지정 AWS KMS 키를 생성하고 사용할 수 있습니다. 생성한 키를 사용하여 Amazon Redshift 대상 엔드포인트를 생성할 때 다음 메커니즘 중 하나를 사용하여 대상 데이터를 암호화할 수 있습니다.

- AWS CLI를 사용하여 `create-endpoint` 명령을 실행할 때 다음 옵션 중 하나를 사용합니다.

```
--redshift-settings '{"EncryptionMode": "SSE_KMS", "ServerSideEncryptionKmsKeyId":
 "your-kms-key-ARN"}'
```

  여기서 *your-kms-key-ARN*은 KMS 키의 Amazon 리소스 이름(ARN)입니다. 자세한 내용은 데이터 암호화 키와 Amazon S3 버킷을 중간 스토리지로 사용 섹션을 참조하세요.

- 추가 연결 속성 `encryptionMode`를 값 `SSE_KMS`로 설정하고, 추가 연결 속성 `serverSideEncryptionKmsKeyId`는 KMS 키의 ARN으로 설정합니다. 자세한 내용은 Amazon Redshift를 AWS DMS의 대상으로 사용 시 엔드포인트 설정 섹션을 참조하세요.

KMS 키를 사용하여 Amazon Redshift 대상 데이터를 암호화하려면 Amazon Redshift 데이터에 액세스할 권한이 있는 AWS Identity and Access Management(IAM) 역할이 필요합니다. 그런 다음, 생성한 암호화 키에 연결된 정책(키 정책)에서 이 IAM 역할에 액세스합니다. 이 작업은 IAM 콘솔에서 다음을 생성하여 수행할 수 있습니다.

- AWS 관리형 정책이 있는 IAM 역할.
- 이 역할을 참조하는 키 정책이 있는 KMS 키.

다음 절차에서 그 방법을 설명합니다.

필요한 AWS 관리형 정책으로 IAM 역할을 생성하려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 여세요.
2. 탐색 창에서 역할을 선택합니다. 역할 페이지가 열립니다.
3. 역할 생성을 선택합니다. 역할 생성 페이지가 열립니다.
4. AWS 서비스를 신뢰할 수 있는 엔터티 유형으로 선택한 상태에서 해당 역할을 사용할 서비스로 DMS를 선택합니다.
5. 다음: 권한을 선택합니다. 권한 정책 연결 페이지가 나타납니다.
6. `AmazonDMSRedshiftS3Role` 정책을 찾아 선택합니다.
7. 다음: 태그를 선택합니다. 태그 추가 페이지가 나타납니다. 여기서 원하는 태그를 추가할 수 있습니다.
8. 다음: 검토를 선택하고 결과를 검토합니다.

9. 필요한 설정이 되었다면 역할의 이름(예: `DMS-Redshift-endpoint-access-role`)과 추가 설명을 입력한 다음 역할 생성을 선택합니다. 역할이 생성되었다는 메시지와 함께 역할 페이지가 열립니다.

이제 암호화를 위해 Amazon Redshift 리소스에 액세스할 수 있는 지정된 이름의 새 역할이 생성되었습니다(예: `DMS-Redshift-endpoint-access-role`).

IAM 역할을 참조하는 키 정책이 있는 AWS KMS 암호화 키를 생성하려면

> **Note**
>
> AWS DMS에서 AWS KMS 암호화 키를 사용하는 방법에 관한 자세한 내용은 암호화 키 설정 및 AWS KMS 권한 지정 단원을 참조하십시오.

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/kms에서 AWS Key Management Service(AWS KMS) 콘솔을 엽니다.
2. AWS 리전을 변경하려면 페이지의 오른쪽 상단 모서리에 있는 리전 선택기를 사용합니다.
3. 탐색 창에서 고객 관리형 키를 선택합니다.
4. 키 생성을 선택합니다. 키 구성 페이지가 열립니다.
5. 키 유형에 대해 대칭을 선택합니다.

   > **Note**
   >
   > Amazon Redshift와 같은 모든 AWS 서비스는 대칭 암호화 키로만 작동하므로 이 키를 생성하면 대칭 키만 생성할 수 있습니다.

6. [Advanced Options]를 선택합니다. 키 구성 요소 오리진에 KMS가 선택되어 있는지 확인하고 다음을 선택합니다. 레이블 추가 페이지가 열립니다.
7. 별칭 및 설명 생성에 키의 별칭(예: `DMS-Redshift-endpoint-encryption-key`)과 추가 설명을 입력합니다.
8. 태그에서 키 식별과 키 사용 추적에 도움이 되는 태그를 추가한 후 다음을 선택합니다. 키 관리 권한 정의 페이지가 열리고 선택할 수 있는 사용자 및 역할 목록이 표시됩니다.
9. 키를 관리할 사용자와 역할을 추가합니다. 키를 관리하는 데 필요한 권한이 이러한 사용자와 역할에 있는지 확인합니다.

10. 키 삭제에서 키 관리자가 키를 삭제할 수 있는지 여부를 선택한 후 다음을 선택합니다. 키 사용 권한 정의 페이지가 열리고 선택할 수 있는 추가 사용자 및 역할 목록이 표시됩니다.
11. 이 계정의 Amazon Redshift 대상에 대해 암호화 작업을 수행할 사용자를 선택합니다. 또한 이전에 역할에서 생성한 역할을 선택하여 Amazon Redshift 대상 객체 암호화를 위한 액세스를 활성화합니다(예: `DMS-Redshift-endpoint-access-role`).
12. 목록에 없는 다른 계정을 추가하여 동일한 액세스 권한을 부여하려면 다른 AWS 계정에 대해 다른 AWS 계정 추가를 선택한 후 다음을 선택합니다. 키 정책 검토 및 편집 페이지가 열리고 기존 JSON을 입력하여 검토하고 편집할 수 있는 키 정책 JSON이 표시됩니다. 여기서 키 정책이 이전 단계에서 선택한 역할 및 사용자(예: `Admin` 및 `User1`)를 참조하는 위치를 확인할 수 있습니다. 다음 예에 나온 것처럼 다양한 보안 주체(사용자 및 역할)에 허용되는 다양한 키 작업도 볼 수 있습니다.

```
{
  "Id": "key-consolepolicy-3",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:root"
        ]
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow access for Key Administrators",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:role/Admin"
        ]
      },
      "Action": [
        "kms:Create*",
        "kms:Describe*",
        "kms:Enable*",
        "kms:List*",
        "kms:Put*",
```

```
        "kms:Update*",
        "kms:Revoke*",
        "kms:Disable*",
        "kms:Get*",
        "kms:Delete*",
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ScheduleKeyDeletion",
        "kms:CancelKeyDeletion"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow use of the key",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:role/DMS-Redshift-endpoint-access-role",
          "arn:aws:iam::111122223333:role/Admin",
          "arn:aws:iam::111122223333:role/User1"
        ]
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow attachment of persistent resources",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:role/DMS-Redshift-endpoint-access-role",
          "arn:aws:iam::111122223333:role/Admin",
          "arn:aws:iam::111122223333:role/User1"
        ]
      },
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
```

```
        "kms:RevokeGrant"
      ],
      "Resource": "*",
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": true
        }
      }
    }
  ]
```

13. 마침을 클릭합니다. AWS KMS key이 생성되었다는 메시지와 함께 암호화 키 페이지가 열립니다.

이제 지정된 별칭의 새 KMS 키가 생성되었습니다(예: `DMS-Redshift-endpoint-encryption-key`). 이 키를 통해 AWS DMS는 Amazon Redshift 대상 데이터를 암호화할 수 있습니다.

Amazon Redshift를 AWS DMS의 대상으로 사용 시 엔드포인트 설정

추가 연결 속성을 사용하는 것과 마찬가지로 엔드포인트 설정을 사용하여 Amazon Redshift 대상을 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI의 `create-endpoint` 명령을 `--redshift-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Amazon Redshift를 대상으로 하여 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

명칭	Description
`MaxFileSize`	데이터를 Amazon Redshift로 전송할 때 사용되는 .csv 파일의 최대 크기(단위: KB)를 지정합니다. 기본값: 32768KB(32MB) 유효값: 1~1,048,576 예제: `--redshift-settings '{"MaxFileSize": 512}'`
`FileTransferUploadStreams`	단일 파일을 업로드할 때 사용되는 스레드 수를 지정합니다. 기본값: 10

명칭	Description
	유효값: 1~64 예제: `--redshift-settings '{"FileTransferUploadStreams": 20}'`
`Acceptanydate`	0000-00-00과 같이 잘못된 날짜 형식을 포함하여 모든 날짜 형식을 수락할지 여부를 지정합니다. 부울 값입니다. 기본값: false 유효값: true/false 예제: `--redshift-settings '{"Acceptanydate": true}'`
`Dateformat`	날짜 형식을 지정합니다. 문자열 입력이며 기본적으로 비어 있습니다. 기본 형식은 YYYY-MM-DD이지만 DD-MM-YYYY 등으로 변경할 수 있습니다. 날짜 또는 시간 값에 다른 형식을 사용할 경우 `auto` 인수와 함께 `Dateformat` 파라미터를 사용하십시오. `auto` 인수는 `Dateformat` 문자열을 사용할 때 지원되지 않는 몇 가지 형식을 인식합니다. `auto` 키워드는 대/소문자를 구별합니다. 기본값: 비어 있음 유효값: '*dateformat_string* ' 또는 `auto` 예제: `--redshift-settings '{"Dateformat": "auto"}'`

명칭	Description
`Timeformat`	시간 형식을 지정합니다. 문자열 입력이며 기본적으로 비어 있습니다. `auto` 인수는 `Timeformat` 문자열을 사용할 때 지원되지 않는 몇 가지 형식을 인식합니다. 날짜 및 시간 값에 사용되는 형식이 서로 다를 경우 `auto` 인수와 함께 `Timeformat` 파라미터를 사용하십시오. 기본값: 10 유효값: '*`Timeformat_string`* ' | 'auto' | 'epochsecs' | 'epochmillisecs' 예제: `--redshift-settings '{"Timeformat": "auto"}'`
`Emptyasnull`	AWS DMS가 비어 있는 CHAR 및 VARCHAR 필드를 null로 마이그레이션할지 지정합니다. true 값은 빈 CHAR 및 VARCHAR 필드를 null로 설정합니다. 기본값: false 유효값: true/false 예제: `--redshift-settings '{"Emptyasnull": true}'`
`TruncateColumns`	열의 데이터를 열 명세에 따라 적합한 수의 문자로 자릅니다. VARCHAR 또는 CHAR 데이터 형식의 열에만 적용되며, 행의 크기는 4MB 이하입니다. 기본값: false 유효값: true/false 예제: `--redshift-settings '{"TruncateColumns": true}'`

명칭	Description
RemoveQuotes	입력 데이터의 문자열에서 묶고 있는 인용 부호를 제거합니다. 인용 부호 안의 문자는 구분자를 포함하여 모두 유지됩니다. Amazon Redshift 대상의 따옴표 제거에 관한 자세한 내용은 Amazon Redshift 데이터베이스 개발자 안내서를 참조하세요. 기본값: false 유효값: true/false 예제: `--redshift-settings '{"RemoveQuotes": true}'`
TrimBlanks	VARCHAR 문자열에서 후행 공백 문자를 제거합니다. 이 파라미터는 VARCHAR 데이터 형식의 열에만 적용됩니다. 기본값: false 유효값: true/false 예제: `--redshift-settings '{"TrimBlanks": true}'`

명칭	Description
EncryptionMode	Amazon Redshift로 복사되기 전에 데이터를 S3로 푸시하는 데 사용할 서버 측 암호화 모드를 지정합니다. 유효 값은 SSE_S3(S3 서버 측 암호화) 또는 SSE_KMS(KMS 키 암호화)입니다. SSE_KMS를 선택하는 경우, ServerSideEncryptionKmsKeyId 파라미터를 암호화에 사용할 KMS 키의 Amazon 리소스 이름(ARN)으로 설정합니다. **Note** 또한 CLI modify-endpoint 명령을 사용하여 기존 엔드포인트의 EncryptionMode 설정 값을 SSE_KMS에서 SSE_S3로 변경할 수 있습니다. 그러나 EncryptionMode 값을 SSE_S3에서 SSE_KMS로 변경할 수 없습니다. 기본 값: SSE_S3 유효값: SSE_S3 또는 SSE_KMS 예제: `--redshift-settings '{"EncryptionMode": "SSE_S3"}'`
ServerSideEncryptionKmsKeyId	EncryptionMode 를 SSE_KMS로 설정하는 경우, 이 파라미터를 KMS 키의 ARN으로 설정합니다. 이 ARN은 계정을 위해 생성한 AWS KMS 키 목록에서 키 별칭을 선택하면 찾을 수 있습니다. 키를 생성할 때 특정 정책과 역할을 키에 연결해야 합니다. 자세한 내용은 Amazon Redshift 대상 데이터 암호화를 위한 AWS KMS 키 생성 및 사용 섹션을 참조하세요. 예제: `--redshift-settings '{"ServerSideEncryptionKmsKeyId":"arn:aws:kms:us-east-1:111122223333:key/11a1a1a1-aaaa-9999-abab-2bbbbbb222a2"}'`

명칭	Description
EnableParallelBatchInMemoryCSVFiles	이 EnableParallelBatchInMemoryCSVFiles 설정은 DMS가 메모리 대신 디스크에 쓰도록 함으로써 대규모 다중 스레드 전체 로드 작업의 성능을 향상시킵니다. 기본 값은 false입니다.
CompressCsvFiles	이 속성을 사용하면 마이그레이션 중에 Amazon Redshift 대상으로 전송되는 데이터를 압축할 수 있습니다. 기본값은 true이며 압축은 기본적으로 활성화되어 있습니다.

데이터 암호화 키와 Amazon S3 버킷을 중간 스토리지로 사용

Amazon Redshift 대상 엔드포인트 설정을 사용하여 다음을 구성할 수 있습니다.

- 사용자 지정 AWS KMS 데이터 암호화 키. 그런 다음, 이 키를 사용하여 Amazon Redshift에 복사되기 전에 Amazon S3로 푸시되는 데이터를 암호화할 수 있습니다.
- Amazon Redshift로 마이그레이션한 데이터의 중간 스토리지인 사용자 지정 S3 버킷.
- PostgreSQL 소스에서 boolean을 부울로 매핑합니다. 기본적으로 부울(BOOLEAN) 형식은 varchar(1)로 마이그레이션됩니다. 다음 예와 같이 Redshift 대상이 부울 형식을 부울로 마이그레이션하도록 MapBooleanAsBoolean을 지정할 수 있습니다.

  ```
  --redshift-settings '{"MapBooleanAsBoolean": true}'
  ```

  단, 이 설정이 적용되려면 소스 엔드포인트와 대상 엔드포인트 모두에서 이 설정을 지정해야 합니다.

데이터 암호화를 위한 KMS 키 설정

다음 예는 S3로 푸시되는 데이터를 암호화하도록 사용자 지정 KMS 키를 구성하는 방법을 보여 줍니다. 먼저 AWS CLI에서 다음 create-endpoint 호출을 할 수 있습니다.

```
aws dms create-endpoint --endpoint-identifier redshift-target-endpoint --endpoint-type
 target
--engine-name redshift --username your-username --password your-password
--server-name your-server-name --port 5439 --database-name your-db-name
--redshift-settings '{"EncryptionMode": "SSE_KMS",
```

```
"ServerSideEncryptionKmsKeyId": "arn:aws:kms:us-east-1:111122223333:key/24c3c5a1-
f34a-4519-a85b-2debbef226d1"}'
```

여기서 `--redshift-settings` 옵션에 의해 지정된 JSON 객체는 다음 두 가지 파라미터를 정의합니다. 하나는 값이 `SSE_KMS`인 `EncryptionMode` 파라미터입니다. 다른 하나는 값이 `arn:aws:kms:us-east-1:111122223333:key/24c3c5a1-f34a-4519-a85b-2debbef226d1`인 `ServerSideEncryptionKmsKeyId` 파라미터입니다. 이 값은 사용자 지정 KMS 키의 Amazon 리소스 이름(ARN)입니다.

기본적으로 S3 데이터 암호화는 S3 서버 측 암호화를 사용하여 수행됩니다. 이전 예의 Amazon Redshift 대상의 경우, 다음 예에서처럼 이것은 엔드포인트 설정을 지정하는 것에 해당합니다.

```
aws dms create-endpoint --endpoint-identifier redshift-target-endpoint --endpoint-type
 target
--engine-name redshift --username your-username --password your-password
--server-name your-server-name --port 5439 --database-name your-db-name
--redshift-settings '{"EncryptionMode": "SSE_S3"}'
```

자세한 내용은 Amazon Simple Storage Service 사용 설명서의 서버 측 암호화를 사용하여 데이터 보호를 참조하세요.

> **Note**
>
> 또한 CLI `modify-endpoint` 명령을 사용하여 기존 엔드포인트의 `EncryptionMode` 파라미터 값을 `SSE_KMS`에서 `SSE_S3`로 변경할 수 있습니다. 그러나 `EncryptionMode` 값을 `SSE_S3`에서 `SSE_KMS`로 변경할 수 없습니다.

Amazon S3 버킷 설정

데이터를 Amazon Redshift 대상 엔드포인트로 마이그레이션할 때 AWS DMS는 마이그레이션된 데이터를 Amazon Redshift로 복사하기 전에 중간 작업 저장 영역으로 기본 Amazon S3 버킷을 사용합니다. 예를 들어 AWS KMS 데이터 암호화 키가 있는 Amazon Redshift 대상 엔드포인트를 생성하는 예제에서는 이 기본 S3 버킷을 사용합니다(데이터 암호화를 위한 KMS 키 설정 참조).

대신 AWS CLI `create-endpoint` 명령의 `--redshift-settings` 옵션 값에 다음 파라미터를 포함하여 이 중간 스토리지에 대한 사용자 지정 S3 버킷을 지정할 수 있습니다.

- `BucketName` – S3 버킷 스토리지의 이름으로 지정하는 문자열. 서비스 액세스 역할이 `AmazonDMSRedshiftS3Role` 정책을 기반으로 하는 경우, 이 값 앞에는 `dms-` 접두사(예를 들면 `dms-my-bucket-name`)가 붙어야 합니다.
- `BucketFolder` – (선택 사항) 지정된 S3 버킷의 스토리지 폴더 이름으로 지정할 수 있는 문자열.
- `ServiceAccessRoleArn` – S3 버킷에 대한 관리 액세스를 허용하는 IAM 역할의 ARN. 일반적으로 `AmazonDMSRedshiftS3Role` 정책에 따라 이 역할을 만듭니다. 예를 들어 Amazon Redshift 대상 데이터 암호화를 위한 AWS KMS 키 생성 및 사용에서 필수 AWS 관리형 정책을 사용하여 IAM 역할을 생성하는 절차를 참조하십시오.

> **Note**
>
> `create-endpoint` 명령의 `--service-access-role-arn` 옵션을 사용하여 다른 IAM 역할의 ARN을 지정하면 이 IAM 역할 옵션이 우선 적용됩니다.

다음 예에서는 AWS CLI를 사용하는 다음 `create-endpoint` 호출에서 이러한 파라미터를 사용하여 사용자 지정 Amazon S3 버킷을 지정하는 방법을 보여줍니다.

```
aws dms create-endpoint --endpoint-identifier redshift-target-endpoint --endpoint-type
 target
--engine-name redshift --username your-username --password your-password
--server-name your-server-name --port 5439 --database-name your-db-name
--redshift-settings '{"ServiceAccessRoleArn": "your-service-access-ARN",
"BucketName": "your-bucket-name", "BucketFolder": "your-bucket-folder-name"}'
```

Amazon Redshift에 대한 멀티스레드 작업 설정

다중 스레드 작업 설정을 사용하여 Amazon Redshift 대상 엔드포인트에 대한 전체 로드 및 변경 데이터 캡처(CDC) 작업의 성능을 개선할 수 있습니다. 이를 통해 버퍼에 저장할 동시 스레드 수와 레코드 수를 지정할 수 있습니다.

Amazon Redshift에 대한 멀티스레드 전체 로드 작업 설정

전체 로드 성능을 높일 수 있도록 다음과 같은 `ParallelLoad*` 작업 설정을 사용할 수 있습니다.

- `ParallelLoadThreads` – 데이터 레코드를 Amazon Redshift 대상 엔드포인트로 푸시하기 위해 전체 로드 중에 DMS가 사용하는 동시 스레드의 수를 지정합니다. 기본값은 0이고 최대값은 32입니다. 자세한 내용은 전체 로드 작업 설정 섹션을 참조하세요.

`ParallelLoadThreads` 작업 설정을 사용할 때 `false`로 설정된 `enableParallelBatchInMemoryCSVFiles` 속성을 사용할 수 있습니다. 이 속성은 DMS가 메모리 대신 디스크에 쓰기를 수행함으로써 대규모 멀티스레드 전체 로드 작업의 성능을 향상시킵니다. 기본 값은 `true`입니다.

- `ParallelLoadBufferSize` – Redshift 대상과 함께 병렬 로드 스레드를 사용하는 동안 최대 데이터 레코드 요청을 지정합니다. 기본값은 100이고 최대값은 1,000입니다. ParallelLoadThreads > 1(1보다 큼)인 경우 이 옵션을 사용하는 것이 좋습니다.

> **Note**
>
> 전체 로드 중에 Amazon Redshift 대상 엔드포인트에서 `ParallelLoad*` 작업 설정의 사용에 대한 지원은 AWS DMS 버전 3.4.5 및 이후 버전에서 제공됩니다.
> `ReplaceInvalidChars` Redshift 엔드포인트 설정은 변경 데이터 캡처(CDC) 중이거나 또는 병렬 로드가 활성화된 전체 로드(FULL LOAD) 마이그레이션 작업 중에는 지원되지 않습니다. 병렬 로드가 활성화되지 않은 경우, 이 설정은 전체 로드(FULL LOAD) 마이그레이션에 대해 지원됩니다. 자세한 내용은 AWS Database Migration Service API 참조의 RedshiftSettings를 참조하세요.

Amazon Redshift에 대한 멀티스레드 CDC 작업 설정

CDC 성능을 높일 수 있도록 다음과 같은 `ParallelApply*` 작업 설정을 사용할 수 있습니다.

- `ParallelApplyThreads` – 데이터 레코드를 Amazon Redshift 대상 엔드포인트로 푸시하기 위해 CDC 로드 중에 AWS DMS가 사용하는 동시 스레드의 수를 지정합니다. 기본값은 0이고 최대값은 32입니다. 최소 권장값은 클러스터 내 조각 수와 같습니다.
- `ParallelApplyBufferSize` – Redshift 대상과 함께 병렬 적용 스레드를 사용하는 동안 최대 데이터 레코드 요청을 지정합니다. 기본값은 100이고 최대값은 1,000입니다. ParallelApplyThreads > 1(1보다 큼)인 경우 이 옵션을 사용하는 것이 좋습니다.

  대상으로서 Redshift의 이점을 최대한 활용하려면 `ParallelApplyBufferSize`의 값을 `ParallelApplyThreads` 값의 두 배 이상으로 설정하는 것이 좋습니다.

> **Note**
>
> CDC 중에 Amazon Redshift 대상 엔드포인트에서 `ParallelApply*` 작업 설정의 사용에 대한 지원은 AWS DMS 버전 3.4.3 및 이후 버전에서 제공됩니다.

적용되는 병렬화 수준은 전체 배치 크기와 데이터 전송에 사용된 최대 파일 크기 간의 상관 관계에 따라 달라집니다. 최대 파일 크기에 비해 배치 크기가 클 때 Redshift 대상과 함께 멀티스레드 CDC 작업 설정을 사용하면 이점이 있습니다. 예를 들어, 다음과 같은 엔드포인트 및 작업 설정 조합을 사용하여 최적의 성능을 발휘하도록 조정할 수 있습니다.

```
// Redshift endpoint setting

        MaxFileSize=250000;

// Task settings

        BatchApplyEnabled=true;
        BatchSplitSize =8000;
        BatchApplyTimeoutMax =1800;
        BatchApplyTimeoutMin =1800;
        ParallelApplyThreads=32;
        ParallelApplyBufferSize=100;
```

이전 예제의 설정을 사용하면 트랜잭션 워크로드가 많은 고객은 8000개의 레코드를 포함하는 배치 버퍼를 1800초 내에 채우고 최대 파일 크기가 250MB인 병렬 스레드 32개를 활용하여 이점을 얻을 수 있습니다.

자세한 내용은 변경 처리 튜닝 설정 섹션을 참조하세요.

> **Note**
>
> Redshift 클러스터로 계속 복제하는 동안 실행되는 DMS 쿼리는 실행 중인 다른 애플리케이션 쿼리와 동일한 WLM(워크로드 관리) 대기열을 공유할 수 있습니다. 따라서 Redshift 대상으로 계속 복제하는 동안 성능에 영향을 미치도록 WLM 속성을 적절하게 구성하는 것이 좋습니다. 예를 들어, 다른 병렬 ETL 쿼리가 실행 중인 경우 DMS는 더 느리게 실행되며 성능 향상은 사라집니다.

Amazon Redshift에 대한 대상 데이터 형식

AWS DMS의 Amazon Redshift 엔드포인트는 대부분의 Amazon Redshift 데이터 형식을 지원합니다. 다음 테이블에는 AWS DMS를 사용하고 기본적으로 AWS DMS 데이터 형식에서 매핑할 때 지원되는 Amazon Redshift 대상 데이터 형식이 나와 있습니다.

AWS DMS 데이터 형식에 관한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

AWS DMS 데이터 유형	Amazon Redshift 데이터 형식
BOOLEAN	BOOL
BYTES	VARCHAR(길이)
날짜	날짜
TIME	VARCHAR(20)
DATETIME	Redshift 대상 열 유형에 따라 크기가 0이상이고 6이하라면 다음 중 하나가 적용됩니다. TIMESTAMP (s) TIMESTAMPTZ (s) — 소스 타임스탬프에 (예를 들면 SQL Server 또는 Oracle 내) 영역 오프셋이 포함된 경우, 삽입/업데이트 시 UTC로 변환됩니다. 오프셋이 없으면 시간은 이미 UTC로 간주됩니다. 크기가 7 이상이고 9 이하인 경우: VARCHAR(37)
INT1	INT2
INT2	INT2
INT4	INT4
INT8	INT8

AWS DMS 데이터 유형	Amazon Redshift 데이터 형식
NUMERIC	크기가 0 이상이고 37 이하인 경우: NUMERIC(p,s) 크기가 38 이상이고 127 이하인 경우: VARCHAR(길이)
REAL4	FLOAT4
REAL8	FLOAT8
STRING	길이가 1~65,535인 경우, VARCHAR(바이트 단위의 길이)를 사용합니다. 길이가 65,536~2,147,483,647인 경우, VARCHAR(65535)를 사용합니다.
UINT1	INT2
UINT2	INT2
UINT4	INT4
UINT8	NUMERIC(20,0)
WSTRING	길이가 1~65,535인 경우, NVARCHAR(바이트 단위의 길이)를 사용합니다. 길이가 65,536~2,147,483,647인 경우, NVARCHAR(65535)를 사용합니다.
BLOB	VARCHAR(최대 LOB 크기 *2) 최대 LOB 크기는 31KB를 초과할 수 없습니다. Amazon Redshift는 64KB를 초과하는 VARCHAR를 지원하지 않습니다.

AWS DMS 데이터 유형	Amazon Redshift 데이터 형식
NCLOB	NVARCHAR(최대 LOB 크기) 최대 LOB 크기는 63KB를 초과할 수 없습니다. Amazon Redshift는 64KB를 초과하는 VARCHAR를 지원하지 않습니다.
CLOB	VARCHAR(최대 LOB 크기) 최대 LOB 크기는 63KB를 초과할 수 없습니다. Amazon Redshift는 64KB를 초과하는 VARCHAR를 지원하지 않습니다.

대상으로서 Amazon Redshift Serverless와 함께 AWS DMS를 사용

AWS DMS에서는 Amazon Redshift Serverless를 대상 엔드포인트로 사용할 수 있습니다. Amazon Redshift Serverless 사용에 관한 자세한 내용은 Amazon Redshift 관리 안내서의 Amazon Redshift Serverless를 참조하십시오.

이 주제에서는 Amazon Redshift Serverless 엔드포인트를 AWS DMS에서 사용하는 방법을 설명합니다.

> **Note**
>
> Amazon Redshift Serverless 엔드포인트를 생성할 때는 RedshiftSettings 엔드포인트 구성의 DatabaseName 필드에 Amazon Redshift 데이터 웨어하우스의 이름 또는 워크그룹 엔드포인트의 이름을 사용합니다. ServerName 필드에는 서버리스 클러스터의 Workgroup 페이지에 표시된 Endpoint 값(예: `default-workgroup.093291321484.us-east-1.redshift-serverless.amazonaws.com`)을 사용합니다. 엔드포인트 생성에 관한 자세한 내용은 소스 및 대상 엔드포인트 생성 단원을 참조하십시오. 워크그룹 엔드포인트에 관한 자세한 내용은 Amazon Redshift Serverless에 연결을 참조하십시오.

대상으로서 Amazon Redshift Serverless를 사용하는 신뢰 정책

Amazon Redshift Serverless를 대상 엔드포인트로 사용하는 경우, 다음과 같이 강조 표시된 섹션을 신뢰 정책에 추가해야 합니다. 이 신뢰 정책은 `dms-access-for-endpoint` 역할에 연결됩니다.

```
{
    "PolicyVersion": {
        "CreateDate": "2016-05-23T16:29:57Z",
        "VersionId": "v3",
        "Document": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": [
                    "ec2:CreateNetworkInterface",
                    "ec2:DescribeAvailabilityZones",
                    "ec2:DescribeInternetGateways",
                    "ec2:DescribeSecurityGroups",
                    "ec2:DescribeSubnets",
                    "ec2:DescribeVpcs",
                    "ec2:DeleteNetworkInterface",
                    "ec2:ModifyNetworkInterfaceAttribute"
                ],
                "Resource": "arn:aws:service:region:account:resourcetype/id",
                "Effect": "Allow"
            },
            {
                "Sid": "",
                "Effect": "Allow",
                "Principal": {
                    "Service": "redshift-serverless.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    },
    "IsDefaultVersion": true
    }
}
```

AWS DMS에서 신뢰 정책을 사용하는 방법에 관한 자세한 내용은 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성 단원을 참조하십시오.

대상으로서 Amazon Redshift Serverless 사용 시 제한 사항

Redshift Serverless를 대상으로 사용할 경우 다음과 같은 제한 사항이 있습니다.

- AWS DMS는 Amazon Redshift Serverless를 지원하는 리전에서만 Amazon Redshift Serverless를 엔드포인트로 지원합니다. Amazon Redshift Serverless를 지원하는 리전에 관한 자세한 내용은 AWS 일반 참조의 Afmazon Redshift 엔드포인트 및 할당량 항목에 있는 Redshift Serverless API를 참조하십시오.
- Enhanced VPC Routing을 사용할 때는 Redshift Serverless 또는 Redshift Provisioned 클러스터와 동일한 VPC에 Amazon S3 엔드포인트를 생성해야 합니다. 자세한 내용은 AWS Database Migration Service에서 대상으로서 Amazon Redshift와 함께 Enhanced VPC Routing 사용 섹션을 참조하세요.
- AWS DMS Serverless는 Amazon Redshift Serverless를 지원하지 않습니다.

SAP ASE 데이터베이스를 AWS Database Migration Service의 대상으로 사용

지원되는 모든 데이터베이스 소스에서 AWS DMS를 사용하여 SAP Adaptive Server Enterprise(ASE, 구 명칭은 Sybase) 데이터베이스로 데이터를 마이그레이션할 수 있습니다.

AWS DMS가 대상으로 지원하는 SAP ASE의 버전에 관한 자세한 내용은 대상: AWS DMS를 참조하십시오.

AWS Database Migration Service에서 SAP ASE 데이터베이스를 대상으로 사용할 때의 사전 조건

AWS DMS에서 SAP ASE 데이터베이스를 대상으로 사용하려면 먼저 다음 사전 조건에 부합하는지 확인해야 합니다.

- AWS DMS 사용자에게 SAP ASE 계정 액세스 권한을 부여합니다. 이 사용자는 SAP ASE 데이터베이스에서 읽기/쓰기 권한이 있어야 합니다.
- 경우에 따라 라틴 문자 외 다른 문자(예: 중국어)로 구성된 Amazon EC2 인스턴스에 설치된 SAP ASE 버전 15.7로 복제할 수 있습니다. 이 경우 AWS DMS를 사용하려면 대상 SAP ASE 시스템에 SAP ASE 15.7 SP121을 설치해야 합니다.

AWS DMS에서 SAP ASE 데이터베이스를 대상으로 사용 시 제한 사항

SAP ASE 데이터베이스를 AWS DMS의 대상으로 사용 시 다음과 같은 제한 사항이 적용됩니다.

- AWS DMS는 다음과 같은 데이터 형식의 필드가 포함된 테이블을 지원하지 않습니다. 이 데이터 형식을 사용하는 복제된 열은 null로 표시됩니다.
  - 사용자 정의 유형(UDT)

SAP ASE를 AWS DMS의 대상으로 사용 시 엔드포인트 설정

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 SAP ASE 대상 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI에서 `create-endpoint` 명령을 `--sybase-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

SAP ASE를 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

이름	설명
`Driver`	ASE 15.7 및 이후 버전에 TLS를 사용하려면 이 속성을 설정하십시오. 기본값: `Adaptive Server Enterprise` 예: `driver=Adaptive Server Enterprise 16.03.06;` 유효값: `Adaptive Server Enterprise 16.03.06`
`AdditionalConnectionProperties`	지정할 추가 ODBC 연결 파라미터입니다.

SAP ASE용 대상 데이터 형식

다음 테이블에는 AWS DMS를 사용하고 기본적으로 AWS DMS 데이터 형식에서 매핑할 때 지원되는 SAP ASE 데이터베이스 대상 데이터 형식이 나와 있습니다.

AWS DMS 데이터 형식에 관한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

AWS DMS 데이터 형식	SAP ASE 데이터 형식
BOOLEAN	BIT
BYTES	VARBINARY(길이)
DATE	DATE

AWS DMS 데이터 형식	SAP ASE 데이터 형식
TIME	TIME
TIMESTAMP	크기가 0 이상이고 6 이하인 경우: BIGDATETIME 크기가 7 이상이고 9 이하인 경우: VARCHAR(37)
INT1	TINYINT
INT2	SMALLINT
INT4	INTEGER
INT8	BIGINT
NUMERIC	NUMERIC(p,s)
REAL4	REAL
REAL8	DOUBLE PRECISION
STRING	VARCHAR(길이)
UINT1	TINYINT
UINT2	UNSIGNED SMALLINT
UINT4	UNSIGNED INTEGER
UINT8	UNSIGNED BIGINT
WSTRING	VARCHAR(길이)
BLOB	IMAGE
CLOB	UNITEXT
NCLOB	TEXT

Amazon S3를 AWS Database Migration Service의 대상으로 사용

지원되는 데이터베이스 소스에서 AWS DMS를 사용하여 Amazon S3에 데이터를 마이그레이션할 수 있습니다. AWS DMS 작업에서 Amazon S3를 대상으로 사용하는 경우, 전체 로드와 변경 데이터 캡처(CDC) 데이터는 기본적으로 쉼표로 구분된 값(.csv) 형식으로 작성됩니다. 보다 간소화된 스토리지 및 보다 빠른 쿼리 옵션을 위해 데이터를 Apache Parquet(.parquet) 형식에 작성할 수도 있습니다.

AWS DMS는 증분 16진수 카운터를 사용하여 전체 로드 중에 생성된 파일의 이름을 지정합니다(예: .csv 파일의 경우 LOAD00001.csv, LOAD00002..., LOAD00009, LOAD0000A 등). AWS DMS는 타임스탬프를 사용하여 CDC 파일의 이름을 지정합니다(예: 20141029-1134010000.csv). 레코드가 포함된 각 소스 테이블에서 AWS DMS는 지정된 대상 폴더 아래에 폴더를 생성합니다(소스 테이블이 비어 있지 않은 경우). AWS DMS는 모든 전체 로드 및 CDC 파일을 지정된 Amazon S3 버킷에 씁니다. MaxFileSize 엔드포인트 설정을 사용하여 AWS DMS가 생성하는 파일의 크기를 제어할 수 있습니다.

파라미터 `bucketFolder`에는 S3 버킷에 업로드되기 전에 .csv 또는 .parquet 파일이 저장되는 위치가 포함되어 있습니다. .csv 파일의 경우, 테이블 데이터는 S3 버킷에 다음 형식으로 저장되며 전체 로드 파일과 함께 표시됩니다.

```
database_schema_name/table_name/LOAD00000001.csv
database_schema_name/table_name/LOAD00000002.csv
...
database_schema_name/table_name/LOAD00000009.csv
database_schema_name/table_name/LOAD0000000A.csv
database_schema_name/table_name/LOAD0000000B.csv
...database_schema_name/table_name/LOAD0000000F.csv
database_schema_name/table_name/LOAD00000010.csv
...
```

추가 연결 속성을 사용하여 열 구분 기호, 행 구분 기호 등 여러 파라미터를 지정할 수 있습니다. 추가 연결 속성에 관한 자세한 내용은 이 섹션의 마지막에 나오는 Amazon S3를 AWS DMS의 대상으로 사용 시 엔드포인트 설정을 참조하십시오.

다음과 같이 `ExpectedBucketOwner` Amazon S3 엔드포인트 설정을 사용하여 버킷 소유자를 지정하고 스나이핑을 방지할 수 있습니다. 그런 다음, 연결을 테스트하거나 마이그레이션을 수행하도록 요청하면 S3는 지정된 파라미터와 비교하여 버킷 소유자의 계정 ID를 확인합니다.

```
--s3-settings='{"ExpectedBucketOwner": "AWS_Account_ID"}'
```

CDC 작업을 이용해 데이터 변경을 복제하기 위해 AWS DMS를 사용하는 경우, .csv 또는 .parquet 출력 파일의 첫 번째 열은 다음 .csv 파일이 보여 주는 것처럼 행 데이터가 어떻게 변경되었는지를 나타냅니다.

```
I,101,Smith,Bob,4-Jun-14,New York
U,101,Smith,Bob,8-Oct-15,Los Angeles
U,101,Smith,Bob,13-Mar-17,Dallas
D,101,Smith,Bob,13-Mar-17,Dallas
```

이 예의 경우, 소스 데이터베이스에 EMPLOYEE 테이블이 있다고 가정합니다. AWS DMS는 다음 이벤트에 응답하여 데이터를 .csv 또는 .parquet 파일에 작성합니다.

- 새 직원(Bob Smith, 직원 ID 101)이 2014년 6월 4일에 뉴욕 지사에 입사했습니다. .csv 또는 .parquet 파일에서 첫 번째 열의 I는 새 행이 소스 데이터베이스의 EMPLOYEE 테이블에 삽입(INSERT)되었음을 나타냅니다.
- 2015년 10월 8일에 Bob은 LA 지사로 전근갑니다. .csv 또는 .parquet 파일에서 U는 Bob의 새 지사 위치를 반영하여 EMPLOYEE 테이블의 해당 행이 업데이트(UPDATE)되었음을 나타냅니다. 이 줄 뒷부분에는 UPDATE 이후에 EMPLOYEE 테이블에서 이 행이 어떻게 표시되는지가 반영됩니다.
- 2017년 3월 13일에 Bob은 Dallas 지사로 다시 전근갑니다. .csv 또는 .parquet 파일에서 U는 이 행이 다시 업데이트(UPDATE)되었음을 나타냅니다. 이 줄 뒷부분에는 UPDATE 이후에 EMPLOYEE 테이블에서 이 행이 어떻게 표시되는지가 반영됩니다.
- Bob은 Dallas에서 얼마간 근무한 후 퇴사합니다. .csv 또는 .parquet 파일에서 D는 소스 테이블에서 이 행이 삭제(DELETE)되었음을 나타냅니다. 이 줄의 나머지에는 삭제 전에 EMPLOYEE 테이블에서 이 행이 어떻게 표시되었는지가 반영됩니다.

CDC의 경우, AWS DMS는 기본적으로 트랜잭션 순서에 관계없이 각 데이터베이스 테이블의 행 변경 내용을 저장합니다. 트랜잭션 순서에 따라 행 변경 내용을 CDC 파일에 저장하려면 S3 엔드포인트 설정을 사용하여 이 경로는 물론, CDC 트랜잭션 파일을 S3 대상에 저장할 폴더 경로도 지정해야 합니다. 자세한 내용은 S3 대상의 트랜잭션 순서를 포함한 변경 데이터 캡처(CDC). 섹션을 참조하세요.

데이터 복제 작업 중에 Amazon S3 대상에 대한 쓰기 빈도를 제어하기 위해 cdcMaxBatchInterval 및 cdcMinFileSize 추가 연결 속성을 구성할 수 있습니다. 따라서 추가 오버헤드 작업 없이 데이터를 분석할 때 성능이 향상될 수 있습니다. 자세한 내용은 Amazon S3를 AWS DMS의 대상으로 사용 시 엔드포인트 설정 단원을 참조하십시오.

주제

- Amazon S3를 대상으로 사용하기 위한 사전 조건
- Amazon S3를 대상으로 사용할 때 적용되는 제한 사항
- 보안
- Apache Parquet를 사용하여 Amazon S3 객체 저장
- Amazon S3 객체 태그 지정
- AWS KMS 키를 생성하여 Amazon S3 대상 객체 암호화
- 날짜 기반 폴더 파티셔닝 사용
- Amazon S3를 AWS DMS의 대상으로 사용할 때 분할된 소스의 병렬 로드
- Amazon S3를 AWS DMS의 대상으로 사용 시 엔드포인트 설정
- AWS DMS의 Amazon S3 대상과 함께 AWS Glue Data Catalog을 사용
- Amazon S3 대상에서 데이터 암호화, parquet 파일 및 CDC 사용
- 마이그레이션된 S3 데이터에 소스 DB 작업 표시
- S3 Parquet의 대상 데이터 유형

Amazon S3를 대상으로 사용하기 위한 사전 조건

Amazon S3를 대상으로 사용하기 전에 다음 사항이 참인지 확인하십시오.

- 대상으로 사용 중인 S3 버킷이 데이터 마이그레이션에 사용 중인 DMS 복제 인스턴스와 동일한 AWS 리전에 있습니다.
- 마이그레이션에 사용하는 AWS 계정에는 대상으로 사용 중인 S3 버킷에 대한 쓰기 및 삭제 액세스 권한이 있는 IAM 역할이 있습니다.
- 이 역할에는 태그 지정 액세스 권한이 있으므로 대상 버킷에 작성되는 S3 객체에 태그를 지정할 수 있습니다.
- IAM 역할에는 DMS(dms.amazonaws.com)가 신뢰할 수 있는 엔터티로 추가되었습니다.

이 계정의 액세스 권한을 설정하려면 마이그레이션 작업 생성에 사용되는 사용자 계정에 할당된 역할에 다음 권한 집합이 있어야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:DeleteObject",
                "s3:PutObjectTagging"
            ],
            "Resource": [
                "arn:aws:s3:::buckettest2/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::buckettest2"
            ]
        }
    ]
}
```

S3를 대상으로 하는 검증을 사용하기 위한 사전 요구 사항은 S3 대상 검증 사전 조건를 참조하십시오.

Amazon S3를 대상으로 사용할 때 적용되는 제한 사항

Amazon S3를 대상으로 사용할 때 적용되는 제한 사항은 다음과 같습니다.

- S3의 버전 관리를 활성화하지 마십시오. S3 버전 관리가 필요한 경우, 수명 주기 정책을 사용하여 이전 버전을 적극적으로 삭제하세요. 그렇지 않으면 S3 `list-object` 직접 호출 시간 초과로 인해 엔드포인트 테스트 연결 실패가 발생할 수 있습니다. S3 버킷의 수명 주기 정책을 생성하려면 스토리지 수명 주기 관리를 참조하세요. S3 객체의 버전을 삭제하려면 버전 관리 활성화 버킷에서 객체 버전 삭제를 참조하세요.
- VPC 지원(게이트웨이 VPC) S3 버킷은 버전 3.4.7 및 이후 버전에서 지원됩니다.
- 변경 데이터 캡처(CDC)에는 테이블 자르기(Truncate Table), 테이블 삭제(Drop Table), 테이블 생성(Create Table), 테이블 이름 바꾸기(Rename Table), 열 추가(Add Column), 열 삭제(Drop Column), 열 이름 바꾸기(Rename Column), 열 데이터 유형 변경(Change Column Data Type)과 같은 데이터 정의 언어(DDL) 명령이 지원됩니다. 소스 데이터베이스에서 열을 추가, 삭제 또는 변경할 때 대상 S3 버킷에는 ALTER 문이 기록되지 않으며 AWS DMS는 이전에 생성된 레코드를 새 구조에 맞게 변경하지 않습니다. 변경 후 AWS DMS는 새 테이블 구조를 사용하여 새 레코드를 생성합니다.

> **Note**
>
> DDL 잘라내기(truncate DDL) 작업을 하면 S3 버킷에서 모든 파일 및 해당 테이블 폴더가 제거됩니다. 작업 설정을 사용하여 해당 동작을 비활성화하고 변경 데이터 캡처(CDC) 중에 DMS가 DDL 동작을 처리하는 방식을 구성할 수 있습니다. 자세한 내용은 변경 처리 DDL을 다루기 위한 작업 설정 섹션을 참조하세요.

- 전체 LOB 모드는 지원되지 않습니다.
- 전체 로드 중에 소스 테이블의 구조 변경은 지원되지 않습니다. 전체 로드 중에 데이터 변경은 지원됩니다.
- 동일한 소스 테이블에서 동일한 대상 S3 엔드포인트 버킷으로 데이터를 복제하는 작업을 여러 번 수행하면 동일한 파일에 해당 작업을 쓰게 됩니다. 동일한 테이블의 데이터 소스인 경우 서로 다른 대상 엔드포인트(버킷)를 지정하는 것이 좋습니다.
- `BatchApply`는 S3 엔드포인트에는 지원되지 않습니다. S3 대상에 대해 Batch Apply(예: `BatchApplyEnabled` 대상 메타데이터 작업 설정)를 사용하면 데이터 손실이 발생할 수 있습니다.
- `DatePartitionEnabled` 또는 `addColumnName`를 `PreserveTransactions` 또는 `CdcPath`와 함께 사용할 수 없습니다.
- AWS DMS는 변환 규칙을 사용하여 여러 소스 테이블의 이름을 동일한 대상 폴더로 바꾸는 것을 지원하지 않습니다.
- 전체 로드 단계에서 소스 테이블에 많은 양의 쓰기가 있는 경우 DMS는 S3 버킷이나 캐시된 변경 사항에 중복 레코드를 쓸 수 있습니다.
- `TargetTablePrepMode`가 `DO_NOTHING`인 작업을 구성하면 전체 로드 단계에서 작업이 갑자기 중지되었다가 재개되는 경우 DMS가 S3 버킷에 중복 레코드를 쓸 수 있습니다.
- `PreserveTransactions`를 `true`로 설정하여 대상 엔드포인트를 구성하는 경우 테이블을 다시 로드해도 이전에 생성된 CDC 파일은 지워지지 않습니다. 자세한 내용은 S3 대상의 트랜잭션 순서를 포함한 변경 데이터 캡처(CDC). 섹션을 참조하세요.

S3를 대상으로 하는 검증을 사용하기 위한 사전 요구 사항은 S3 대상 검증 사용에 대한 제한 사항를 참조하십시오.

보안

Amazon S3를 대상으로 사용하려면 마이그레이션에 사용되는 계정에는 대상으로 사용되는 Amazon S3 버킷에 대한 쓰기 및 삭제 액세스 권한이 있어야 합니다. Amazon S3에 액세스하는 데 필요한 권한을 가진 IAM 역할의 Amazon 리소스 이름(ARN)을 지정합니다.

AWS DMS는 Amazon S3에 대해 사전 정의된 권한 부여 집합 즉, 미리 제공된 ACL(액세스 제어 목록)을 지원합니다. 각각의 미리 제공된 ACL에는 Amazon S3 버킷에 대한 권한을 설정하는 데 사용할 수 있는 피부여자 및 권한 집합이 있습니다. S3 대상 엔드포인트에 대한 연결 문자열 속성에서 `cannedAclForObjects`를 사용하여 미리 제공된 ACL을 지정할 수 있습니다. 추가 연결 속성 `cannedAclForObjects` 사용에 관한 자세한 내용은 Amazon S3를 AWS DMS의 대상으로 사용 시 엔드포인트 설정 단원을 참조하십시오. Amazon S3에서 미리 제공된 ACL에 관한 자세한 내용은 미리 제공된 ACL 단원을 참조하십시오.

마이그레이션에 사용하는 IAM 역할이 `s3:PutObjectAcl` API 작업을 수행할 수 있어야 합니다.

Apache Parquet를 사용하여 Amazon S3 객체 저장

쉼표로 구분된 값(.csv) 형식이 Amazon S3 대상 객체의 기본 스토리지 형식입니다. 보다 간소화된 스토리지와 보다 빠른 쿼리를 위해 Apache Parquet(.parquet)를 스토리지 형식으로 대신 사용할 수 있습니다.

Apache Parquet는 원래 하둡용으로 개발된 오픈 소스 파일 스토리지 형식입니다. Apache Parquet에 관한 자세한 내용은 https://parquet.apache.org/를 참조하십시오.

.parquet를 마이그레이션된 S3 대상 객체의 스토리지 형식으로 설정하기 위해 다음 메커니즘을 사용할 수 있습니다.

- AWS CLI 또는 API for AWS DMS를 사용하여 엔드포인트를 생성할 때 JSON 객체의 파라미터로 제공하는 엔드포인트 설정. 자세한 내용은 Amazon S3 대상에서 데이터 암호화, parquet 파일 및 CDC 사용 섹션을 참조하세요.
- 엔드포인트를 생성할 때 세미콜론으로 구분된 목록으로 제공하는 추가 연결 속성. 자세한 내용은 Amazon S3를 AWS DMS의 대상으로 사용 시 엔드포인트 설정 섹션을 참조하세요.

Amazon S3 객체 태그 지정

작업-테이블 매핑 규칙의 일환으로 적절한 JSON 객체m를 지정하여 복제 인스턴스가 생성하는 Amazon S3 객체에 태그를 지정할 수 있습니다. 유효한 태그 이름을 포함하여 S3 객체 태그 지정의 요구 사항 및 옵션에 관한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 객체 태그 지정을 참조하십시오. JSON을 사용한 테이블 매핑에 관한 자세한 내용은 JSON을 사용하여 테이블 선택 및 변환 지정 단원을 참조하십시오.

`selection` 규칙 유형의 JSON 객체를 하나 이상 사용하여 지정된 테이블 및 스키마를 위해 생성된 S3 객체에 태그를 지정합니다. 그런 다음, `add-tag` 작업을 사용하여 `post-processing` 규칙 유형의 JSON 객체 중 하나 이상에서 이 `selection` 객체(또는 객체들)를 따릅니다. 이러한 사후 처리 규칙

은 태그 지정하려는 S3 객체를 식별하고 이러한 S3 객체에 추가하려는 태그의 이름과 값을 지정합니다.

다음 테이블에서 post-processing 규칙 유형의 JSON 객체에 지정할 파라미터를 찾을 수 있습니다.

파라미터	가능한 값	Description
rule-type	post-processing	생성된 대상 객체에 사후 처리 작업을 적용하는 값입니다. 하나 이상의 사후 처리 규칙을 지정하여 선택한 S3 객체에 태그를 지정할 수 있습니다.
rule-id	숫자 값.	규칙을 식별하기 위한 고유한 숫자 값입니다.
rule-name	영숫자 값입니다.	규칙을 식별하기 위한 고유한 이름입니다.
rule-action	add-tag	S3 객체에 적용할 사후 처리 작업입니다. add-tag 작업의 단일 JSON 사후 처리 객체를 사용하여 하나 이상의 태그를 추가할 수 있습니다.
object-locator	schema-name – 테이블 스키마의 이름입니다. table-name – 테이블의 이름입니다.	규칙이 적용되는 각 스키마와 테이블의 이름입니다. 각 object-locator 파라미터 값의 전부 또는 일부에 '%'(퍼센트 기호)를 와일드카드로 사용할 수 있습니다. 따라서 다음 항목들을 일치시킬 수 있습니다. • 단일 스키마의 단일 테이블 • 일부 또는 전체 스키마의 단일 테이블 • 단일 스키마의 일부 또는 전체 테이블 • 일부 또는 전체 스키마의 일부 또는 전체 테이블

파라미터	가능한 값	Description
`tag-set`	`key` – 단일 태그의 유효한 이름입니다. `value` – 이 태그의 유효한 JSON 값입니다.	지정된 `object-locator` 와 일치하는 생성된 각 S3 객체에 설정하려는 하나 이상의 태그 이름과 값입니다. 하나의 `tag-set` 파라미터 객체에서 최대 10개의 키-값 페어를 지정할 수 있습니다. S3 객체 태그 지정에 관한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 객체 태그 지정을 참조하세요. `${dyn-value }`를 사용하여 태그의 `key` 및 `value` 파라미터 값 전부 또는 일부에 동적 값을 지정할 수도 있습니다. 여기서 `${dyn-value }`는 `${schema-name}` 또는 `${table-name}` 일 수 있습니다. 따라서 현재 선택된 스키마 또는 테이블의 이름을 파라미터 값의 전체 또는 일부로 삽입할 수 있습니다. **Note** **Important** `key` 파라미터의 동적 값을 삽입하는 경우, 사용 방법에 따라 S3 객체의 중복된 이름이 있는 태그를 생성할 수 있습니다. 이 경우, 중복 태그 설정 중 하나만 객체에 추가됩니다.

여러 post-processing 규칙 유형을 지정하여 일련의 선택된 S3 객체에 태그를 지정하는 경우, 각 S3 객체에는 하나의 사후 처리 규칙으로부터 단 하나의 tag-set 객체만 사용하여 태그가 지정됩니다. 주어진 S3 객체에 태그를 지정하는 데 사용되는 특정 태그 세트는 연결된 객체 로케이터가 해당 S3 객체와 가장 잘 일치하는 사후 처리 규칙의 태그 세트입니다.

예를 들어, 2개의 사후 처리 규칙이 동일한 S3 객체를 식별한다고 가정해 보겠습니다. 또한 한 규칙의 객체 로케이터가 와일드카드를 사용하며 다른 한 규칙의 객체 로케이터는 정확한 일치를 사용하여 S3 객체를 (와일드카드 없이) 식별하는 경우를 가정해 보겠습니다. 이 경우, 정확한 일치를 사용하는 사후 처리 규칙에 연결된 태그 세트가 S3 객체의 태그 지정에 사용됩니다. 여러 사후 처리 규칙이 주어진 S3 객체를 똑같이 잘 일치시키는 경우, 이러한 첫 번째 사후 처리 규칙에 연결된 태그 세트가 객체 태그 지정에 사용됩니다.

Example 단일 테이블 및 스키마를 위해 생성된 S3 객체에 정적 태그 추가

다음 선택 및 사후 처리 규칙은 3개의 태그(tag_1, tag_2, tag_3 및 해당 정적 값 value_1, value_2, value_3)를 생성된 S3 객체에 추가합니다. 이 S3 객체는 스키마 이름이 aat2인 STOCK이라는 이름의 소스에 있는 단일 테이블에 해당됩니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "5",
            "rule-name": "5",
            "object-locator": {
                "schema-name": "aat2",
                "table-name": "STOCK"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "post-processing",
            "rule-id": "41",
            "rule-name": "41",
            "rule-action": "add-tag",
            "object-locator": {
                "schema-name": "aat2",
                "table-name": "STOCK"
            },
            "tag-set": [
              {
                "key": "tag_1",
```

```
                "value": "value_1"
              },
              {
                "key": "tag_2",
                "value": "value_2"
              },
              {
                "key": "tag_3",
                "value": "value_3"
              }
           ]
        }
    ]
}
```

Example 여러 테이블 및 스키마를 위해 생성된 S3 객체에 정적 및 동적 태그 추가

다음 예에는 1개의 선택 규칙과 2개의 사후 처리 규칙이 있습니다. 여기서 소스의 입력에는 모든 테이블과 테이블의 모든 스키마가 포함됩니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "post-processing",
            "rule-id": "21",
            "rule-name": "21",
            "rule-action": "add-tag",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%",
            },
            "tag-set": [
              {
```

```
                "key": "dw-schema-name",
                "value":"${schema-name}"
              },
              {
                "key": "dw-schema-table",
                "value": "my_prefix_${table-name}"
              }
            ]
        },
        {
            "rule-type": "post-processing",
            "rule-id": "41",
            "rule-name": "41",
            "rule-action": "add-tag",
            "object-locator": {
                "schema-name": "aat",
                "table-name": "ITEM",
            },
            "tag-set": [
              {
                "key": "tag_1",
                "value": "value_1"
              },
              {
                "key": "tag_2",
                "value": "value_2"
              }           ]
        }
    ]
}
```

첫 번째 사후 처리 규칙은 2개의 태그(dw-schema-name과 dw-schema-table) 및 해당 동적 값(${schema-name}과 my_prefix_${table-name})을 대상에 생성된 거의 모든 S3 객체에 추가합니다. 예외는 두 번째 사후 처리 규칙으로 식별되고 태그 지정된 S3 객체입니다. 따라서 와일드카드 객체 로케이터에 의해 식별된 각각의 S3 객체는 소스에서 상응하는 스키마와 테이블을 식별하는 태그를 사용하여 생성됩니다.

두 번째 사후 처리 규칙은 tag_1 및 tag_2와 해당 정적 값 value_1 및 value_2를 정확한 일치 객체 로케이터에 의해 식별되는 생성된 S3 객체에 추가합니다. 생성된 이 S3 객체는 따라서 스키마 이름이 aat인 ITEM이라는 이름의 소스에 있는 단일 테이블에 상응합니다. 정확한 일치로 인해 이러한 태그는 와일드카드만으로 S3 객체를 일치시키는 첫 번째 사후 처리 규칙에서 추가된 이 객체의 모든 태그를 대체합니다.

Example S3 객체에 동적 태그 이름과 값을 모두 추가

다음 예에는 2개의 선택 규칙과 1개의 사후 처리 규칙이 있습니다. 여기서 소스의 입력에는 `retail` 또는 `wholesale` 스키마의 `ITEM` 테이블만 포함됩니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "retail",
                "table-name": "ITEM"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "wholesale",
                "table-name": "ITEM"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "post-processing",
            "rule-id": "21",
            "rule-name": "21",
            "rule-action": "add-tag",
            "object-locator": {
                "schema-name": "%",
                "table-name": "ITEM",
            },
            "tag-set": [
              {
                "key": "dw-schema-name",
                "value":"${schema-name}"
              },
              {
                "key": "dw-schema-table",
                "value": "my_prefix_ITEM"
```

```
            },
            {
              "key": "${schema-name}_ITEM_tag_1",
              "value": "value_1"
            },
            {
              "key": "${schema-name}_ITEM_tag_2",
              "value": "value_2"
            }
          ]
    ]
}
```

사후 처리 규칙의 태그 세트는 2개의 태그(dw-schema-name 및 dw-schema-table)를 대상의 ITEM 테이블을 위해 생성된 모든 S3 객체에 추가합니다. 첫 번째 태그는 동적 값 "${schema-name}"를 갖고 두 번째 태그는 정적 값 "my_prefix_ITEM"을 갖습니다. 따라서 각각의 대상 S3 객체는 소스에서 상응하는 스키마와 테이블을 식별하는 태그를 사용하여 생성됩니다.

또한 태그 세트는 동적 이름이 있는 2개의 태그(${schema-name}_ITEM_tag_1 및 "${schema-name}_ITEM_tag_2")를 추가합니다. 이러한 태그는 상응하는 정적 값 value_1과 value_2를 갖습니다. 따라서 이러한 태그는 각각 현재의 스키마 retail 또는 wholesale에 대해 이름이 지정됩니다. 각 객체가 단일한 고유 스키마 이름을 위해 생성되기 때문에 이 객체에는 중복된 동적 태그 이름을 생성할 수 없습니다. 이 스키마 이름은 다른 방식으로 고유한 태그 이름을 생성하는 데 사용됩니다.

AWS KMS 키를 생성하여 Amazon S3 대상 객체 암호화

사용자 지정 AWS KMS 키를 생성하고 사용하여 Amazon S3 대상 객체를 암호화할 수 있습니다. KMS 키를 생성한 후 S3 대상 엔드포인트를 만들 때 다음 방법 중 하나로 이 키를 사용하여 객체를 암호화할 수 있습니다.

- AWS CLI를 사용하여 create-endpoint 명령을 실행할 때 (기본 .csv 파일 스토리지 형식의) S3 대상 객체에 다음 옵션을 사용하십시오.

```
--s3-settings '{"ServiceAccessRoleArn": "your-service-access-ARN",
"CsvRowDelimiter": "\n", "CsvDelimiter": ",", "BucketFolder": "your-bucket-folder",
"BucketName": "your-bucket-name", "EncryptionMode": "SSE_KMS",
"ServerSideEncryptionKmsKeyId": "your-KMS-key-ARN"}'
```

  여기서 *your-KMS-key-ARN*은 KMS 키의 Amazon 리소스 이름(ARN)입니다. 자세한 내용은 Amazon S3 대상에서 데이터 암호화, parquet 파일 및 CDC 사용 섹션을 참조하세요.

- 추가 연결 속성 encryptionMode를 값 SSE_KMS로 설정하고, 추가 연결 속성 serverSideEncryptionKmsKeyId는 KMS 키의 ARN으로 설정합니다. 자세한 내용은 Amazon S3를 AWS DMS의 대상으로 사용 시 엔드포인트 설정 섹션을 참조하세요.

KMS 키를 사용하여 Amazon S3 대상 객체를 암호화하려면 S3 버킷에 액세스할 권한이 있는 IAM 역할이 필요합니다. 그러면 생성한 암호화 키에 연결된 정책(키 정책)에서 이 IAM 역할에 액세스합니다. 이 작업은 IAM 콘솔에서 다음을 생성하여 수행할 수 있습니다.

- Amazon S3 버킷에 액세스할 권한이 있는 정책.
- 이 정책이 있는 IAM 역할.
- 이 역할을 참조하는 키 정책이 있는 KMS 키 암호화 키.

다음 절차에서 그 방법을 설명합니다.

Amazon S3 버킷에 액세스할 권한이 있는 IAM 정책을 생성하려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 여세요.
2. 탐색 창에서 정책을 선택합니다. 정책 페이지가 열립니다.
3. 정책 생성을 선택합니다. 정책 생성 페이지가 열립니다.
4. 서비스를 선택하고 S3를 선택합니다. 작업 권한 목록이 나타납니다.
5. 모두 확장을 선택하여 목록을 확장하고 최소한 다음 권한을 선택합니다.

   - ListBucket
   - PutObject
   - DeleteObject

   필요한 그 밖의 권한을 선택한 다음 모두 축소를 선택하여 목록을 축소합니다.
6. 리소스를 선택하여 액세스하려는 리소스를 지정합니다. 최소한 모든 리소스를 선택하여 일반적인 S3 리소스 액세스 권한을 제공합니다.
7. 필요한 그 밖의 조건 또는 권한을 추가한 다음 정책 검토를 선택합니다. 정책 검토 페이지에서 결과를 확인합니다.
8. 필요한 설정이 되었다면 정책의 이름(예: DMS-S3-endpoint-access)과 추가 설명을 입력한 다음 정책 생성을 선택합니다. 정책이 생성되었다는 메시지와 함께 정책 페이지가 열립니다.

9. 정책 목록에서 정책 이름을 검색하고 선택합니다. 다음과 비슷한 정책의 JSON이 표시되는 요약 페이지가 나타납니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:ListBucket",
                "s3:DeleteObject"
            ],
            "Resource": "*"
        }
    ]
}
```

이제 암호화를 위해 Amazon S3 리소스에 액세스할 수 있는 지정된 이름의 새 정책이 생성되었습니다(예: `DMS-S3-endpoint-access`).

이 정책을 사용하여 IAM 역할을 생성하려면

1. IAM 콘솔의 탐색 창에서 역할을 선택합니다. 역할 세부 정보 페이지가 열립니다.
2. 역할 생성을 선택합니다. 역할 생성 페이지가 열립니다.
3. AWS 서비스를 신뢰할 수 있는 엔터티 유형으로 선택한 상태에서 해당 IAM 역할을 사용할 서비스로 DMS를 선택합니다.
4. 다음: 권한을 선택합니다. 권한 정책 연결 뷰는 역할 생성 페이지에 표시됩니다.
5. 이전 절차(`DMS-S3-endpoint-access`)에서 생성한 IAM 역할에 대한 IAM 정책을 찾아 선택합니다.
6. 다음: 태그를 선택합니다. 태그 추가 뷰는 역할 생성 페이지에 나타납니다. 여기서 원하는 태그를 추가할 수 있습니다.
7. 다음: 검토를 선택합니다. 검토 뷰는 역할 생성 페이지에 나타납니다. 여기서 결과를 확인할 수 있습니다.

8. 필요한 설정이 되었다면 역할의 이름(필수, 예를 들어 `DMS-S3-endpoint-access-role`)과 추가 설명을 입력한 다음, 역할 생성을 선택합니다. 역할이 생성되었다는 메시지와 함께 역할 세부 정보 페이지가 열립니다.

이제 암호화를 위해 Amazon S3 리소스에 액세스할 수 있는 지정된 이름의 새 역할이 생성되었습니다(예: `DMS-S3-endpoint-access-role`).

IAM 역할을 참조하는 키 정책이 있는 KMS 키 암호화 키를 생성하려면

> **Note**
>
> AWS DMS에서 AWS KMS 암호화 키를 사용하는 방법에 관한 자세한 내용은 암호화 키 설정 및 AWS KMS 권한 지정 단원을 참조하십시오.

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/kms에서 AWS Key Management Service(AWS KMS) 콘솔을 엽니다.
2. AWS 리전을 변경하려면 페이지의 오른쪽 상단 모서리에 있는 리전 선택기를 사용합니다.
3. 탐색 창에서 고객 관리형 키를 선택합니다.
4. 키 생성을 선택합니다. 키 구성 페이지가 열립니다.
5. 키 유형에 대해 대칭을 선택합니다.

   > **Note**
   >
   > Amazon S3와 같은 모든 AWS 서비스는 대칭 암호화 키로만 작동하므로 이 키를 생성하면 대칭 키만 생성할 수 있습니다.

6. [Advanced Options]를 선택합니다. 키 구성 요소 오리진에 KMS가 선택되어 있는지 확인하고 다음을 선택합니다. 레이블 추가 페이지가 열립니다.
7. 별칭 및 설명 생성에 키의 별칭(예: `DMS-S3-endpoint-encryption-key`)과 추가 설명을 입력합니다.
8. 태그에서 키 식별과 키 사용 추적에 도움이 되는 태그를 추가한 후 다음을 선택합니다. 키 관리 권한 정의 페이지가 열리고 선택할 수 있는 사용자 및 역할 목록이 표시됩니다.
9. 키를 관리할 사용자와 역할을 추가합니다. 키를 관리하는 데 필요한 권한이 이러한 사용자와 역할에 있는지 확인합니다.

10. 키 삭제에서 키 관리자가 키를 삭제할 수 있는지 여부를 선택한 후 다음을 선택합니다. 키 사용 권한 정의 페이지가 열리고 선택할 수 있는 추가 사용자 및 역할 목록이 표시됩니다.
11. 이 계정의 Amazon S3 대상에 대해 암호화 작업을 수행할 사용자를 선택합니다. 또한 이전에 역할에서 생성한 역할을 선택하여 Amazon S3 대상 객체 암호화를 위한 액세스를 활성화합니다(예: `DMS-S3-endpoint-access-role`).
12. 목록에 없는 다른 계정을 추가하여 동일한 액세스 권한을 부여하려면 다른 AWS 계정에 대해 다른 AWS 계정 추가를 선택한 후 다음을 선택합니다. 키 정책 검토 및 편집 페이지가 열리고 기존 JSON을 입력하여 검토하고 편집할 수 있는 키 정책 JSON이 표시됩니다. 여기서 키 정책이 이전 단계에서 선택한 역할 및 사용자(예: `Admin` 및 `User1`)를 참조하는 위치를 확인할 수 있습니다. 다음 예제에 나온 것처럼 다양한 보안 주체(사용자 및 역할)에 허용되는 다양한 키 작업도 볼 수 있습니다.

```
{
  "Id": "key-consolepolicy-3",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:root"
        ]
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow access for Key Administrators",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:role/Admin"
        ]
      },
      "Action": [
        "kms:Create*",
        "kms:Describe*",
        "kms:Enable*",
        "kms:List*",
        "kms:Put*",
```

```
        "kms:Update*",
        "kms:Revoke*",
        "kms:Disable*",
        "kms:Get*",
        "kms:Delete*",
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ScheduleKeyDeletion",
        "kms:CancelKeyDeletion"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow use of the key",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:role/DMS-S3-endpoint-access-role",
          "arn:aws:iam::111122223333:role/Admin",
          "arn:aws:iam::111122223333:role/User1"
        ]
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow attachment of persistent resources",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::111122223333:role/DMS-S3-endpoint-access-role",
          "arn:aws:iam::111122223333:role/Admin",
          "arn:aws:iam::111122223333:role/User1"
        ]
      },
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
```

```
        "kms:RevokeGrant"
      ],
      "Resource": "*",
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": true
        }
      }
    }
  ]
```

13. 마침을 클릭합니다. KMS 키가 생성되었다는 메시지와 함께 암호화 키 페이지가 열립니다.

이제 지정된 별칭의 새 KMS 키가 생성되었습니다(예: `DMS-S3-endpoint-encryption-key`). 이 키를 통해 AWS DMS가 Amazon S3 대상 객체를 암호화할 수 있습니다.

날짜 기반 폴더 파티셔닝 사용

AWS DMS는 Amazon S3를 대상 엔드포인트로 사용하는 경우 트랜잭션 커밋 날짜를 기반으로 S3 폴더 파티션을 지원합니다. 날짜 기반 폴더 파티셔닝을 사용하면 단일 소스 테이블의 데이터를 S3 버킷의 시간 계층 폴더 구조에 쓸 수 있습니다. S3 대상 엔드포인트를 생성할 때 폴더를 파티셔닝하면 다음을 수행할 수 있습니다.

- S3 객체를 더 잘 관리할 수 있습니다.
- 각 S3 폴더의 크기를 제한하십시오.
- 데이터 레이크 쿼리 또는 기타 후속 작업을 최적화합니다.

S3 대상 엔드포인트를 생성할 때 날짜 기반 폴더 파티셔닝을 활성화할 수 있습니다. 기존 데이터를 마이그레이션하고 진행 중인 변경 사항을 복제(전체 로드+CDC)하거나 데이터 변경 사항만 복제(CDC만 해당)할 때 활성화할 수 있습니다. 다음과 같은 대상 엔드포인트 설정을 사용합니다.

- `DatePartitionEnabled` – 날짜를 기준으로 파티셔닝을 지정합니다. 이 부울 옵션을 `true`로 설정하여 트랜잭션 커밋 날짜를 기준으로 S3 버킷 폴더를 파티셔닝합니다.

  이 설정은 `PreserveTransactions` 또는 `CdcPath`와 함께 사용할 수 없습니다.

  기본 값은 `false`입니다.

- DatePartitionSequence – 폴더 파티셔닝 중 사용할 날짜 형식의 순서를 식별합니다. 이 ENUM 옵션을 YYYYMMDD, YYYYMMDDHH, YYYYMM, MMYYYYDD 또는 DDMMYYYY로 설정합니다. 기본 값은 YYYYMMDD입니다. DatePartitionEnabled이 true.로 설정된 경우 이 설정을 사용합니다.
- DatePartitionDelimiter – 폴더 파티셔닝 중 사용할 날짜 구분 기호를 지정합니다. 이 ENUM 옵션을 SLASH, DASH, UNDERSCORE 또는 NONE로 설정합니다. 기본 값은 SLASH입니다. DatePartitionEnabled가 true로 설정된 경우 이 설정을 사용합니다.

다음 예제는 데이터 파티션 시퀀스의 기본값과 구분 기호를 사용하여 날짜 기반 폴더 파티셔닝을 활성화하는 방법을 보여줍니다. AWS CLI.create-endpoint 명령의 --s3-settings '{*json-settings*}' 옵션을 사용합니다.

```
  --s3-settings '{"DatePartitionEnabled": true,"DatePartitionSequence":
"YYYYMMDD","DatePartitionDelimiter": "SLASH"}'
```

Amazon S3를 AWS DMS의 대상으로 사용할 때 분할된 소스의 병렬 로드

분할된 데이터 소스를 Amazon S3 대상에 병렬로 전체 로드하도록 구성할 수 있습니다. 이 접근 방식은 지원되는 소스 데이터베이스 엔진에서 S3 대상으로 분할된 데이터를 마이그레이션할 때 로드 시간을 개선합니다. 분할된 소스 데이터의 로드 시간을 개선하려면 소스 데이터베이스에 있는 모든 테이블의 파티션에 매핑된 S3 대상 하위 폴더를 생성합니다. 파티션에 바인딩된 이러한 하위 폴더를 사용하면 AWS DMS는 병렬 프로세스를 실행하여 대상의 각 하위 폴더를 채울 수 있습니다.

S3 대상의 병렬 전체 로드를 구성하기 위해 S3는 테이블 매핑의 table-settings 규칙에 대해 세 가지 parallel-load 규칙 유형을 지원합니다.

- partitions-auto
- partitions-list
- ranges

이러한 병렬 로드 규칙 유형에 관한 자세한 내용은 테이블 및 컬렉션 설정 규칙과 작업 단원을 참조하십시오.

partitions-auto 및 partitions-list 규칙 유형의 경우, AWS DMS는 다음과 같이 소스 엔드포인트의 각 파티션 이름을 사용하여 대상 하위 폴더 구조를 식별합니다.

```
bucket_name/bucket_folder/database_schema_name/table_name/partition_name/
LOADseq_num.csv
```

여기서 데이터가 S3 대상으로 마이그레이션되어 저장되는 하위 폴더 경로에는 이름이 같은 소스 파티션에 해당하는 추가 *partition_name* 하위 폴더가 포함됩니다. 그러면 이 *partition_name* 하위 폴더는 지정된 소스 파티션에서 마이그레이션된 데이터를 포함하는 하나 이상의 LOAD*seq_num*.csv 파일을 저장합니다. 여기서 *seq_num*은 .csv 파일 이름의 시퀀스 번호 접미사입니다(예: .csv 파일 LOAD00000001.csv의 이름에 포함된 00000001).

하지만 MongoDB, DocumentDB 같은 일부 데이터베이스 엔진에는 파티션이라는 개념이 없습니다. 이러한 데이터베이스 엔진의 경우, AWS DMS는 다음과 같이 실행 중인 소스 세그먼트 인덱스를 대상 .csv 파일 이름에 접두사로 s추가합니다.

```
.../database_schema_name/table_name/SEGMENT1_LOAD00000001.csv
.../database_schema_name/table_name/SEGMENT1_LOAD00000002.csv
...
.../database_schema_name/table_name/SEGMENT2_LOAD00000009.csv
.../database_schema_name/table_name/SEGMENT3_LOAD0000000A.csv
```

여기서 파일 SEGMENT1_LOAD00000001.csv 및 SEGMENT1_LOAD00000002.csv는 실행 중인 소스 세그먼트 인덱스 접두사 SEGMENT1과 동일한 접두사를 포함하여 이름이 지정됩니다. 이 두 개의.csv 파일에 대해 마이그레이션된 소스 데이터는 실행 중인 동일한 소스 세그먼트 인덱스와 연결되어 있기 때문에 그와 같은 이름이 지정됩니다. 반면, 대상 SEGMENT2_LOAD00000009.csv 및 SEGMENT3_LOAD0000000A.csv 파일에 각각 저장된 마이그레이션된 데이터는 서로 다른 실행 소스 세그먼트 인덱스와 연결됩니다. 각 파일의 파일 이름 앞에는 실행 중인 세그먼트 인덱스의 이름 SEGMENT2 및 SEGMENT3이 접두사로 붙습니다.

ranges 병렬 로드 유형의 경우, table-settings 규칙의 columns 및 boundaries 설정을 사용하여 열 이름과 열 값을 정의합니다. 이러한 규칙을 사용하여 다음과 같이 세그먼트 이름에 해당하는 파티션을 지정할 수 있습니다.

```
"parallel-load": {
    "type": "ranges",
    "columns": [
         "region",
         "sale"
    ],
    "boundaries": [
          [
               "NORTH",
```

```
                "1000"
            ],
            [
                "WEST",
                "3000"
            ]
    ],
    "segment-names": [
            "custom_segment1",
            "custom_segment2",
            "custom_segment3"
    ]
}
```

여기서 segment-names 설정은 S3 대상에서 데이터를 병렬로 마이그레이션하기 위한 세 개의 파티션 이름을 정의합니다. 마이그레이션된 데이터는 다음과 같이 병렬로 로드되어 파티션 하위 폴더 아래의 .csv 파일에 순서대로 저장됩니다.

```
.../database_schema_name/table_name/custom_segment1/LOAD[00000001...].csv
.../database_schema_name/table_name/custom_segment2/LOAD[00000001...].csv
.../database_schema_name/table_name/custom_segment3/LOAD[00000001...].csv
```

여기서 AWS DMS는 세 파티션 하위 폴더에 각각 일련의 .csv 파일을 순서대로 저장합니다. 각 파티션 하위 폴더에 있는 일련의 .csv 파일은 모든 데이터가 마이그레이션될 때까지 LOAD00000001.csv를 시작으로 단계적으로 이름이 지정됩니다.

경우에 따라서는 segment-names 설정을 사용하여 ranges 병렬 로드 유형에 대한 파티션 하위 폴더의 이름을 명시적으로 지정하지 않을 수 있습니다. 이 경우 AWS DMS는 해당 *table_name* 하위 폴더 아래에 일련의 .csv 파일을 각각 생성하는 기본값을 적용합니다. 여기서 AWS DMS는 다음과 같이 일련의 .csv 파일의 s파일 이름 앞에 실행 중인 소스 세그먼트 인덱스의 이름을 접두사로 붙입니다.

```
.../database_schema_name/table_name/SEGMENT1_LOAD[00000001...].csv
.../database_schema_name/table_name/SEGMENT2_LOAD[00000001...].csv
.../database_schema_name/table_name/SEGMENT3_LOAD[00000001...].csv
...
.../database_schema_name/table_name/SEGMENTZ_LOAD[00000001...].csv
```

Amazon S3를 AWS DMS의 대상으로 사용 시 엔드포인트 설정

추가 연결 속성을 사용하는 것과 마찬가지로 엔드포인트 설정을 사용하여 Amazon S3 대상을 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI의 create-endpoint 명령을 --s3-

`settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Amazon S3를 대상으로 하여 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

옵션	설명
`CsvNullValue`	AWS DMS가 null 값을 처리하는 방법을 지정하는 선택적 파라미터입니다. null 값을 처리하는 동안 이 파라미터를 사용하여 대상에 쓸 때 사용자 정의 문자열을 null로 전달할 수 있습니다. 예를 들어, 대상 열이 null을 허용하는 경우, 이 옵션을 사용하여 빈 문자열 값과 null 값을 구분할 수 있습니다. 따라서 이 파라미터 값을 빈 문자열(" " 또는 '')로 설정하면 AWS DMS는 빈 문자열을 NULL 대신 null 값으로 처리합니다. 기본 값: `NULL` 유효값: 모든 유효한 문자열 예제: `--s3-settings '{"CsvNullValue": " "}'`
`AddColumnName`	`true` 또는 `y`로 설정할 때 .csv 출력 파일에 열 이름 정보를 추가하는 데 사용할 수 있는 선택적 파라미터입니다. 이 파라미터는 `PreserveTransactions` 또는 `CdcPath`와 함께 사용할 수 없습니다. 기본 값: `false` 유효값: `true`, `false`, `y`, `n` 예제: `--s3-settings '{"AddColumnName": true}'`
`AddTrailingPaddingCharacter`	S3 대상 엔드포인트 설정 `AddTrailingPaddingCharacter`를 사용하여 문자열 데이터에 패딩을 추가합니다. 기본 값은 `false`입니다. 유형: 부울 예제: `--s3-settings '{"AddTrailingPaddingCharacter": true}'`

옵션	설명
`BucketFolder`	S3 버킷의 폴더 이름을 설정하는 선택적 파라미터입니다. 제공되는 경우, 대상 객체는 경로 *`BucketFolder`* `/`*`schema_name`* `/`*`table_name`* `/`에서 .csv 또는 .parquet 파일로 생성됩니다. 이 파라미터가 지정되지 않은 경우, 사용되는 경로는 *`schema_name`* `/`*`table_name`* `/`입니다. 예제: `--s3-settings '{"BucketFolder": "testFolder"}'`
`BucketName`	S3 대상 객체가 .csv 또는 .parquet 파일로 생성되는 S3 버킷의 이름입니다. 예제: `--s3-settings '{"BucketName": "buckettest"}'`
`CannedAcl ForObjects`	AWS DMS가 S3 버킷에서 .csv 또는 .parquet 파일로 생성되는 객체에 대한 사전 정의된(미리 제공된) 액세스 제어 목록을 지정할 수 있도록 하는 값입니다. Amazon S3에서 미리 제공된 ACL에 관한 자세한 내용은 Amazon S3 개발자 안내서의 미리 제공된 ACL을 참조하세요. 기본값: NONE 이 속성의 유효값은 NONE, PRIVATE, PUBLIC_READ, PUBLIC_READ_WRITE, AUTHENTICATED_READ, AWS_EXEC_READ, BUCKET_OWNER_READ, BUCKET_OWNER_FULL_CONTROL입니다. 예제: `--s3-settings '{"CannedAclForObjects": "PUBLIC_READ"}'`

옵션	설명
CdcInsertsOnly	변경 데이터 캡처(CDC) 로드 중 쉼표로 구분된 값(.csv) 또는 컬럼 방식 스토리지(.parquet) 출력 파일에 INSERT 작업만을 작성하는 선택적 파라미터입니다. 기본적으로(false 설정), .csv 또는 .parquet 레코드의 첫 번째 필드에는 문자 I(INSERT), U(UPDATE) 또는 D(DELETE)가 포함됩니다. 이 문자는 대상에 대한 CDC 로드를 위해 소스 데이터베이스에 행이 삽입, 업데이트 또는 삭제되었는지 여부를 나타냅니다. cdcInsertsOnly 가 true 또는 y로 설정되면 소스 데이터베이스의 INSERT만 .csv 또는 .parquet 파일로 마이그레이션됩니다. .csv 형식의 경우에만 IncludeOpForFullLoad 의 값에 따라 이 INSERT가 기록되는 방식이 달라집니다. IncludeOpForFullLoad 가 true로 설정되면 모든 CDC 레코드의 첫 번째 필드는 소스에서 INSERT 작업을 나타내기 위해 I로 설정됩니다. IncludeOpForFullLoad 가 false로 설정되면 모든 CDC 레코드는 소스에서 INSERT 작업을 나타내는 첫 번째 필드 없이 작성됩니다. 이러한 파라미터가 상호 작용하는 방식에 관한 자세한 내용은 마이그레이션된 S3 데이터에 소스 DB 작업 표시 단원을 참조하십시오. 기본 값: false 유효값: true, false, y, n 예제: --s3-settings '{"CdcInsertsOnly": true}'

옵션	설명
CdcInsertsAndUpdates	변경 데이터 캡처(CDC) 로드를 사용하여 INSERT 및 UPDATE 작업을 .csv 또는 .parquet(열 기반 스토리지) 출력 파일에 기록할 수 있습니다. 기본 설정은 false지만 cdcInsertsAndUpdates 를 true 또는 y로 설정하면 소스 데이터베이스의 INSERT 및 UPDATE가 .csv 또는 .parquet 파일로 마이그레이션됩니다. .csv 파일 형식의 경우에만 이러한 INSERT 및 UPDATE가 기록되는 방법이 includeOpForFullLoad 파라미터 값에 따라 달라집니다. includeOpForFullLoad 를 true로 설정하면 모든 CDC 레코드의 첫 번째 필드가 소스에서 INSERT 및 UPDATE 작업을 나타내도록 I 또는 U로 설정됩니다. 그러나 includeOpForFullLoad 를 false로 설정하면 소스에서 INSERT 또는 UPDATE 작업 표시 없이 CDC 레코드가 기록됩니다. 이러한 파라미터가 상호 작용하는 방식에 관한 자세한 내용은 마이그레이션된 S3 데이터에 소스 DB 작업 표시 단원을 참조하십시오. **Note** CdcInsertsOnly 및 cdcInsertsAndUpdates 는 동일한 엔드 포인트에서 둘 다 true로 설정할 수 없습니다. 동일한 엔드포인트에 대해 cdcInsertsOnly 또는 cdcInsertsAndUpdates 를 true로 설정합니다. 단, 둘 다 설정하지는 않습니다. 기본 값: false 유효값: true, false, y, n 예제: --s3-settings '{"CdcInsertsAndUpdates": true}'

옵션	설명
CdcPath	CDC 파일의 폴더 경로를 지정합니다. S3 소스의 경우, 태스크가 변경 데이터를 캡처하는 경우 이 설정이 필요합니다. 그렇지 않으면 선택 사항입니다. CdcPath가 설정된 경우 DMC는 이 경로에서 CDC 파일을 읽고 데이터 변경 사항을 대상 엔드포인트로 복제합니다. S3 대상의 경우 PreserveTransactions 를 true로 설정하면 DMS는 이 파라미터를 S3 대상의 폴더 경로로 설정했는지 확인합니다. 폴더 경로에서는 DMS가 CDC 로드에 대한 트랜잭션 순서를 저장할 수 있습니다. DMS는 이 CDC 폴더 경로를 S3 대상 작업 디렉터리 또는 BucketFolder 및 BucketName 으로 지정된 S3 대상 위치에 생성합니다. 이 파라미터는 DatePartitionEnabled 또는 AddColumnName 과 함께 사용할 수 없습니다. 유형: 문자열 예를 들어, CdcPath를 MyChangedData 로 지정하고 BucketName 을 MyTargetBucket 으로 지정하지만 BucketFolder 를 지정하지 않으면 DMS는 CDC 폴더 경로(MyTargetBucket/MyChangedData)를 생성합니다. 동일한 CdcPath를 지정하고 BucketName 을 MyTargetBucket 으로, BucketFolder 를 MyTargetData 로 지정하면 DMS는 CDC 폴더 경로(MyTargetBucket/MyTargetData/MyChangedData)를 생성합니다. **Note** 이 설정은 AWS DMS 버전 3.4.2 및 이후 버전에서 지원됩니다. 트랜잭션 순서대로 데이터 변경을 캡처할 때 DMS는 대상의 DataFormat S3 설정 값에 관계없이 항상 행 변경 내용을 .csv 파일에 저장합니다. DMS는 .parquet 파일을 사용하여 데이터 변경 사항을 트랜잭션 순서대로 저장하지 않습니다.

옵션	설명
CdcMaxBatchInterval	파일을 Amazon S3에 출력할 때까지의 최대 간격(초)입니다. 기본값: 60초 CdcMaxBatchInterval 이 지정되고 CdcMinFileSize 가 지정되면 파일 쓰기는 가장 먼저 충족되는 파라미터 조건에 의해 트리거됩니다.
CdcMinFileSize	Amazon S3에 파일을 출력하기 위한 최소 파일 크기 조건(단위: 킬로바이트[KB]) 기본값: 32000 KB CdcMinFileSize 가 지정되고 CdcMaxBatchInterval 이 지정되면 파일 쓰기는 가장 먼저 충족되는 파라미터 조건에 의해 트리거됩니다.
PreserveTransactions	true로 설정된 경우 DMS는 CdcPath로 지정된 Amazon S3 대상의 변경 데이터 캡처(CDC)에 대한 트랜잭션 순서를 저장합니다. 이 파라미터는 DatePartitionEnabled 또는 AddColumnName 와 함께 사용할 수 없습니다. 유형: 부울 트랜잭션 순서대로 데이터 변경을 캡처할 때 DMS는 대상의 DataFormat S3 설정 값에 관계없이 항상 행 변경 내용을 .csv 파일에 저장합니다. DMS는 .parquet 파일을 사용하여 데이터 변경 사항을 트랜잭션 순서대로 저장하지 않습니다. **Note** 이 설정은 AWS DMS 버전 3.4.2 및 이후 버전에서 지원됩니다.

옵션	설명
IncludeOp ForFullLoad	전체 로드 중 쉼표로 구분된 값(.csv) 출력 파일에만 INSERT 작업을 쓰기 위한 선택적 파라미터입니다. 전체 로드의 경우, 레코드는 삽입만 가능합니다. 기본적으로(false 설정) 전체 로드에 대한 이 출력 파일에는 행이 소스 데이터베이스에 삽입되었음을 나타내는 레코드가 기록되어 있지 않습니다. IncludeOp ForFullLoad 가 true 또는 y로 설정되면 .csv 파일의 첫 번째 필드에 INSERT가 I 주석으로 기록됩니다. **Note** 이 파라미터는 CdcInsertsOnly 또는 CdcInsert sAndUpdates 와 함께 .csv 파일의 출력에만 적용됩니다. 이러한 파라미터가 상호 작용하는 방식에 관한 자세한 내용은 마이그레이션된 S3 데이터에 소스 DB 작업 표시 단원을 참조하십시오. 기본 값: false 유효값: true, false, y, n 예제: --s3-settings '{"IncludeOpForFullLoad": true}'
CompressionType	GZIP으로 설정하면 GZIP을 사용하여 대상 .csv 또는 .parquet 파일을 압축하는 선택적 파라미터입니다. 이 파라미터를 기본값으로 설정하면 파일을 압축되지 않은 상태로 둡니다. 기본 값: NONE 유효값: GZIP 또는 NONE 예제: --s3-settings '{"CompressionType": "GZIP"}'
CsvDelimiter	.csv 소스 파일에서 열을 구분하는 데 사용되는 구분 기호입니다. 기본값은 쉼표(,)입니다. 예제: --s3-settings '{"CsvDelimiter": ","}'

옵션	설명
`CsvRowDelimiter`	.csv 소스 파일에서 행을 구분하는 데 사용되는 구분 기호입니다. 기본값은 줄 바꿈(\n)입니다. 예제: `--s3-settings '{"CsvRowDelimiter": "\n"}'`
`MaxFileSize`	전체 로드 중에 S3 대상으로 마이그레이션하는 동안 생성할 .csv 파일의 최대 크기(KB 단위)를 지정하는 값입니다. 기본값: 1,048,576KB(1GB) 유효값: 1~1,048,576 예제: `--s3-settings '{"MaxFileSize": 512}'`
`Rfc4180`	.csv 파일 형식만 사용하여 Amazon S3로 마이그레이션되는 데이터의 RFC 규정 준수 동작을 설정하는 데 사용되는 선택적 파라미터입니다. Amazon S3을 대상으로 사용하여 이 값을 `true` 또는 `y`로 설정하는 경우, 데이터에 따옴표나 쉼표 또는 줄 바꿈 문자가 있다면 AWS DMS는 전체 열을 큰따옴표(")로 묶습니다. 데이터 내 모든 따옴표는 두 번 반복됩니다. 이 형식은 RFC 4180을 준수합니다. 기본 값: `true` 유효값: `true`, `false`, `y`, `n` 예제: `--s3-settings '{"Rfc4180": false}'`

옵션	설명
EncryptionMode	S3로 복사되는 .csv 또는 .parquet 객체 파일을 암호화할 서버 측 암호화 모드입니다. 유효값은 SSE_S3(S3 서버 측 암호화) 또는 SSE_KMS(KMS 키 암호화)입니다. SSE_KMS를 선택하는 경우, ServerSideEncryptionKmsKeyId 파라미터를 암호화에 사용할 KMS 키의 Amazon 리소스 이름(ARN)으로 설정합니다. **Note** 또한 CLI modify-endpoint 명령을 사용하여 기존 엔드포인트의 EncryptionMode 속성 값을 SSE_KMS에서 SSE_S3로 변경할 수 있습니다. 그러나 EncryptionMode 값을 SSE_S3에서 SSE_KMS로 변경할 수 없습니다. 기본 값: SSE_S3 유효값: SSE_S3 또는 SSE_KMS 예제: --s3-settings '{"EncryptionMode": SSE_S3}'
ServerSideEncryptionKmsKeyId	EncryptionMode 를 SSE_KMS로 설정하는 경우, 이 파라미터를 KMS 키의 Amazon 리소스 이름(ARN)으로 설정합니다. 이 ARN은 계정을 위해 생성한 AWS KMS 키 목록에서 키 별칭을 선택하면 찾을 수 있습니다. 키를 생성할 때 특정 정책과 역할을 이 KMS 키와 연결해야 합니다. 자세한 내용은 AWS KMS 키를 생성하여 Amazon S3 대상 객체 암호화 섹션을 참조하세요. 예제: --s3-settings '{"ServerSideEncryptionKmsKeyId":"arn:aws:kms:us-east-1:111122223333:key/11a1a1a1-aaaa-9999-abab-2bbbbbb222a2"}'

옵션	설명
DataFormat	AWS DMS가 S3 객체를 생성하는 데 사용하는 파일의 출력 형식입니다. Amazon S3 대상의 경우, AWS DMS는 .csv 또는 .parquet 파일을 지원합니다. .parquet 파일은 효율적인 압축 옵션과 보다 빠른 쿼리 성능을 갖춘 이진 열 방식 스토리지 형식을 갖고 있습니다. .parquet 파일에 관한 자세한 내용은 https://parquet.apache.org/를 참조하십시오. 기본 값: csv 유효값: csv 또는 parquet 예제: --s3-settings '{"DataFormat": "parquet"}'
EncodingType	Parquet 인코딩 유형입니다. 인코딩 유형 옵션에는 다음이 포함됩니다. • rle-dictionary – 이 사전식 인코딩은 반복되는 값을 보다 효율적으로 저장하기 위해 비트 압축과 실행 길이 인코딩의 조합을 사용합니다. • plain – 인코딩 없음. • plain-dictionary – 이 사전식 인코딩은 주어진 열에서 발생하는 값의 사전을 빌드합니다. 사전은 각 열 청크의 사전 페이지에 저장됩니다. 기본 값: rle-dictionary 유효값: rle-dictionary , plain 또는 plain-dictionary 예제: --s3-settings '{"EncodingType": "plain-dictionary"}'

옵션	설명
DictPageSizeLimit	.parquet 파일에서 허용되는 사전 페이지의 최대 크기(단위: 바이트)입니다. 사전 페이지가 이 값을 초과하면 해당 페이지는 일반 인코딩을 사용합니다. 기본값: 1,024,000(1MB) 유효값: 유효한 정수 값 예제: `--s3-settings '{"DictPageSizeLimit": 2,048,000}'`
RowGroupLength	.parquet 파일의 한 행 그룹에 있는 행 수입니다. 기본값: 10,024(10KB) 유효값: 유효한 정수 값 예제: `--s3-settings '{"RowGroupLength": 20,048}'`
DataPageSize	.parquet 파일에서 허용되는 데이터 페이지의 최대 크기(단위: 바이트)입니다. 기본값: 1,024,000(1MB) 유효값: 유효한 정수 값 예제: `--s3-settings '{"DataPageSize": 2,048,000}'`
ParquetVersion	.parquet 파일 형식의 버전입니다. 기본 값: PARQUET_1_0 유효값: PARQUET_1_0 또는 PARQUET_2_0 예제: `--s3-settings '{"ParquetVersion": "PARQUET_2_0"}'`

옵션	설명
EnableStatistics	true 또는 y로 설정하여 .parquet 파일 페이지와 행 그룹에 관한 통계를 활성화합니다. 기본 값: true 유효값: true, false, y, n 예제: --s3-settings '{"EnableStatistics": false}'
TimestampColumnName	S3 대상 엔드포인트 데이터에 타임스탬프 열을 포함시키기 위한 선택적 파라미터입니다. TimestampColumnName 을 비어 있지 않은 값으로 설정하면 AWS DMS는 마이그레이션된 데이터의 .csv 또는 .parquet 객체 파일에 추가 STRING 열을 포함시킵니다. 전체 로드의 경우, 타임스탬프 열의 각 행에는 DMS에 의해 소스에서 대상으로 데이터가 전송된 시점에 대한 타임스탬프가 포함됩니다. CDC 로드의 경우, 타임스탬프 열의 각 행에는 소스 데이터베이스의 해당 행 커밋에 대한 타임스탬프가 있습니다. 이 타임스탬프 열 값의 문자열 형식은 yyyy-MM-dd HH:mm:ss.SSSSSS 입니다. 기본적으로 이 값의 정밀도는 마이크로초입니다. CDC 로드의 경우, 정밀도의 라운딩은 소스 데이터베이스에 대해 DMS가 지원하는 커밋 타임스탬프에 따라 다릅니다. AddColumnName 파라미터가 true로 설정되면 DMS는 비어 있지 않은 TimestampColumnName 값으로 설정한 타임스탬프 열의 이름도 포함합니다. 예제: --s3-settings '{"TimestampColumnName": "TIMESTAMP"}'

옵션	설명
UseTaskStartTimeForFullLoadTimestamp	true로 설정하면 이 파라미터는 데이터를 대상에 쓰는 시간 대신 태스크 시작 시간을 타임스탬프 열 값으로 사용합니다. 전체 로드의 경우, UseTaskStartTimeForFullLoadTimestamp 가 true로 설정되면 타임스탬프 열의 각 행에 태스크 시작 시간이 포함됩니다. CDC 로드의 경우 타임스탬프 열의 각 행에는 트랜잭션 커밋 시간이 포함됩니다. UseTaskStartTimeForFullLoadTimestamp 가 false로 설정되면 타임스탬프 열의 전체 로드 타임스탬프는 데이터가 대상에 도착하는 시간과 함께 증가합니다. 기본 값: false 유효값: true, false 예제: --s3-settings '{"UseTaskStartTimeForFullLoadTimestamp": true}' UseTaskStartTimeForFullLoadTimestamp: true 는 전체 로드에 대한 S3 대상 TimestampColumnName 을 CDC 로드 시 TimestampColumnName 과 함께 정렬할 수 있도록 도움을 줍니다.

옵션	설명
ParquetTimestampInMillisecond	.parquet 형식의 S3 객체 파일에 기록된 모든 TIMESTAMP 열 값의 정밀도를 지정하는 선택적 파라미터입니다. 이 속성을 true 또는 y로 설정하면 AWS DMS는 모든 TIMESTAMP 열을 .parquet 형식 파일에 밀리초 정밀도로 기록합니다. 그렇지 않으면 DMS는 해당 열을 마이크로초 정밀도로 기록합니다. 현재 Amazon Athena 및 AWS Glue은 TIMESTAMP 값에 대한 밀리초 정밀도만 처리할 수 있습니다. Athena 또는 AWS Glue를 사용하여 데이터를 쿼리하거나 처리할 계획인 경우에만 .parquet 형식의 S3 엔드포인트 객체 파일에 대해 이 속성을 true로 설정합니다. **Note** • AWS DMS는 .csv 형식 S3 파일에 기록된 모든 TIMESTAMP 열 값을 마이크로초 정밀도로 기록합니다. • 이 속성의 설정은 TimestampColumnName 속성을 설정하여 삽입된 타임스탬프 열 값의 문자열 형식에는 영향을 미치지 않습니다. 기본 값: false 유효값: true, false, y, n 예제: --s3-settings '{"ParquetTimestampInMillisecond": true}'

옵션	설명
GlueCatalogGeneration	AWS Glue Data Catalog을 생성하려면 이 엔드포인트 설정을 true로 설정하십시오. 기본 값: false 유효값: true, false. 예제: --s3-settings '{"GlueCatalogGeneration": true}' 참고: GlueCatalogGeneration 을 PreserveTransactions 및 CdcPath와 함께 사용하지 마십시오.

AWS DMS의 Amazon S3 대상과 함께 AWS Glue Data Catalog을 사용

AWS Glue는 데이터를 분류하는 간단한 방법을 제공하는 서비스로서, AWS Glue Data Catalog으로 알려진 메타데이터 리포지토리로 구성되어 있습니다. AWS Glue Data Catalog을 Amazon S3 d대상 엔드포인트와 통합하고 Amazon Athena 같은 다른 AWS 서비스를 통해 Amazon S3 데이터를 쿼리할 수 있습니다. Amazon Redshift는 AWS Glue와 함께 연동하지만 AWS DMS는 이를 사전 빌드된 옵션으로 지원하지 않습니다.

데이터 카탈로그를 생성하려면 다음 AWS CLI 예와 같이 GlueCatalogGeneration 엔드포인트 설정을 true로 설정합니다.

```
aws dms create-endpoint --endpoint-identifier s3-target-endpoint
            --engine-name s3 --endpoint-type target--s3-settings
 '{"ServiceAccessRoleArn":
            "your-service-access-ARN", "BucketFolder": "your-bucket-folder",
 "BucketName":
            "your-bucket-name", "DataFormat": "parquet", "GlueCatalogGeneration":
 true}'
```

csv 형식 데이터가 포함된 전체 부하 복제 작업의 경우, IncludeOpForFullLoad를 true로 설정합니다.

참고: GlueCatalogGeneration을 PreserveTransactions 및 CdcPath와 함께 사용하지 마십시오. AWS Glue 크롤러는 지정된 CdcPath에 저장된 파일의 여러 스키마를 조정할 수 없습니다.

Amazon Athena가 Amazon S3 데이터를 인덱싱하고 Amazon Athena를 통해 표준 SQL 쿼리를 사용하여 데이터를 쿼리하려면 엔드포인트에 연결된 IAM 역할에 다음 정책이 있어야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:ListMultipartUploadParts",
                "s3:AbortMultipartUpload"
            ],
            "Resource": [
                "arn:aws:s3:::bucket123",
                "arn:aws:s3:::bucket123/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateDatabase",
                "glue:GetDatabase",
                "glue:CreateTable",
                "glue:DeleteTable",
                "glue:UpdateTable",
                "glue:GetTable",
                "glue:BatchCreatePartition",
                "glue:CreatePartition",
                "glue:UpdatePartition",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:BatchGetPartition"
            ],
            "Resource": [
                "arn:aws:glue:*:111122223333:catalog",
                "arn:aws:glue:*:111122223333:database/*",
                "arn:aws:glue:*:111122223333:table/*"
            ]
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "athena:StartQueryExecution",
                "athena:GetQueryExecution",
                "athena:CreateWorkGroup"
            ],
            "Resource": "arn:aws:athena:*:111122223333:workgroup/
glue_catalog_generation_for_task_*"
        }
    ]
}
```

참조

- AWS Glue에 관한 자세한 내용은 AWS Glue 개발자 안내서의 개념 섹션을 참조하세요.
- AWS Glue Data Catalog에 관한 자세한 내용은 AWS Glue 개발자 안내서의 구성 요소를 참조하세요.

Amazon S3 대상에서 데이터 암호화, parquet 파일 및 CDC 사용

S3 대상 엔드포인트 설정을 사용하여 다음을 구성할 수 있습니다.

- S3 대상 객체를 암호화하기 위한 사용자 지정 KMS 키.
- S3 대상 객체의 스토리지 형식인 parquet 파일
- S3 대상의 트랜잭션 순서를 포함한 변경 데이터 캡처(CDC).
- AWS Glue Data Catalog을 Amazon S3 대상 엔드포인트와 통합하고 Amazon Athena 같은 다른 서비스를 통해 Amazon S3 데이터를 쿼리할 수 있습니다.

데이터 암호화를 위한 AWS KMS 키 설정

다음 예는 S3 대상 객체를 암호화하도록 사용자 지정 KMS 키를 구성하는 방법을 보여 줍니다. 시작하려면 다음 `create-endpoint` CLI 명령을 실행합니다.

```
aws dms create-endpoint --endpoint-identifier s3-target-endpoint --engine-name s3 --
endpoint-type target
--s3-settings '{"ServiceAccessRoleArn": "your-service-access-ARN", "CsvRowDelimiter":
 "\n",
"CsvDelimiter": ",", "BucketFolder": "your-bucket-folder",
```

```
"BucketName": "your-bucket-name",
"EncryptionMode": "SSE_KMS",
"ServerSideEncryptionKmsKeyId": "arn:aws:kms:us-
east-1:111122223333:key/72abb6fb-1e49-4ac1-9aed-c803dfcc0480"}'
```

여기서 `--s3-settings` 옵션에 의해 지정된 JSON 객체는 다음 두 가지 파라미터를 정의합니다. 하나는 값이 `SSE_KMS`인 `EncryptionMode` 파라미터입니다. 다른 하나는 값이 `arn:aws:kms:us-east-1:111122223333:key/72abb6fb-1e49-4ac1-9aed-c803dfcc0480`인 `ServerSideEncryptionKmsKeyId` 파라미터입니다. 이 값은 사용자 지정 KMS 키의 Amazon 리소스 이름(ARN)입니다. S3 대상의 경우, 추가 설정도 지정할 수 있습니다. 이러한 설정은 서버 액세스 역할을 식별하고 기본 CSV 객체 스토리지 형식의 구분 기호를 제공하며 S3 대상 객체를 저장할 버킷 위치와 이름을 제공합니다.

기본적으로 S3 데이터 암호화는 S3 서버 측 암호화를 사용하여 수행됩니다. 이전 예의 S3 대상의 경우, 다음 예에서처럼 이것은 엔드포인트 설정을 지정하는 것에 해당합니다.

```
aws dms create-endpoint --endpoint-identifier s3-target-endpoint --engine-name s3 --
endpoint-type target
--s3-settings '{"ServiceAccessRoleArn": "your-service-access-ARN", "CsvRowDelimiter":
 "\n",
"CsvDelimiter": ",", "BucketFolder": "your-bucket-folder",
"BucketName": "your-bucket-name",
"EncryptionMode": "SSE_S3"}'
```

S3 서버 측 암호화 작업에 관한 자세한 내용은 서버 측 암호화를 사용하여 데이터 보호 단원을 참조하십시오.

> **Note**
>
> 또한 CLI `modify-endpoint` 명령을 사용하여 기존 엔드포인트의 `EncryptionMode` 파라미터 값을 `SSE_KMS`에서 `SSE_S3`로 변경할 수 있습니다. 그러나 `EncryptionMode` 값을 `SSE_S3`에서 `SSE_KMS`로 변경할 수 없습니다.

.parquet 파일을 사용하여 S3 대상 객체를 저장하기 위한 설정

S3 대상 객체 생성의 기본 형식은 .csv 파일입니다. 다음 예는 S3 대상 객체 생성을 위한 형식으로 .parquet 파일을 지정하기 위한 몇 가지 엔드포인트 설정을 보여 줍니다. 다음 예에서처럼 .parquet 파일 형식을 모두 기본값으로 지정할 수 있습니다.

```
aws dms create-endpoint --endpoint-identifier s3-target-endpoint --engine-name s3 --
endpoint-type target
--s3-settings '{"ServiceAccessRoleArn": "your-service-access-ARN", "DataFormat":
 "parquet"}'
```

여기서는 모든 S3 기본값으로 형식을 활성화하기 위해 DataFormat 파라미터가 parquet로 설정됩니다. 이러한 기본값에는 반복되는 값을 보다 효율적으로 저장하기 위해 비트 압축과 실행 길이 인코딩의 조합을 사용하는 사전식 인코딩("EncodingType: "rle-dictionary")이 포함됩니다.

다음 예에서처럼 기본값이 아닌 옵션의 추가 설정을 추가할 수 있습니다.

```
aws dms create-endpoint --endpoint-identifier s3-target-endpoint --engine-name s3 --
endpoint-type target
--s3-settings '{"ServiceAccessRoleArn": "your-service-access-ARN", "BucketFolder":
 "your-bucket-folder",
"BucketName": "your-bucket-name", "CompressionType": "GZIP", "DataFormat": "parquet",
 "EncodingType: "plain-dictionary", "DictPageSizeLimit": 3,072,000,
"EnableStatistics": false }'
```

여기서 몇 가지 표준 S3 버킷 옵션의 파라미터와 DataFormat 파라미터 외에 다음의 추가 .parquet 파일 파라미터가 설정됩니다.

- EncodingType – 사전 페이지의 열당 청크의 각 열에서 발생하는 값을 저장하는 사전식 인코딩(plain-dictionary)으로 설정됩니다.
- DictPageSizeLimit – 사전 페이지 최대 크기인 3MB로 설정됩니다.
- EnableStatistics – Parquet 파일 페이지와 행 그룹에 관한 통계 수집을 활성화하는 기본값을 비활성화합니다.

S3 대상의 트랜잭션 순서를 포함한 변경 데이터 캡처(CDC).

기본적으로 AWS DMS가 CDC 작업을 실행하면 소스 데이터베이스(또는 데이터베이스)에 기록된 모든 행 변경 내용이 각 테이블에 대해 하나 이상의 파일에 저장됩니다. 동일한 테이블에 대한 변경 내용을 포함하는 각 파일 세트는 해당 테이블과 연결된 단일 대상 디렉터리에 있습니다. AWS DMS는 Amazon S3 대상 엔드포인트로 마이그레이션된 데이터베이스 테이블 수만큼 대상 디렉터리를 생성합니다. 파일은 트랜잭션 순서에 관계없이 S3 대상의 이러한 디렉터리에 저장됩니다. 파일 명명 규칙, 데이터 내용 및 형식에 관한 자세한 내용은 Amazon S3를 AWS Database Migration Service의 대상으로 사용을 참조하십시오.

트랜잭션 순서도 캡처하는 방식으로 소스 데이터베이스 변경 사항을 캡처하려면 트랜잭션 크기에 따라 생성되는 하나 이상의 .csv 파일에 모든 데이터베이스 테이블의 행 변경 내용을 저장하도록 AWS DMS에 지시하는 S3 엔드포인트 설정을 지정할 수 있습니다. 이러한 .csv 트랜잭션 파일에는 각 트랜잭션과 관련된 모든 테이블에 대해 트랜잭션 순서대로 나열된 모든 행 변경 내용이 포함되어 있습니다. 이러한 트랜잭션 파일은 사용자가 S3 대상에서도 지정하는 단일 트랜잭션 디렉터리에 함께 있습니다. 각 트랜잭션 파일에서 트랜잭션 작업과 각 행 변경에 대한 데이터베이스 및 소스 테이블의 ID는 다음과 같이 행 데이터의 일부로 저장됩니다.

```
operation,table_name,database_schema_name,field_value,...
```

여기서 *operation*은 변경된 행의 트랜잭션 작업이고 *table_name*은 행이 변경된 데이터베이스 테이블의 이름이며 *database_schema_name*은 테이블이 있는 데이터베이스 스키마의 이름이고 *field_value*는 행의 데이터를 지정하는 하나 이상의 필드 값 중 첫 번째 필드 값입니다.

다음 트랜잭션 파일 예제는 두 테이블을 포함하는 하나 이상의 트랜잭션에 대해 변경된 행을 보여줍니다.

```
I,Names_03cdcad11a,rdsTempsdb,13,Daniel
U,Names_03cdcad11a,rdsTempsdb,23,Kathy
D,Names_03cdcad11a,rdsTempsdb,13,Cathy
I,Names_6d152ce62d,rdsTempsdb,15,Jane
I,Names_6d152ce62d,rdsTempsdb,24,Chris
I,Names_03cdcad11a,rdsTempsdb,16,Mike
```

여기서 각 행의 트랜잭션 작업은 첫 번째 열에 I(삽입), U(업데이트) 또는 D(삭제)로 표시됩니다. 테이블 이름은 두 번째 열 값입니다(예:Names_03cdcad11a). 데이터베이스 스키마의 이름은 세 번째 열의 값입니다(예:rdsTempsdb). 그리고 나머지 열은 사용자 고유의 행 데이터(예:13,Daniel)로 채워집니다.

또한 AWS DMS는 다음 명명 규칙에 따라 타임스탬프를 사용하여 Amazon S3 대상에서 생성하는 트랜잭션 파일의 이름을 지정합니다.

```
CDC_TXN-timestamp.csv
```

여기서 *timestamp*는 다음 예와 같이 트랜잭션 파일이 생성된 시간입니다.

```
CDC_TXN-20201117153046033.csv
```

파일 이름의 이 타임스탬프를 사용하면 트랜잭션 디렉터리에 트랜잭션 파일을 나열할 때 트랜잭션 파일이 트랜잭션 순서대로 생성되고 나열됩니다.

> **Note**
>
> 트랜잭션 순서대로 데이터 변경을 캡처할 때 AWS DMS는 대상의 `DataFormat` S3 설정 값에 관계없이 항상 행 변경 내용을 .csv 파일에 저장합니다. AWS DMS는 .parquet 파일을 사용하여 트랜잭션 순서대로 데이터 변경 사항을 저장하지 않습니다.

데이터 복제 작업 중에 Amazon S3 대상에 대한 쓰기 빈도를 제어하기 위해 `CdcMaxBatchInterval` 및 `CdcMinFileSize` 설정을 구성할 수 있습니다. 따라서 추가 오버헤드 작업 없이 데이터를 분석할 때 성능이 향상될 수 있습니다. 자세한 내용은 Amazon S3를 AWS DMS의 대상으로 사용 시 엔드포인트 설정 단원을 참조하십시오.

모든 행 변경 사항을 트랜잭션 순서대로 저장하도록 AWS DMS에 지시하려면

1. 대상의 `PreserveTransactions` S3 설정을 `true`로 설정합니다.
2. 대상의 `CdcPath` S3 설정을 AWS DMS가 .csv 트랜잭션 파일을 저장하려는 상대 폴더 경로로 설정합니다.

   AWS DMS는 기본 S3 대상 버킷 및 작업 디렉터리 아래 또는 대상의 `BucketName` 및 `BucketFolder` S3 설정을 사용하여 지정한 버킷 및 버킷 폴더 아래에 이 경로를 생성합니다.

마이그레이션된 S3 데이터에 소스 DB 작업 표시

AWS DMS가 S3 대상으로 레코드를 마이그레이션하면 마이그레이션된 각 레코드에 추가 필드를 만들 수 있습니다. 이 추가 필드는 소스 데이터베이스의 레코드에 적용된 작업을 나타냅니다. AWS DMS가 이 첫 번째 필드를 생성하고 설정하는 방법은 `includeOpForFullLoad`, `cdcInsertsOnly` 및 `cdcInsertsAndUpdates`의 마이그레이션 작업 유형과 설정에 따라 달라집니다.

`includeOpForFullLoad`가 `true`인 전체 로드의 경우, AWS DMS는 항상 각 .csv 레코드에 첫 번째 추가 필드를 생성합니다. 이 필드에는 행이 소스 데이터베이스에 삽입되었음을 나타내는 문자 I(INSERT)가 들어 있습니다. `cdcInsertsOnly`가 `false`(기본값)인 CDC 로드의 경우, AWS DMS는 항상 각 .csv 또는 .parquet 레코드에 첫 번째 필드를 추가로 만듭니다. 이 필드에는 소스 데이터베이스에서 행이 삽입, 업데이트 또는 삭제되었는지 여부를 나타내는 문자 I(INSERT), U(UPDATE) 또는 D(DELETE)가 포함됩니다.

다음 표에서 includeOpForFullLoad 및 cdcInsertsOnly 속성의 설정들이 작용하여 마이그레이션된 레코드의 설정에 영향을 주는 것을 볼 수 있습니다.

이 파라미터 설정을 사용할 경우		DMS는 .csv 및 .parquet 출력에 다음과 같이 대상 레코드를 설정합니다.	
includeOpForFullLoad	cdcInsertsOnly	전체 로드	CDC 로드
true	true	추가된 첫 번째 필드 값은 I로 설정	추가된 첫 번째 필드 값은 I로 설정
false	false	추가된 필드 없음	추가된 첫 번째 필드 값은 I, U 또는 D로 설정
false	true	추가된 필드 없음	추가된 필드 없음
true	false	추가된 첫 번째 필드 값은 I로 설정	추가된 첫 번째 필드 값은 I, U 또는 D로 설정

includeOpForFullLoad 및 cdcInsertsOnly가 같은 값으로 설정되면 대상 레코드는 현재 마이그레이션 유형에 대한 레코드 설정을 제어하는 속성에 따라 설정됩니다. 이 속성은 전체 로드의 경우 includeOpForFullLoad이고 CDC 로드의 경우 cdcInsertsOnly입니다.

includeOpForFullLoad와 cdcInsertsOnly가 서로 다른 값으로 설정되면 AWS DMS는 대상 레코드 설정을 CDC 및 전체 로드에 일관되게 만듭니다. 이를 위해 includeOpForFullLoad에서 이전에 지정한 전체 로드에 대한 레코드 설정에 부합하도록 CDC 로드에 대한 레코드 설정을 지정합니다.

즉, 삽입된 레코드를 나타내는 첫 번째 필드를 추가하도록 전체 로드가 설정되었다고 가정하십시오. 이 경우 다음 CDC 로드는 삽입, 업데이트 또는 삭제된 레코드가 소스에 적절함을 나타내는 첫 번째 필드를 추가하도록 설정됩니다. 이와 달리, 삽입된 레코드를 나타내는 첫 번째 필드를 추가하지 않도록 전체 로드가 설정된다고 가정하십시오. 이 경우 CDC 로드 역시, 소스에서의 해당 레코드 작업에 관계없이 각 레코드에 첫 번째 필드를 추가하지 않도록 설정됩니다.

마찬가지로 DMS가 첫 번째 추가 필드를 생성하고 설정하는 방법은 includeOpForFullLoad 및 cdcInsertsAndUpdates의 설정에 따라 다릅니다. 다음 표에서 includeOpForFullLoad 및

`cdcInsertsAndUpdates` 속성의 설정들이 작용하여 이 형식으로 마이그레이션된 레코드의 설정에 영향을 주는 것을 볼 수 있습니다.

이 파라미터 설정을 사용할 경우		DMS는 .csv 출력에 다음과 같이 대상 레코드를 설정합니다.	
includeOpForFullLoad	cdcInsertsAndUpdates	전체 로드	CDC 로드
`true`	`true`	추가된 첫 번째 필드 값은 `I`로 설정	추가된 첫 번째 필드 값은 `I` 또는 `U`로 설정
`false`	`false`	추가된 필드 없음	추가된 첫 번째 필드 값은 `I`, `U` 또는 `D`로 설정
`false`	`true`	추가된 필드 없음	추가된 첫 번째 필드 값은 `I` 또는 `U`로 설정
`true`	`false`	추가된 첫 번째 필드 값은 `I`로 설정	추가된 첫 번째 필드 값은 `I`, `U` 또는 `D`로 설정

S3 Parquet의 대상 데이터 유형

다음 테이블에서는 AWS DMS를 사용할 때 지원되는 Parquet 대상 데이터 형식과 AWS DMS 데이터 형식에 따른 기본 매핑을 보여줍니다.

AWS DMS 데이터 형식에 관한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

AWS DMS 데이터 유형	S3 Parquet 데이터 형식
`BYTES`	`BINARY`
`DATE`	`DATE32`
`TIME`	`TIME32`

AWS DMS 데이터 유형	S3 Parquet 데이터 형식
DATETIME	TIMESTAMP
INT1	INT8
INT2	INT16
INT4	INT32
INT8	INT64
NUMERIC	DECIMAL
REAL4	FLOAT
REAL8	DOUBLE
STRING	STRING
UINT1	UINT8
UINT2	UINT16
UINT4	UINT32
UINT8	UINT64
WSTRING	STRING
BLOB	BINARY
NCLOB	STRING
CLOB	STRING
BOOLEAN	BOOL

AWS Database Migration Service 대상으로 Amazon DynamoDB 데이터베이스 사용

AWS DMS를 사용하여 Amazon DynamoDB 테이블로 데이터를 마이그레이션할 수 있습니다. Amazon DynamoDB는 완벽한 확장성과 함께 빠르고 예측 가능한 성능을 제공하는 완전 관리형 NoSQL 데이터베이스 서비스입니다. AWS DMS는 관계형 데이터베이스 또는 MongoDB를 소스로 사용할 수 있도록 지원합니다.

DynamoDB에서 테이블, 항목 및 속성은 작업 시 사용하는 핵심 구성 요소입니다. 테이블은 항목의 컬렉션이고 각 항목은 속성의 컬렉션입니다. DynamoDB는 파티션 키라는 기본 키를 사용하여 테이블의 각 항목을 고유하게 식별합니다. 또한 키와 보조 인덱스를 사용할 수 있어 쿼리 유연성이 향상됩니다.

객체 매핑을 사용하여 소스 데이터베이스에서 대상 DynamoDB 테이블로 데이터를 마이그레이션할 수 있습니다. 객체 매핑을 통해 소스 데이터가 대상에 배치되는 위치를 결정할 수 있습니다.

AWS DMS는 DynamoDB 대상 엔드포인트에서 테이블을 생성하는 경우 소스 데이터베이스 엔드포인트에 있는 만큼 테이블을 생성합니다. AWS DMS는 여러 DynamoDB 파라미터 값을 설정하기도 합니다. 테이블 생성 비용은 마이그레이션할 테이블 수와 데이터 양에 따라 달라집니다.

> **Note**
>
> AWS DMS 콘솔 또는 API의 SSL 모드 옵션은 Kinesis 및 DynamoDB와 같은 일부 데이터 스트리밍 및 NoSQL 서비스에는 적용되지 않습니다. 이러한 스트리밍과 서비스는 기본적으로 안전하므로 AWS DMS에서는 SSL 모드 설정이 없음으로 표시됩니다(SSL 모드=없음). 엔드포인트에서 SSL을 사용하기 위한 추가 구성을 제공할 필요는 없습니다. 예를 들어 DynamoDB를 대상 엔드포인트로 사용하는 경우 기본적으로 안전합니다. DynamoDB에 대한 모든 API 직접 호출은 SSL을 사용하므로 AWS DMS 엔드포인트에서는 추가 SSL 옵션이 필요하지 않습니다. DynamoDB 데이터베이스에 연결할 때 기본적으로 AWS DMS가 사용하는 HTTPS 프로토콜을 사용하여 SSL 엔드포인트를 통해 안전하게 데이터를 넣고 검색할 수 있습니다.

전송 속도를 향상하는 데 도움이 되도록 AWS DMS는 DynamoDB 대상 인스턴스에 대한 멀티스레드 전체 로드를 지원합니다. DMS는 다음과 같은 작업 설정을 통해 이 멀티스레딩을 지원합니다.

- `MaxFullLoadSubTasks` – 병렬로 로드할 최대 소스 테이블 수를 표시하려면 이 옵션을 사용합니다. DMS는 전용 하위 작업을 사용하여 각 테이블을 해당 DynamoDB 대상 테이블에 로드합니다. 기본값은 8입니다. 최대값은 49입니다.

- `ParallelLoadThreads` – AWS DMS에서 각 테이블을 DynamoDB 대상 테이블에 로드하는 데 사용하는 스레드 수를 지정하려면 이 옵션을 사용합니다. 기본값은 0입니다(단일 스레드). 최대값은 200입니다. 이 최대 한도를 증가시키도록 요청할 수 있습니다.

  > **Note**
  >
  > DNMS는 로딩을 위해 테이블의 각 세그먼트를 고유의 스레드에 할당합니다. 따라서 `ParallelLoadThreads`를 소스의 테이블에 지정하는 최대 세그먼트 수로 설정합니다.

- `ParallelLoadBufferSize` – 병렬 로드 스레드에서 데이터를 DynamoDB 대상에 로드하기 위해 사용하는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 이 설정은 `ParallelLoadThreads`와 함께 사용하십시오. `ParallelLoadBufferSize`는 둘 이상의 스레드가 있는 경우에만 유효합니다.
- 개별 테이블에 대한 테이블 매핑 설정 – `table-settings` 규칙을 사용하여 소스에서 병렬로 로드할 개별 테이블을 식별합니다. 또한 멀티스레드 로딩을 위해 각 테이블의 행을 세그먼트화하는 방법도 이 규칙을 사용하여 지정합니다. 자세한 내용은 테이블 및 컬렉션 설정 규칙과 작업 섹션을 참조하세요.

> **Note**
>
> AWS DMS가 마이그레이션 작업에 대해 DynamoDB 파라미터 값을 설정하는 경우 기본 읽기 용량 단위(RCU) 파라미터 값이 200으로 설정됩니다.
> 쓰기 용량 단위(WCU) 파라미터 값도 설정되지만 이 파라미터의 값은 다른 여러 설정에 따라 달라집니다.
>
> - WCU 파라미터의 기본값은 200입니다.
> - `ParallelLoadThreads` 작업 설정이 1(기본값은 0)보다 크게 설정되면 WCU 파라미터는 `ParallelLoadThreads` 값의 200배로 설정됩니다.
> - 사용하는 리소스에는 표준 AWS DMS 사용 요금이 적용됩니다.

관계형 데이터베이스에서 DynamoDB 테이블로 마이그레이션

AWS DMS는 DynamoDB 스칼라 데이터 형식으로의 데이터 마이그레이션을 지원합니다. Oracle 또는 MySQL 같은 관계형 데이터베이스에서 DynamoDB로 마이그레이션할 때에는 이 데이터를 저장하는 방법을 다시 구성하는 것이 좋습니다.

현재 AWS DMS는 단일 테이블 간 DynamoDB 스칼라 형식 속성으로의 재구성을 지원합니다. 관계형 데이터베이스 테이블에서 데이터를 DynamoDB로 마이그레이션하려는 경우, 테이블에서 데이터를 가져와서 DynamoDB 스칼라 데이터 형식 속성으로 재지정합니다. 이 속성은 여러 열에서 데이터를 수락하여 열을 속성에 직접 매핑할 수 있습니다.

AWS DMS는 다음과 같은 DynamoDB 스칼라 데이터 형식을 지원합니다.

- 문자열
- 숫자
- 부울

> **Note**
>
> 소스의 NULL 데이터는 대상에서 무시됩니다.

DynamoDB를 AWS Database Migration Service의 대상으로 사용하기 위한 사전 조건

DynamoDB 데이터베이스를 AWS DMS의 대상으로 사용하기에 앞서 IAM 역할을 생성해야 합니다. 이 IAM 역할은 AWS DMS가 역할을 맡도록 허용해야 하며, 마이그레이션되고 있는 DynamoDB 테이블에 대한 액세스 권한을 부여합니다. 최소 액세스 권한 집합이 다음 IAM 정책에 나와 있습니다.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "",
         "Effect": "Allow",
         "Principal": {
            "Service": "dms.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
      }
   ]
}
```

DynamoDB로의 마이그레이션에 사용하는 역할에는 다음 권한이 있어야 합니다.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Action": [
        "dynamodb:PutItem",
        "dynamodb:CreateTable",
        "dynamodb:DescribeTable",
        "dynamodb:DeleteTable",
        "dynamodb:DeleteItem",
        "dynamodb:UpdateItem"
    ],
    "Resource": [
    "arn:aws:dynamodb:us-west-2:account-id:table/name1",
    "arn:aws:dynamodb:us-west-2:account-id:table/OtherName*",
    "arn:aws:dynamodb:us-west-2:account-id:table/awsdms_apply_exceptions",
    "arn:aws:dynamodb:us-west-2:account-id:table/awsdms_full_load_exceptions"
    ]
    },
    {
        "Effect": "Allow",
        "Action": [
        "dynamodb:ListTables"
        ],
        "Resource": "*"
    }
    ]
}
```

AWS Database Migration Service의 대상으로서 DynamoDB 사용 시 제한 사항

DynamoDB를 대상으로 사용할 때에는 다음 제한 사항이 적용됩니다.

- DynamoDB는 숫자 데이터 형식의 정밀도를 38자리로 제한합니다. 정밀도가 더 높은 모든 데이터 형식은 문자열로 저장합니다. 객체 매핑 기능을 사용하여 이것을 명시적으로 지정해야 합니다.
- DynamoDB에는 날짜 데이터 형식이 없기 때문에 날짜 데이터 형식을 사용한 데이터는 문자열로 변환됩니다.
- DynamoDB에서는 기본 키 속성 업데이트를 허용하지 않습니다. 대상에 원치 않는 데이터가 발생할 수 있으므로 변경 데이터 캡처(CDC)와 함께 지속적인 복제 사용 시 이 제한이 중요합니다. 객체 매핑 형태에 따라 기본 키를 업데이트하는 CDC 작업은 두 가지 중 하나를 수행할 수 있습니다. 즉, 실패하거나 업데이트된 키와 불완전한 데이터로 새 항목을 삽입할 수 있습니다.

- AWS DMS는 비복합 기본 키가 테이블의 복제만을 지원합니다. 예외는 사용자 지정 파티션 키나 정렬 키 또는 두 키를 모두 사용하여 대상 테이블의 객체 매핑을 지정하는 경우입니다.
- AWS DMS는 CLOB가 아닌 LOB 데이터를 지원하지 않습니다. AWS DMS는 데이터를 마이그레이션할 때 CLOB 데이터를 DynamoDB 문자열로 변환합니다.
- DynamoDB를 대상으로 사용하는 경우 예외 적용 제어 테이블(`dmslogs.awsdms_apply_exceptions`)만 지원됩니다. 제어 테이블에 대한 자세한 내용은 제어 테이블 작업 설정 섹션을 참조하세요.
- AWS DMS는 DynamoDB에 대한 작업 설정 `TargetTablePrepMode=TRUNCATE_BEFORE_LOAD`를 대상으로 지원하지 않습니다.
- AWS DMS는 DynamoDB에 대한 작업 설정 `TaskRecoveryTableEnabled`를 대상으로 지원하지 않습니다.

객체 매핑을 사용하여 데이터를 DynamoDB로 마이그레이션

AWS DMS는 테이블 매핑 규칙을 사용하여 데이터를 소스에서 대상 DynamoDB 테이블로 매핑합니다. 데이터를 DynamoDB 대상에 매핑하려면 객체 매핑이라는 테이블 매핑 규칙 유형을 사용합니다. 객체 매핑을 통해 마이그레이션할 속성 이름과 데이터를 정의할 수 있습니다. 객체 매핑을 사용할 때는 선택 규칙이 있어야 합니다.

DynamoDB에는 파티션 키와 정렬 키(선택 사항) 외에 사전 설정 구조가 없습니다. 비복합 기본 키가 있는 경우, AWS DMS는 이 기본 키를 사용합니다. 복합 기본 키가 있거나 정렬 키를 사용하려는 경우, 대상 DynamoDB 테이블에서 이러한 키와 기타 속성을 정의하십시오.

객체 매핑 규칙을 만들려면 `rule-type`을 object-mapping으로 지정합니다. 이 규칙은 사용할 객체 매핑의 유형을 지정합니다.

이 규칙의 구조는 다음과 같습니다.

```
{ "rules": [
    {
      "rule-type": "object-mapping",
      "rule-id": "<id>",
      "rule-name": "<name>",
      "rule-action": "<valid object-mapping rule action>",
      "object-locator": {
      "schema-name": "<case-sensitive schema name>",
      "table-name": ""
      },
      "target-table-name": "<table_name>"
```

```
    }
  ]
}
```

AWS DMS는 현재 `map-record-to-record`와 `map-record-to-document`를 `rule-action` 파라미터의 유일한 유효값으로 지원합니다. 이러한 값은 `exclude-columns` 속성 목록의 일부로 제외되지 않은 레코드에 대해 AWS DMS가 기본적으로 수행하는 작업을 지정합니다. 이러한 값은 어떤 식으로든 속성 매핑에 영향을 미치지 않습니다.

- 관계형 데이터베이스에서 DynamoDB로 마이그레이션할 때 `map-record-to-record`를 사용할 수 있습니다. 이것은 DynamoDB에서 파티션 키로서 관계형 데이터베이스의 기본 키를 사용하고 소스 데이터베이스의 각 열마다 속성을 생성합니다. `map-record-to-record`를 사용할 때 `exclude-columns` 속성 목록에 소스 테이블의 열이 나열되지 않은 경우, AWS DMS는 대상 DynamoDB 인스턴스에 해당 속성을 생성합니다. 이 속성은 소스 열이 속성 매핑에 사용되는지 여부와 상관없이 생성됩니다.
- 속성 이름 '_doc'를 사용하여 대상에 있는 단일한 플랫 DynamoDB 맵에 소스 열을 배치하려면 `map-record-to-document`를 사용합니다. `map-record-to-document`를 사용할 때 AWS DMS는 해당 데이터를 소스에 있는 단일한 플랫 맵 속성에 배치합니다. 이 속성을 '_doc'라고 합니다. 이 배치는 `exclude-columns` 속성 목록에 나열되지 않은 소스 테이블의 모든 열에 적용됩니다.

`rule-action` 파라미터 `map-record-to-record`와 `map-record-to-document`의 차이를 이해하는 한 가지 방법은 두 파라미터를 실행하는 것입니다. 이 예에서는 다음 구조와 데이터를 사용하여 관계형 데이터베이스 테이블 행에서 시작한다고 가정합니다.

FirstName	LastName	NickName	WorkAddress	WorkPhone	HomeAddress	HomePhone	income
Daniel	Sheridan	Dan	101 Main St Cambridge, MA	800-867-5309	100 Secret St, Unknownville, MA	123-456-7890	12345678

이 정보를 DynamoDB로 마이그레이션하려면 데이터를 DynamoDB 테이블 항목으로 매핑하는 규칙을 생성해야 합니다. `exclude-columns` 파라미터에 대해 나열된 열을 기록합니다. 이러한 열은 대상에 직접 매핑되지 않습니다. 그 대신 속성 매핑을 사용하여 데이터를 새 항목으로 결합합니다. 예를 들어 FirstName과 LastName이 함께 그룹화되어 DynamoDB 대상에서 CustomerName이 됩니다. NickName과 income은 제외되지 않습니다.

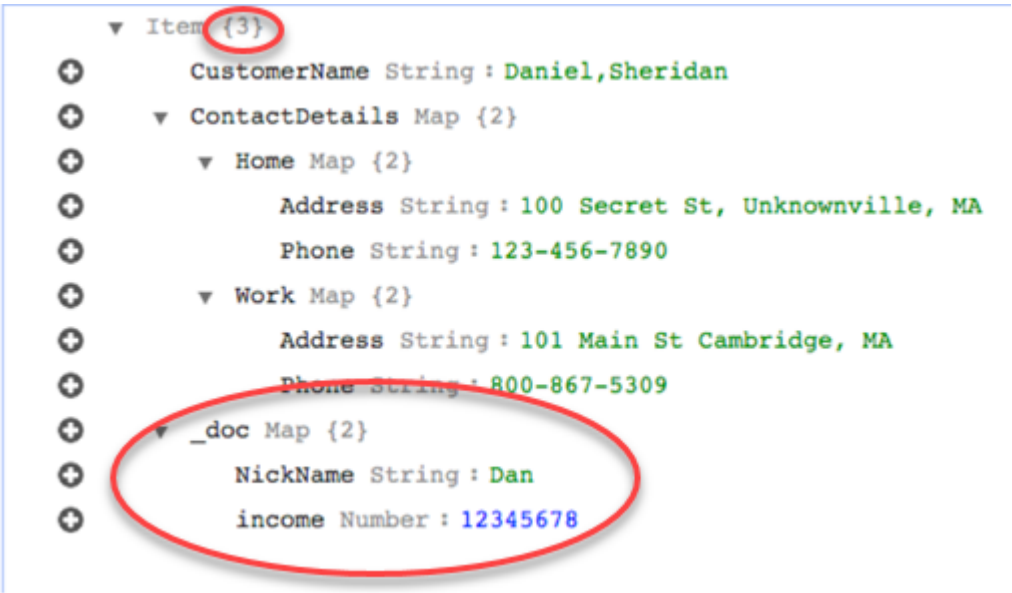
```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
```

```
            "rule-name": "1",
            "object-locator": {
                "schema-name": "test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "2",
            "rule-name": "TransformToDDB",
            "rule-action": "map-record-to-record",
            "object-locator": {
                "schema-name": "test",
                "table-name": "customer"
            },
            "target-table-name": "customer_t",
            "mapping-parameters": {
                "partition-key-name": "CustomerName",
                "exclude-columns": [
                    "FirstName",
                    "LastName",
                    "HomeAddress",
                    "HomePhone",
                    "WorkAddress",
                    "WorkPhone"
                ],
                "attribute-mappings": [
                    {
                        "target-attribute-name": "CustomerName",
                        "attribute-type": "scalar",
                        "attribute-sub-type": "string",
                        "value": "${FirstName},${LastName}"
                    },
                    {
                        "target-attribute-name": "ContactDetails",
                        "attribute-type": "document",
                        "attribute-sub-type": "dynamodb-map",
                        "value": {
                            "M": {
                                "Home": {
                                    "M": {
                                        "Address": {
                                            "S": "${HomeAddress}"
```

```
                                    },
                                    "Phone": {
                                        "S": "${HomePhone}"
                                    }
                                }
                            },
                            "Work": {
                                "M": {
                                    "Address": {
                                        "S": "${WorkAddress}"
                                    },
                                    "Phone": {
                                        "S": "${WorkPhone}"
                                    }
                                }
                            }
                        }
                    }
                ]
            }
        }
    ]
}
```

`rule-action` 파라미터 map-record-to-record를 사용하여 NickName 및 income의 데이터가 DynamoDB 대상에 있는 동일한 이름의 항목에 매핑됩니다.

단, 동일한 규칙을 사용하되 `rule-action` 파라미터를 map-record-to-document로 변경한다고 가정합니다. 이 경우, `exclude-columns` 파라미터에 나열되어 있지 않은 열인 NickName 및 income은 _doc 항목으로 매핑됩니다.



객체 매핑과 함께 사용자 지정 조건식 사용

조건식이라는 DynamoDB의 기능을 사용하여 DynamoDB 테이블에 작성되고 있는 데이터를 조작할 수 있습니다. DynamoDB의 조건식에 대한 자세한 내용은 조건식 단원을 참조하십시오.

조건식 멤버의 구성 요소는 다음과 같습니다.

- 표현식(필수)
- 표현식 속성 값(선택 사항). 속성 값의 json 구조를 지정합니다.
- 표현식 속성 이름(선택 사항)
- 조건식을 사용할 때 수반되는 옵션(선택 사항). 기본값은 apply-during-cdc = false와 apply-during-full-load = true입니다.

이 규칙의 구조는 다음과 같습니다.

```
"target-table-name": "customer_t",
      "mapping-parameters": {
        "partition-key-name": "CustomerName",
        "condition-expression": {
          "expression":"<conditional expression>",
```

```
            "expression-attribute-values": [
                {
                  "name":"<attribute name>",
                  "value":<attribute value>
                }
            ],
            "apply-during-cdc":<optional Boolean value>,
            "apply-during-full-load": <optional Boolean value>
          }
```

다음 샘플은 조건식에 사용되는 섹션을 강조한 것입니다.

객체 매핑과 함께 속성 매핑 사용

속성 매핑을 통해 소스 열 이름을 사용한 템플릿 문자열을 지정하여 대상에서 데이터를 재구성할 수 있습니다. 사용자가 템플릿에서 지정한 항목 외에 완료한 서식 변경이 없습니다.

다음 예는 소스 데이터베이스의 구조와 필요한 DynamoDB 대상 구조를 보여 줍니다. 첫 번째는 소스(이 경우, Oracle 데이터베이스)의 구조를, 그 다음은 필요한 DynamoDB 데이터 구조를 나타냅니다. 이 예는 필요한 대상 구조를 생성하는 데 사용되는 JSON으로 끝납니다.

Oracle 데이터의 구조는 다음과 같습니다.

First	Las	St	Home/ ss	HomeF	WorkAddre ss	Work	DateOfBirth
기본 키				N/A			
Ran	Mar	5	221B Baker Street	123456 0	31 Spooner Street, Quahog	9876 0	1988/02/29

DynamoDB 데이터의 구조는 다음과 같습니다.

Custor ame	StoreId	ContactDetails	DateOfBirth
파티션 키	정렬 키	N/A	
Randy sh	5	{ "Name": "Randy", "Home": { "Address": "221B Baker Street", "Phone": 1234567890 }, "Work": { "Address": "31 Spooner Street, Quahog", "Phone": 9876541230 } }	02/29/1988

다음 JSON은 DynamoDB 구조를 달성하는 데 사용되는 객체 매핑과 열 매핑을 보여 줍니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "2",
            "rule-name": "TransformToDDB",
            "rule-action": "map-record-to-record",
            "object-locator": {
                "schema-name": "test",
                "table-name": "customer"
            },
            "target-table-name": "customer_t",
            "mapping-parameters": {
                "partition-key-name": "CustomerName",
                "sort-key-name": "StoreId",
                "exclude-columns": [
                    "FirstName",
                    "LastName",
                    "HomeAddress",
                    "HomePhone",
                    "WorkAddress",
                    "WorkPhone"
                ],
                "attribute-mappings": [
                    {
                        "target-attribute-name": "CustomerName",
                        "attribute-type": "scalar",
                        "attribute-sub-type": "string",
                        "value": "${FirstName},${LastName}"
                    },
                    {
```

```
                        "target-attribute-name": "StoreId",
                        "attribute-type": "scalar",
                        "attribute-sub-type": "string",
                        "value": "${StoreId}"
                    },
                    {
                        "target-attribute-name": "ContactDetails",
                        "attribute-type": "scalar",
                        "attribute-sub-type": "string",
                        "value": "{\"Name\":\"${FirstName}\",\"Home\":{\"Address
\":\"${HomeAddress}\",\"Phone\":\"${HomePhone}\"}, \"Work\":{\"Address\":
\"${WorkAddress}\",\"Phone\":\"${WorkPhone}\"}}"
                    }
                ]
            }
        }
    ]
}
```

열 매핑을 사용하는 또 다른 방법은 DynamoDB 형식을 문서 형식으로 사용하는 것입니다. 다음 코드 예제에서는 dynamodb-map을 속성 매핑을 위한 `attribute-sub-type`으로 사용합니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "2",
            "rule-name": "TransformToDDB",
            "rule-action": "map-record-to-record",
            "object-locator": {
                "schema-name": "test",
```

```
            "table-name": "customer"
        },
        "target-table-name": "customer_t",
        "mapping-parameters": {
            "partition-key-name": "CustomerName",
            "sort-key-name": "StoreId",
            "exclude-columns": [
                "FirstName",
                "LastName",
                "HomeAddress",
                "HomePhone",
                "WorkAddress",
                "WorkPhone"
            ],
            "attribute-mappings": [
                {
                    "target-attribute-name": "CustomerName",
                    "attribute-type": "scalar",
                    "attribute-sub-type": "string",
                    "value": "${FirstName},${LastName}"
                },
                {
                    "target-attribute-name": "StoreId",
                    "attribute-type": "scalar",
                    "attribute-sub-type": "string",
                    "value": "${StoreId}"
                },
                {
                    "target-attribute-name": "ContactDetails",
                    "attribute-type": "document",
                    "attribute-sub-type": "dynamodb-map",
                    "value": {
                      "M": {
                        "Name": {
                          "S": "${FirstName}"
                        },
                        "Home": {
                                "M": {
                                    "Address": {
                                        "S": "${HomeAddress}"
                                    },
                                    "Phone": {
                                        "S": "${HomePhone}"
                                    }
```

```
                                    }
                                },
                                "Work": {
                                    "M": {
                                        "Address": {
                                            "S": "${WorkAddress}"
                                        },
                                        "Phone": {
                                            "S": "${WorkPhone}"
                                        }
                                    }
                                }
                            }
                        }
                    }
                ]
            }
        }
    ]
}
```

dynamodb-map 대신 다음 예제와 같이 속성 매핑을 위한 속성-하위 유형으로 dynamodb-list를 사용할 수 있습니다.

```
{
"target-attribute-name": "ContactDetailsList",
"attribute-type": "document",
"attribute-sub-type": "dynamodb-list",
"value": {
    "L": [
            {
                "N": "${FirstName}"
            },
            {
                "N": "${HomeAddress}"
            },
            {
                "N": "${HomePhone}"
            },
            {
                "N": "${WorkAddress}"
```

```
            },
            {
                "N": "${WorkPhone}"
            }
        ]
    }
}
```

예제 1: 객체 매핑과 함께 속성 매핑 사용

다음 예제는 두 MySQL 데이터베이스 테이블인 nfl_data 및 sport_team에서 NFLTeams와 SportTeam이라는 두 개의 DynamoDB 테이블로 데이터를 마이그레이션합니다. 이 테이블의 구조와 MySQL 데이터베이스 테이블의 데이터를 DynamoDB 테이블로 매핑하는 데 사용된 JSON은 다음과 같습니다.

MySQL 데이터베이스 테이블 nfl_data의 구조는 다음과 같습니다.

```
mysql> desc nfl_data;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| Position      | varchar(5)  | YES  |     | NULL    |       |
| player_number | smallint(6) | YES  |     | NULL    |       |
| Name          | varchar(40) | YES  |     | NULL    |       |
| status        | varchar(10) | YES  |     | NULL    |       |
| stat1         | varchar(10) | YES  |     | NULL    |       |
| stat1_val     | varchar(10) | YES  |     | NULL    |       |
| stat2         | varchar(10) | YES  |     | NULL    |       |
| stat2_val     | varchar(10) | YES  |     | NULL    |       |
| stat3         | varchar(10) | YES  |     | NULL    |       |
| stat3_val     | varchar(10) | YES  |     | NULL    |       |
| stat4         | varchar(10) | YES  |     | NULL    |       |
| stat4_val     | varchar(10) | YES  |     | NULL    |       |
| team          | varchar(10) | YES  |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
```

MySQL 데이터베이스 테이블 sport_team의 구조는 다음과 같습니다.

```
mysql> desc sport_team;
+--------------------------+-------------+------+-----+---------+----------------+
| Field                    | Type        | Null | Key | Default | Extra          |
+--------------------------+-------------+------+-----+---------+----------------+
| id                       | mediumint(9) | NO   | PRI | NULL    | auto_increment |
| name                     | varchar(30)  | NO   |     | NULL    |                |
| abbreviated_name         | varchar(10)  | YES  |     | NULL    |                |
| home_field_id            | smallint(6)  | YES  | MUL | NULL    |                |
| sport_type_name          | varchar(15)  | NO   | MUL | NULL    |                |
| sport_league_short_name  | varchar(10)  | NO   |     | NULL    |                |
| sport_division_short_name | varchar(10)  | YES  |     | NULL    |                |
```

테이블 2개를 DynamoDB 테이블 2개에 매핑하는 데 사용된 테이블 매핑 규칙은 다음과 같습니다.

```
{
  "rules":[
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "dms_sample",
        "table-name": "nfl_data"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "selection",
      "rule-id": "2",
      "rule-name": "2",
      "object-locator": {
        "schema-name": "dms_sample",
        "table-name": "sport_team"
      },
      "rule-action": "include"
    },
    {
      "rule-type":"object-mapping",
      "rule-id":"3",
      "rule-name":"MapNFLData",
      "rule-action":"map-record-to-record",
```

```
      "object-locator":{
        "schema-name":"dms_sample",
        "table-name":"nfl_data"
      },
      "target-table-name":"NFLTeams",
      "mapping-parameters":{
        "partition-key-name":"Team",
        "sort-key-name":"PlayerName",
        "exclude-columns": [
          "player_number", "team", "name"
        ],
        "attribute-mappings":[
          {
            "target-attribute-name":"Team",
            "attribute-type":"scalar",
            "attribute-sub-type":"string",
            "value":"${team}"
          },
          {
            "target-attribute-name":"PlayerName",
            "attribute-type":"scalar",
            "attribute-sub-type":"string",
            "value":"${name}"
          },
          {
            "target-attribute-name":"PlayerInfo",
            "attribute-type":"scalar",
            "attribute-sub-type":"string",
            "value":"{\"Number\": \"${player_number}\",\"Position\": \"${Position}\",
\"Status\": \"${status}\",\"Stats\": {\"Stat1\": \"${stat1}:${stat1_val}\",\"Stat2\":
 \"${stat2}:${stat2_val}\",\"Stat3\": \"${stat3}:${
stat3_val}\",\"Stat4\": \"${stat4}:${stat4_val}\"}"
          }
        ]
      }
    },
    {
      "rule-type":"object-mapping",
      "rule-id":"4",
      "rule-name":"MapSportTeam",
      "rule-action":"map-record-to-record",
      "object-locator":{
        "schema-name":"dms_sample",
        "table-name":"sport_team"
```

```
      },
      "target-table-name":"SportTeams",
      "mapping-parameters":{
        "partition-key-name":"TeamName",
        "exclude-columns": [
          "name", "id"
        ],
        "attribute-mappings":[
          {
            "target-attribute-name":"TeamName",
            "attribute-type":"scalar",
            "attribute-sub-type":"string",
            "value":"${name}"
          },
          {
            "target-attribute-name":"TeamInfo",
            "attribute-type":"scalar",
            "attribute-sub-type":"string",
            "value":"{\"League\": \"${sport_league_short_name}\",\"Division\":
 \"${sport_division_short_name}\"}"
          }
        ]
      }
    }
  ]
}
```

NFLTeams DynamoDB 테이블의 샘플 출력은 다음과 같습니다.

```
 "PlayerInfo": "{\"Number\": \"6\",\"Position\": \"P\",\"Status\": \"ACT\",\"Stats\":
{\"Stat1\": \"PUNTS:73\",\"Stat2\": \"AVG:46\",\"Stat3\": \"LNG:67\",\"Stat4\": \"IN
20:31\"}",
 "PlayerName": "Allen, Ryan",
 "Position": "P",
 "stat1": "PUNTS",
 "stat1_val": "73",
 "stat2": "AVG",
 "stat2_val": "46",
 "stat3": "LNG",
 "stat3_val": "67",
 "stat4": "IN 20",
```

```
  "stat4_val": "31",
  "status": "ACT",
  "Team": "NE"
}
```

SportsTeams DynamoDB 테이블의 샘플 출력은 다음과 같습니다.

```
{
  "abbreviated_name": "IND",
  "home_field_id": 53,
  "sport_division_short_name": "AFC South",
  "sport_league_short_name": "NFL",
  "sport_type_name": "football",
  "TeamInfo": "{\"League\": \"NFL\",\"Division\": \"AFC South\"}",
  "TeamName": "Indianapolis Colts"
}
```

DynamoDB에 대한 대상 데이터 형식

AWS DMS용 DynamoDB 엔드포인트는 대부분의 DynamoDB 데이터 형식을 지원합니다. 다음 테이블에는 AWS DMS를 사용하고 기본적으로 AWS DMS 데이터 형식에서 매핑할 때 지원되는 Amazon AWS DMS 대상 데이터 형식이 나와 있습니다.

AWS DMS 데이터 형식에 관한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

AWS DMS에서 다른 형식의 데이터베이스에서 가져온 데이터를 마이그레이션할 때 소스 데이터베이스의 데이터 형식을 AWS DMS 데이터 형식이라고 하는 중간 데이터 형식에 매핑합니다. 그런 다음, 중간 데이터 형식을 대상 데이터 형식에 매핑합니다. 다음 테이블에는 각 AWS DMS 데이터 형식과 DynamoDB에서 매핑되는 데이터 형식이 나와 있습니다.

AWS DMS 데이터 유형	DynamoDB 데이터 형식
문자열	문자열
WString	문자열
부울	부울

AWS DMS 데이터 유형	DynamoDB 데이터 형식
날짜	문자열
DateTime	문자열
INT1	숫자
INT2	숫자
INT4	숫자
INT8	숫자
숫자	숫자
Real4	숫자
Real8	숫자
UINT1	숫자
UINT2	숫자
UINT4	숫자
UINT8	숫자
CLOB	문자열

Amazon Kinesis Data Streams를 대상으로 사용 AWS Database Migration Service

를 AWS DMS 사용하여 데이터를 Amazon Kinesis 데이터 스트림으로 마이그레이션할 수 있습니다. Amazon Kinesis 데이터 스트림은 Amazon Kinesis Data Streams 서비스의 일부입니다. Kinesis 데이터 스트림을 사용하여 대규모 데이터 레코드 스트림을 실시간으로 수집하고 처리할 수 있습니다.

Kinesis 데이터 스트림은 샤드로 구성됩니다. 샤드는 스트림에서 고유하게 식별되는 데이터 레코드 시퀀스입니다. Amazon Kinesis Data Streams 내 샤드에 관한 자세한 내용은 Amazon Kinesis Data Streams 개발자 안내서의 샤드를 참조하세요.

AWS Database Migration Service JSON을 사용하여 Kinesis 데이터 스트림에 레코드를 게시합니다. 변환 중 AWS DMS 는 각 레코드를 소스 데이터베이스로부터 JSON 형식 또는 JSON_UNFORMATTED 메시지 형식의 속성-값 페어로 직렬화합니다. JSON_UNFORMATTED 메시지 형식은 줄 바꿈 구분 기호가 있는 한 줄 JSON 문자열입니다. 이를 통해 Amazon Data Firehose는 Kinesis 데이터를 Amazon S3 대상으로 전송한 다음 Amazon Athena를 비롯한 다양한 쿼리 엔진을 사용하여 쿼리할 수 있습니다.

객체 매핑을 사용하여 데이터를 지원되는 데이터 소스에서 대상 스트림으로 마이그레이션합니다. 객체 매핑을 통해 데이터 레코드를 스트림에 구조화하는 방법을 결정합니다. 또한 각 테이블에 대한 파티션 키를 정의합니다. Kinesis Data Streams는 이러한 키를 사용해 데이터를 샤드로 그룹화합니다.

Kinesis Data Streams 대상 엔드포인트에서 테이블을 생성하면 AWS DMS 소스 데이터베이스 엔드포인트와 동일한 수의 테이블이 생성됩니다. AWS DMS 또한 여러 Kinesis Data Streams 파라미터 값을 설정합니다. 테이블 생성 비용은 마이그레이션할 테이블 수와 데이터 양에 따라 달라집니다.

> **Note**
>
> AWS DMS 콘솔 또는 API의 SSL 모드 옵션은 Kinesis 및 DynamoDB와 같은 일부 데이터 스트리밍 및 NoSQL 서비스에는 적용되지 않습니다. 기본적으로 안전하므로 SSL 모드 설정이 없음으로 AWS DMS 표시됩니다 (SSL 모드=없음). 엔드포인트에서 SSL을 사용하기 위한 추가 구성을 제공할 필요는 없습니다. 예를 들어 Kinesis를 대상 엔드포인트로 사용하는 경우, 기본적으로 안전합니다. Kinesis에 대한 모든 API 호출은 SSL을 사용하므로 엔드포인트에 추가 SSL 옵션이 필요하지 않습니다. AWS DMS Kinesis 데이터 스트림에 연결할 때 기본적으로 AWS DMS 가 사용하는 HTTPS 프로토콜을 사용하여 SSL 엔드포인트를 통해 안전하게 데이터를 넣고 검색할 수 있습니다.

Kinesis Data Streams 엔드포인트 설정

Kinesis Data Streams 대상 엔드포인트를 사용하는 경우 API의 옵션을 사용하여 `KinesisSettings` 트랜잭션 및 제어 세부 정보를 가져올 수 있습니다. AWS DMS

연결을 설정하는 방법은 다음과 같습니다.

- AWS DMS 콘솔에서 엔드포인트 설정 사용.
- CLI에서 명령의 `kinesis-settings` CreateEndpoint옵션을 사용합니다.

CLI에서 다음 `kinesis-settings` 옵션의 요청 파라미터를 사용합니다.

> **Note**
>
> `IncludeNullAndEmpty` 엔드포인트 설정에 대한 지원은 AWS DMS 버전 3.4.1 및 이후 버전에서 제공됩니다. 그러나 Kinesis Data Streams 대상에 대한 다음 다른 엔드포인트 설정에 대한 지원은 에서 사용할 수 있습니다 AWS DMS.

- `MessageFormat` – 엔드포인트에 생성된 레코드의 출력 형식입니다. 메시지 형식은 `JSON`(기본값) 또는 `JSON_UNFORMATTED`(탭이 없는 한 줄)입니다.
- `IncludeControlDetails` – Kinesis 메시지 출력에서 테이블 정의, 열 정의, 테이블 및 열 변경 사항에 대한 자세한 제어 정보를 표시합니다. 기본값은 `false`입니다.
- `IncludeNullAndEmpty` – NULL 및 빈 열을 대상에 포함합니다. 기본값은 `false`입니다.
- `IncludePartitionValue` – 파티션 유형이 `schema-table-type`이 아닌 경우, Kinesis 메시지 출력 내에 파티션 값을 표시합니다. 기본값은 `false`입니다.
- `IncludeTableAlterOperations` – 제어 데이터의 테이블을 변경하는 데이터 정의 언어(DDL) 작업(예: `rename-table`, `drop-table`, `add-column`, `drop-column` 및 `rename-column`)을 포함합니다. 기본값은 `false`입니다.
- `IncludeTransactionDetails` – 소스 데이터베이스의 자세한 트랜잭션 정보를 제공합니다. 이 정보에는 커밋 타임스탬프, 로그 위치 및 `transaction_id`, `previous_transaction_id`, `transaction_record_id` (트랜잭션 내의 레코드 오프셋)에 대한 값이 포함됩니다. 기본값은 `false`입니다.
- `PartitionIncludeSchemaTable` – 파티션 유형이 `primary-key-type`인 경우 스키마 및 테이블 이름을 파티션 값에 접두사로 지정합니다. 이렇게 하면 Kinesis 샤드 간의 데이터 분산이 증가합니다. 예를 들어 SysBench 스키마에 수천 개의 테이블이 있고 각 테이블에 기본 키의 범위가 제한되어 있다고 가정하겠습니다. 이 경우 동일한 기본 키가 수천 개의 테이블에서 동일한 샤드로 전송되어 제한이 발생합니다. 기본값은 `false`입니다.

다음 예제는 AWS CLI를 사용하여 실행한 예제 `create-endpoint` 명령과 함께 사용 중인 `kinesis-settings` 옵션을 보여줍니다.

```
aws dms create-endpoint --endpoint-identifier=$target_name --engine-name kinesis --
endpoint-type target
--region us-east-1 --kinesis-settings
 ServiceAccessRoleArn=arn:aws:iam::333333333333:role/dms-kinesis-role,
StreamArn=arn:aws:kinesis:us-east-1:333333333333:stream/dms-kinesis-target-
doc,MessageFormat=json-unformatted,
```

```
IncludeControlDetails=true,IncludeTransactionDetails=true,IncludePartitionValue=true,PartitionI
IncludeTableAlterOperations=true
```

멀티스레드 전체 로드 작업 설정

전송 속도를 높이기 위해 Kinesis Data Streams 대상 인스턴스에 대한 멀티스레드 전체 로드를 AWS DMS 지원합니다. DMS는 다음과 같은 작업 설정을 통해 이 멀티스레딩을 지원합니다.

- `MaxFullLoadSubTasks` – 병렬로 로드할 최대 소스 테이블 수를 표시하려면 이 옵션을 사용합니다. DMS는 전용 하위 태스크를 사용하여 각 테이블을 해당 Kinesis 대상 테이블에 로드합니다. 기본값은 8이며 최대값은 49입니다.
- `ParallelLoadThreads`— 이 옵션을 사용하여 각 테이블을 Kinesis 대상 테이블에 로드하는 데 AWS DMS 사용하는 스레드 수를 지정합니다. Kinesis Data Streams 대상의 최대값은 32입니다. 이 최대 한도를 증가시키도록 요청할 수 있습니다.
- `ParallelLoadBufferSize` – 이 옵션을 사용하여 병렬 로드 스레드에서 데이터를 Kinesis 대상에 로드하기 위해 사용하는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 이 설정은 `ParallelLoadThreads`와 함께 사용하십시오. `ParallelLoadBufferSize`는 둘 이상의 스레드가 있는 경우에만 유효합니다.
- `ParallelLoadQueuesPerThread` – 각 동시 스레드가 액세스하는 대기열 수를 지정하여 대기열에서 데이터 레코드를 가져오고 대상에 대한 배치 로드를 생성하려면 이 옵션을 사용합니다. 기본값은 1입니다. 그러나 다양한 페이로드 크기의 Kinesis 대상에서 스레드당 대기열 수의 유효 범위는 5~512입니다.

멀티스레드 CDC 로드 작업 설정

`PutRecords` API 직접 호출의 동작을 수정하는 데 작업 설정을 사용하여 Kinesis와 같은 실시간 데이터 스트리밍 대상 엔드포인트에 대한 변경 데이터 캡처(CDC)의 성능을 향상시킬 수 있습니다. 이렇게 하려면 `ParallelApply*` 작업 설정을 사용하여 동시 스레드 수, 스레드당 대기열 및 버퍼에 저장할 레코드 수를 지정해야 합니다. 예를 들어 CDC 로드를 수행하고 128개의 스레드를 병렬로 적용한다고 가정하겠습니다. 또한 버퍼당 50개 레코드가 저장된 스레드당 64개 대기열에 액세스하려고 합니다.

CDC 성능을 높이기 위해 AWS DMS 은 다음과 같은 작업 설정을 지원합니다.

- `ParallelApplyThreads`— CDC 로드 중에 데이터 레코드를 Kinesis 대상 엔드포인트로 푸시하는 데 AWS DMS 사용하는 동시 스레드 수를 지정합니다. 기본값은 0이고 최대값은 32입니다.
- `ParallelApplyBufferSize` – CDC 로드 중에 Kinesis 대상 엔드포인트로 푸시할 동시 스레드에 대한 각 버퍼 대기열에 저장할 최대 레코드 수를 지정합니다. 기본값은 100이고 최대값은 1,000입니다. `ParallelApplyThreads`가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.

- `ParallelApplyQueuesPerThread` – 각 스레드가 대기열에서 데이터 레코드를 가져오고 CDC 중에 Kinesis 엔드포인트에 대한 배치 로드를 생성하기 위한 대기열 수를 지정합니다. 기본값은 1이고 최대값은 512입니다.

`ParallelApply*` 작업 설정을 사용할 때 `partition-key-type` 기본값은 테이블의 `schema-name.table-name`가 아니라 `primary-key`입니다.

Kinesis 데이터 스트림 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기

Kinesis 같은 데이터 스트리밍 대상에 CDC 업데이트를 작성하는 경우 업데이트를 통한 변경 전에 소스 데이터베이스 행의 원래 값을 볼 수 있습니다. 이를 가능하게 하기 위해 소스 데이터베이스 엔진에서 제공하는 데이터를 기반으로 업데이트 이벤트 전 이미지를 AWS DMS 채웁니다.

소스 데이터베이스 엔진에 따라 이전 이미지에 대해 서로 다른 양의 정보를 제공합니다.

- Oracle은 열이 변경되는 경우에만 열에 대한 업데이트를 제공합니다.
- PostgreSQL은 기본 키의 일부인 열에 대한 데이터(변경 여부에 관계 없음)만 제공합니다. 모든 열에 대한 데이터를 제공(변경 여부에 관계 없음)하려면 `REPLICA_IDENTITY`를 `DEFAULT` 대신 `FULL`로 설정해야 합니다. 참고로 각 테이블의 `REPLICA_IDENTITY` 설정을 신중하게 선택해야 합니다. `REPLICA_IDENTITY`를 `FULL`로 설정하면 모든 열 값이 계속해서 미리 쓰기 로깅(WAL)에 기록됩니다. 이로 인해 자주 업데이트되는 테이블에 성능 또는 리소스 문제가 발생할 수 있습니다.
- MySQL은 일반적으로 BLOB 및 CLOB 데이터 형식을 제외한 모든 열에 대한 데이터(변경 여부에 관계 없음)를 제공합니다.

이전 이미징을 활성화하여 소스 데이터베이스의 원래 값을 AWS DMS 출력에 추가하려면 `BeforeImageSettings` 태스크 설정 또는 `add-before-image-columns` 파라미터를 사용합니다. 이 파라미터는 열 변환 규칙을 적용합니다.

`BeforeImageSettings`는 다음과 같이 소스 데이터베이스 시스템에서 수집된 값을 사용하여 모든 업데이트 작업에 새 JSON 속성을 추가합니다.

```
"BeforeImageSettings": {
    "EnableBeforeImage": boolean,
    "FieldName": string,
    "ColumnFilter": pk-only (default) / non-lob / all (but only one)
}
```

> **Note**
>
> 전체 로드와 CDC AWS DMS 작업 (기존 데이터를 마이그레이션하고 진행 중인 변경 사항을 복제) 과 같이 CDC 구성 요소가 포함된 작업이나 CDC 전용 작업 (데이터 변경 사항만 복제) 에만 적용됩니다`BeforeImageSettings`. 전체 로드 전용 작업에는 `BeforeImageSettings`를 적용하지 마십시오.

`BeforeImageSettings` 옵션의 경우, 다음이 적용됩니다.

- 이전 이미징을 활성화하려면 `EnableBeforeImage` 옵션을 `true`로 설정합니다. 기본값은 `false`입니다.
- `FieldName` 옵션을 사용하여 새 JSON 속성에 이름을 지정합니다. `EnableBeforeImage`가 `true`인 경우 `FieldName`은 필수이며 비워 둘 수 없습니다.
- `ColumnFilter` 옵션은 이전 이미징을 사용하여 추가할 열을 지정합니다. 테이블 기본 키의 일부에 속하는 열만 추가하려면 기본값 `pk-only`를 사용하고, 이전 이미지 값이 있는 모든 열을 추가하려면 `all`을 사용합니다. 이전 이미지에는 CLOB 또는 BLOB와 같은 LOB 데이터 형식의 열이 포함되지 않는다는 점에 유의하십시오.

```
"BeforeImageSettings": {
    "EnableBeforeImage": true,
    "FieldName": "before-image",
    "ColumnFilter": "pk-only"
  }
```

> **Note**
>
> Amazon S3 대상은 `BeforeImageSettings`를 지원하지 않습니다. S3 대상의 경우 `add-before-image-columns` 변환 규칙만 사용하여 CDC 중에 이전 이미징을 수행합니다.

이전 이미지 변환 규칙 사용

작업 설정 대신 열 변환 규칙을 적용하는 `add-before-image-columns` 파라미터를 사용할 수 있습니다. 이 파라미터를 사용하면 Kinesis 같은 데이터 스트리밍 대상에서 CDC 중에 이전 이미징을 활성화할 수 있습니다.

변환 규칙에 add-before-image-columns를 사용하면 이전 이미지 결과를 보다 세밀하게 제어할 수 있습니다. 변환 규칙을 통해 객체 로케이터를 사용하여 규칙에 대해 선택한 테이블을 제어할 수 있습니다. 또한 변환 규칙을 함께 연결하여 테이블마다 서로 다른 규칙을 적용할 수 있습니다. 그런 다음, 다른 규칙을 사용하여 생성된 열을 조작할 수 있습니다.

> **Note**
>
> 동일한 작업 내에서 BeforeImageSettings 작업 설정과 함께 add-before-image-columns 파라미터를 사용해서는 안 됩니다. 단일 작업에는 이 파라미터 또는 설정 중 하나만 사용하고 둘 다 사용하지는 마십시오.

해당 열에 대해 add-before-image-columns 파라미터가 있는 transformation 규칙 유형이 before-image-def 단원을 제공해야 합니다. 다음은 그 한 예입니다.

```
{
  "rule-type": "transformation",
  …
  "rule-target": "column",
  "rule-action": "add-before-image-columns",
  "before-image-def":{
    "column-filter": one-of  (pk-only / non-lob / all),
    "column-prefix": string,
    "column-suffix": string,
  }
}
```

column-prefix의 값은 열 이름 앞에 추가되며, column-prefix의 기본값은 BI_입니다. column-suffix의 값은 열 이름 뒤에 추가되며, 기본값은 비어 있습니다. column-prefix와 column-suffix를 모두 빈 문자열로 설정하지 마십시오.

column-filter에 대해 하나의 값을 선택합니다. 테이블 기본 키의 일부인 열만 추가하려면 pk-only를 선택하고, LOB 유형이 아닌 열만 추가하려면 non-lob를 선택하고, 이전 이미지 값이 있는 모든 열을 추가하려면 all을 선택합니다.

이전 이미지 변환 규칙의 예

다음 예의 변환 규칙은 대상에서 BI_emp_no라는 새 열을 추가합니다. UPDATE employees SET emp_no = 3 WHERE emp_no = 1; 같은 문은 BI_emp_no 필드를 1로 채웁니다. Amazon S3 대상에 CDC 업데이트를 작성할 때 BI_emp_no 열을 통해 업데이트된 원래 행을 알 수 있습니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "%",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-target": "column",
      "object-locator": {
        "schema-name": "%",
        "table-name": "employees"
      },
      "rule-action": "add-before-image-columns",
      "before-image-def": {
        "column-prefix": "BI_",
        "column-suffix": "",
        "column-filter": "pk-only"
      }
    }
  ]
}
```

`add-before-image-columns` 규칙 작업 사용에 관한 자세한 내용은 변환 규칙 및 작업 단원을 참조하십시오.

Kinesis 데이터 스트림을 대상으로 사용하기 위한 사전 요구 사항 AWS Database Migration Service

Kinesis 데이터 스트림을 대상으로 사용하기 위한 IAM 역할 AWS Database Migration Service

Kinesis 데이터 스트림을 대상으로 AWS DMS 설정하기 전에 먼저 IAM 역할을 생성해야 합니다. 이 역할은 마이그레이션되는 Kinesis 데이터 스트림에 대한 액세스 권한을 위임하고 권한을 부여할 수 있어야 합니다. AWS DMS 최소 액세스 권한 집합이 다음 IAM 정책에 나와 있습니다.

```
{
   "Version": "2012-10-17",
   "Statement": [
   {
     "Sid": "1",
     "Effect": "Allow",
     "Principal": {
        "Service": "dms.amazonaws.com"
     },
   "Action": "sts:AssumeRole"
   }
]
}
```

Kinesis 데이터 스트림으로 마이그레이션할 때 사용하는 역할에는 다음 권한이 있어야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStream",
                "kinesis:PutRecord",
                "kinesis:PutRecords"
            ],
            "Resource": "arn:aws:kinesis:region:accountID:stream/streamName"
        }
    ]
}
```

Kinesis 데이터 스트림에 대한 타겟으로 액세스 AWS Database Migration Service

AWS DMS 버전 3.4.7 이상에서 Kinesis 엔드포인트에 연결하려면 다음 중 하나를 수행해야 합니다.

- VPC 엔드포인트를 사용하도록 DMS를 구성합니다. VPC 엔드포인트를 사용하도록 DMS를 구성하는 방법은 VPC 엔드포인트를 AWS DMS 소스 및 대상 엔드포인트로 구성 섹션을 참조하세요.

- 퍼블릭 경로를 사용하도록 DMS를 구성합니다. 즉, 복제 인스턴스를 퍼블릭으로 설정합니다. 퍼블릭 복제 인스턴스에 대한 자세한 내용은 퍼블릭 및 프라이빗 복제 인스턴스 섹션을 참조하세요.

Kinesis Data Streams를 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

Kinesis Data Streams를 대상으로 사용할 때에는 다음 제한 사항이 적용됩니다.

- AWS DMS 트랜잭션과 관계없이 소스 데이터베이스의 단일 레코드에 대한 각 업데이트를 지정된 Kinesis 데이터 스트림의 단일 데이터 레코드로 게시합니다. 그러나 `KinesisSettings` API의 관련 파라미터를 사용하여 각 데이터 레코드에 대한 트랜잭션 세부 정보를 포함할 수 있습니다.
- 전체 LOB 모드는 지원되지 않습니다.
- 지원되는 최대 LOB 크기는 1MB입니다.
- Kinesis 데이터 스트림은 중복 제거를 지원하지 않습니다. 스트림의 데이터를 소비하는 애플리케이션은 중복 레코드를 처리해야 합니다. 자세한 내용은 Amazon Kinesis Data Streams 개발자 안내서의 중복 레코드 처리를 참조하세요.
- AWS DMS 다음과 같은 두 가지 형식의 파티션 키를 지원합니다.
  - `SchemaName.TableName`: 스키마와 테이블 이름의 조합입니다.
  - `${AttributeName}`: JSON 내 한 필드의 값 또는 소스 데이터베이스 내 테이블의 기본 키입니다.
- Kinesis Data Streams 내 저장 데이터를 암호화하는 방법에 관한 자세한 내용은 AWS Key Management Service 개발자 안내서의 Kinesis Data Streams의 데이터 보호를 참조하십시오.
- `BatchApply`는 Kinesis 엔드포인트에는 지원되지 않습니다. Kinesis 대상에 대해 Batch Apply(예: `BatchApplyEnabled` 대상 메타데이터 작업 설정)를 사용하면 데이터 손실이 발생할 수 있습니다.
- Kinesis 타겟은 복제 인스턴스와 동일한 AWS 계정의 Kinesis 데이터 스트림에만 지원됩니다. AWS 리전
- MySQL 원본에서 마이그레이션하는 경우 데이터에는 CLOB 및 BLOB 데이터 BeforeImage 형식이 포함되지 않습니다. 자세한 설명은 Kinesis 데이터 스트림 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기 섹션을 참조하세요.
- AWS DMS 16자리가 넘는 `BigInt` 데이터 유형 값의 마이그레이션은 지원하지 않습니다. 이 제한을 해결하기 위해 다음 변환 규칙을 사용하여 `BigInt` 열을 문자열로 변환할 수 있습니다. 변환 규칙에 대한 자세한 내용은 변환 규칙 및 작업 섹션을 참조하세요.

```
{
    "rule-type": "transformation",
```

```
    "rule-id": "id",
    "rule-name": "name",
    "rule-target": "column",
    "object-locator": {
        "schema-name": "valid object-mapping rule action",
        "table-name": "",
        "column-name": ""
    },
    "rule-action": "change-data-type",
    "data-type": {
        "type": "string",
        "length": 20
    }
}
```

객체 매핑을 사용하여 데이터를 Kinesis 데이터 스트림으로 마이그레이션

AWS DMS 는 테이블 매핑 규칙을 사용하여 소스의 데이터를 대상 Kinesis 데이터 스트림으로 매핑합니다. 데이터를 대상 스트림에 매핑하기 위해 객체 매핑이라는 테이블 매핑 규칙 유형을 사용합니다. 객체 매핑을 사용하여 소스의 데이터 레코드를 Kinesis 데이터 스트림에 게시된 데이터 레코드로 매핑하는 방법을 지정합니다.

Kinesis 데이터 스트림에는 파티션 키 외에 별다른 사전 설정 구조가 없습니다. 객체 매핑 규칙에서 데이터 레코드의 `partition-key-type`에 사용할 수 있는 값은 `schema-table`, `transaction-id`, `primary-key`, `constant` 및 `attribute-name`입니다.

객체 매핑 규칙을 만들려면 `rule-type`을 `object-mapping`으로 지정합니다. 이 규칙은 사용할 객체 매핑의 유형을 지정합니다.

이 규칙의 구조는 다음과 같습니다.

```
{
    "rules": [
        {
            "rule-type": "object-mapping",
            "rule-id": "id",
            "rule-name": "name",
            "rule-action": "valid object-mapping rule action",
            "object-locator": {
                "schema-name": "case-sensitive schema name",
                "table-name": ""
```

```
            }
        }
    ]
}
```

AWS DMS 현재 파라미터에 대한 유일한 유효 값은 `map-record-to-record` 및 `map-record-to-document` 입니다. `rule-action` 이러한 설정은 `exclude-columns` 속성 목록의 일부로 제외되지 않은 값에 영향을 줍니다. `map-record-to-record`및 `map-record-to-document` 값은 이러한 레코드를 기본적으로 AWS DMS 처리하는 방법을 지정합니다. 이러한 값은 어떤 식으로든 속성 매핑에 영향을 미치지 않습니다.

관계형 데이터베이스에서 Kinesis 데이터 스트림으로 마이그레이션할 때 `map-record-to-record`를 사용합니다. 이러한 규칙 유형은 관계형 데이터베이스의 `taskResourceId.schemaName.tableName` 값을 Kinesis 데이터 스트림의 파티션 키로 사용하며 소스 데이터베이스의 각 열마다 속성을 하나씩 생성합니다.

`map-record-to-record`를 사용하는 경우 다음 사항에 유의하세요.

- 이 설정은 `exclude-columns` 목록에서 제외된 열에만 영향을 줍니다.
- 이러한 모든 열에 대해 대상 주제에 해당 속성을 AWS DMS 만듭니다.
- AWS DMS 소스 열이 속성 매핑에 사용되는지 여부에 관계없이 해당하는 이 속성을 생성합니다.

속성 이름 "_doc"를 사용하여 소스 열을 적절한 대상 스트림의 단일 플랫 문서에 넣는 데 `map-record-to-document`를 사용합니다. AWS DMS 는 "_doc"라는 소스의 단일 플랫 맵에 데이터를 배치합니다. 이 배치는 `exclude-columns` 속성 목록에 나열되지 않은 소스 테이블의 모든 열에 적용됩니다.

`map-record-to-record`를 이해하는 한 가지 방법은 작업 중일 때 관찰하는 것입니다. 이 예에서는 다음 구조와 데이터를 사용하여 관계형 데이터베이스 테이블 행에서 시작한다고 가정합니다.

FirstName	LastName	StoreId	HomeAddress	HomePhone	WorkAddress	WorkPhone	DateofBirth
Randy	Marsh	5	221B Baker Street	1234567890	31 Spooner Street, Quahog	9876543210	1988/02/29

이 정보를 Test라는 스키마에서 Kinesis 데이터 스트림으로 마이그레이션하려면 데이터를 대상 스트림으로 매핑하는 규칙을 생성해야 합니다. 다음 규칙은 그 매핑 과정을 보여 줍니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "rule-action": "include",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            }
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "2",
            "rule-name": "DefaultMapToKinesis",
            "rule-action": "map-record-to-record",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "Customers"
            }
        }
    ]
}
```

다음은 Kinesis 데이터 스트림에 생성되는 레코드 형식을 예시합니다.

- StreamName: XXX
- PartitionKey: 테스트. 고객 //스키마 이름. 테이블 이름
- 날짜: //다음의 JSON 메시지

```
{
   "FirstName": "Randy",
   "LastName": "Marsh",
   "StoreId":  "5",
   "HomeAddress": "221B Baker Street",
   "HomePhone": "1234567890",
   "WorkAddress": "31 Spooner Street, Quahog",
```

```
      "WorkPhone": "9876543210",
      "DateOfBirth": "02/29/1988"
   }
```

동일한 규칙을 사용하지만 rule-action 파라미터를 map-record-to-document로 변경하거나 특정 열을 제외한다고 가정해 보겠습니다. 다음 규칙은 그 매핑 과정을 보여 줍니다.

```
{
 "rules": [
    {
   "rule-type": "selection",
   "rule-id": "1",
   "rule-name": "1",
   "rule-action": "include",
   "object-locator": {
    "schema-name": "Test",
    "table-name": "%"
   }
  },
  {
   "rule-type": "object-mapping",
   "rule-id": "2",
   "rule-name": "DefaultMapToKinesis",
   "rule-action": "map-record-to-document",
   "object-locator": {
    "schema-name": "Test",
    "table-name": "Customers"
   },
   "mapping-parameters": {
    "exclude-columns": [
     "homeaddress",
     "homephone",
     "workaddress",
     "workphone"
    ]
   }
  }
 ]
}
```

이 경우, exclude-columns 파라미터에 나열되어 있지 않은 열인 FirstName, LastName, StoreId 및 DateOfBirth는 _doc로 매핑됩니다. 다음은 생성되는 레코드 형식을 보여줍니다.

```
      {
           "data":{
               "_doc":{
                   "FirstName": "Randy",
                   "LastName": "Marsh",
                   "StoreId":  "5",
                   "DateOfBirth": "02/29/1988"
               }
           }
       }
```

속성 매핑으로 날짜 재구성

속성 맵을 사용하여 날짜를 Kinesis 데이터 스트림으로 마이그레이션하는 동안 날짜를 재구성할 수 있습니다. 예를 들어 소스의 필드 몇 개를 대상의 단일 필드로 묶어야 하는 경우도 있을 것입니다. 다음의 속성 맵은 날짜를 재구성하는 방법을 보여줍니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "rule-action": "include",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            }
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "2",
            "rule-name": "TransformToKinesis",
            "rule-action": "map-record-to-record",
            "target-table-name": "CustomerData",
            "object-locator": {
                "schema-name": "Test",
```

```
                "table-name": "Customers"
            },
            "mapping-parameters": {
                "partition-key-type": "attribute-name",
                "partition-key-name": "CustomerName",
                "exclude-columns": [
                    "firstname",
                    "lastname",
                    "homeaddress",
                    "homephone",
                    "workaddress",
                    "workphone"
                ],
                "attribute-mappings": [
                    {
                        "target-attribute-name": "CustomerName",
                        "attribute-type": "scalar",
                        "attribute-sub-type": "string",
                        "value": "${lastname}, ${firstname}"
                    },
                    {
                        "target-attribute-name": "ContactDetails",
                        "attribute-type": "document",
                        "attribute-sub-type": "json",
                        "value": {
                            "Home": {
                                "Address": "${homeaddress}",
                                "Phone": "${homephone}"
                            },
                            "Work": {
                                "Address": "${workaddress}",
                                "Phone": "${workphone}"
                            }
                        }
                    }
                ]
            }
        }
    ]
}
```

partition-key에 상수 값을 설정하려면 partition-key 값을 지정합니다. 예를 들어 이 방법을 통해 모든 데이터를 단일 샤드에 저장할 수 있습니다. 다음 매핑은 이러한 접근 방식을 보여줍니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "1",
            "rule-name": "TransformToKinesis",
            "rule-action": "map-record-to-document",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "Customer"
            },
            "mapping-parameters": {
                "partition-key": {
                    "value": "ConstantPartitionKey"
                },
                "exclude-columns": [
                    "FirstName",
                    "LastName",
                    "HomeAddress",
                    "HomePhone",
                    "WorkAddress",
                    "WorkPhone"
                ],
                "attribute-mappings": [
                    {
                        "attribute-name": "CustomerName",
                        "value": "${FirstName},${LastName}"
                    },
                    {
                        "attribute-name": "ContactDetails",
                        "value": {
                            "Home": {
                                "Address": "${HomeAddress}",
```

```
                            "Phone": "${HomePhone}"
                        },
                        "Work": {
                            "Address": "${WorkAddress}",
                            "Phone": "${WorkPhone}"
                        }
                    }
                },
                {
                    "attribute-name": "DateOfBirth",
                    "value": "${DateOfBirth}"
                }
            ]
        }
    }
  ]
}
```

> **Note**
>
> 특정 테이블을 위한 제어 레코드에 대한 `partition-key` 값은 `TaskId.SchemaName.TableName`입니다. 특정 작업을 위한 제어 레코드에 대한 `partition-key` 값은 해당 레코드의 `TaskId`입니다. `partition-key` 값을 객체 매핑에 지정해도 제어 레코드에 대한 `partition-key`에는 영향이 없습니다.

Kinesis Data Streams에 대한 메시지 형식

JSON 출력은 단지 키-값 페어의 목록입니다. JSON_UNFORMATTED 메시지 형식은 줄 바꿈 구분 기호가 있는 한 줄 JSON 문자열입니다.

AWS DMS Kinesis Data Streams의 데이터를 더 쉽게 사용할 수 있도록 다음과 같은 예약된 필드를 제공합니다.

RecordType

레코드 유형은 데이터나 제어일 수 있습니다. 데이터 레코드는 소스 내의 실제 행을 나타냅니다. 제어 레코드는 스트림 내의 중요 이벤트를 나타냅니다(예: 작업의 재시작).

Operation

데이터 레코드의 경우, 작업은 `load`, `insert`, `update` 또는 `delete`일 수 있습니다.

데이터 레코드의 경우, 작업은 `create-table`, `rename-table`, `drop-table`, `change-columns`, `add-column`, `drop-column`, `rename-column` 또는 `column-type-change`일 수 있습니다.

SchemaName

레코드에 대한 원본 스키마입니다. 제어 레코드의 경우, 이 필드는 비워둘 수 있습니다.

TableName

레코드에 대한 소스 테이블입니다. 제어 레코드의 경우, 이 필드는 비워둘 수 있습니다.

Timestamp

JSON 메시지가 구성된 경우의 타임스탬프입니다. 이 필드는 ISO 8601 형식으로 구성되었습니다.

아파치 카프카를 타겟으로 사용 AWS Database Migration Service

를 사용하여 데이터를 Apache AWS DMS Kafka 클러스터로 마이그레이션할 수 있습니다. Apache Kafka는 분산 스트리밍 플랫폼입니다. Apache Kafka를 사용하여 스트리밍 데이터의 실시간 수집 및 처리를 수행할 수 있습니다.

AWS 또한 타겟으로 사용할 수 있는 Apache Kafka용 아마존 매니지드 스트리밍 (Amazon MSK) 도 제공합니다. AWS DMS Amazon MSK는 Apache Kafka 인스턴스의 구현 및 관리를 단순화하는 완전 관리형 Apache Kafka 스트리밍 서비스입니다. 오픈 소스 Apache Kafka 버전에서 작동하며, 다른 Apache Kafka 인스턴스와 마찬가지로 Amazon MSK 인스턴스에 AWS DMS 대상으로 액세스할 수 있습니다. 자세한 내용은 Amazon Managed Streaming for Apache Kafka 개발자 안내서의 Amazon MSK란 무엇인가요?를 참조하세요.

Kafka 클러스터는 여러 파티션으로 구분된 주제라는 범주에 레코드 스트림을 저장합니다. 파티션은 주제에서 고유하게 식별되는 데이터 레코드(메시지) 시퀀스입니다. 파티션은 주제 레코드의 병렬 처리가 가능하도록 한 클러스터의 여러 브로커로 분산될 수 있습니다. Apache Kafka의 주제 및 파티션과 분산에 관한 자세한 내용은 주제 및 로그와 분산을 참조하십시오.

Kafka 클러스터는 Amazon MSK 인스턴스, Amazon EC2 인스턴스에서 실행되는 클러스터 또는 온프레미스 클러스터일 수 있습니다. Amazon MSK 인스턴스 또는 Amazon EC2 인스턴스의 클러스터는 동일한 VPC에 있을 수도 있고 다른 VPC에 있을 수도 있습니다. 클러스터가 온프레미스인 경우 복제 인스턴스용 자체 온프레미스 이름 서버를 사용하여 클러스터의 호스트 이름을 확인할 수 있습니다. 복제 인스턴스에서 이름 서버를 설정하는 방법에 관한 자세한 내용은 자체 온프레미스 이름 서버 사용 단원을 참조하십시오. 네트워크 설정에 관한 자세한 내용은 복제 인스턴스용으로 네트워크 설정 단원을 참조하십시오.

Amazon MSK 클러스터를 사용할 때는 해당 보안 그룹이 복제 인스턴스에서의 액세스를 허용하는지 확인하십시오. Amazon MSK 클러스터의 보안 그룹 변경에 관한 자세한 내용은 Amazon MSK 클러스터의 보안 그룹 변경을 참조하십시오.

AWS Database Migration Service JSON을 사용하여 Kafka 주제에 레코드를 게시합니다. 변환 중 AWS DMS 는 각 레코드를 소스 데이터베이스로부터 JSON 형식의 속성-값 페어로 직렬화합니다.

지원되는 데이터 소스에서 대상 Kafka 클러스터로 데이터를 마이그레이션하려면 객체 매핑을 사용합니다. 객체 매핑을 통해 데이터 레코드를 대상 주제에 구조화하는 방법을 결정합니다. 또한 각 테이블에 대한 파티션 키를 정의합니다. Apache Kafka가 이러한 테이블을 사용해 데이터를 파티션으로 그룹화합니다.

현재는 작업당 단일 주제를 AWS DMS 지원합니다. 테이블이 여러 개 있는 단일 작업의 경우 모든 메시지가 단일 주제로 이동합니다. 각 메시지에는 대상 스키마와 테이블을 식별하는 메타데이터 섹션이 포함되어 있습니다. AWS DMS 버전 3.4.6 이상에서는 객체 매핑을 사용한 멀티토픽 복제를 지원합니다. 자세한 설명은 객체 매핑을 사용한 멀티주제 복제 섹션을 참조하세요.

Apache Kafka 엔드포인트 설정

AWS DMS 콘솔의 엔드포인트 설정 또는 CLI의 `--kafka-settings` 옵션을 통해 연결 세부 정보를 지정할 수 있습니다. 각 설정의 요구 사항은 다음과 같습니다.

- `Broker` – Kafka 클러스터에 있는 하나 이상의 브로커 위치를 쉼표로 구분된 각 *`broker-hostname`*`:`*`port`*의 목록 형식으로 지정합니다. 예를 들면, `"ec2-12-345-678-901.compute-1.amazonaws.com:2345,ec2-10-987-654-321.compute-1.a` 니다. 이 설정은 클러스터 내 일부 또는 모든 브로커의 위치를 지정할 수 있습니다. 모든 클러스터 브로커는 주제로 마이그레이션된 데이터 레코드의 분할을 처리하기 위해 통신합니다.
- `Topic` – (선택 사항) 최대 길이 255개 문자 및 기호로 주제 이름을 지정합니다. 마침표(.), 밑줄(_) 및 빼기(-)를 사용할 수 있습니다. 마침표(.) 또는 밑줄(_)이 있는 주제 이름이 내부 데이터 구조에서 충돌할 수 있습니다. 주제 이름에 이러한 기호 중 하나만 사용하고 둘 다 사용하지는 마십시오. 주제 이름을 지정하지 않는 경우 를 마이그레이션 주제로 AWS DMS 사용합니다`"kafka-default-topic"`.

> **Note**
>
> 지정한 마이그레이션 주제 또는 기본 주제를 AWS DMS 만들려면 Kafka 클러스터 구성의 `auto.create.topics.enable = true` 일부로 설정하십시오. 자세한 내용은 Apache Kafka를 대상으로 사용할 때의 제한 사항 AWS Database Migration Service 단원을 참조하세요.

- `MessageFormat` – 엔드포인트에 생성된 레코드의 출력 형식입니다. 메시지 형식은 `JSON`(기본값) 또는 `JSON_UNFORMATTED`(탭이 없는 한 줄)입니다.
- `MessageMaxBytes` – 엔드포인트에 생성된 레코드의 최대 크기(바이트)입니다. 기본값은 1,000,000입니다.

> **Note**
>
> AWS CLI/SDK를 사용하여 기본값이 아닌 값으로 변경할 `MessageMaxBytes` 수만 있습니다. 예를 들어 기존 Kafka 엔드포인트를 수정하고 `MessageMaxBytes`를 변경하려면 다음 명령을 사용합니다.
>
> ```
> aws dms modify-endpoint --endpoint-arn your-endpoint
> --kafka-settings Broker="broker1-server:broker1-port,broker2-server:broker2-
> port,...",
> Topic=topic-name,MessageMaxBytes=integer-of-max-message-size-in-bytes
> ```

- `IncludeTransactionDetails` – 소스 데이터베이스의 자세한 트랜잭션 정보를 제공합니다. 이 정보에는 커밋 타임스탬프, 로그 위치 및 `transaction_id`, `previous_transaction_id`, `transaction_record_id`(트랜잭션 내의 레코드 오프셋)에 대한 값이 포함됩니다. 기본값은 `false`입니다.
- `IncludePartitionValue` – 파티션 유형이 `schema-table-type`이 아닌 경우 Kafka 메시지 출력에 파티션 값을 표시합니다. 기본값은 `false`입니다.
- `PartitionIncludeSchemaTable` – 파티션 유형이 `primary-key-type`인 경우 스키마 및 테이블 이름을 파티션 값에 접두사로 지정합니다. 이렇게 하면 Kafka 파티션 간의 데이터 분산이 증가합니다. 예를 들어 SysBench 스키마에 수천 개의 테이블이 있고 각 테이블에 기본 키의 범위가 제한되어 있다고 가정하겠습니다. 이 경우 동일한 기본 키가 수천 개의 테이블에서 동일한 파티션으로 전송되어 제한이 발생합니다. 기본값은 `false`입니다.
- `IncludeTableAlterOperations` – 제어 데이터의 테이블을 변경하는 데이터 정의 언어(DDL) 작업(예: `rename-table`, `drop-table`, `add-column`, `drop-column` 및 `rename-column`)을 포함합니다. 기본값은 `false`입니다.
- `IncludeControlDetails` – Kafka 메시지 출력에서 테이블 정의, 열 정의, 테이블 및 열 변경 사항에 관한 자세한 제어 정보를 표시합니다. 기본값은 `false`입니다.
- `IncludeNullAndEmpty` – NULL 및 빈 열을 대상에 포함합니다. 기본값은 `false`입니다.
- `SecurityProtocol` – 전송 계층 보안(TLS)을 사용하여 Kafka 대상 엔드포인트에 대한 보안 연결을 설정합니다. 옵션에는 `ssl-authentication`, `ssl-encryption` 및 `sasl-ssl`이 포함됩니다. `sasl-ssl`을 사용하려면 `SaslUsername`과 `SaslPassword`가 필요합니다.

- `SslEndpointIdentificationAlgorithm`— 인증서에 대한 호스트 이름 확인을 설정합니다. 이 설정은 AWS DMS 버전 3.5.1 이상에서 지원됩니다. 옵션에는 다음 사항이 포함됩니다.
  - `NONE`: 클라이언트 연결에서 브로커의 호스트 이름 확인을 비활성화합니다.
  - `HTTPS`: 클라이언트 연결에서 브로커의 호스트 이름 확인을 활성화합니다.

설정을 사용하여 전송 속도를 높일 수 있습니다. 이를 위해 AWS DMS 는 Apache Kafka 대상 클러스터에 대한 멀티스레드 전체 로드를 지원합니다. AWS DMS 는 다음을 포함하는 작업 설정으로 이 멀티 스레딩을 지원합니다.

- `MaxFullLoadSubTasks`— 이 옵션을 사용하여 병렬로 로드할 소스 테이블의 최대 수를 지정합니다. AWS DMS 전용 하위 작업을 사용하여 각 테이블을 해당 Kafka 대상 테이블에 로드합니다. 기본값은 8이며 최대값은 49입니다.
- `ParallelLoadThreads`— 이 옵션을 사용하여 각 테이블을 Kafka 대상 테이블에 로드하는 데 AWS DMS 사용하는 스레드 수를 지정합니다. Apache Kafka 대상의 최대값은 32입니다. 이 최대 한도를 증가시키도록 요청할 수 있습니다.
- `ParallelLoadBufferSize` – 이 옵션을 사용하여 병렬 로드 스레드에서 데이터를 Kafka 대상에 로드하기 위해 사용하는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 이 설정은 `ParallelLoadThreads`와 함께 사용하십시오. `ParallelLoadBufferSize`는 둘 이상의 스레드가 있는 경우에만 유효합니다.
- `ParallelLoadQueuesPerThread` – 각 동시 스레드가 액세스하는 대기열 수를 지정하여 대기열에서 데이터 레코드를 가져오고 대상에 대한 배치 로드를 생성하려면 이 옵션을 사용합니다. 기본 값은 1입니다. 최대 한도는 512입니다.

병렬 스레드 및 대량 작업에 대한 작업 설정을 조정하여 Kafka 엔드포인트의 변경 데이터 캡처(CDC) 성능을 개선할 수 있습니다. 이렇게 하려면 `ParallelApply*` 작업 설정을 사용하여 동시 스레드 수, 스레드당 대기열 및 버퍼에 저장할 레코드 수를 지정해야 합니다. 예를 들어 CDC 로드를 수행하고 128개의 스레드를 병렬로 적용한다고 가정하겠습니다. 또한 버퍼당 50개 레코드가 저장된 스레드당 64개 대기열에 액세스하려고 합니다.

CDC 성능을 높이기 위해 AWS DMS 은 다음과 같은 작업 설정을 지원합니다.

- `ParallelApplyThreads`— CDC 로드 중에 데이터 레코드를 Kafka 대상 엔드포인트로 푸시하는 데 AWS DMS 사용하는 동시 스레드 수를 지정합니다. 기본값은 0이고 최대값은 32입니다.
- `ParallelApplyBufferSize` – CDC 로드 중에 Kafka 대상 엔드포인트로 푸시할 동시 스레드에 대한 각 버퍼 대기열에 저장할 최대 레코드 수를 지정합니다. 기본값은 100이고 최대값은 1,000입니다. `ParallelApplyThreads`가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.

- `ParallelApplyQueuesPerThread` – 각 스레드가 대기열에서 데이터 레코드를 가져오고 CDC 중에 Kafka 엔드포인트에 대한 배치 로드를 생성하기 위한 대기열 수를 지정합니다. 기본 값은 1입니다. 최대 한도는 512입니다.

`ParallelApply*` 작업 설정을 사용할 때 `partition-key-type` 기본값은 테이블의 `schema-name.table-name`가 아니라 `primary-key`입니다.

전송 계층 보안(TLS)을 사용하여 Kafka에 연결

Kafka 클러스터는 전송 계층 보안(TLS)을 사용한 보안 연결만 허용합니다. DMS를 사용하면 다음 세 가지 보안 프로토콜 옵션 중 하나를 사용하여 Kafka 엔드포인트 연결을 보호할 수 있습니다.

SSL 암호화(**`server-encryption`**)

클라이언트는 서버의 인증서를 통해 서버 ID를 확인합니다. 그러면 서버와 클라이언트 간에 암호화된 연결이 이루어집니다.

SSL 인증(**`mutual-authentication`**)

서버와 클라이언트는 자체 인증서를 통해 서로 ID를 확인합니다. 그러면 서버와 클라이언트 간에 암호화된 연결이 이루어집니다.

SASL-SSL(**`mutual-authentication`**)

SASL(Simple Authentication and Security Layer) 메서드는 클라이언트 인증서를 사용자 이름 및 암호로 대체하여 클라이언트 ID를 검증합니다. 특히 서버가 클라이언트의 ID를 검증할 수 있도록 서버가 등록한 사용자 이름과 암호를 제공합니다. 그러면 서버와 클라이언트 간에 암호화된 연결이 이루어집니다.

> **Important**
>
> Apache Kafka와 Amazon MSK는 확인된 인증서를 수락합니다. 이는 Kafka와 Amazon MSK에서 해결해야 하는 알려진 제한 사항입니다. 자세한 내용은 Apache Kafka 문제, KAFKA-3700단원을 참조하세요.
>
> Amazon MSK를 사용하는 경우 이 알려진 제한에 대한 해결 방법으로 액세스 제어 목록(ACL)을 사용하는 것이 좋습니다. ACL 사용에 관한 자세한 내용은 Amazon Managed Streaming for Apache Kafka 개발자 안내서의 Apache Kafka ACL 단원을 참조하세요.
>
> 자체 관리형 Kafka 클러스터를 사용하는 경우, 클러스터 구성에 관한 자세한 내용은 2018년 10월 21일자 설명을 참조하십시오.

Amazon MSK 또는 자체 관리형 Kafka 클러스터에서 SSL 암호화 사용

Amazon MSK 또는 자체 관리형 Kafka 클러스터에 대한 엔드포인트 연결을 보호하는 데 SSL 암호화를 사용할 수 있습니다. SSL 암호화 인증 방법을 사용하면 클라이언트가 서버의 인증서를 통해 서버의 ID를 검증합니다. 그러면 서버와 클라이언트 간에 암호화된 연결이 이루어집니다.

SSL 암호화를 사용하여 Amazon MSK에 연결하려면

- 대상 Kafka 엔드포인트를 생성할 때 `ssl-encryption` 옵션을 사용하여 보안 프로토콜 엔드포인트 설정(`SecurityProtocol`)을 설정합니다.

  다음 JSON 예제는 보안 프로토콜을 SSL 암호화로 설정합니다.

```
"KafkaSettings": {
    "SecurityProtocol": "ssl-encryption",
}
```

자체 관리형 Kafka 클러스터에 SSL 암호화를 사용하려면

1. 온프레미스 Kafka 클러스터에서 프라이빗 CA(인증 기관)를 사용하는 경우 프라이빗 CA 인증서를 업로드하고 Amazon 리소스 이름(ARN)을 받으십시오.
2. 대상 Kafka 엔드포인트를 생성할 때 `ssl-encryption` 옵션을 사용하여 보안 프로토콜 엔드포인트 설정(`SecurityProtocol`)을 설정합니다. 다음 JSON 예제는 보안 프로토콜을 `ssl-encryption`으로 설정합니다.

   ```
   "KafkaSettings": {
       "SecurityProtocol": "ssl-encryption",
   }
   ```

3. 프라이빗 CA를 사용하는 경우 위의 첫 번째 단계에서 얻은 ARN에 `SslCaCertificateArn`을 설정하십시오.

SSL 인증 사용

Amazon MSK 또는 자체 관리형 Kafka 클러스터에 대한 엔드포인트 연결을 보호하는 데 SSL 인증을 사용할 수 있습니다.

Amazon MSK에 연결할 때 SSL 인증을 사용한 클라이언트 인증 및 암호화를 활성화하려면 다음을 수행하십시오.

- Kafka용 프라이빗 키와 퍼블릭 인증서를 준비합니다.
- 인증서를 DMS Certificate Manager에 업로드합니다.
- Kafka 엔드포인트 설정에 지정된 해당 인증서 ARN을 사용하여 Kafka 대상 엔드포인트를 생성합니다.

Amazon MSK용 프라이빗 키와 퍼블릭 인증서를 준비하려면

1. Amazon Managed Streaming for Apache Kafka 개발자 안내서의 클라이언트 인증 단원에 있는 1~9단계에 설명된 대로 EC2 인스턴스를 생성하고 클라이언트가 인증을 사용하도록 설정합니다.

   이러한 단계를 완료하면 인증서-ARN(ACM에 저장된 공개 인증서 ARN)과 `kafka.client.keystore.jks` 파일에 포함된 프라이빗 키를 갖게 됩니다.
2. 다음 명령을 사용하여 퍼블릭 인증서를 가져오고 인증서를 `signed-certificate-from-acm.pem` 파일에 복사합니다.

   ```
   aws acm-pca get-certificate --certificate-authority-arn Private_CA_ARN --
   certificate-arn Certificate_ARN
   ```

   이 명령은 다음 예와 유사한 정보를 반환합니다.

   ```
   {"Certificate": "123", "CertificateChain": "456"}
   ```

   그런 다음, `"123"`과 동일한 데이터를 `signed-certificate-from-acm.pem` 파일에 복사합니다.
3. 다음 예제와 같이 `kafka.client.keystore.jks to keystore.p12`에서 `msk-rsa` 키를 가져와서 프라이빗 키를 받습니다.

   ```
   keytool -importkeystore \
   -srckeystore kafka.client.keystore.jks \
   -destkeystore keystore.p12 \
   ```

```
-deststoretype PKCS12 \
-srcalias msk-rsa-client \
-deststorepass test1234 \
-destkeypass test1234
```

4. 다음 명령을 사용하여 keystore.p12를 .pem 형식으로 내보냅니다.

```
Openssl pkcs12 -in keystore.p12 -out encrypted-private-client-key.pem -nocerts
```

Enter PEM 암호문 메시지가 나타나고 인증서를 암호화하는 데 적용되는 키를 식별합니다.

5. .pem 파일에서 백 속성 및 키 속성을 제거하여 첫 번째 줄이 다음 문자열로 시작되도록 합니다.

```
                                ---BEGIN ENCRYPTED PRIVATE KEY---
```

퍼블릭 인증서와 프라이빗 키를 DMS 인증서 관리자에 업로드하고 Amazon MSK에 대한 연결을 테스트하려면

1. 다음 명령을 사용하여 DMS 인증서 관리자에 업로드합니다.

```
aws dms import-certificate --certificate-identifier signed-cert --certificate-pem
 file://path to signed cert
aws dms import-certificate --certificate-identifier private-key –certificate-pem
 file://path to private key
```

2. Amazon MSK 대상 엔드포인트를 생성하고 연결을 테스트하여 TLS 인증이 작동하는지 확인합니다.

```
aws dms create-endpoint --endpoint-identifier $endpoint-identifier --engine-name
 kafka --endpoint-type target --kafka-settings
'{"Broker": "b-0.kafka260.aaaaa1.a99.kafka.us-east-1.amazonaws.com:0000",
 "SecurityProtocol":"ssl-authentication",
"SslClientCertificateArn": "arn:aws:dms:us-east-1:012346789012:cert:",
"SslClientKeyArn": "arn:aws:dms:us-
east-1:0123456789012:cert:","SslClientKeyPassword":"test1234"}'
aws dms test-connection -replication-instance-arn=$rep_inst_arn –endpoint-arn=
$kafka_tar_arn_msk
```

> **Important**
>
> 자체 관리형 Kafka 클러스터에 대한 연결을 보호하는 데 SSL 인증을 사용할 수 있습니다. 경우에 따라 온프레미스 Kafka 클러스터에서 프라이빗 CA(인증 기관)를 사용할 수 있습니다. 그렇다면 CA 체인, 퍼블릭 인증서, 프라이빗 키를 DMS 인증서 관리자에게 업로드하세요. 그런 다음 온프레미스 Kafka 대상 엔드포인트를 생성할 때 엔드포인트 설정에서 해당 Amazon 리소스 이름(ARN)을 사용합니다.

자체 관리형 Kafka 클러스터에 사용할 프라이빗 키와 서명된 인증서를 준비하려면

1. 다음 예제에서 보는 것처럼 키 페어를 생성합니다.

```
keytool -genkey -keystore kafka.server.keystore.jks -validity 300 -storepass your-
keystore-password
-keypass your-key-passphrase -dname "CN=your-cn-name"
-alias alias-of-key-pair -storetype pkcs12 -keyalg RSA
```

2. 인증서 서명 요청(CSR)을 생성합니다.

```
keytool -keystore kafka.server.keystore.jks -certreq -file server-cert-sign-
request-rsa -alias on-premise-rsa -storepass your-key-store-password
-keypass your-key-password
```

3. 클러스터 트러스트 스토어의 CA를 사용하여 CSR에 서명합니다. CA가 없는 경우 자체 프라이빗 CA를 생성할 수 있습니다.

```
openssl req -new -x509 -keyout ca-key -out ca-cert -days validate-days
```

4. `ca-cert`를 서버 트러스트 스토어 및 키 스토어로 가져옵니다. 신뢰 스토어가 없는 경우에는 다음 명령을 사용하여 트러스트 스토어를 생성하고 `ca-cert` 를 해당 스토어로 가져옵니다.

```
keytool -keystore kafka.server.truststore.jks -alias CARoot -import -file ca-cert
keytool -keystore kafka.server.keystore.jks -alias CARoot -import -file ca-cert
```

5. 인증서에 서명합니다.

```
openssl x509 -req -CA ca-cert -CAkey ca-key -in server-cert-sign-request-rsa -out
 signed-server-certificate.pem
-days validate-days -CAcreateserial -passin pass:ca-password
```

6. 서명된 인증서를 키 스토어로 가져옵니다.

```
keytool -keystore kafka.server.keystore.jks -import -file signed-certificate.pem -
alias on-premise-rsa -storepass your-keystore-password
-keypass your-key-password
```

7. 다음 명령을 사용하여 kafka.server.keystore.jks에서 keystore.p12으로 on-premise-rsa 키를 가져옵니다.

```
keytool -importkeystore \
-srckeystore kafka.server.keystore.jks \
-destkeystore keystore.p12 \
-deststoretype PKCS12 \
-srcalias on-premise-rsa \
-deststorepass your-truststore-password \
-destkeypass your-key-password
```

8. 다음 명령을 사용하여 keystore.p12를 .pem 형식으로 내보냅니다.

```
Openssl pkcs12 -in keystore.p12 -out encrypted-private-server-key.pem –nocerts
```

9. encrypted-private-server-key.pem, signed-certificate.pem 및 ca-cert를 DMS 인증서 관리자에 업로드합니다.
10. 반환된 ARN을 사용하여 엔드포인트를 생성합니다.

```
aws dms create-endpoint --endpoint-identifier $endpoint-identifier --engine-name
 kafka --endpoint-type target --kafka-settings
'{"Broker": "b-0.kafka260.aaaaa1.a99.kafka.us-east-1.amazonaws.com:9092",
 "SecurityProtocol":"ssl-authentication",
"SslClientCertificateArn": "your-client-cert-arn","SslClientKeyArn": "your-client-
key-arn","SslClientKeyPassword":"your-client-key-password",
"SslCaCertificateArn": "your-ca-certificate-arn"}'
```

```
aws dms test-connection -replication-instance-arn=$rep_inst_arn –endpoint-arn=
$kafka_tar_arn_msk
```

SASL-SSL 인증을 사용하여 Amazon MSK에 연결

SASL(Simple Authentication and Security Layer) 메서드는 사용자 이름과 암호를 사용하여 클라이언트 ID를 검증하고 서버와 클라이언트 간에 암호화된 연결을 만듭니다.

SASL을 사용하려면 Amazon MSK 클러스터를 설정할 때 먼저 안전한 사용자 이름과 암호를 생성해야 합니다. Amazon MSK 클러스터의 보안 사용자 이름과 암호를 설정하는 방법에 관한 설명은 Amazon Managed Streaming for Apache Kafka 개발자 안내서의 Amazon MSK 클러스터용 SASL/SCRAM 인증 설정을 참조하세요.

그런 다음, Kafka 대상 엔드포인트를 생성할 때 `sasl-ssl` 옵션을 사용하여 보안 프로토콜 엔드포인트 설정(`SecurityProtocol`)을 설정합니다. `SaslUsername` 및 `SaslPassword` 옵션도 설정합니다. Amazon MSK 클러스터를 처음 설정할 때 다음 JSON 예제에서 보는 것처럼 이 옵션 설정이 보안 사용자 이름 및 암호와 일치하는지 확인하십시오.

```
"KafkaSettings": {
    "SecurityProtocol": "sasl-ssl",
    "SaslUsername":"Amazon MSK cluster secure user name",
    "SaslPassword":"Amazon MSK cluster secure password"
}
```

Note

- 현재는 공용 CA 지원 AWS DMS SASL-SSL만 지원합니다. DMS는 프라이빗 CA가 지원하는 자체 관리형 Kafka와 함께 사용할 수 있는 SASL-SSL을 지원하지 않습니다.
- SASL-SSL 인증의 경우 기본적으로 SCRAM-SHA-512 메커니즘을 AWS DMS 지원합니다. AWS DMS 버전 3.5.0 이상에서는 일반 메커니즘도 지원합니다. Plain 메커니즘을 지원하려면 `KafkaSettings` API 데이터 형식의 `SaslMechanism` 파라미터를 `PLAIN`으로 설정하십시오.

Apache Kafka 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기

Kafka 같은 데이터 스트리밍 대상에 CDC 업데이트를 작성하는 경우 업데이트를 통한 변경 전에 소스 데이터베이스 행의 원래 값을 볼 수 있습니다. 이를 가능하게 하기 위해 소스 데이터베이스 엔진에서 제공한 데이터를 기반으로 업데이트 이벤트의 이전 이미지를 AWS DMS 채웁니다.

소스 데이터베이스 엔진에 따라 이전 이미지에 대해 서로 다른 양의 정보를 제공합니다.

- Oracle은 열이 변경되는 경우에만 열에 대한 업데이트를 제공합니다.
- PostgreSQL은 기본 키의 일부인 열에 대한 데이터(변경 여부에 관계 없음)만 제공합니다. 논리적 복제를 사용 중이고 원본 테이블에 대해 REPLICA IDENTITY FULL이 설정된 경우 WAL에 기록된 행의 전체 이전 정보 및 이후 정보를 얻을 수 있으며 여기에서 확인할 수 있습니다.
- MySQL은 일반적으로 모든 열에 대한 데이터(변경 여부에 관계 없음)를 제공합니다.

이전 이미징을 활성화하여 소스 데이터베이스의 원래 값을 AWS DMS 출력에 추가하려면 `BeforeImageSettings` 태스크 설정 또는 `add-before-image-columns` 파라미터를 사용합니다. 이 파라미터는 열 변환 규칙을 적용합니다.

`BeforeImageSettings`는 다음과 같이 소스 데이터베이스 시스템에서 수집된 값을 사용하여 모든 업데이트 작업에 새 JSON 속성을 추가합니다.

```
"BeforeImageSettings": {
    "EnableBeforeImage": boolean,
    "FieldName": string,
    "ColumnFilter": pk-only (default) / non-lob / all (but only one)
}
```

> **Note**
>
> 전체 로드와 CDC 작업(기존 데이터 마이그레이션 및 지속적인 변경 사항 복제) 또는 CDC 전용 작업(데이터 변경 사항만 복제)에 `BeforeImageSettings`를 적용합니다. 전체 로드 전용 작업에는 `BeforeImageSettings`를 적용하지 마십시오.

`BeforeImageSettings` 옵션의 경우, 다음이 적용됩니다.

- 이전 이미징을 활성화하려면 EnableBeforeImage 옵션을 true로 설정합니다. 기본값은 false입니다.
- FieldName 옵션을 사용하여 새 JSON 속성에 이름을 지정합니다. EnableBeforeImage가 true인 경우 FieldName은 필수이며 비워 둘 수 없습니다.
- ColumnFilter 옵션은 이전 이미징을 사용하여 추가할 열을 지정합니다. 테이블 기본 키의 일부에 속하는 열만 추가하려면 기본값 pk-only를 사용하고, LOB 유형이 아닌 열만 추가하려면 non-lob를 사용하고, 이전 이미지 값이 있는 모든 열을 추가하려면 all을 사용합니다.

```
"BeforeImageSettings": {
    "EnableBeforeImage": true,
    "FieldName": "before-image",
    "ColumnFilter": "pk-only"
  }
```

이전 이미지 변환 규칙 사용

작업 설정 대신 열 변환 규칙을 적용하는 add-before-image-columns 파라미터를 사용할 수 있습니다. 이 파라미터를 사용하면 Kafka 같은 데이터 스트리밍 대상에서 CDC 중에 이전 이미징을 활성화할 수 있습니다.

변환 규칙에 add-before-image-columns를 사용하면 이전 이미지 결과를 보다 세밀하게 제어할 수 있습니다. 변환 규칙을 통해 객체 로케이터를 사용하여 규칙에 대해 선택한 테이블을 제어할 수 있습니다. 또한 변환 규칙을 함께 연결하여 테이블마다 서로 다른 규칙을 적용할 수 있습니다. 그런 다음, 다른 규칙을 사용하여 생성된 열을 조작할 수 있습니다.

> **Note**
>
> 동일한 작업 내에서 BeforeImageSettings 작업 설정과 함께 add-before-image-columns 파라미터를 사용해서는 안 됩니다. 단일 작업에는 이 파라미터 또는 설정 중 하나만 사용하고 둘 다 사용하지는 마십시오.

해당 열에 대해 add-before-image-columns 파라미터가 있는 transformation 규칙 유형이 before-image-def 단원을 제공해야 합니다. 다음은 그 한 예입니다.

```
  {
    "rule-type": "transformation",
```

```
      …
      "rule-target": "column",
      "rule-action": "add-before-image-columns",
      "before-image-def":{
        "column-filter": one-of  (pk-only / non-lob / all),
        "column-prefix": string,
        "column-suffix": string,
      }
    }
```

column-prefix의 값은 열 이름 앞에 추가되며, column-prefix의 기본값은 BI_입니다. column-suffix의 값은 열 이름 뒤에 추가되며, 기본값은 비어 있습니다. column-prefix와 column-suffix를 모두 빈 문자열로 설정하지 마십시오.

column-filter에 대해 하나의 값을 선택합니다. 테이블 기본 키의 일부인 열만 추가하려면 pk-only를 선택하고, LOB 유형이 아닌 열만 추가하려면 non-lob를 선택하고, 이전 이미지 값이 있는 모든 열을 추가하려면 all을 선택합니다.

이전 이미지 변환 규칙의 예

다음 예의 변환 규칙은 대상에서 BI_emp_no라는 새 열을 추가합니다. UPDATE employees SET emp_no = 3 WHERE emp_no = 1; 같은 문은 BI_emp_no 필드를 1로 채웁니다. Amazon S3 대상에 CDC 업데이트를 작성할 때 BI_emp_no 열을 통해 업데이트된 원래 행을 알 수 있습니다.

```
{
  "rules": [
    {
      "rule-type": "selection",
      "rule-id": "1",
      "rule-name": "1",
      "object-locator": {
        "schema-name": "%",
        "table-name": "%"
      },
      "rule-action": "include"
    },
    {
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-target": "column",
      "object-locator": {
```

```
        "schema-name": "%",
        "table-name": "employees"
      },
      "rule-action": "add-before-image-columns",
      "before-image-def": {
        "column-prefix": "BI_",
        "column-suffix": "",
        "column-filter": "pk-only"
      }
    }
  ]
}
```

`add-before-image-columns` 규칙 작업 사용에 관한 자세한 내용은 변환 규칙 및 작업 단원을 참조하십시오.

Apache Kafka를 대상으로 사용할 때의 제한 사항 AWS Database Migration Service

Apache Kafka를 대상으로 사용할 때 다음 제한 사항이 적용됩니다.

- AWS DMS Kafka 대상 엔드포인트는 Apache Kafka용 아마존 매니지드 스트리밍 (Amazon MSK) 에 대한 IAM 액세스 제어를 지원하지 않습니다.
- 전체 LOB 모드는 지원되지 않습니다.
- 새 주제를 자동으로 생성할 수 있는 속성을 포함하는 클러스터의 Kafka 구성 파일을 지정하십시오. AWS DMS `auto.create.topics.enable = true` 설정을 포함합니다. Amazon MSK 설정을 사용하는 경우 Kafka 클러스터를 만들 때 기본 구성을 지정한 다음 `auto.create.topics.enable` 설정을 `true`로 변경할 수 있습니다. 기본 구성 설정에 관한 자세한 내용은 Amazon Managed Streaming for Apache Kafka 개발자 안내서의 기본 Amazon MSK 구성을 참조하세요. Amazon MSK를 사용하여 만든 기존 Kafka 클러스터를 수정해야 하는 경우 다음 예와 같이 AWS CLI 명령을 `aws kafka create-configuration` 실행하여 Kafka 구성을 업데이트하십시오.

```
14:38:41 $ aws kafka create-configuration --name "kafka-configuration" --kafka-
versions "2.2.1" --server-properties file://~/kafka_configuration
{
    "LatestRevision": {
        "Revision": 1,
        "CreationTime": "2019-09-06T14:39:37.708Z"
    },
    "CreationTime": "2019-09-06T14:39:37.708Z",
```

```
    "Name": "kafka-configuration",
    "Arn": "arn:aws:kafka:us-east-1:111122223333:configuration/kafka-
configuration/7e008070-6a08-445f-9fe5-36ccf630ecfd-3"
}
```

여기서, //~/kafka_configuration은 필요한 속성 설정으로 만든 구성 파일입니다.

Amazon EC2에 설치된 자체 Kafka 인스턴스를 사용하는 경우, 인스턴스와 함께 auto.create.topics.enable = true 제공된 옵션을 사용하여 새 주제를 자동으로 생성할 AWS DMS 수 있는 설정으로 Kafka 클러스터 구성을 수정하십시오.

- AWS DMS 트랜잭션과 상관없이 각 업데이트를 소스 데이터베이스의 단일 레코드에 지정된 Kafka 주제에 하나의 데이터 레코드 (메시지) 로 게시합니다.
- AWS DMS 다음과 같은 두 가지 형식의 파티션 키를 지원합니다.
  - SchemaName.TableName: 스키마와 테이블 이름의 조합입니다.
  - ${AttributeName}: JSON 내 한 필드의 값 또는 소스 데이터베이스 내 테이블의 기본 키입니다.
- BatchApply는 Kafka 엔드포인트에는 지원되지 않습니다. Kafka 대상에 대해 Batch Apply(예: BatchApplyEnabled 대상 메타데이터 작업 설정)를 사용하면 데이터 손실이 발생할 수 있습니다.
- AWS DMS 16자리가 넘는 BigInt 데이터 유형 값의 마이그레이션을 지원하지 않습니다. 이 제한을 해결하기 위해 다음 변환 규칙을 사용하여 BigInt 열을 문자열로 변환할 수 있습니다. 변환 규칙에 대한 자세한 내용은 변환 규칙 및 작업 섹션을 참조하세요.

```
{
    "rule-type": "transformation",
    "rule-id": "id",
    "rule-name": "name",
    "rule-target": "column",
    "object-locator": {
        "schema-name": "valid object-mapping rule action",
        "table-name": "",
        "column-name": ""
    },
    "rule-action": "change-data-type",
    "data-type": {
        "type": "string",
        "length": 20
    }
}
```

객체 매핑을 사용하여 데이터를 Kafka 주제로 마이그레이션

AWS DMS 테이블 매핑 규칙을 사용하여 원본의 데이터를 대상 Kafka 주제에 매핑합니다. 데이터를 대상 주제에 매핑하려면 객체 매핑이라는 테이블 매핑 규칙 유형을 사용합니다. 객체 매핑을 사용하여 원본의 데이터 레코드를 Kafka 주제에 게시된 데이터 레코드로 매핑하는 방법을 지정합니다.

Kafka 주제에는 파티션 키 외에 별다른 사전 설정 구조가 없습니다.

> **Note**
>
> 객체 매핑을 사용할 필요는 없습니다. 일반 테이블 매핑을 다양한 변환에 사용할 수 있습니다. 하지만 파티션 키 유형은 다음과 같은 기본 동작을 따릅니다.
>
> - 기본 키는 전체 로드의 파티션 키로 사용됩니다.
> - paralle-apply 작업 설정을 사용하지 않는 경우, CDC의 파티션 키로 `schema.table`이 사용됩니다.
> - paralle-apply 작업 설정을 사용하지 않는 경우, CDC의 파티션 키로 기본 키가 사용됩니다.

object-mapping 규칙을 만들려면 `rule-type`을 `object-mapping`으로 지정합니다. 이 규칙은 사용할 객체 매핑의 유형을 지정합니다.

이 규칙의 구조는 다음과 같습니다.

```
{
    "rules": [
        {
            "rule-type": "object-mapping",
            "rule-id": "id",
            "rule-name": "name",
            "rule-action": "valid object-mapping rule action",
            "object-locator": {
                "schema-name": "case-sensitive schema name",
                "table-name": ""
            }
        }
    ]
}
```

AWS DMS 현재 매개 변수에 대해 `map-record-to-record` 및 `map-record-to-document` 만 유효한 값으로 지원합니다. `rule-action` 이러한 설정은 `exclude-columns` 속성 목록의 일부로 제외

되지 않은 값에 영향을 줍니다. `map-record-to-record`및 `map-record-to-document` 값은 이러한 레코드를 기본적으로 AWS DMS 처리하는 방법을 지정합니다. 이러한 값은 어떤 식으로든 속성 매핑에 영향을 미치지 않습니다.

관계형 데이터베이스에서 Kafka 주제로 마이그레이션할 때 `map-record-to-record`를 사용합니다. 이러한 규칙 유형은 Kafka 주제의 파티션 키로 관계형 데이터베이스의 `taskResourceId.schemaName.tableName` 값을 사용하며 소스 데이터베이스의 각 열마다 속성을 하나씩 생성합니다.

`map-record-to-record`을 사용하는 경우 다음 사항에 유의하세요.

- 이 설정은 `exclude-columns` 목록에서 제외된 열에만 영향을 줍니다.
- 이러한 모든 열에 대해 대상 주제에 해당 속성을 AWS DMS 만듭니다.
- AWS DMS 소스 열이 속성 매핑에 사용되는지 여부에 관계없이 해당하는 이 속성을 생성합니다.

`map-record-to-record`를 이해하는 한 가지 방법은 작업 중일 때 관찰하는 것입니다. 이 예에서는 다음 구조와 데이터를 사용하여 관계형 데이터베이스 테이블 행에서 시작한다고 가정합니다.

FirstName	LastName	StoreId	HomeAddress	HomePhone	WorkAddress	WorkPhone	DateofBirth
Randy	Marsh	5	221B Baker Street	1234567890	31 Spooner Street, Quahog	9876543210	1988/02/29

이 정보를 `Test`라는 스키마에서 Kafka 주제로 마이그레이션하려면 데이터를 대상 주제로 매핑하는 규칙을 생성합니다. 다음 규칙은 그 매핑 과정을 보여 줍니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "rule-action": "include",
            "object-locator": {
```

```
                "schema-name": "Test",
                "table-name": "%"
            }
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "2",
            "rule-name": "DefaultMapToKafka",
            "rule-action": "map-record-to-record",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "Customers"
            }
        }
    ]
}
```

다음은 지정된 Kafka 주제와 파티션 키(이 경우 `taskResourceId.schemaName.tableName`)로 Kafka 대상 주제의 샘플 데이터를 사용하여 결과 레코드 형식을 보여줍니다.

```
  {
     "FirstName": "Randy",
     "LastName": "Marsh",
     "StoreId":  "5",
     "HomeAddress": "221B Baker Street",
     "HomePhone": "1234567890",
     "WorkAddress": "31 Spooner Street, Quahog",
     "WorkPhone": "9876543210",
     "DateOfBirth": "02/29/1988"
  }
```

주제

- 속성 매핑으로 날짜 재구성
- 객체 매핑을 사용한 멀티주제 복제
- Apache Kafka 메시지 형식

속성 매핑으로 날짜 재구성

속성 맵을 사용하여 데이터를 Kafka 주제로 마이그레이션하는 동안 데이터를 재구성할 수 있습니다. 예를 들어 소스의 필드 몇 개를 대상의 단일 필드로 묶어야 하는 경우도 있을 것입니다. 다음의 속성 맵은 날짜를 재구성하는 방법을 보여줍니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "rule-action": "include",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            }
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "2",
            "rule-name": "TransformToKafka",
            "rule-action": "map-record-to-record",
            "target-table-name": "CustomerData",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "Customers"
            },
            "mapping-parameters": {
                "partition-key-type": "attribute-name",
                "partition-key-name": "CustomerName",
                "exclude-columns": [
                    "firstname",
                    "lastname",
                    "homeaddress",
                    "homephone",
                    "workaddress",
                    "workphone"
                ],
                "attribute-mappings": [
                    {
                        "target-attribute-name": "CustomerName",
                        "attribute-type": "scalar",
```

```
                    "attribute-sub-type": "string",
                    "value": "${lastname}, ${firstname}"
                },
                {
                    "target-attribute-name": "ContactDetails",
                    "attribute-type": "document",
                    "attribute-sub-type": "json",
                    "value": {
                        "Home": {
                            "Address": "${homeaddress}",
                            "Phone": "${homephone}"
                        },
                        "Work": {
                            "Address": "${workaddress}",
                            "Phone": "${workphone}"
                        }
                    }
                }
            ]
        }
    ]
}
```

partition-key에 상수 값을 설정하려면 partition-key 값을 지정합니다. 예를 들어 이 방법을 통해 모든 데이터를 단일 파티션에 저장할 수 있습니다. 다음 매핑은 이러한 접근 방식을 보여줍니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "1",
            "rule-name": "TransformToKafka",
```

```
            "rule-action": "map-record-to-document",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "Customer"
            },
            "mapping-parameters": {
                "partition-key": {
                    "value": "ConstantPartitionKey"
                },
                "exclude-columns": [
                    "FirstName",
                    "LastName",
                    "HomeAddress",
                    "HomePhone",
                    "WorkAddress",
                    "WorkPhone"
                ],
                "attribute-mappings": [
                    {
                        "attribute-name": "CustomerName",
                        "value": "${FirstName},${LastName}"
                    },
                    {
                        "attribute-name": "ContactDetails",
                        "value": {
                            "Home": {
                                "Address": "${HomeAddress}",
                                "Phone": "${HomePhone}"
                            },
                            "Work": {
                                "Address": "${WorkAddress}",
                                "Phone": "${WorkPhone}"
                            }
                        }
                    },
                    {
                        "attribute-name": "DateOfBirth",
                        "value": "${DateOfBirth}"
                    }
                ]
            }
        }
    ]
```

```
}
```

> **Note**
>
> 특정 테이블을 위한 제어 레코드에 대한 partition-key 값은 TaskId.SchemaName.TableName입니다. 특정 작업을 위한 제어 레코드에 대한 partition-key 값은 해당 레코드의 TaskId입니다. partition-key 값을 객체 매핑에 지정해도 제어 레코드에 대한 partition-key에는 영향이 없습니다.

객체 매핑을 사용한 멀티주제 복제

기본적으로 AWS DMS 작업은 모든 소스 데이터를 다음 Kafka 주제 중 하나로 마이그레이션합니다.

- AWS DMS 대상 엔드포인트의 Topic 필드에 지정된 대로
- 대상 엔드포인트의 Topic 필드가 채워지지 않고 Kafka auto.create.topics.enable 설정이 true로 설정된 경우에는 kafka-default-topic에서 지정한 대로 마이그레이션합니다.

AWS DMS 엔진 버전 3.4.6 이상에서는 kafka-target-topic 속성을 사용하여 마이그레이션된 각 소스 테이블을 별도의 주제에 매핑할 수 있습니다. 예를 들어, 다음 객체 매핑 규칙은 원본 테이블 Customer과 Address를 Kafka 주제 customer_topic 및 address_topic에 각각 마이그레이션합니다. 동시에 Test 스키마의 테이블을 포함하여 다른 모든 원본 테이블을 대상 Bills 엔드포인트에 지정된 주제로 AWS DMS 마이그레이션합니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "rule-action": "include",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            }
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "2",
            "rule-name": "MapToKafka1",
```

```
            "rule-action": "map-record-to-record",
            "kafka-target-topic": "customer_topic",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "Customer"
            },
            "partition-key": {"value": "ConstantPartitionKey" }
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "3",
            "rule-name": "MapToKafka2",
            "rule-action": "map-record-to-record",
            "kafka-target-topic": "address_topic",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "Address"
            },
            "partition-key": {"value": "HomeAddress" }
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "4",
            "rule-name": "DefaultMapToKafka",
            "rule-action": "map-record-to-record",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "Bills"
            }
        }
    ]
}
```

Kafka 멀티주제 복제를 사용하면 단일 복제 작업을 사용하여 원본 테이블을 그룹화하여 별도의 Kafka 주제로 마이그레이션할 수 있습니다.

Apache Kafka 메시지 형식

JSON 출력은 단지 키-값 페어의 목록입니다.

RecordType

레코드 유형은 데이터나 제어일 수 있습니다. 데이터 레코드는 소스 내의 실제 행을 나타냅니다. 제어 레코드는 스트림 내의 중요 이벤트를 나타냅니다(예: 작업의 재시작).

Operation

데이터 레코드의 경우, 작업은 `load`, `insert`, `update` 또는 `delete`일 수 있습니다.

데이터 레코드의 경우, 작업은 `create-table`, `rename-table`, `drop-table`, `change-columns`, `add-column`, `drop-column`, `rename-column` 또는 `column-type-change`일 수 있습니다.

SchemaName

레코드에 대한 원본 스키마입니다. 제어 레코드의 경우, 이 필드는 비워둘 수 있습니다.

TableName

레코드에 대한 소스 테이블입니다. 제어 레코드의 경우, 이 필드는 비워둘 수 있습니다.

Timestamp

JSON 메시지가 구성된 경우의 타임스탬프입니다. 이 필드는 ISO 8601 형식으로 구성되었습니다.

다음 JSON 메시지 예제는 모든 추가 메타데이터가 포함된 데이터 유형 메시지를 보여줍니다.

```
{
   "data":{
      "id":100000161,
      "fname":"val61s",
      "lname":"val61s",
      "REGION":"val61s"
   },
   "metadata":{
      "timestamp":"2019-10-31T22:53:59.721201Z",
      "record-type":"data",
      "operation":"insert",
      "partition-key-type":"primary-key",
      "partition-key-value":"sbtest.sbtest_x.100000161",
      "schema-name":"sbtest",
      "table-name":"sbtest_x",
      "transaction-id":9324410911751,
      "transaction-record-id":1,
      "prev-transaction-id":9324410910341,
      "prev-transaction-record-id":10,
      "commit-timestamp":"2019-10-31T22:53:55.000000Z",
      "stream-position":"mysql-bin-
changelog.002171:36912271:0:36912333:9324410911751:mysql-bin-changelog.002171:36912209"
```

```
    }
}
```

다음 JSON 메시지 예제는 컨트롤 유형 메시지를 보여줍니다.

```
{
   "control":{
      "table-def":{
         "columns":{
            "id":{
               "type":"WSTRING",
               "length":512,
               "nullable":false
            },
            "fname":{
               "type":"WSTRING",
               "length":255,
               "nullable":true
            },
            "lname":{
               "type":"WSTRING",
               "length":255,
               "nullable":true
            },
            "REGION":{
               "type":"WSTRING",
               "length":1000,
               "nullable":true
            }
         },
         "primary-key":[
            "id"
         ],
         "collation-name":"latin1_swedish_ci"
      }
   },
   "metadata":{
      "timestamp":"2019-11-21T19:14:22.223792Z",
      "record-type":"control",
      "operation":"create-table",
      "partition-key-type":"task-id",
      "schema-name":"sbtest",
```

```
      "table-name":"sbtest_t1"
   }
}
```

Amazon OpenSearch Service 클러스터를 AWS Database Migration Service의 대상으로 사용

Amazon OpenSearch Service(OpenSearch Service)에 데이터를 마이그레이션하는 데 AWS DMS를 사용할 수 있습니다. OpenSearch Service는 OpenSearch Service 클러스터를 손쉽게 배포, 운영 및 확장할 수 있도록 해주는 관리형 서비스입니다.

OpenSearch Service에서는 인덱스와 문서로 작업할 수 있습니다. 인덱스는 문서 컬렉션이며 문서는 스칼라 값, 배열 및 기타 객체를 포함하는 JSON 객체입니다. OpenSearch은 인덱스의 데이터를 쿼리하고 해당 문서를 검색할 수 있도록 JSON 기반 쿼리 언어를 제공합니다.

AWS DMS가 OpenSearch Service의 대상 엔드포인트에 대한 인덱스를 생성할 때는 소스 엔드포인트의 각 테이블에 대해 1개의 인덱스를 생성합니다. OpenSearch Service 인덱스를 생성하기 위한 비용은 다음과 같은 다양한 요인에 따라 달라집니다. 그러한 요인으로는 생성되는 인덱스 수, 이러한 인덱스 내 데이터의 총량, OpenSearch이 문서마다 저장하는 소량의 메타데이터 등이 있습니다.

마이그레이션 범위에 적합한 컴퓨팅 및 스토리지 리소스를 갖춘 OpenSearch Service 클러스터를 구성합니다. 사용할 복제 작업에 따라 다음 요소를 고려하는 것이 좋습니다.

- 전체 데이터 로드의 경우, 마이그레이션할 데이터의 총량과 전송 속도를 고려합니다.
- 지속적인 변경 사항을 복제하는 경우, 업데이트 빈도와 종단 간 지연 시간 요건을 고려합니다.

또한 문서 개수에 주의하며 OpenSearch 클러스터의 인덱스 설정을 구성합니다.

멀티스레드 전체 로드 작업 설정

전송 속도를 높이기 위해 AWS DMS는 OpenSearch Service 대상 클러스터에 대한 멀티스레드 전체 로드를 지원합니다. AWS DMS는 다음을 포함하는 작업 설정으로 이 멀티스레딩을 지원합니다.

- `MaxFullLoadSubTasks` – 병렬로 로드할 최대 소스 테이블 수를 표시하려면 이 옵션을 사용합니다. DMS는 전용 하위 작업을 사용하여 각 테이블을 해당 OpenSearch Service 대상 인덱스에 로드합니다. 기본값은 8이며, 최대값은 49입니다.
- `ParallelLoadThreads` – AWS DMS에서 각 테이블을 OpenSearch Service 대상 인덱스에 로드하는 데 사용하는 스레드 수를 지정하려면 이 옵션을 사용합니다. OpenSearch Service 대상의 최대값은 32입니다. 이 최대 한도를 증가시키도록 요청할 수 있습니다.

> **Note**
>
> `ParallelLoadThreads`를 기본값(0)에서 변경하지 않으면 AWS DMS는 한 번에 하나의 레코드를 전송합니다. 이 접근 방식은 OpenSearch Service 클러스터에서 고유의 로드를 생성합니다. 이 옵션을 1 이상으로 설정해야 합니다.

- `ParallelLoadBufferSize` – 병렬 로드 스레드에서 데이터를 OpenSearch Service 대상에 로드하기 위해 사용하는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 이 설정은 `ParallelLoadThreads`와 함께 사용하십시오. `ParallelLoadBufferSize`는 둘 이상의 스레드가 있는 경우에만 유효합니다.

DMS가 멀티스레드를 사용하여 OpenSearch Service 클러스터를 로드하는 방법에 관한 자세한 내용은 AWS 블로그 게시물 AWS Database Migration Service 마이그레이션을 위한 Amazon OpenSearch Service 확장을 참조하십시오.

멀티스레드 CDC 로드 작업 설정

작업 설정을 사용하여 `PutRecords` API 직접 호출의 동작을 수정하여 OpenSearch Service 대상 클러스터의 변경 데이터 캡처(CDC) 성능을 개선할 수 있습니다. 이렇게 하려면 `ParallelApply*` 작업 설정을 사용하여 동시 스레드 수, 스레드당 대기열 및 버퍼에 저장할 레코드 수를 지정해야 합니다. 예를 들어 CDC 로드를 수행하고 32개의 스레드를 병렬로 적용한다고 가정하겠습니다. 또한 버퍼당 50개 레코드가 저장된 스레드당 64개 대기열에 액세스하려고 합니다.

> **Note**
>
> CDC 중에 Amazon OpenSearch Service 대상 엔드포인트에서 `ParallelApply*` 작업 설정의 사용에 대한 지원은 AWS DMS 버전 3.4.0 이상에서 제공됩니다.

CDC 성능을 승격하기 위해 AWS DMS에서는 다음 작업 설정을 지원합니다.

- `ParallelApplyThreads` – 데이터 레코드를 OpenSearch Service 대상 엔드포인트로 푸시하기 위해 CDC 로드 중에 AWS DMS가 사용하는 동시 스레드 수를 지정합니다. 기본값은 0이고 최대값은 32입니다.
- `ParallelApplyBufferSize` – CDC 로드 중에 OpenSearch Service 대상 엔드포인트로 푸시할 동시 스레드에 대한 각 버퍼 대기열에 저장할 최대 레코드 수를 지정합니다. 기본값은 100이고 최대

값은 1,000입니다. `ParallelApplyThreads`가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.

- `ParallelApplyQueuesPerThread` – 각 스레드가 대기열에서 데이터 레코드를 가져오고 CDC 중에 OpenSearch Service 엔드포인트에 대한 배치 로드를 생성하기 위한 대기열 수를 지정합니다.

`ParallelApply*` 작업 설정을 사용할 때 `partition-key-type` 기본값은 테이블의 `schema-name.table-name`가 아니라 `primary-key`입니다.

관계형 데이터베이스 테이블에서 OpenSearch Service 인덱스로 마이그레이션

AWS DMS에서는 OpenSearch Service의 스칼라 데이터 형식으로 데이터 마이그레이션을 지원합니다. Oracle 또는 MySQL 같은 관계형 데이터베이스에서 OpenSearch Service로 마이그레이션할 때 필요하다면 이 데이터를 저장하는 방법을 다시 구성할 수 있습니다.

AWS DMS는 다음 OpenSearch Service 스칼라 데이터 형식을 지원합니다.

- 부울
- 날짜
- Float
- 정수
- 문자열

AWS DMS는 날짜 형식의 데이터를 문자열 형식으로 변환합니다. 사용자 지정 매핑을 지정하여 이러한 날짜를 해석할 수 있습니다.

AWS DMS는 LOB 데이터 형식의 마이그레이션을 지원하지 않습니다.

Amazon OpenSearch Service 서비스를 AWS Database Migration Service의 대상으로 사용하기 위한 사전 요구 사항

OpenSearch Service 데이터베이스를 AWS DMS의 대상으로 사용하기에 앞서 AWS Identity and Access Management(IAM) 역할을 생성해야 합니다. 이 역할은 AWS DMS가 대상 엔드포인트에서 OpenSearch Service 인덱스에 액세스하도록 해야 합니다. 최소 액세스 권한 집합이 다음 IAM 정책에 나와 있습니다.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
                "Service": "dms.amazonaws.com"
        },
        "Action": "sts:AssumeRole"
        }
    ]
}
```

OpenSearch Service로의 마이그레이션에 사용하는 역할에는 다음 권한이 있어야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                      "es:ESHttpDelete",
                      "es:ESHttpGet",
                      "es:ESHttpHead",
                      "es:ESHttpPost",
                      "es:ESHttpPut"
                    ],
            "Resource": "arn:aws:es:region:account-id:domain/domain-name/*"
        }
    ]
}
```

앞의 예에서는 *region*을 AWS 리전 식별자로 교체하고, *account-id*를 AWS 계정 ID로 교체하고, *domain-name*을 Amazon OpenSearch Service 도메인 이름으로 교체합니다. `arn:aws:es:us-west-2:123456789012:domain/my-es-domain`를 예로 들 수 있음

OpenSearch Service를 AWS DMS의 대상으로 사용할 경우 엔드포인트 설정

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 OpenSearch Service 대상 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI의 `create-endpoint` 명

령을 `--elasticsearch-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

OpenSearch Service를 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

속성 이름	유효값	기본값과 설명
`FullLoadErrorPercentage`	0보다 크고 100 이하인 양의 정수입니다.	10 - 전체 로드 작업의 경우 이 속성이 작업 실패 전에 허용되는 오류의 임계값을 결정합니다. 예를 들어 소스 엔드포인트에 1,500개 행이 있으며 이 파라미터가 10으로 설정되어 있다고 가정합니다. 이 경우, AWS DMS가 대상 엔드포인트에 작성할 때 150개(행 수의 10%)를 초과하는 오류를 포착하면 작업이 실패합니다.
`ErrorRetryDuration`	0보다 큰 양의 정수입니다.	300 - 대상 엔드포인트에 오류가 발생하면 AWS DMS가 이를 몇 초 동안 재시도합니다. 그렇지 않으면 작업이 실패한 것입니다.

Amazon OpenSearch Service를 AWS Database Migration Service의 대상으로 사용할 때의 제한 사항

Amazon OpenSearch Service를 대상으로 사용할 때에는 다음 제한 사항이 적용됩니다.

- OpenSearch Service는 동적 매핑(자동 추정)을 사용하여 마이그레이션된 데이터에 사용할 데이터 형식을 결정합니다.
- OpenSearch Service는 각 문서를 고유한 ID로 저장합니다. 다음은 ID의 예입니다.

```
"_id": "D359F8B537F1888BC71FE20B3D79EAE6674BE7ACA9B645B0279C7015F6FF19FD"
```

  각 문서 ID의 길이는 스토리지 요구 사항에 부합하도록 64바이트입니다. 예를 들어, AWS DMS 소스로부터 10만 개 행을 마이그레이션한다면 그에 따른 OpenSearch Service 인덱스에는 640만 바이트의 스토리지가 추가로 필요하게 됩니다.

- OpenSearch Service에서는 기본 키 속성 업데이트를 허용하지 않습니다. 대상에 원치 않는 데이터가 발생할 수 있으므로 변경 데이터 캡처(CDC)와 함께 지속적인 복제 사용 시 이 제한이 중요합니다. CDC 모드에서 기본 키는 32바이트 길이의 SHA256 값으로 매핑됩니다. 이는 사람이 읽을 수 있는 64바이트 문자열로 변환되며 OpenSearch Service 문서 ID로 사용됩니다.

- AWS DMS가 마이그레이션할 수 없는 항목을 인식하면 Amazon CloudWatch Logs에 오류 메시지를 작성합니다. 이러한 동작은 오류 메시지를 예외 테이블에 작성하는 다른 AWS DMS 대상 엔드포인트의 경우와는 다릅니다.
- AWS DMS는 마스터 사용자 및 암호를 사용하여 세분화된 액세스 제어가 활성화된 Amazon ES 클러스터에 대한 연결을 지원하지 않습니다.
- AWS DMS는 OpenSearch Service 서버리스는 지원하지 않습니다.
- OpenSearch Service는 기존 인덱스에 데이터를 쓰는 것을 지원하지 않습니다.

Amazon OpenSearch Service의 대상 데이터 유형

AWS DMS에서 다른 형식의 데이터베이스에서 가져온 데이터를 마이그레이션할 때, 원본 데이터베이스의 데이터 형식을 AWS DMS 데이터 형식이라고 하는 중간 데이터 형식에 매핑합니다. 그런 다음, 중간 데이터 형식을 대상 데이터 형식에 매핑합니다. 다음 테이블에는 각 AWS DMS 데이터 형식과 OpenSearch Service에서 매핑되는 데이터 형식이 나와 있습니다.

AWS DMS 데이터 형식	OpenSearch Service 데이터 형식
부울	부울
날짜	문자열
시간	날짜
타임스탬프	날짜
INT4	integer
Real4	float
UINT4	integer

AWS DMS 데이터 형식에 관한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

Amazon DocumentDB를 AWS Database Migration Service의 대상으로 사용

AWS DMS가 지원하는 Amazon DocumentDB(MongoDB와 호환)에 관한 자세한 내용은 대상: AWS DMS을 참조하십시오. AWS DMS를 사용하면 AWS DMS가 지원하는 소스 데이터 엔진에서 Amazon

DocumentDB(MongoDB 호환)로 데이터를 마이그레이션할 수 있습니다. 소스 엔진은 Amazon RDS, Aurora 또는 Amazon S3와 같은 AWS 관리형 서비스에서 사용할 수 있습니다. 또는 Amazon EC2 또는 온프레미스에서 실행되는 MongoDB와 같이 자체 관리형 데이터베이스를 기반으로 할 수 있습니다.

AWS DMS를 사용하여 Amazon DocumentDB 데이터베이스, 컬렉션 또는 문서에 소스 데이터를 복제할 수 있습니다.

> **Note**
>
> 소스 엔드포인트가 MongoDB 또는 Amazon DocumentDB인 경우 문서 모드에서 마이그레이션을 실행하십시오.

MongoDB는 이진 JSON(BSON) 형식으로 데이터를 저장합니다. AWS DMS는 Amazon DocumentDB에서 지원되는 BSON 데이터 형식을 모두 지원합니다. 이러한 데이터 형식의 목록은 Amazon DocumentDB 개발자 안내서의 지원되는 MongoDB API, 작업 및 데이터 형식을 참조하십시오.

소스 엔드포인트가 관계형 데이터베이스인 경우, AWS DMS는 다음과 같이 Amazon DocumentDB에 데이터베이스 객체를 매핑합니다.

- 관계형 데이터베이스 또는 데이터베이스 스키마는 Amazon DocumentDB 데이터베이스에 매핑됩니다.
- 관계형 데이터베이스 내 테이블은 Amazon DocumentDB의 컬렉션에 매핑됩니다.
- 관계형 테이블의 레코드는 Amazon DocumentDB의 문서에 매핑됩니다. 각 문서는 소스 레코드의 데이터로 구성됩니다.

소스 엔드포인트가 Amazon S3이면 그 결과로 만들어지는 Amazon DocumentDB 객체는 Amazon S3을 위한 AWS DMS 매핑 규칙에 해당합니다. 예를 들어 다음 URI를 고려해 보십시오.

```
s3://mybucket/hr/employee
```

이 경우, AWS DMS는 다음과 같이 Amazon DocumentDB의 객체를 `mybucket`로 매핑합니다.

- URI의 최상위 부분(`hr`)은 Amazon DocumentDB 데이터베이스로 매핑됩니다.
- 다음 URI 부분(`employee`)은 Amazon DocumentDB 컬렉션으로 매핑됩니다.
- `employee`의 각 객체는 Amazon DocumentDB의 문서에 매핑됩니다.

Amazon S3의 매핑 규칙에 관한 자세한 내용은 Amazon S3를 AWS DMS 소스로 사용 단원을 참조하십시오.

Amazon DocumentDB 엔드포인트 설정

AWS DMS 버전 3.5.0 및 이후 버전에서는 병렬 스레드 및 대량 작업에 대한 작업 설정을 조정하여 Amazon DocumentDB 엔드포인트의 변경 데이터 캡처(CDC) 성능을 개선할 수 있습니다. 이렇게 하려면 `ParallelApply*` 작업 설정을 사용하여 동시 스레드 수, 스레드당 대기열 및 버퍼에 저장할 레코드 수를 지정해야 합니다. 예를 들어 CDC 로드를 수행하고 128개의 스레드를 병렬로 적용한다고 가정하겠습니다. 또한 버퍼당 50개 레코드가 저장된 스레드당 64개 대기열에 액세스하려고 합니다.

CDC 성능을 승격하기 위해 AWS DMS에서는 다음 작업 설정을 지원합니다.

- `ParallelApplyThreads` – 데이터 레코드를 Amazon DocumentDB 대상 엔드포인트로 푸시하기 위해 CDC 로드 중에 AWS DMS가 사용하는 동시 스레드 수를 지정합니다. 기본값은 0이고 최대값은 32입니다.
- `ParallelApplyBufferSize` – CDC 로드 중에 Amazon DocumentDB 대상 엔드포인트로 푸시할 동시 스레드에 대한 각 버퍼 대기열에 저장할 최대 레코드 수를 지정합니다. 기본값은 100이고 최대값은 1,000입니다. `ParallelApplyThreads`가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.
- `ParallelApplyQueuesPerThread` – 각 스레드가 대기열에서 데이터 레코드를 가져오고 CDC 중에 Amazon DocumentDB 엔드포인트에 대한 배치 로드를 생성하기 위한 대기열 수를 지정합니다. 기본값은 1입니다. 최대 한도는 512입니다.

Amazon DocumentDB를 AWS DMS의 대상으로 사용하여 작업하는 방법에 관한 자세한 내용은 다음 단원을 참조하십시오.

주제

- 소스에서 Amazon DocumentDB 대상으로 데이터 매핑
- Amazon DocumentDB Elastic Clusters에 대상으로 연결
- Amazon DocumentDB를 대상으로 지속적 복제
- Amazon DocumentDB를 대상으로 사용할 때 적용되는 제한 사항
- Amazon DocumentDB를 대상으로 하는 엔드포인트 설정 사용
- Amazon DocumentDB의 대상 데이터 형식

> **Note**
>
> 마이그레이션 프로세스를 단계별로 살펴보려면 AWS Database Migration Service 단계별 마이그레이션 가이드의 MongoDB에서 Amazon DocumentDB로 마이그레이션을 참조하십시오.

소스에서 Amazon DocumentDB 대상으로 데이터 매핑

AWS DMS는 소스 엔드포인트에서 레코드를 읽고 판독한 데이터를 토대로 JSON 문서를 구성합니다. 각 JSON 문서에서 AWS DMS는 고유한 식별자 역할을 하는 `_id` 필드를 확인해야 합니다. 그런 다음, `_id` 필드를 기본 키로 사용하여 JSON 문서를 Amazon DocumentDB 컬렉션에 기록합니다.

단일 열인 소스 데이터

단일 열로 구성된 소스 데이터는 문자열 형식으로 간주됩니다. (소스 엔진에 따라 실제 데이터 형식은 VARCHAR, NVARCHAR, TEXT, LOB, CLOB 등이 될 수 있습니다.) AWS DMS는 데이터가 유효한 JSON 문서라고 가정하고 데이터를 Amazon DocumentDB에 있는 그대로 복제합니다.

그 결과로 생성된 JSON 문서에 `_id`라는 필드가 포함되어 있는 경우, 해당 필드는 Amazon DocumentDB에서 고유한 `_id`로 사용됩니다.

JSON에 `_id` 필드가 포함되어 있지 않으면 Amazon DocumentDB에서 `_id` 값이 자동으로 생성됩니다.

다중 열인 소스 데이터

소스 데이터가 여러 개의 열로 구성된 경우, AWS DMS는 이러한 모든 열에서 JSON 문서를 구성합니다. 해당 문서에 대한 `_id` 필드를 확인하기 위해 AWS DMS는 다음과 같이 진행됩니다.

- 열 중 하나의 이름이 `_id`이면 해당 열의 데이터가 대상 `_id`로 사용됩니다.
- `_id` 열은 없지만 소스 데이터가 기본 키나 고유한 인덱스를 가지고 있으면 AWS DMS는 `_id` 값으로 해당 키 또는 인덱스 값을 사용합니다. 또한 기본 키 또는 고유 인덱스에서 나온 데이터는 JSON 문서에 명시적 필드 형태로 나타납니다.
- `_id` 열과 기본 키 또는 고유 인덱스가 없으면 Amazon DocumentDB에서 `_id` 값이 자동으로 생성됩니다.

대상 엔드포인트에서 데이터 형식을 강제로 적용

AWS DMS는 Amazon DocumentDB 대상 엔드포인트에 기록할 때 데이터 구조를 수정할 수 있습니다. 소스 엔드포인트에서 열과 테이블의 이름을 변경하거나 작업 실행 시 적용되는 변환 규칙을 제공하여 이러한 변경을 요청할 수 있습니다.

중첩 JSON 문서(json_ prefix) 사용

데이터 형식을 강제로 적용하려면 직접, 또는 변환 기능을 사용해 소스 열 이름에 `json_`(즉, `json_`*`columnName`*) 접두사를 붙일 수 있습니다. 이 경우, 열은 문자열 필드가 아니라 대상 문서 내에 중첩 JSON 문서 형태로 생성됩니다.

예를 들어 MongoDB 소스 엔드포인트에서 다음 문서를 마이그레이션하고 싶다고 가정해 봅시다.

```
{
    "_id": "1",
    "FirstName": "John",
    "LastName": "Doe",
    "ContactDetails": "{"Home": {"Address": "Boston","Phone": "1111111"},"Work":
 { "Address": "Boston", "Phone": "2222222222"}}"
}
```

소스 데이터 형식을 강제로 적용하지 않으면 기본적으로 포함된 `ContactDetails` 문서가 문자열 형태로 마이그레이션됩니다.

```
{
    "_id": "1",
    "FirstName": "John",
    "LastName": "Doe",
    "ContactDetails": "{\"Home\": {\"Address\": \"Boston\",\"Phone\": \"1111111\"},
\"Work\": { \"Address\": \"Boston\", \"Phone\": \"2222222222\"}}"
}
```

하지만 JSON 객체에 `ContactDetails`를 강제 적용하기 위한 변환 규칙을 추가할 수 있습니다. 예를 들어 원래의 소스 열 이름이 `ContactDetails`라고 가정해 봅시다. 데이터 형식을 중첩된 JSON으로 강제 변환하려면 소스에 “*json_*” 접두사를 수동으로 추가하거나 변환 규칙을 통해 소스 엔드포인트의 열 이름을 “JSON_ContactDetails”로 변경해야 합니다. 예를 들어, 아래의 변환 규칙을 사용할 수 있습니다.

```
{
    "rules": [
    {
    "rule-type": "transformation",
    "rule-id": "1",
    "rule-name": "1",
    "rule-target": "column",
    "object-locator": {
    "schema-name": "%",
    "table-name": "%",
    "column-name": "ContactDetails"
     },
    "rule-action": "rename",
    "value": "json_ContactDetails",
    "old-value": null
    }
    ]
}
```

AWS DMS는 다음과 같이 ContactDetails 필드를 중첩된 JSON으로 복제합니다.

```
{
    "_id": "1",
    "FirstName": "John",
    "LastName": "Doe",
    "ContactDetails": {
        "Home": {
            "Address": "Boston",
            "Phone": "1111111111"
        },
        "Work": {
            "Address": "Boston",
            "Phone": "2222222222"
        }
    }
}
```

JSON 배열(array_ prefix) 사용

데이터 형식을 강제로 적용하려면 직접, 또는 변환 기능을 사용해 열 이름에 `array_`(즉, `array_`*`columnName`*) 접두사를 붙일 수 있습니다. 이 경우, AWS DMS는 해당 열을 JSON 어레이로 간주하고 대상 문서에서와 같이 이를 생성합니다.

MongoDB 소스 엔드포인트에서 다음 문서를 마이그레이션하고 싶다고 가정해 봅시다.

```
{
    "_id" : "1",
    "FirstName": "John",
    "LastName": "Doe",

    "ContactAddresses": ["Boston", "New York"],

    "ContactPhoneNumbers": ["1111111111", "2222222222"]
}
```

소스 데이터 형식을 강제로 적용하지 않으면 기본적으로 포함된 `ContactDetails` 문서가 문자열 형태로 마이그레이션됩니다.

```
{
    "_id": "1",
    "FirstName": "John",
    "LastName": "Doe",

    "ContactAddresses": "[\"Boston\", \"New York\"]",

    "ContactPhoneNumbers": "[\"1111111111\", \"2222222222\"]"

}
```

하지만 JSON 어레이에 `ContactAddress` 및 `ContactPhoneNumbers`를 강제 적용하기 위한 변환 규칙을 추가할 수 있습니다(아래 표 참조).

원래의 소스 열 이름	이름이 변경된 소스 열
`ContactAddress`	`array_ContactAddress`
`ContactPhoneNumbers`	`array_ContactPhoneNumbers`

AWS DMS는 다음과 같이 `ContactAddress` 및 `ContactPhoneNumbers`를 복제합니다.

```
{
    "_id": "1",
    "FirstName": "John",
```

```
    "LastName": "Doe",
    "ContactAddresses": [
        "Boston",
        "New York"
    ],
    "ContactPhoneNumbers": [
        "1111111111",
        "2222222222"
    ]
}
```

TLS를 사용하여 Amazon DocumentDB에 연결

기본적으로 새로 생성한 Amazon DocumentDB 클러스터는 전송 계층 보안(TLS)을 사용한 보안 연결만 허용합니다. TLS가 활성화되면 Amazon DocumentDB에 대한 모든 연결에 퍼블릭 키가 필요합니다.

AWS에서 호스팅된 Amazon S3 버킷에서 `rds-combined-ca-bundle.pem` 파일을 다운로드하여 Amazon DocumentDB의 퍼블릭 키를 검색할 수 있습니다. 이 파일을 다운로드하는 방법에 관한 자세한 내용은 Amazon DocumentDB 개발자 안내서의 TLS를 사용한 연결 암호화를 참조하십시오.

이 .pem 파일을 다운로드한 후 다음 설명에 따라 파일에 포함된 퍼블릭 키를 AWS DMS로 가져올 수 있습니다.

AWS Management Console

퍼블릭 키(.pem) 파일을 가져오려면

1. https://console.aws.amazon.com/dmsAWS DMS에서 콘솔을 엽니다.
2. 탐색 창에서 [Certificates]를 선택합니다.
3. 인증서 가져오기를 선택하고 다음과 같이 합니다.

   - 인증서 식별자에 해당 인증서의 고유 이름(예: `docdb-cert`)을 입력합니다.
   - 파일 가져오기에서 .pem 파일이 저장된 위치로 이동합니다.

   원하는 대로 설정이 되었으면 새 CA 인증서 추가를 선택합니다.

AWS CLI

다음 예와 같이 `aws dms import-certificate` 명령을 사용하십시오.

```
aws dms import-certificate \
    --certificate-identifier docdb-cert \
    --certificate-pem file://./rds-combined-ca-bundle.pem
```

AWS DMS 대상 엔드포인트를 생성할 때 인증서 식별자(예: `docdb-cert`)를 제공합니다. 또한 SSL 모드 파라미터를 `verify-full`로 설정합니다.

Amazon DocumentDB Elastic Clusters에 대상으로 연결

AWS DMS 버전 3.4.7 이상에서는 Amazon DocumentDB 대상 엔드포인트를 Elastic 클러스터로 생성할 수 있습니다. 대상 엔드포인트를 Elastic 클러스터로 생성하는 경우 기존 SSL 인증서는 작동하지 않으므로 Amazon DocumentDB Elastic 클러스터 엔드포인트에 새 SSL 인증서를 연결해야 합니다.

Amazon DocumentDB Elastic 클러스터 엔드포인트에 새 SSL 인증서를 연결하려면

1. 브라우저에서 https://www.amazontrust.com/repository/SFSRootCAG2.pem을 열고 콘텐츠를 고유한 파일 이름(예: `SFSRootCAG2.pem`)을 가진 `.pem` 파일에 저장합니다. 이 파일은 다음 단계에서 가져와야 하는 인증서 파일입니다.
2. Elastic 클러스터 엔드포인트를 생성한 후 다음 옵션을 설정합니다.
    a. 엔드포인트 구성에서 새 CA 인증서 추가를 선택합니다.
    b. 인증서 식별자에 **`SFSRootCAG2.pem`**을 입력합니다.
    c. 인증서 파일 가져오기에서 파일 선택을 선택하고 이전에 다운로드한 `SFSRootCAG2.pem` 파일로 이동합니다.
    d. `SFSRootCAG2.pem` 파일 다운로드를 선택하고 엽니다.
    e. [Import certificate]를 선택합니다.
    f. 인증서 선택 드롭다운에서 SFSRootCag2.pem을 선택합니다.

다운로드한 `SFSRootCAG2.pem` 파일의 새 SSL 인증서가 이제 Amazon DocumentDB Elastic 클러스터 엔드포인트에 연결됩니다.

Amazon DocumentDB를 대상으로 지속적 복제

Amazon DocumentDB를 대상으로 지속적인 복제(변경 데이터 캡처, CDC)를 활성화한 경우 AWS DMS 버전 3.5.0 이상에서는 이전 릴리스보다 20배 향상된 성능 향상을 제공합니다. AWS DMS가 초당 최대 250개의 레코드를 처리하던 이전 릴리스에서 AWS DMS는 이제 초당 약 5000개의 레코드를 처리합니다. 또한 AWS DMS는 Amazon DocumentDB의 문서가 소스과 동기화된 상태를 유지하도

록 합니다. 소스 레코드가 생성 또는 업데이트되면 AWS DMS는 먼저 다음을 수행하여 어떤 Amazon DocumentDB 레코드가 영향을 받는지 확인해야 합니다.

- 소스 레코드에 `_id`라는 값이 있으면 해당 열의 값이 Amazon DocumentDB 컬렉션에서 해당되는 `_id`를 확인합니다.
- `_id` 열은 없지만 소스 데이터가 기본 키나 고유 인덱스를 가지고 있으면 AWS DMS는 Amazon DocumentDB 컬렉션을 위한 `_id`로 해당 키나 인덱스 값을 사용합니다.
- 소스 레코드에 `_id` 열, 기본 키 또는 고유 인덱스가 없으면 AWS DMS는 Amazon DocumentDB 컬렉션의 해당 필드에 모든 소스 열을 일치시킵니다.

새로운 소스 레코드가 생성되면 AWS DMS는 해당 문서를 Amazon DocumentDB에 기록합니다. 기존 소스 레코드가 업데이트되면 AWS DMS는 Amazon DocumentDB의 대상 문서에서 해당 필드를 업데이트합니다. 대상 문서에는 존재하지만 소스 레코드에 존재하지 않는 필드는 그대로 남아 있습니다.

소스 레코드가 삭제되면 AWS DMS는 Amazon DocumentDB에서 해당 문서를 삭제합니다.

소스에서 구조 변경(DDL)

지속적 복제가 이루어지면서 소스 데이터 구조(테이블, 열 등)의 변경 사항이 Amazon DocumentDB의 구조에 전파됩니다. 관계형 데이터베이스에서 이러한 변경 사항은 데이터 정의 언어(DDL) 명령문을 사용해 시작됩니다. AWS DMS가 이러한 변경 사항을 Amazon DocumentDB에 어떻게 전파하는지 다음 표에서 확인할 수 있습니다.

소스의 DDL	Amazon DocumentDB 타겟에서의 효과
`CREATE TABLE`	빈 컬렉션을 생성합니다.
테이블 이름을 변경하는 명령문(`RENAME TABLE`, `ALTER TABLE...RENAME` 등)	컬렉션의 이름을 바꿉니다.
`TRUNCATE TABLE`	해당 컬렉션에서 모든 문서를 제거합니다(단, `HandleSourceTableTruncated` 가 `true`인 경우에만). 자세한 내용은 변경 처리 DDL을 다루기 위한 작업 설정 섹션을 참조하세요.
`DROP TABLE`	해당 컬렉션을 삭제합니다(단, `HandleSourceTableDropped` 가 `true`인 경우에만). 자

소스의 DDL	Amazon DocumentDB 타겟에서의 효과
	세한 내용은 변경 처리 DDL을 다루기 위한 작업 설정 섹션을 참조하세요.
테이블에 열을 추가하는 명령문(ALTER TABLE...ADD 등)	DDL 명령문이 무시되면서 경고가 발행됩니다. 첫 번째 INSERT가 소스에서 수행될 때 대상 문서에 새 필드가 추가됩니다.
ALTER TABLE...RENAME COLUMN	DDL 명령문이 무시되면서 경고가 발행됩니다. 첫 번째 INSERT가 소스에서 수행될 때 대상 문서에 새로운 이름의 필드가 추가됩니다.
ALTER TABLE...DROP COLUMN	DDL 명령문이 무시되면서 경고가 발행됩니다.
열 데이터 형식을 변경하는 명령문(ALTER COLUMN...MODIFY 등)	DDL 명령문이 무시되면서 경고가 발행됩니다. 첫 번째 INSERT가 소스에서 새 데이터 형식으로 수행될 때 이러한 새 데이터 형식을 가진 필드와 함께 대상 문서가 생성됩니다.

Amazon DocumentDB를 대상으로 사용할 때 적용되는 제한 사항

Amazon DocumentDB를 AWS DMS의 대상으로 사용할 때는 다음 제한 사항이 적용됩니다.

- Amazon DocumentDB에서 컬렉션 이름에는 달러 기호($)가 포함될 수 없습니다. 또한 데이터베이스 이름에는 어떤 Unicode 문자도 포함될 수 없습니다.
- AWS DMS는 여러 소스 테이블을 단일 Amazon DocumentDB 컬렉션에 병합하는 기능을 지원하지 않습니다.
- AWS DMS가 기본 키가 없는 소스 데이터에서 변경 사항을 처리할 때 해당 테이블의 모든 LOB 열이 무시됩니다.
- 변경 테이블 옵션이 활성화되고 AWS DMS에 "_id"라는 소스 열이 나타나면 해당 열은 변경 테이블에 "__id"(2개의 밑줄)로 표시됩니다.
- 소스 엔드포인트로 Oracle을 선택하면 Oracle 소스에서 전체 보충 로깅을 활성화해야 합니다. 그렇지 않으면 변경되지 않은 소스의 열이 있을 경우에 Amazon DocumentDB에 데이터가 null 값으로 로드됩니다.
- 복제 작업 설정 TargetTablePrepMode:TRUNCATE_BEFORE_LOAD는 DocumentDB 대상 엔드포인트와 함께 사용할 수 없습니다.

Amazon DocumentDB를 대상으로 하는 엔드포인트 설정 사용

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Amazon DocumentDB 대상을 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI의 `create-endpoint` 명령을 `--doc-db-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Amazon DocumentDB를 대상으로 사용할 수 있는 엔드포인트 설정이 다음 테이블에 나와 있습니다.

속성 이름	유효값	기본값과 설명

`replicateShardCollections`	부울
	
	`true`
	
	`false`
		값이 `true`일 때 이 엔드포인트 설정은 다음과 같은 영향을 미치며 다음과 같은 제한이 적용됩니다.
		• AWS DMS는 대상 샤드 컬렉션에 데이터를 복제할 수 있습니다. 이 설정은 대상 DocumentDB 엔드포인트가 Elastic 클러스터인 경우에만 적용됩니다.
		• `TargetTablePrepMode` 를 `DO_NOTHING` 으로 설정해야 합니다.
		• AWS DMS는 마이그레이션 중에 `useUpdateLookup` 을 자동으로 `false`로 설정합니다.

Amazon DocumentDB의 대상 데이터 형식

다음 테이블에서 AWS DMS를 사용할 때 지원되는 Amazon DocumentDB 대상 데이터 형식과 AWS DMS 데이터 형식에서의 기본 매핑을 확인할 수 있습니다. AWS DMS 데이터 형식에 관한 자세한 내용은 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

AWS DMS 데이터 형식	Amazon DocumentDB 데이터 형식
BOOLEAN	부울
BYTES	이진 데이터
DATE	날짜
TIME	문자열(UTF8)

AWS DMS 데이터 형식	Amazon DocumentDB 데이터 형식
DATETIME	날짜
INT1	32비트 정수
INT2	32비트 정수
INT4	32비트 정수
INT8	64비트 정수
NUMERIC	문자열(UTF8)
REAL4	Double
REAL8	Double
STRING	데이터가 JSON으로 인식되면 AWS DMS는 데이터를 Amazon DocumentDB에 문서 형태로 마이그레이션합니다. 그렇지 않으면 데이터가 문자열(UTF8)로 매핑됩니다.
UINT1	32비트 정수
UINT2	32비트 정수
UINT4	64비트 정수
UINT8	문자열(UTF8)
WSTRING	데이터가 JSON으로 인식되면 AWS DMS는 데이터를 Amazon DocumentDB에 문서 형태로 마이그레이션합니다. 그렇지 않으면 데이터가 문자열(UTF8)로 매핑됩니다.
BLOB	이진수
CLOB	데이터가 JSON으로 인식되면 AWS DMS는 데이터를 Amazon DocumentDB에 문서 형태로 마이그레이션합니다. 그렇지 않으면 데이터가 문자열(UTF8)로 매핑됩니다.

AWS DMS 데이터 형식	Amazon DocumentDB 데이터 형식
NCLOB	데이터가 JSON으로 인식되면 AWS DMS는 데이터를 Amazon DocumentDB에 문서 형태로 마이그레이션합니다. 그렇지 않으면 데이터가 문자열(UTF8)로 매핑됩니다.

Amazon Neptune을 AWS Database Migration Service의 대상으로 사용

Amazon Neptune은 빠르고 안정적인 종합 관리형 그래프 데이터베이스 서비스로, 고도로 연결된 데이터 세트를 사용하는 애플리케이션을 쉽게 빌드하고 실행할 수 있습니다. Neptune의 핵심은 특별한 용도의 고성능 그래프 데이터베이스 엔진입니다. 이 엔진은 수십억 개의 관계를 저장하고 몇 밀리초의 지연 시간으로 그래프를 쿼리하도록 최적화되었습니다. Neptune은 인기 있는 그래프 쿼리 언어인 Apache TinkerPop Gremlin과 W3C의 SPARQL을 지원합니다. Amazon Neptune에 관한 자세한 내용은 Amazon Neptune 사용 설명서의 Amazon Neptune이란 무엇인가? 섹션을 참조하세요.

Neptune과 같은 그래프 데이터베이스가 없더라도 관계형 데이터베이스에서 고도로 연결된 데이터를 모델링할 수 있습니다. 데이터에 동적 연결이 있을 가능성이 있기 때문에 이러한 데이터 소스를 사용하는 애플리케이션은 연결된 데이터 쿼리를 SQL로 모델링해야 합니다. 이 방법을 사용하려면 그래프 쿼리를 SQL로 변환하기 위해 추가 계층을 작성해야 합니다. 또한 관계형 데이터베이스에는 스키마 엄격성이 있습니다. 변경되는 연결을 모델링하기 위해 스키마를 변경하면 가동 중지가 발생하고 새 스키마를 지원하기 위해 쿼리 변환을 추가로 유지 관리해야 합니다. 쿼리 성능은 애플리케이션을 설계하는 동안 고려해야 할 또 하나의 큰 제약 조건입니다.

그래프 데이터베이스는 이러한 상황을 크게 단순화할 수 있습니다. 스키마 없이 서식 있는 그래프 쿼리 계층(Gremlin 또는 SPARQL) 및 그래프 쿼리에 최적화된 인덱스는 유연성과 성능을 향상시킵니다. Amazon Neptune 그래프 데이터베이스에는 유휴 시 암호화, 보안 승인 계층, 기본 백업, 다중 AZ 지원, 읽기 전용 복제 지원 등의 엔터프라이즈 기능도 있습니다.

AWS DMS를 사용하면 고도로 연결된 그래프를 모델링하는 관계형 데이터를 지원되는 모든 SQL 데이터베이스의 DMS 소스 엔드포인트에서 Neptune 대상 엔드포인트로 마이그레이션할 수 있습니다.

자세한 내용은 다음을 참조하십시오.

주제

- 대상으로서 Amazon Neptune으로 마이그레이션 개요
- 대상으로서 Amazon Neptune에 대한 엔드포인트 설정 지정
- Amazon Neptune에 대상으로 액세스하기 위한 IAM 서비스 역할 생성

- Amazon Neptune에 대한 Gremlin 및 R2RML을 대상으로 사용하여 그래프 매핑 규칙 지정
- 대상으로서 Amazon Neptune으로 Gremlin 및 R2RML을 마이그레이션하기 위한 데이터 형식
- Amazon Neptune을 대상으로 사용 시 제한 사항

대상으로서 Amazon Neptune으로 마이그레이션 개요

AWS 대상으로 마이그레이션을 시작하기 전에 계정에 다음 리소스를 생성합니다.

- 대상 엔드포인트에 대한 Neptune 클러스터.
- 소스 엔드포인트에 대해 AWS DMS에서 지원하는 SQL 관계형 데이터베이스.
- 대상 엔드포인트용 Amazon S3 버킷. Neptune 클러스터와 동일한 AWS 리전에 이 S3 버킷을 생성합니다. AWS DMS는 이 S3 버킷을 Neptune 데이터베이스에 대량 로드하는 대상 데이터에 대한 중간 파일 스토리지로 사용합니다. Amazon S3 버킷 생성에 관한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 생성 섹션을 참조하세요.
- Neptune 클러스터와 동일한 VPC에 있는 S3에 대한 가상 프라이빗 클라우드(VPC) 엔드포인트.
- IAM 정책을 포함하는 AWS Identity and Access Management(IAM) 역할. 이 정책은 대상 엔드포인트에 대한 S3 버킷에 `GetObject`, `PutObject`, `DeleteObject` 및 `ListObject` 권한을 지정해야 합니다. 이 역할은 대상 S3 버킷과 Neptune 데이터베이스 모두에 대한 IAM 액세스 권한을 가진 AWS DMS 및 Neptune 모두에 위임됩니다. 자세한 내용은 Amazon Neptune에 대상으로 액세스하기 위한 IAM 서비스 역할 생성 섹션을 참조하세요.

이러한 리소스를 확보한 후 Neptune 대상으로의 마이그레이션을 설정하고 시작하는 방법은 콘솔 또는 DMS API를 사용하는 전체 로드 마이그레이션과 유사합니다. 그러나 Neptune 대상으로 마이그레이션하려면 몇 가지 고유한 단계가 필요합니다.

AWS DMS 관계형 데이터베이스를 Neptune으로 마이그레이션하려면

1. 복제 인스턴스 생성에 설명된 대로 복제 인스턴스를 생성합니다.
2. AWS DMS에서 소스 엔드포인트에 대해 지원하는 SQL 관계형 데이터베이스를 만들고 테스트합니다.
3. Neptune 데이터베이스의 대상 엔드포인트를 만들고 테스트합니다.

   대상 엔드포인트를 Neptune 데이터베이스에 연결하려면 Neptune 클러스터 엔드포인트 또는 Neptune 라이터 인스턴스 엔드포인트에 대한 서버 이름을 지정합니다. 또한 AWS DMS에 대한 S3 버킷 폴더를 지정하여 대량 로드할 중간 파일을 Neptune 데이터베이스에 저장합니다.

마이그레이션하는 동안 AWS DMS에서는 마이그레이션된 모든 대상 데이터를 사용자가 지정한 최대 파일 크기까지 이 S3 버킷 폴더에 저장합니다. 이 파일 스토리지가 최대 크기에 도달하면 AWS DMS는 저장된 S3 데이터를 대상 데이터베이스에 대량으로 로드합니다. 그러면 폴더를 지워서 이후에 대상 데이터베이스에 로드할 추가 대상 데이터를 저장할 수 있습니다. 이러한 설정 지정에 관한 자세한 내용은 대상으로서 Amazon Neptune에 대한 엔드포인트 설정 지정 단원을 참조하십시오.

4. 1~3단계에서 만든 리소스를 사용하여 전체 로드 복제 작업을 생성한 후 다음을 수행합니다.

   a. 평소와 같이 작업 테이블 매핑을 사용하여 적절한 선택 및 변환 규칙을 사용하여 관계형 데이터베이스에서 마이그레이션할 특정 소스 스키마, 테이블 및 뷰를 식별합니다. 자세한 내용은 작업 설정을 지정하기 위한 테이블 매핑 사용 섹션을 참조하세요.

   b. 다음 중 하나를 선택하여 소스 테이블 및 뷰에서 대상 데이터베이스 그래프로의 매핑 규칙을 지정하여 Neptune 대상 매핑을 지정합니다.

      - Gremlin JSON — Gremlin JSON을 사용하여 Neptune 데이터베이스를 로드하는 방법에 관한 자세한 내용은 Amazon Neptune 사용 설명서의 Gremlin 로드 데이터 형식을 참조하세요.
      - SPARQL RDB에서 R2RML(Resource Description Framework Mapping Language)로 – SPARQL R2RML 사용에 관한 자세한 내용은 W3C 사양 R2RML: RDB-RDF 매핑 언어를 참조하십시오.

   c. 다음 중 하나를 수행하세요.

      - AWS DMS 콘솔을 사용하여 데이터베이스 마이그레이션 작업 생성 페이지의 그래프 매핑 규칙을 사용하여 그래프 매핑 옵션을 지정합니다.
      - AWS DMS API를 사용하면서 `CreateReplicationTask` API 직접 호출의 `TaskData` 요청 파라미터를 사용하여 이러한 옵션을 지정합니다.

      Gremlin JSON 및 SPARQL R2RML을 사용하여 그래프 매핑 규칙을 지정하는 방법에 관한 자세한 내용과 예제는 Amazon Neptune에 대한 Gremlin 및 R2RML을 대상으로 사용하여 그래프 매핑 규칙 지정 단원을 참조하십시오.

5. 마이그레이션 작업에 대한 복제를 시작합니다.

대상으로서 Amazon Neptune에 대한 엔드포인트 설정 지정

대상 엔드포인트를 만들거나 수정하려면 콘솔이나 `CreateEndpoint` 또는 `ModifyEndpoint` API 작업을 사용할 수 있습니다.

AWS DMS 콘솔 내 Neptune 대상의 경우, 엔드포인트 생성 또는 엔드포인트 수정 콘솔 페이지에서 엔드포인트별 설정을 지정합니다. `CreateEndpoint`, `ModifyEndpoint`의 경우 `NeptuneSettings` 옵션에 대한 요청 파라미터를 지정합니다. 다음 예에서는 CLI를 사용하여 이 작업을 수행하는 방법을 보여줍니다.

```
dms create-endpoint --endpoint-identifier my-neptune-target-endpoint
--endpoint-type target --engine-name neptune
--server-name my-neptune-db.cluster-cspckvklbvgf.us-east-1.neptune.amazonaws.com
--port 8192
--neptune-settings
     '{"ServiceAccessRoleArn":"arn:aws:iam::123456789012:role/myNeptuneRole",
       "S3BucketName":"my-bucket",
       "S3BucketFolder":"my-bucket-folder",
       "ErrorRetryDuration":57,
       "MaxFileSize":100,
       "MaxRetryCount": 10,
       "IAMAuthEnabled":false}‘
```

여기서 CLI `--server-name` 옵션은 Neptune 클러스터 라이터 엔드포인트의 서버 이름을 지정합니다. 또는 Neptune 라이터 인스턴스 엔드포인트에 대한 서버 이름을 지정할 수 있습니다.

`--neptune-settings` 옵션 요청 파라미터는 다음과 같습니다.

- `ServiceAccessRoleArn` – (필수) Neptune 대상 엔드포인트에 대해 생성한 서비스 역할의 Amazon 리소스 이름(ARN)입니다. 자세한 내용은 Amazon Neptune에 대상으로 액세스하기 위한 IAM 서비스 역할 생성 섹션을 참조하세요.
- `S3BucketName` – (필수) DMS가 마이그레이션된 그래프 데이터를 Neptune 대상 데이터베이스에 대량으로 로드하기 전에 .csv 파일에 임시로 저장할 수 있는 S3 버킷의 이름입니다. DMS는 이 .csv 파일에 저장하기 전에 SQL 소스 데이터를 그래프 데이터에 매핑합니다.
- `S3BucketFolder` – (필수) DMS가 `S3BucketName`에 의해 지정된 S3 버킷에 마이그레이션된 그래프 데이터를 저장할 때 사용할 폴더 경로입니다.
- `ErrorRetryDuration` – (옵션) DMS는 이 시간(밀리초) 동안 기다렸다가 마이그레이션된 그래프 데이터를 Neptune 대상 데이터베이스에 대량 로드하고자 재시도한 다음 오류를 발생시킵니다. 기본값은 250입니다.

- `MaxFileSize` – (옵션) DMS가 Neptune 대상 데이터베이스에 데이터를 대량으로 로드하기 전에 .csv 파일에 저장되어 있던 마이그레이션된 그래프 데이터의 최대 크기(KB)입니다. 기본값은 1,048,576KB(1GB)입니다. 성공하면 DMS는 버킷을 지우고 마이그레이션된 그래프 데이터의 다음 배치를 저장할 준비가 됩니다.
- `MaxRetryCount` – (옵션) DMS는 이 횟수만큼 Neptune 대상 데이터베이스로 마이그레이션된 그래프 데이터의 대량 로드를 재시도한 뒤에 오류를 발생시킵니다. 기본값은 5입니다.
- `IAMAuthEnabled` – (옵션) 이 엔드포인트에 대해 IAM 권한 부여를 활성화하려면 이 파라미터를 `true`로 설정하고 `ServiceAccessRoleArn`에서 지정한 서비스 역할에 적절한 IAM 정책 문서를 연결합니다. 기본값은 `false`입니다.

Amazon Neptune에 대상으로 액세스하기 위한 IAM 서비스 역할 생성

Neptune에 대상으로 액세스하려면 IAM을 사용하여 서비스 역할을 만듭니다. Neptune 엔드포인트 구성에 따라 이 역할에 IAM 정책의 일부 또는 전부를 연결하고 다음에 설명된 문서를 신뢰합니다. Neptune 엔드포인트를 생성할 때 이 서비스 역할의 ARN을 제공합니다. 이렇게 하면 Neptune과 그에 연결된 Amazon S3 버킷 모두에 액세스할 수 있는 권한을 AWS DMS 및 Amazon Neptune이 수임할 수 있습니다.

Neptune 엔드포인트 구성에서 `NeptuneSettings`의 `IAMAuthEnabled` 파라미터를 `true`로 설정한 경우, 다음과 같은 IAM 정책을 서비스 역할에 연결합니다. `IAMAuthEnabled`를 `false`로 설정하면 이 정책을 무시할 수 있습니다.

```
// Policy to access Neptune

    {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "VisualEditor0",
                "Effect": "Allow",
                "Action": "neptune-db:*",
                "Resource": "arn:aws:neptune-db:us-east-1:123456789012:cluster-
CLG7H7FHK54AZGHEH6MNS55JKM/*"
            }
        ]
    }
```

앞의 IAM 정책은 `Resource`에서 지정한 Neptune 대상 클러스터에 대한 전체 액세스를 허용합니다.

다음과 같은 IAM 정책을 서비스 역할에 연결합니다. 이 정책을 통해 DMS는 Neptune 대상 데이터베이스에 대량 로드하기 위해 생성한 S3 버킷에 마이그레이션된 그래프 데이터를 임시로 저장할 수 있습니다.

```
//Policy to access S3 bucket

{
 "Version": "2012-10-17",
 "Statement": [{
   "Sid": "ListObjectsInBucket0",
   "Effect": "Allow",
   "Action": "s3:ListBucket",
   "Resource": [
    "arn:aws:s3:::my-bucket"
   ]
  },
  {
   "Sid": "AllObjectActions",
   "Effect": "Allow",
   "Action": ["s3:GetObject",
    "s3:PutObject",
    "s3:DeleteObject"
   ],

   "Resource": [
    "arn:aws:s3:::my-bucket/"
   ]
  },
  {
   "Sid": "ListObjectsInBucket1",
   "Effect": "Allow",
   "Action": "s3:ListBucket",
   "Resource": [
    "arn:aws:s3:::my-bucket",
    "arn:aws:s3:::my-bucket/"
   ]
  }
 ]
}
```

앞의 IAM 정책을 사용하면 계정에서 Neptune 대상에 대해 생성된 S3 버킷(`arn:aws:s3:::my-bucket`)의 콘텐츠를 쿼리할 수 있습니다. 또한 계정에서 모든 버킷 파일 및 폴더(`arn:aws:s3:::my-bucket/`)의 콘텐츠에 대해 완벽하게 작동할 수 있습니다.

신뢰 관계를 편집하고 다음 IAM 역할을 서비스 역할에 연결하여 AWS DMS 및 Amazon Neptune 데이터베이스 서비스가 역할을 수임할 수 있도록 합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    },
    {
      "Sid": "neptune",
      "Effect": "Allow",
      "Principal": {
        "Service": "rds.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Neptune 대상 엔드포인트에 대해 이 서비스 역할을 지정하는 방법에 관한 자세한 내용은 대상으로서 Amazon Neptune에 대한 엔드포인트 설정 지정 단원을 참조하십시오.

Amazon Neptune에 대한 Gremlin 및 R2RML을 대상으로 사용하여 그래프 매핑 규칙 지정

생성하는 그래프 매핑 규칙은 SQL 관계형 데이터베이스 소스에서 추출된 데이터를 Neptune 데이터베이스 클러스터 대상으로 로드하는 방법을 지정합니다. 이러한 매핑 규칙의 형식은 규칙이 Apache TinkerPop Greglin을 사용하여 속성-그래프 데이터를 로드하는지 또는 R2RML을 사용하여 RDF(Resource Description Framework) 데이터를 로드하는지 여부에 따라 다릅니다. 다음에서는 이러한 형식에 대한 정보와 확인할 위치를 찾을 수 있습니다.

콘솔 또는 DMS API를 사용하여 마이그레이션 작업을 생성할 때 이러한 매핑 규칙을 지정할 수 있습니다.

콘솔을 사용하면서 데이터베이스 마이그레이션 작업 생성 페이지의 그래프 매핑 규칙을 사용하여 이러한 그래프 매핑 규칙을 지정합니다. 그래프 매핑 규칙에서 제공된 편집기를 사용하여 매핑 규칙을 직접 입력하고 편집할 수 있습니다. 또는 적절한 그래프 매핑 형식으로 매핑 규칙이 포함된 파일을 찾아볼 수 있습니다.

API를 사용하여 `CreateReplicationTask` API 직접 호출의 `TaskData` 요청 파라미터를 사용하여 이러한 옵션을 지정합니다. 적절한 그래프 매핑 형식의 매핑 규칙이 포함된 파일의 경로로 `TaskData`를 설정합니다.

Gremlin을 사용하여 속성-그래프 데이터를 생성하기 위한 그래프 매핑 규칙

Gremlin을 사용하여 속성-그래프 데이터를 생성하고 소스 데이터에서 생성할 각 그래프 엔터티에 대한 매핑 규칙이 있는 JSON 객체를 지정합니다. 이 JSON의 형식은 특별히 Amazon Neptune 대량 로드를 위해 정의됩니다. 다음 템플릿에서는 이 객체의 각 규칙이 어떻게 표시되는지 보여줍니다.

```
{
    "rules": [
        {
            "rule_id": "(an identifier for this rule)",
            "rule_name": "(a name for this rule)",
            "table_name": "(the name of the table or view being loaded)",
            "vertex_definitions": [
                {
                    "vertex_id_template": "{col1}",
                    "vertex_label": "(the vertex to create)",
                    "vertex_definition_id": "(an identifier for this vertex)",
                    "vertex_properties": [
                        {
                            "property_name": "(name of the property)",
                            "property_value_template": "{col2} or text",
                            "property_value_type": "(data type of the property)"
                        }
                    ]
                }
            ]
        },
        {
            "rule_id": "(an identifier for this rule)",
            "rule_name": "(a name for this rule)",
```

```
            "table_name": "(the name of the table or view being loaded)",
            "edge_definitions": [
                {
                    "from_vertex": {
                        "vertex_id_template": "{col1}",
                        "vertex_definition_id": "(an identifier for the vertex
 referenced above)"
                    },
                    "to_vertex": {
                        "vertex_id_template": "{col3}",
                        "vertex_definition_id": "(an identifier for the vertex
 referenced above)"
                    },
                    "edge_id_template": {
                        "label": "(the edge label to add)",
                        "template": "{col1}_{col3}"
                    },
                    "edge_properties":[
                        {
                            "property_name": "(the property to add)",
                            "property_value_template": "{col4} or text",
                            "property_value_type": "(data type like String, int,
 double)"
                        }
                    ]
                }
            ]
        }
    ]
}
```

버텍스 레이블이 있으면 여기에 버텍스가 만들어지고 있음을 의미합니다. 없는 경우, 버텍스가 다른 소스에 의해 생성된다는 것을 의미하며 이 정의는 버텍스 속성만 추가하는 것입니다. 전체 관계형 데이터베이스 소스에 대한 매핑을 지정하는 데 필요한 만큼 버텍스 및 엣지 정의를 지정합니다.

`employee` 테이블에 대한 샘플 규칙은 다음과 같습니다.

```
{
    "rules": [
        {
            "rule_id": "1",
```

```
            "rule_name": "vertex_mapping_rule_from_nodes",
            "table_name": "nodes",
            "vertex_definitions": [
                {
                    "vertex_id_template": "{emp_id}",
                    "vertex_label": "employee",
                    "vertex_definition_id": "1",
                    "vertex_properties": [
                        {
                            "property_name": "name",
                            "property_value_template": "{emp_name}",
                            "property_value_type": "String"
                        }
                    ]
                }
            ]
        },
        {
            "rule_id": "2",
            "rule_name": "edge_mapping_rule_from_emp",
            "table_name": "nodes",
            "edge_definitions": [
                {
                    "from_vertex": {
                        "vertex_id_template": "{emp_id}",
                        "vertex_definition_id": "1"
                    },
                    "to_vertex": {
                        "vertex_id_template": "{mgr_id}",
                        "vertex_definition_id": "1"
                    },
                    "edge_id_template": {
                        "label": "reportsTo",
                        "template": "{emp_id}_{mgr_id}"
                    },
                    "edge_properties":[
                        {
                            "property_name": "team",
                            "property_value_template": "{team}",
                            "property_value_type": "String"
                        }
                    ]
                }
            ]
```

```
        }
    ]
}
```

여기서, 버텍스 및 엣지 정의는 직원 ID(EmpID)의 employee 노드와 관리자 ID(managerId)의 employee 노드에서 보고 관계를 매핑합니다.

Gremlin JSON을 사용하여 그래프 매핑 규칙을 생성하는 방법에 관한 자세한 내용은 Amazon Neptune 사용 설명서의 Gremlin 로드 데이터 형식을 참조하십시오.

RDF/SPARQL 데이터 생성을 위한 그래프 매핑 규칙

SPARQL을 사용하여 쿼리할 RDF 데이터를 로드하는 경우 R2RML로 그래프 매핑 규칙을 작성합니다. R2RML은 RDF에 관계형 데이터를 매핑하기 위한 표준 W3C 언어입니다. R2RML 파일에서 triples 맵(예: 다음 <#TriplesMap1>)은 논리 테이블의 각 행을 0개 이상의 RDF triples로 변환하는 규칙을 지정합니다. 주제 맵(예: 다음 rr:subjectMap)은 트리플 맵에 의해 생성된 RDF triples의 주제를 생성하기 위한 규칙을 지정합니다. 조건자-객체 맵(예: 다음 rr:predicateObjectMap)은 논리 테이블의 각 논리 테이블 행에 대해 하나 이상의 조건자-객체 쌍을 만드는 함수입니다.

nodes 테이블에 대한 간단한 예는 다음과 같습니다.

```
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix ex: <http://example.com/ns#>.

<#TriplesMap1>
    rr:logicalTable [ rr:tableName "nodes" ];
    rr:subjectMap [
        rr:template "http://data.example.com/employee/{id}";
        rr:class ex:Employee;
    ];
    rr:predicateObjectMap [
        rr:predicate ex:name;
        rr:objectMap [ rr:column "label" ];
    ]
```

앞의 예에서 매핑은 직원 테이블에서 매핑된 그래프 노드를 정의합니다.

Student 테이블에 대한 또 다른 간단한 예는 다음과 같습니다.

```
@prefix rr: <http://www.w3.org/ns/r2rml#>.
```

```
@prefix ex: <http://example.com/#>.
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.

<#TriplesMap2>
    rr:logicalTable [ rr:tableName "Student" ];
    rr:subjectMap   [ rr:template "http://example.com/{ID}{Name}";
                      rr:class foaf:Person ];
    rr:predicateObjectMap [
        rr:predicate ex:id ;
        rr:objectMap  [ rr:column "ID";
                        rr:datatype xsd:integer ]
    ];
    rr:predicateObjectMap [
        rr:predicate foaf:name ;
        rr:objectMap  [ rr:column "Name" ]
    ].
```

앞의 예에서 매핑은 `Student` 테이블의 사람 사이의 친구 관계를 매핑하는 그래프 노드를 정의합니다.

SPARQL R2RML을 사용하여 그래프 매핑 규칙을 생성하는 방법에 관한 자세한 내용은 W3C 사양 R2RML: RDB-RDF 매핑 언어를 참조하십시오.

대상으로서 Amazon Neptune으로 Gremlin 및 R2RML을 마이그레이션하기 위한 데이터 형식

AWS DMS는 두 가지 방법 중 하나로 SQL 소스 엔드포인트에서 Neptune 대상으로 데이터 형식 매핑을 수행합니다. 사용하는 방법은 Neptune 데이터베이스를 로드하는 데 사용하는 그래프 매핑 형식에 따라 다릅니다.

- Apache TinkerPop Gremlin, 마이그레이션 데이터의 JSON 표현을 사용.
- W3C의 SPARQL, 마이그레이션 데이터의 R2RML 표현을 사용.

이러한 두 그래프 매핑 형식에 관한 자세한 내용은 Amazon Neptune에 대한 Gremlin 및 R2RML을 대상으로 사용하여 그래프 매핑 규칙 지정 단원을 참조하십시오.

다음에서는 각 형식에 대한 데이터 형식 매핑에 대한 설명을 찾을 수 있습니다.

SQL 소스를 Gremlin 대상 데이터 형식 매핑으로

다음 표에서는 SQL 소스에서 Gremlin 형식 대상으로의 데이터 형식 매핑을 보여줍니다.

AWS DMS는 나열되지 않은 SQL 소스 데이터 형식을 Gremlin `String`에 매핑합니다.

SQL 소스 데이터 형식	Gremlin 대상 데이터 형식
NUMERIC(및 변형)	Double
DECIMAL	
TINYINT	Byte
SMALLINT	Short
INT, INTEGER	Int
BIGINT	Long
FLOAT	Float
DOUBLE PRECISION	
REAL	Double
BIT	Boolean
BOOLEAN	
DATE	Date
TIME	
TIMESTAMP	
CHARACTER (및 변형)	String

Neptune을 로드하기 위한 Gremlin 데이터 형식에 관한 자세한 내용은 Neptune 사용 설명서의 Gremlin 데이터 형식을 참조하십시오.

SQL 소스를 R2RML(RDF) 대상 데이터 형식 매핑으로

다음 표에서는 SQL 소스에서 R2RML 형식 대상으로의 데이터 형식 매핑을 보여줍니다.

나열된 모든 RDF 데이터 형식은 RDF 리터럴을 제외하고 대소문자를 구분합니다. AWS DMS는 나열되지 않은 SQL 소스 데이터 형식을 RDF 리터럴에 매핑합니다.

RDF 리터럴은 다양한 리터럴 어휘 형태와 데이터 형식 중 하나입니다. 자세한 내용은 W3C 사양 리소스 설명 프레임워크 (RDF): 개념 및 추상 구문의 RDF 리터럴을 참조하십시오.

SQL 소스 데이터 형식	R2RML(RDF) 대상 데이터 형식
BINARY(및 변형)	xsd:hexBinary
NUMERIC(및 변형)	xsd:decimal
DECIMAL	
TINYINT	xsd:integer
SMALLINT	
INT, INTEGER	
BIGINT	
FLOAT	xsd:double
DOUBLE PRECISION	
REAL	
BIT	xsd:boolean
BOOLEAN	
DATE	xsd:date
TIME	xsd:time
TIMESTAMP	xsd:dateTime
CHARACTER (및 변형)	RDF 리터럴

Neptune을 로드하기 위한 RDF 데이터 형식 및 SQL 소스 데이터 형식으로의 매핑에 관한 자세한 내용은 W3C 사양 R2RML: RDB에서 RDF로 매핑 언어의 데이터 형식 변환을 참조하십시오.

Amazon Neptune을 대상으로 사용 시 제한 사항

Neptune을 대상으로 사용할 때에는 다음 제한 사항이 적용됩니다.

- AWS DMS는 현재 Neptune 대상으로의 마이그레이션에 대해서만 전체 로드 작업을 지원합니다. Neptune 대상으로의 CDC(데이터 캡처 변경) 마이그레이션은 지원되지 않습니다.
- 다음 예와 같이 마이그레이션 작업을 시작하기 전에 대상 Neptune 데이터베이스에서 모든 데이터를 수동으로 지워야 합니다.

  그래프 내의 모든 데이터(버텍스 및 엣지)를 삭제하려면 다음 Gremlin 명령을 실행합니다.

  ```
  gremlin> g.V().drop().iterate()
  ```

  `'customer'` 레이블이 있는 버텍스를 삭제하려면 다음 Gremlin 명령을 실행합니다.

  ```
  gremlin> g.V().hasLabel('customer').drop()
  ```

  > **Note**
  >
  > 대용량 데이터 세트를 삭제하려면 약간의 시간이 걸릴 수 있습니다. 필요하다면 제한을 두어 `drop()`을 반복할 수도 있습니다(예: `limit(1000)`).

  `'rated'` 레이블이 있는 엣지를 삭제하려면 다음 Gremlin 명령을 실행합니다.

  ```
  gremlin> g.E().hasLabel('rated').drop()
  ```

  > **Note**
  >
  > 대용량 데이터 세트를 삭제하려면 약간의 시간이 걸릴 수 있습니다. 필요하다면 제한을 두어 `drop()`을 반복할 수도 있습니다(예: `limit(1000)`).

- DMS API 연산 `DescribeTableStatistics`는 Neptune 그래프 데이터 구조의 속성으로 인해 주어진 테이블에 관한 부정확한 결과를 반환할 수 있습니다.

마이그레이션하는 동안 AWS DMS에서는 각 소스 테이블을 스캔하고 그래프 매핑을 사용하여 소스 데이터를 Neptune 그래프로 변환합니다. 변환된 데이터는 먼저 대상 엔드포인트에 지정된 S3 버킷 폴더에 저장됩니다. 소스가 검색되고 이 중간 S3 데이터가 성공적으로 생성되면 `DescribeTableStatistics`는 데이터가 Neptune 대상 데이터베이스에 성공적으로 로드되었다고 가정합니다. 그러나 이것이 항상 사실인 것은 아닙니다. 지정된 테이블에 대해 데이터가 올바르게 로드되었는지 확인하려면 해당 테이블에 대한 마이그레이션의 양측에서 `count()` 반환 값을 비교합니다.

다음 예에서는 AWS DMS가 소스 데이터베이스에서 `customer` 테이블을 로드했습니다. 여기에서 대상 Neptune 데이터베이스 그래프에 `'customer'` 레이블이 지정됩니다. 그러면 이 레이블이 대상 데이터베이스에 확실하게 기록됩니다. 이렇게 하려면 소스 데이터베이스에서 사용할 수 있는 `customer` 행 수를 작업이 완료된 후 Neptune 대상 데이터베이스에 로드된 `'customer'` 레이블이 지정된 행 수와 비교합니다.

SQL을 사용하여 소스 데이터베이스에서 사용 가능한 고객 행 수를 가져오려면 다음을 실행합니다.

```
select count(*) from customer;
```

Gremlin을 사용하여 대상 데이터베이스 그래프에 로드된 `'customer'` 레이블이 지정된 행 수를 얻으려면 다음을 실행하십시오.

```
gremlin> g.V().hasLabel('customer').count()
```

- 현재 단일 테이블이 로드되지 않으면 전체 작업이 실패합니다. 관계형 데이터베이스 대상과는 달리 Neptune의 데이터는 고도로 연결되어 있으므로 대부분의 경우 작업을 다시 시작할 수 없습니다. 이러한 유형의 데이터 로드 실패로 인해 작업을 성공적으로 재개할 수 없는 경우, 로드하지 못한 테이블을 로드하는 새 작업을 생성합니다. 이 새 작업을 실행하기 전에 Neptune 대상에서 부분적으로 로드된 테이블을 수동으로 지웁니다.

  > **Note**
  >
  > 장애를 복구할 수 있는 경우(예: 네트워크 전송 오류) Neptune 대상으로의 마이그레이션에 실패한 작업을 재개할 수 있습니다.

- AWS DMS는 R2RML에 대한 대부분의 표준을 지원합니다. 그러나 AWS DMS에서는 역 표현식, 조인 및 뷰를 비롯한 특정 R2RML 표준을 지원하지 않습니다. R2RML 뷰의 차선책은 소스 데이터베이스에 해당 사용자 지정 SQL 뷰를 만드는 것입니다. 마이그레이션 작업에서 테이블 매핑을 사용하여

뷰를 입력으로 선택합니다. 그런 다음, 테이블에 뷰를 매핑하면 R2RML이 이를 사용하여 그래프 데이터를 생성합니다.

- 지원되지 않는 SQL 데이터 형식을 사용하여 소스 데이터를 마이그레이션하면 결과 대상 데이터의 정밀도가 손실될 수 있습니다. 자세한 내용은 대상으로서 Amazon Neptune으로 Gremlin 및 R2RML을 마이그레이션하기 위한 데이터 형식 섹션을 참조하세요.
- AWS DMS는 LOB 데이터를 Neptune 대상으로 마이그레이션하는 것을 지원하지 않습니다.

Redis를 AWS Database Migration Service의 대상으로 사용

Redis는 데이터베이스, 캐시 및 메시지 브로커로 사용되는 오픈 소스 인메모리 데이터 구조 저장소입니다. 데이터를 인메모리로 관리하면 읽기 또는 쓰기 작업의 소요 시간이 밀리초 미만으로 단축되며 초당 수억 건의 작업을 수행할 수 있습니다. Redis는 인메모리 데이터 스토어로서 밀리초 미만의 응답 시간을 요구하는 가장 까다로운 애플리케이션을 지원합니다.

AWS DMS를 사용하면 가동 중지 시간을 최소화하면서 지원되는 모든 소스 데이터베이스의 데이터를 대상 Redis 데이터 스토어로 마이그레이션할 수 있습니다. Redis에 관한 추가 정보는 Redis 설명서를 참조하십시오.

AWS Database Migration Service은 온프레미스 Redis와 함께 다음을 지원합니다.

- 대상 데이터 스토어로 사용되는 Amazon ElastiCache for Redis ElastiCache for Redis는 Redis 클라이언트와 연동하며 개방형 Redis 데이터 형식을 사용하여 데이터를 저장합니다.
- 대상 데이터 스토어로 사용되는 Amazon MemoryDB for Redis MemoryDB는 Redis와 호환되며 현재 사용 중인 모든 Redis 데이터 구조, API 및 명령을 사용하여 애플리케이션을 구축할 수 있습니다.

Redis를 AWS DMS의 대상으로 연동하는 방법에 관한 자세한 내용은 다음 섹션을 참조하세요.

주제

- Redis 클러스터를 AWS DMS에 대한 대상으로 사용하기 위한 사전 조건
- AWS Database Migration Service의 대상으로서 Redis 사용 시 제한 사항
- 관계형 또는 비관계형 데이터베이스에서 Redis 대상으로 데이터 마이그레이션
- 대상으로서 Redis에 대한 엔드포인트 설정 지정

Redis 클러스터를 AWS DMS에 대한 대상으로 사용하기 위한 사전 조건

DMS는 독립형 구성의 온프레미스 Redis 대상을 지원하며 또는 데이터가 여러 노드에 걸쳐 자동으로 분할(샤딩)되는 Redis 클러스터로 지원합니다. 샤딩은 데이터를 여러 서버 또는 노드에 분산된 '샤드'라는 작은 청크로 분할하는 프로세스입니다. 실제로 하나의 샤드는 전체 데이터 세트의 하위 집합을 포함하며 전체 워크로드의 일부를 차지하는 데이터 파티션입니다.

Redis는 키-값 NoSQL 데이터 스토어이므로 소스가 관계형 데이터베이스일 때 사용하는 Redis 키 명명 규칙은 schema-name.table-name.primary key입니다. Redis에서는 키와 값에 특수 문자 %가 포함되어서는 안 됩니다. 그렇지 않으면 DMS는 레코드를 건너뜁니다.

> **Note**
>
> ElastiCache for Redis를 대상으로 사용하는 경우, DMS는 클러스터 모드가 활성화된 구성만 지원합니다. ElastiCache for Redis 버전 6.x 또는 이후 버전을 사용하여 클러스터 모드가 활성화된 대상 데이터 스토어를 생성하는 방법에 관한 자세한 내용은 Amazon ElastiCache for Redis 사용 설명서의 시작하기를 참조하세요.

데이터베이스 마이그레이션을 시작하기 전에 다음 기준에 따라 Redis 클러스터를 시작하십시오.

- 클러스터에 샤드가 하나 이상 있습니다.
- ElastiCache for Redis 대상을 사용하는 경우, 클러스터에서 IAM 역할 기반 액세스 제어를 사용하지 않도록 하세요. 그 대신 Redis 인증을 사용하여 사용자를 인증하세요.
- 다중 AZ(가용 영역)를 활성화합니다.
- 데이터베이스에서 마이그레이션할 데이터에 맞는 충분한 메모리가 클러스터에 있는지 확인하십시오.
- 초기 마이그레이션 작업을 시작하기 전에 대상 Redis 클러스터에서 모든 데이터가 삭제되었는지 확인하십시오.

클러스터 구성을 생성하기 전에 데이터 마이그레이션에 대한 보안 요구 사항을 결정해야 합니다. DMS는 암호화 구성과 상관없이 대상 복제 그룹으로의 마이그레이션을 지원합니다. 하지만 클러스터 구성을 생성할 때만 암호화를 활성화하거나 비활성화할 수 있습니다.

AWS Database Migration Service의 대상으로서 Redis 사용 시 제한 사항

Redis를 대상으로 사용할 때에는 다음 제한 사항이 적용됩니다.

- Redis는 키-값 NoSQL 데이터 스토어이므로 소스가 관계형 데이터베이스일 때 사용하는 Redis 키 명명 규칙은 `schema-name.table-name.primary-key`입니다.
- Redis에서는 키-값에 특수 문자 %가 포함되어서는 안 됩니다. 그렇지 않으면 DMS는 레코드를 건너뜁니다.
- DMS는 특수 문자가 포함된 행을 마이그레이션하지 않습니다.
- DMS는 필드 이름에 특수 문자가 포함된 필드를 마이그레이션하지 않습니다.
- 전체 LOB 모드는 지원되지 않습니다.
- ElastiCache for Redis를 대상으로 사용할 때는 프라이빗 인증 기관(CA)이 지원되지 않습니다.

관계형 또는 비관계형 데이터베이스에서 Redis 대상으로 데이터 마이그레이션

모든 소스 SQL 또는 NoSQL 데이터 스토어의 데이터를 Redis 대상으로 직접 마이그레이션할 수 있습니다. Redis 대상으로의 마이그레이션을 설정하고 시작하는 방법은 DMS 콘솔 또는 API를 사용하는 전체 로드 및 데이터 캡처 변경(CDC) 마이그레이션과 유사합니다. Redis 대상으로 데이터베이스 마이그레이션을 수행하려면 다음과 같이 하십시오.

- 마이그레이션을 위한 모든 프로세스를 수행하는 복제 인스턴스를 생성합니다. 자세한 내용은 복제 인스턴스 생성 섹션을 참조하세요.
- 소스 엔드포인트를 지정합니다. 자세한 내용은 소스 및 대상 엔드포인트 생성 섹션을 참조하세요.
- 클러스터의 DNS 이름과 포트 번호를 찾습니다.
- SSL 연결을 확인하는 데 사용할 수 있는 인증서 번들을 다운로드합니다.
- 아래 설명에 따라 대상 엔드포인트를 지정합니다.
- 하나의 작업이나 작업 집합을 생성하여 사용할 테이블과 복제 프로세스를 정의합니다. 자세한 내용은 작업 생성 섹션을 참조하세요.
- 소스 데이터베이스의 데이터를 대상 클러스터로 마이그레이션합니다.

데이터베이스 마이그레이션은 다음의 두 가지 방법으로 시작할 수 있습니다.

1. AWS DMS 콘솔을 선택하고 이 콘솔에서 각 단계를 수행하면 됩니다.
2. AWS Command Line Interface(AWS CLI)을 사용해도 됩니다. CLI를 AWS DMS와 함께 사용하는 방법에 관한 자세한 내용은 AWS CLI for AWS DMS 단원을 참조하세요.

클러스터의 DNS 이름과 포트 번호를 찾으려면

- 다음 AWS CLI 명령을 사용하여 복제 그룹의 이름과 함께 `replication-group-id`를 입력합니다.

```
aws elasticache describe-replication-groups --replication-group-id myreplgroup
```

여기서 출력에는 `Address` 속성에 있는 DNS 이름과 클러스터의 기본 노드 `Port` 속성에 있는 포트 번호가 표시됩니다.

```
 ...
"ReadEndpoint": {
"Port": 6379,
"Address": "myreplgroup-
111.1abc1d.1111.uuu1.cache.example.com"
}
...
```

MemoryDB for Redis를 대상으로 사용하는 경우 다음 AWS CLI 명령을 사용하여 Redis 클러스터에 엔드포인트 주소를 제공하십시오.

```
aws memorydb describe-clusters --clusterid clusterid
```

SSL 연결을 확인하는 데 사용할 인증서 번들을 다운로드합니다.

- 명령줄에 다음 `wget` 명령을 입력합니다. Wget은 인터넷에서 파일을 다운로드하는 데 사용되는 무료 GNU 명령줄 유틸리티 도구입니다.

```
wget https://s3.aws-api-domain/rds-downloads/rds-combined-ca-bundle.pem
```

여기서 *aws-api-domain*은 지정된 S3 버킷과 해당 버킷이 제공하는 rds-combined-ca-bundle.pem 파일에 액세스하는 데 필요한 AWS 리전의 Amazon S3 도메인을 완성합니다.

AWS DMS 콘솔을 사용하여 대상 엔드포인트를 생성하려면

이 엔드포인트는 이미 실행 중인 Redis 대상에 사용됩니다.

- 콘솔의 탐색 창에서 엔드포인트를 선택하고 엔드포인트 생성을 선택합니다. 다음 표는 설정에 대한 설명입니다.

옵션	조치
[Endpoint type]	대상 엔드포인트 유형을 선택합니다.
[Endpoint identifier]	엔드포인트 이름을 입력합니다. 예를 들어, 이름에 엔드포인트 유형(예: **`my-redis-target`**)을 포함하십시오.
[Target Engine]	이 엔드포인트에서 연결하려는 데이터베이스 엔진 유형으로 Redis를 선택합니다.
클러스터 이름	Redis 클러스터의 DNS 이름을 입력합니다.
포트	포트에 Redis 클러스터의 포트 번호를 입력합니다.
SSL 보안 프로토콜	일반 텍스트 또는 SSL 암호화를 선택합니다. 일반 텍스트 - 이 옵션은 엔드포인트와 데이터베이스 간의 트래픽에 대해 전송 계층 보안(TLS) 암호화를 제공하지 않습니다. SSL 암호화 - 이 옵션을 선택하는 경우, SSL 인증 기관(CA) 인증서 ARN을 입력하여 서버의 인증서를 확인하고 암호화된 연결을 설정합니다. 온프레미스 Redis의 경우, DMS는 퍼블릭 및 프라이빗 CA 인증 기관(CA)을 모두 지원합니다. ElastiCache for Redis의 경우, DMS는 퍼블릭 CA만 지원합니다.
인증 유형	Redis 대상에 연결할 때 수행할 인증 유형을 선택합니다. 옵션에는 없음, 인증 역할 및 인증 토큰이 포함됩니다.

옵션	조치
	인증 역할을 선택하는 경우, 인증 사용자 이름과 인증 암호를 제공하십시오. 인증 토큰을 선택하는 경우, 인증 암호만 제공하십시오.
복제 인스턴스	[선택 사항] 연결을 테스트하려는 경우에만 이전에 복제 인스턴스 생성 페이지에 입력한 복제 인스턴스의 이름을 선택합니다.

엔드포인트에 대한 모든 정보를 모두 제공했다면 AWS DMS는 데이터베이스 마이그레이션 중에 사용할 Redis 대상 엔드포인트를 생성합니다.

마이그레이션 작업 생성 및 데이터베이스 마이그레이션 시작에 관한 자세한 내용은 작업 생성을 참조하십시오.

대상으로서 Redis에 대한 엔드포인트 설정 지정

대상 엔드포인트를 만들거나 수정하려면 콘솔이나 `CreateEndpoint` 또는 `ModifyEndpoint` API 작업을 사용할 수 있습니다.

AWS DMS 콘솔 내 Redis 대상의 경우, 엔드포인트 생성 또는 엔드포인트 수정 콘솔 페이지에서 엔드포인트별 설정을 지정합니다.

`CreateEndpoint` 및 `ModifyEndpoint` API 작업을 사용할 때는 `RedisSettings` 옵션에 대한 요청 파라미터를 지정하십시오. 다음 예에서는 AWS CLI를 사용하여 이 작업을 수행하는 방법을 보여줍니다.

```
aws dms create-endpoint --endpoint-identifier my-redis-target
--endpoint-type target --engine-name redis --redis-settings
'{"ServerName":"sample-test-
sample.zz012zz.cluster.eee1.cache.bbbxxx.com","Port":6379,"AuthType":"auth-token",
 "SslSecurityProtocol":"ssl-encryption", "AuthPassword":"notanactualpassword"}'

{
    "Endpoint": {
        "EndpointIdentifier": "my-redis-target",
        "EndpointType": "TARGET",
```

```
        "EngineName": "redis",
        "EngineDisplayName": "Redis",
        "TransferFiles": false,
        "ReceiveTransferredFiles": false,
        "Status": "active",
        "KmsKeyId": "arn:aws:kms:us-east-1:999999999999:key/x-b188188x",
        "EndpointArn": "arn:aws:dms:us-
east-1:555555555555:endpoint:ABCDEFGHIJKLMONOPQRSTUVWXYZ",
        "SslMode": "none",
        "RedisSettings": {
            "ServerName": "sample-test-sample.zz012zz.cluster.eee1.cache.bbbxxx.com",
            "Port": 6379,
            "SslSecurityProtocol": "ssl-encryption",
            "AuthType": "auth-token"
        }
    }
}
```

`--redis-settings` 파라미터는 다음과 같습니다.

- `ServerName` – (필수) `string` 유형에서 데이터가 마이그레이션되고 동일한 VPC에 있는 Redis 클러스터를 지정합니다.
- `Port` – (필수) `number` 유형에서 엔드포인트에 액세스하는 데 사용되는 포트 값입니다.
- `SslSecurityProtocol` - (선택) 유효값에는 `plaintext` 및 `ssl-encryption`이 포함됩니다. 기본값은 `ssl-encryption`입니다.

  `plaintext` 옵션은 엔드포인트와 데이터베이스 간의 트래픽에 대해 전송 계층 보안(TLS) 암호화를 제공하지 않습니다.

  암호화된 연결을 만드는 데 `ssl-encryption`을 사용합니다. `ssl-encryption`은 서버의 인증서를 확인하는 데 SSL 인증 기관(CA) ARN이 필요하지 않지만 `SslCaCertificateArn` 설정을 사용하여 선택적으로 식별할 수 있습니다. 인증 기관(CA) ARN이 없다면 DMS는 Amazon 루트 CA를 사용합니다.

  온프레미스 Redis 대상을 사용하는 경우, 퍼블릭 또는 프라이빗 인증 기관(CA)을 DMS로 가져오고 서버 인증 시 해당 ARN을 제공하는 데 `SslCaCertificateArn`을 사용할 수 있습니다. ElastiCache for Redis를 대상으로 사용할 때는 프라이빗 CA가 지원되지 않습니다.
- `AuthType` – (필수) Redis에 연결할 때 수행할 인증 유형을 나타냅니다. 유효값에는 `none`, `auth-token` 및 `auth-role`이 있습니다.

`auth-token` 옵션에는 '*AuthPassword*'를 입력해야 하는 반면, `auth-role` 옵션에는 '*AuthUserName*'과 '*AuthPassword*'를 입력해야 합니다.

Babelfish를 AWS Database Migration Service 대상으로 사용

AWS Database Migration Service를 사용하여 Microsoft SQL Server 소스 데이터베이스의 데이터를 Babelfish 대상으로 마이그레이션할 수 있습니다.

Babelfish for Aurora PostgreSQL은 Microsoft SQL Server 클라이언트에서 데이터베이스 연결을 수락할 수 있는 기능을 사용하여 Amazon Aurora PostgreSQL 호환 버전 데이터베이스를 확장합니다. 이렇게 하면 원래 SQL Server용으로 구축된 애플리케이션이 기존 마이그레이션에 비해 거의 코드 변경 없이, 또 데이터베이스 드라이버 변경 없이 Aurora PostgreSQL에서 직접 작동할 수 있습니다.

AWS DMS가 대상으로 지원하는 Babelfish 버전에 대한 자세한 내용은 대상: AWS DMS 섹션을 참조하세요. Aurora PostgreSQL의 이전 Babelfish 버전은 Babelfish 엔드포인트를 사용하기 전에 업그레이드가 필요합니다.

> **Note**
>
> Aurora PostgreSQL 대상 엔드포인트는 데이터를 Babelfish로 마이그레이션하는 데 선호되는 방법입니다. 자세한 내용은 대상으로 Babelfish for Aurora PostgreSQL 사용 섹션을 참조하세요.

Babelfish를 데이터베이스 엔드포인트로 사용하는 방법에 대한 자세한 내용은 Aurora용 Amazon Aurora 사용 설명서에서 Babelfish for Aurora PostgreSQL을 참조하세요.

Babelfish를 AWS DMS 대상으로 사용하기 위한 사전 요구 사항

데이터를 마이그레이션하기 전에 테이블을 생성하여 AWS DMS가 올바른 데이터 유형과 테이블 메타데이터를 사용하도록 해야 합니다. 마이그레이션을 실행하기 전에 대상에 테이블을 생성하지 않으면 AWS DMS는 잘못된 데이터 유형 및 권한으로 테이블을 생성할 수 있습니다. 예를 들어 AWS DMS는 타임스탬프 열을 바이너리(8)로 대신 생성하고, 예상 타임스탬프/행 버전 기능을 제공하지 않습니다.

마이그레이션 전에 테이블을 준비하고 생성하려면

1. 고유한 제약 조건, 프라이머리 키 또는 기본 제약 조건을 포함하는 create table DDL 문을 실행합니다.

뷰, 저장 프로시저, 함수 또는 트리거 같은 객체에 대한 DDL 문이나 외래 키 제약 조건을 포함하지 마세요. 이들은 소스 데이터베이스를 마이그레이션한 후에 적용할 수 있습니다.

2. 테이블의 ID 열, 계산된 열 또는 행 버전이나 타임스탬프 데이터 형식이 포함된 열을 식별합니다. 그런 다음 마이그레이션 작업을 실행할 때 알려진 문제를 처리하는 데 필요한 변환 규칙을 생성합니다. 자세한 내용은 변환 규칙 및 작업 단원을 참조하십시오.
3. Babelfish가 지원하지 않는 데이터 형식의 열을 식별합니다. 그런 다음 지원되는 데이터 형식을 사용하도록 대상 테이블의 영향을 받는 열을 변경하거나, 마이그레이션 작업 중에 해당 열을 제거하는 변환 규칙을 생성합니다. 자세한 내용은 변환 규칙 및 작업 단원을 참조하십시오.

   다음 표에는 Babelfish에서 지원하지 않는 소스 데이터 형식과 사용할 해당 권장 대상 데이터 형식이 나열되어 있습니다.

소스 데이터 유형	권장 Babelfish 데이터 형식
HEIRARCHYID	NVARCHAR(250)
GEOMETRY	VARCHAR(MAX)
GEOGRAPHY	VARCHAR(MAX)

Aurora PostgreSQL Serverless V2 소스 데이터베이스의 Aurora 용량 단위(ACU) 수준을 설정하려면

최소 ACU 값을 설정하여 AWS DMS 마이그레이션 작업을 실행하기 전에 성능을 개선할 수 있습니다.

- Serverless v2 용량 설정 창에서 최소 ACU를 **2** 또는 Aurora DB 클러스터에 적합한 수준으로 설정합니다.

  Aurora 용량 단위 설정에 대한 자세한 내용은 Amazon Aurora 사용 설명서의 Aurora 클러스터의 Aurora Serverless v2 용량 범위 선택을 참조하세요.

AWS DMS 마이그레이션 작업을 실행한 후 Aurora PostgreSQL Serverless V2 소스 데이터베이스에 적합한 수준으로 ACU의 최소값을 재설정할 수 있습니다.

Babelfish를 AWS Database Migration Service 대상으로 사용 시 보안 요구 사항

다음은 Babelfish를 대상으로 하여 AWS DMS를 사용할 경우의 보안 요구 사항에 대한 설명입니다.

- 데이터베이스를 생성하는 데 사용된 관리자 사용자 이름(관리자 사용자).
- 충분한 SELECT, INSERT, UPDATE, REFERENCES, REFERENCES 권한을 가진 PSQL 로그인 및 사용자.

Babelfish를 AWS DMS 대상으로 사용하기 위한 사용자 권한

> **Important**
>
> 보안을 위해 데이터 마이그레이션에 사용되는 사용자 계정은 대상으로 사용하는 모든 Babelfish 데이터베이스에서 등록된 사용자여야 합니다.

Babelfish 대상 엔드포인트에는 AWS DMS 마이그레이션을 실행하기 위한 최소 사용자 권한이 필요합니다.

로그인 및 권한이 낮은 Transact-SQL(T-SQL) 사용자를 생성하려면

1. 서버에 연결할 때 사용할 로그인과 암호를 생성합니다.

```
CREATE LOGIN dms_user WITH PASSWORD = 'password';
GO
```

2. Babelfish 클러스터의 가상 데이터베이스를 생성합니다.

```
CREATE DATABASE my_database;
GO
```

3. 대상 데이터베이스의 T-SQL 사용자를 생성합니다.

```
USE my_database
GO
CREATE USER dms_user FOR LOGIN dms_user;
GO
```

4. Babelfish 데이터베이스의 각 테이블에 권한을 부여합니다.

```
GRANT SELECT, DELETE, INSERT, REFERENCES, UPDATE ON [dbo].[Categories] TO dms_user;
```

Babelfish를 AWS Database Migration Service 대상으로 사용 시 제한 사항

다음 제한 사항은 Babelfish 데이터베이스를 AWS DMS 대상으로 사용할 때 적용됩니다.

- 테이블 준비 모드 “Do Nothing“만 지원됩니다.
- ROWVERSION 데이터 형식에는 마이그레이션 작업 중에 테이블에서 열 이름을 제거하는 테이블 매핑 규칙이 필요합니다.
- sql_variant 데이터 형식은 지원되지 않습니다.
- 전체 LOB 모드가 지원됩니다. SQL Server를 소스 엔드포인트로 사용하려면 SQL Server 엔드포인트 연결 속성 설정 `ForceFullLob=True`을 설정해야 LOB를 대상 엔드포인트로 마이그레이션할 수 있습니다.
- 복제 작업 설정에는 다음과 같은 제한 사항이 있습니다.

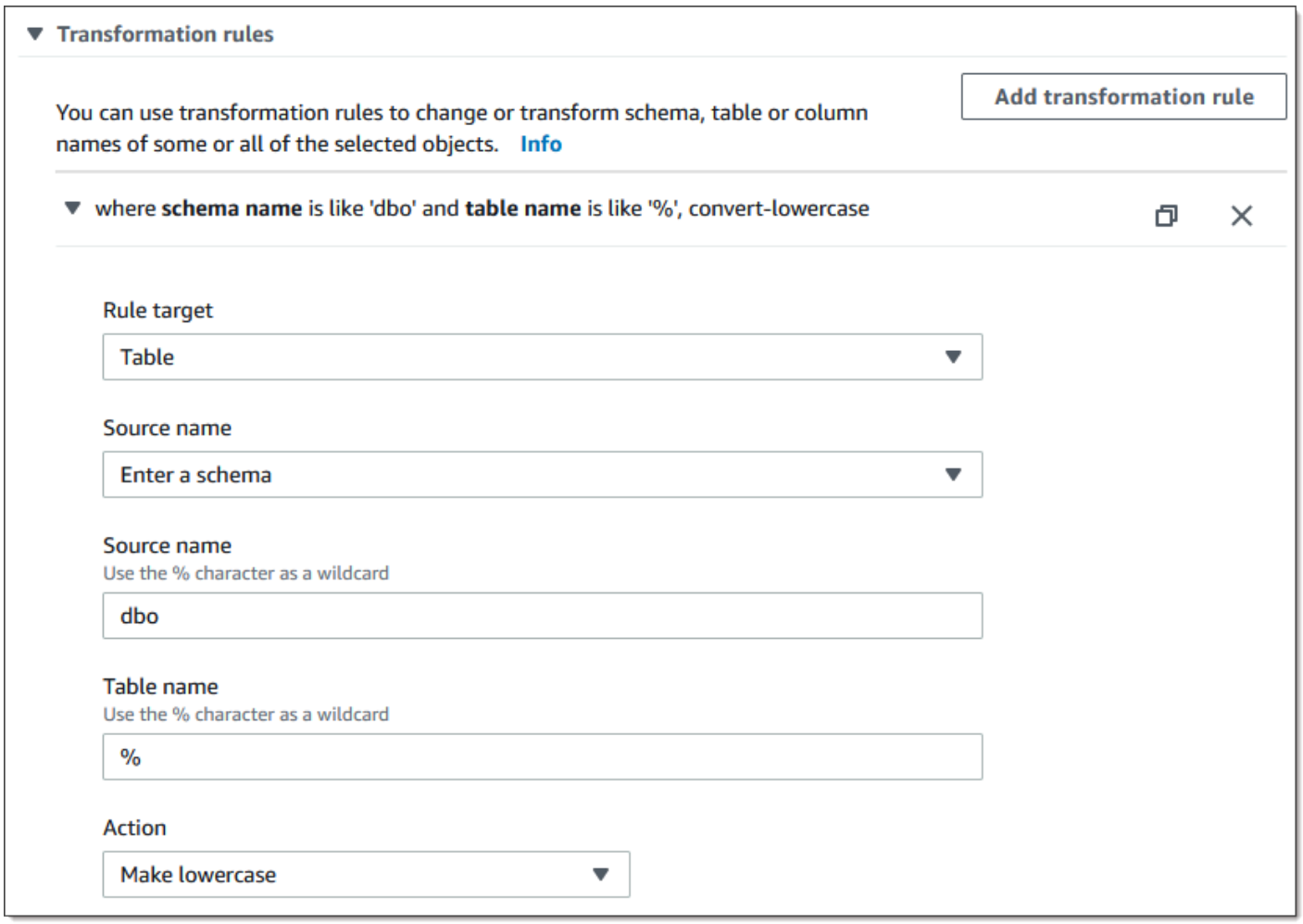
```
{
   "FullLoadSettings": {
      "TargetTablePrepMode": "DO_NOTHING",
      "CreatePkAfterFullLoad": false,
      }.

}
```

- Babelfish의 TIME(7), DATETIME2(7), DATETIMEOFFSET(7) 데이터 형식은 시간의 초 부분 정밀도를 6자리로 제한합니다. 이러한 데이터 형식을 사용할 때는 대상 테이블에 정밀도 값 6을 사용하는 것이 좋습니다. Babelfish 버전 2.2.0 이상에서 TIME(7) 및 DATETIME2(7)를 사용할 경우, 정밀도의 일곱 번째 자리는 항상 0입니다.
- DO_NOTHING 모드에서 DMS는 테이블이 이미 존재하는지 확인합니다. 대상 스키마에 테이블이 없으면 DMS는 소스 테이블 정의를 기반으로 테이블을 생성하고, 사용자 정의 데이터 형식을 기본 데이터 형식에 매핑합니다.
- Babelfish 대상으로의 AWS DMS 마이그레이션 작업은 ROWVERSION 또는 TIMESTAMP 데이터 형식을 사용하는 열이 있는 테이블을 지원하지 않습니다. 전송 프로세스 중에 테이블에서 열 이름을 제거하는 테이블 매핑 규칙을 사용할 수 있습니다. 다음 변환 규칙 예제에서 소스에 있는 `Actor`라는 테이블이 변환되어 대상의 `Actor` 테이블에서 `col` 문자로 시작하는 모든 열을 제거합니다.

```
{
  "rules": [{
  "rule-type": "selection",is
  "rule-id": "1",
```

```
 "rule-name": "1",
 "object-locator": {
  "schema-name": "test",
  "table-name": "%"
 },
 "rule-action": "include"
}, {
 "rule-type": "transformation",
 "rule-id": "2",
 "rule-name": "2",
 "rule-action": "remove-column",
 "rule-target": "column",
 "object-locator": {
  "schema-name": "test",
  "table-name": "Actor",
  "column-name": "col%"
 }
}]
}
```

- 대상 테이블이 Categories처럼 대소문자가 혼용된 이름을 사용하는 ID 열 또는 계산된 열이 있는 테이블의 경우, DMS 작업을 위해 테이블 이름을 소문자로 변환하는 변환 규칙 작업을 생성해야 합니다. 다음 예제는 AWS DMS 콘솔을 사용하여 Make lowercase 변환 규칙 작업을 생성하는 방법을 보여 줍니다. 자세한 내용은 변환 규칙 및 작업 섹션을 참조하세요.



- Babelfish 버전 2.2.0 이전에 DMS는 Babelfish 대상 엔드포인트에 복제할 수 있는 열 수를 20개로 제한했습니다. Babelfish 2.2.0에서는 이 한도가 100개로 늘어났습니다. 하지만 Babelfish 버전 2.4.0 이상에서는 복제할 수 있는 열 수가 다시 늘어납니다. SQL Server 데이터베이스에 대해 다음 코드 샘플을 실행하여 어떤 테이블이 너무 긴지 확인할 수 있습니다.

```
USE myDB;
GO
DECLARE @Babelfish_version_string_limit INT = 8000; -- Use 380 for Babelfish versions
 before 2.2.0
WITH bfendpoint
AS (
SELECT
 [TABLE_SCHEMA]
      ,[TABLE_NAME]
   , COUNT( [COLUMN_NAME] ) AS NumberColumns
   , ( SUM( LEN( [COLUMN_NAME] ) + 3)
  + SUM( LEN( FORMAT(ORDINAL_POSITION, 'N0') ) + 3 )
```

```
     + LEN( TABLE_SCHEMA ) + 3
  + 12 -- INSERT INTO string
  + 12)  AS InsertIntoCommandLength -- values string
      , CASE WHEN ( SUM( LEN( [COLUMN_NAME] ) + 3)
  + SUM( LEN( FORMAT(ORDINAL_POSITION, 'N0') ) + 3 )
     + LEN( TABLE_SCHEMA ) + 3
  + 12 -- INSERT INTO string
  + 12)  -- values string
   >= @Babelfish_version_string_limit
   THEN 1
   ELSE 0
  END AS IsTooLong
FROM [INFORMATION_SCHEMA].[COLUMNS]
GROUP BY [TABLE_SCHEMA], [TABLE_NAME]
)
SELECT *
FROM bfendpoint
WHERE IsTooLong = 1
ORDER BY TABLE_SCHEMA, InsertIntoCommandLength DESC, TABLE_NAME
;
```

Babelfish의 대상 데이터 형식

다음 테이블에는 AWS DMS를 사용할 때 지원되는 Babelfish 대상 데이터 형식과 AWS DMS 데이터 형식에서의 기본 매핑이 나와 있습니다.

AWS DMS 데이터 형식에 대한 추가 정보는 AWS Database Migration Service에서 사용되는 데이터 형식 단원을 참조하십시오.

AWS DMS 데이터 형식	Babelfish 데이터 형식
BOOLEAN	TINYINT
BYTES	VARBINARY(길이)
DATE	DATE
TIME	TIME
INT1	SMALLINT

AWS DMS 데이터 형식	Babelfish 데이터 형식
INT2	SMALLINT
INT4	INT
INT8	BIGINT
NUMERIC	NUMERIC(p,s)
REAL4	REAL
REAL8	FLOAT
STRING	열이 날짜 또는 시간 열인 경우, 다음 작업을 수행합니다. • SQL Server 2008 이상에서는 DATETIME2를 사용합니다. • 이전 버전의 경우에 크기가 3이면 DATETIME을 사용합니다. 다른 모든 경우에서는 VARCHAR (37)을 사용합니다. 열이 날짜 또는 시간 열이 아닌 경우, VARCHAR(길이)을 사용합니다.
UINT1	TINYINT
UINT2	SMALLINT
UINT4	INT
UINT8	BIGINT
WSTRING	NVARCHAR(length)

AWS DMS 데이터 형식	Babelfish 데이터 형식
BLOB	VARBINARY(max) DMS에 이 데이터 형식을 사용하려면 특정 작업에서 BLOB 사용을 활성화해야 합니다. DMS는 프라이머리 키를 포함하는 테이블에서만 BLOB 데이터 형식을 지원합니다.
CLOB	VARCHAR(최대) DMS에 이 데이터 형식을 사용하려면 특정 작업에서 CLOB 사용을 활성화해야 합니다.
NCLOB	NVARCHAR(최대) DMS에 이 데이터 형식을 사용하려면 특정 작업에서 NCLOB 사용을 활성화해야 합니다. CDC 중에 DMS는 프라이머리 키를 포함하는 테이블에서만 NCLOB 데이터 형식을 지원합니다.

Amazon Timestream을 AWS Database Migration Service의 대상으로 사용

전체 로드 및 CDC 데이터 마이그레이션을 지원하는 AWS Database Migration Service를 사용하여 소스 데이터베이스의 데이터를 Amazon Timestream 대상 엔드포인트로 마이그레이션 수 있습니다.

Amazon Timestream은 대량 데이터 모으기를 위해 설계된 빠르고 확장 가능한 서버리스 시계열 데이터베이스 서비스입니다. 시계열 데이터는 일정 기간 동안 수집되는 일련의 데이터 포인트로, 시간이 지남에 따라 변화하는 이벤트를 측정하는 데 사용됩니다. IoT 애플리케이션, 애플리케이션 및 분석 애플리케이션에서 메트릭을 수집, DevOps 저장 및 분석하는 데 사용됩니다. Timestream에 데이터를 저장하면 데이터의 추세와 패턴을 거의 실시간으로 시각화하고 식별할 수 있습니다. Amazon Timestream에 대한 자세한 내용은 Amazon Timestream 개발자 가이드에서 Amazon Timestream이란 무엇인가요?를 참조하세요.

주제

- Amazon Timestream을 AWS Database Migration Service의 대상으로 사용하기 위한 사전 조건
- 멀티스레드 전체 로드 작업 설정
- 멀티스레드 CDC 로드 작업 설정

- Timestream을 AWS DMS의 대상으로 사용할 경우 엔드포인트 설정
- Amazon Timestream 대상 엔드포인트 생성 및 수정
- 객체 매핑을 사용하여 데이터를 Timestream 주제로 마이그레이션
- AWS Database Migration Service의 대상으로 Amazon Timestream 사용 시 제한 사항

Amazon Timestream을 AWS Database Migration Service의 대상으로 사용하기 위한 사전 조건

Amazon Timestream을 AWS DMS의 대상으로 설정하기 전에 IAM 역할을 생성합니다. 이 역할은 AWS DMS가 Amazon Timestream으로 마이그레이션되는 데이터에 액세스할 수 있도록 허용해야 합니다. Timestream으로 마이그레이션하는 데 사용하는 역할에 대한 최소 액세스 권한 세트가 다음 IAM 정책에 나와 있습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowDescribeEndpoints",
            "Effect": "Allow",
            "Action": [
                "timestream:DescribeEndpoints"
            ],
            "Resource": "*"
        },
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "timestream:ListTables",
                "timestream:DescribeDatabase"
            ],
            "Resource": "arn:aws:timestream:region:account_id:database/DATABASE_NAME"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "timestream:DeleteTable",
                "timestream:WriteRecords",
                "timestream:UpdateTable",
```

```
                "timestream:CreateTable"
            ],
            "Resource": "arn:aws:timestream:region:account_id:database/DATABASE_NAME/
table/TABLE_NAME"
        }
    ]
}
```

모든 테이블을 마이그레이션하려는 경우 위 예시에서 *TABLE_NAME*에 *를 사용합니다.

Timestream을 대상으로 사용할 때는 다음 지침을 참고하세요.

- 타임스탬프가 1년 전을 초과하는 기록 데이터를 수집하려는 경우 AWS DMS를 사용하여 Amazon S3에 쉼표로 구분된 값(csv) 형식으로 데이터를 쓰는 것이 좋습니다. 그런 다음 Timestream의 배치 로드를 사용하여 데이터를 Timestream으로 수집합니다. 자세한 내용은 Amazon Timestream 개발자 안내서의 Using batch load in Timestream을 참조하세요.
- 1년 전 미만 데이터의 전체 로드 데이터 마이그레이션인 경우 Timestream 테이블의 메모리 스토어 보존 기간을 가장 오래된 타임스탬프보다 크거나 같게 설정하는 것이 좋습니다. 그런 다음 마이그레이션이 완료된 후 테이블의 메모리 스토어 보존 기간을 원하는 값으로 편집합니다. 예를 들어 가장 오래된 타임스탬프가 2개월 전인 데이터를 마이그레이션하려면 다음과 같이 합니다.
  - Timestream 대상 테이블의 메모리 스토어 보존 기간을 2개월로 설정합니다.
  - AWS DMS를 사용하여 데이터 마이그레이션을 시작합니다.
  - 데이터 마이그레이션이 완료되면 대상 Timestream 테이블의 보존 기간을 원하는 값으로 변경합니다.

  마이그레이션 전에 다음 페이지의 정보를 사용하여 메모리 스토어 비용을 추정하는 것이 좋습니다.
  - Amazon Timestream 요금
  - AWS 요금 계산기
- CDC 데이터 마이그레이션인 경우 수집된 데이터가 메모리 스토어 보존 한도 내에 포함되도록 대상 테이블의 메모리 스토어 보존 기간을 설정하는 것이 좋습니다. 자세한 내용은 Amazon Timestream 개발자 안내서의 Writes Best Practices를 참조하세요.

멀티스레드 전체 로드 작업 설정

데이터 전송 속도를 높이기 위해 AWS DMS는 다음과 같은 작업 설정을 사용하여 Timestream 대상 엔드포인트로의 멀티스레드 전체 로드 마이그레이션 작업을 지원합니다.

- `MaxFullLoadSubTasks` – 병렬로 로드할 최대 소스 테이블 수를 표시하려면 이 옵션을 사용합니다. DMS는 전용 하위 작업을 사용하여 각 테이블을 해당 Amazon Timestream 대상 테이블에 로드합니다. 기본값은 8이며 최대값은 49입니다.
- `ParallelLoadThreads` – AWS DMS가 각 테이블을 Amazon Timestream 대상 테이블에 로드하는 데 사용하는 스레드 수를 지정하려면 이 옵션을 사용합니다. Timestream 대상의 최대값은 32입니다. 이 최대 한도를 증가시키도록 요청할 수 있습니다.
- `ParallelLoadBufferSize` – 이 옵션을 사용하여 병렬 로드 스레드에서 데이터를 Amazon Timestream 대상에 로드하기 위해 사용하는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 이 설정은 `ParallelLoadThreads`와 함께 사용하십시오. `ParallelLoadBufferSize`는 둘 이상의 스레드가 있는 경우에만 유효합니다.
- `ParallelLoadQueuesPerThread` – 각 동시 스레드가 액세스하는 대기열 수를 지정하여 대기열에서 데이터 레코드를 가져오고 대상에 대한 배치 로드를 생성하려면 이 옵션을 사용합니다. 기본 값은 1입니다. 그러나 다양한 페이로드 크기의 Amazon Timestream 대상에서 스레드당 대기열 수의 유효 범위는 5~512입니다.

멀티스레드 CDC 로드 작업 설정

CDC 성능을 승격하기 위해 AWS DMS에서는 다음 작업 설정을 지원합니다.

- `ParallelApplyThreads` – 데이터 레코드를 Timestream 대상 엔드포인트로 푸시하기 위해 CDC 로드 중에 AWS DMS가 사용하는 동시 스레드 수를 지정합니다. 기본값은 0이고 최대값은 32입니다.
- `ParallelApplyBufferSize` – CDC 로드 중에 Timestream 대상 엔드포인트로 푸시할 동시 스레드에 대한 각 버퍼 대기열에 저장할 최대 레코드 수를 지정합니다. 기본값은 100이고 최대값은 1,000입니다. `ParallelApplyThreads`가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.
- `ParallelApplyQueuesPerThread` – 각 스레드가 대기열에서 데이터 레코드를 가져오고 CDC 중에 Timestream 엔드포인트에 대한 배치 로드를 생성하기 위한 대기열 수를 지정합니다. 기본값은 1이고 최대값은 512입니다.

Timestream을 AWS DMS의 대상으로 사용할 경우 엔드포인트 설정

추가 연결 속성을 사용하는 것과 마찬가지로 엔드포인트 설정을 사용하여 Timestream 대상을 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI의 `create-endpoint` 명령을 `--timestream-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Timestream을 대상으로 하여 사용할 수 있는 엔드포인트 설정은 다음 테이블에 나와 있습니다.

명칭	설명
`MemoryDuration`	Timestream의 메모리 스토어에 마이그레이션된 데이터를 저장할 보존 한도를 지정하려면 이 속성을 설정합니다. 시간은 시간 단위로 측정됩니다. Timestream의 메모리 스토어는 높은 수집 처리량과 빠른 액세스에 최적화되어 있습니다. 기본값: 24(시간) 유효한 값: 1~8,736(시간 단위로 측정된 1시간~12개월) 예제: `--timestream-settings '{"MemoryDuration": 20}'`
`DatabaseName`	대상 Timestream 데이터베이스 이름을 지정하려면 이 속성을 설정합니다. 유형: 문자열 예제: `--timestream-settings '{"DatabaseName": " db_name"}'`
`TableName`	대상 Timestream 테이블 이름을 지정하려면 이 속성을 설정합니다. 유형: 문자열 예제: `--timestream-settings '{"TableName": " table_name "}'`
`MagneticDuration`	Timestream 테이블에 적용되는 마그네틱 스토어 저장 기간을 일 단위로 지정하려면 이 속성을 설정합니다. 수집된 데이터의 보존 한도입니다. Timestream은 보존 한도를 초과하는 모든 타임스탬프를 삭제합니다. 자세한 내용은 Amazon Timestream 개발자 안내서의 Storage를 참조하세요.

명칭	설명
	예제: `--timestream-settings '{"MagneticDuration": "3"}'`
`CdcInsertsAndUpdates`	AWS DMS가 삽입과 업데이트만 적용하고 삭제는 적용하지 않도록 지정하려면 이 속성을 `true`로 설정합니다. Timestream은 레코드 삭제를 허용하지 않으므로 이 값이 `false`이면 AWS DMS가 Timestream 데이터베이스에서 해당 레코드를 삭제하지 않고 무효화합니다. 자세한 내용은 제한 사항 단원을 참조하십시오. 기본 값: `false` 예제: `--timestream-settings '{"CdcInsertsAndUpdates": "true"}'`
`EnableMagneticStoreWrites`	마그네틱 스토어 쓰기를 활성화하려면 이 속성을 `true`로 설정합니다. 이 값이 `false`인 경우, Timestream은 기본적으로 마그네틱 스토어 쓰기를 허용하지 않으므로 AWS DMS는 대상 테이블의 메모리 스토어 보존 기간보다 오래된 타임스탬프가 있는 레코드를 기록하지 않습니다. 자세한 내용은 Amazon Timestream 개발자 안내서의 Writes Best Practices를 참조하세요. 기본 값: `false` 예제: `--timestream-settings '{"EnableMagneticStoreWrites": "true"}'`

Amazon Timestream 대상 엔드포인트 생성 및 수정

IAM 역할을 생성하고 최소 액세스 권한 세트를 설정한 후에는 AWS DMS 콘솔을 사용하거나 AWS CLI의 `create-endpoint` 명령을 `--timestream-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 Amazon Timestream 대상 엔드포인트를 생성할 수 있습니다.

다음 예시에서는 AWS CLI를 사용하여 Timestream 대상 엔드포인트를 생성하고 수정하는 방법을 보여줍니다.

Timestream 대상 엔드포인트 생성 명령

```
aws dms create-endpoint –endpoint-identifier timestream-target-demo
--endpoint-type target –engine-name timestream
--service-access-role-arn arn:aws:iam::123456789012:role/my-role
--timestream-settings
{
    "MemoryDuration": 20,
    "DatabaseName":"db_name",
    "MagneticDuration": 3,
    "CdcInsertsAndUpdates": true,
    "EnableMagneticStoreWrites": true,
}
```

Timestream 대상 엔드포인트 수정 명령

```
aws dms modify-endpoint –endpoint-identifier timestream-target-demo
--endpoint-type target –engine-name timestream
--service-access-role-arn arn:aws:iam::123456789012:role/my-role
--timestream-settings
{
    "MemoryDuration": 20,
    "MagneticDuration": 3,
}
```

객체 매핑을 사용하여 데이터를 Timestream 주제로 마이그레이션

AWS DMS는 테이블 매핑 규칙을 사용하여 데이터를 소스에서 대상 Timestream 주제에 매핑합니다. 데이터를 대상 주제에 매핑하려면 객체 매핑이라는 테이블 매핑 규칙 유형을 사용합니다. 객체 매핑을 사용하여 원본의 데이터 레코드를 Timestream 주제에 게시된 데이터 레코드로 매핑하는 방법을 지정합니다.

Timestream 주제에는 파티션 키 외에 별다른 사전 설정 구조가 없습니다.

> **Note**
>
> 객체 매핑을 사용할 필요는 없습니다. 일반 테이블 매핑을 다양한 변환에 사용할 수 있습니다. 하지만 파티션 키 유형은 다음과 같은 기본 동작을 따릅니다.
>
> - 기본 키는 전체 로드의 파티션 키로 사용됩니다.

- parallel-apply 작업 설정을 사용하지 않는 경우, CDC의 파티션 키로 `schema.table`이 사용됩니다.
- paralle-apply 작업 설정을 사용하지 않는 경우, CDC의 파티션 키로 기본 키가 사용됩니다.

object-mapping 규칙을 만들려면 `rule-type`을 `object-mapping`으로 지정합니다. 이 규칙은 사용할 객체 매핑의 유형을 지정합니다. 이 규칙의 구조는 다음과 같습니다.

```
{
    "rules": [
        {
            "rule-type": "object-mapping",
            "rule-id": "id",
            "rule-name": "name",
            "rule-action": "valid object-mapping rule action",
            "object-locator": {
                "schema-name": "case-sensitive schema name",
                "table-name": ""
            }
        }
    ]
}
```

```
{
    "rules": [
        {
            "rule-type": "object-mapping",
            "rule-id": "1",
            "rule-name": "timestream-map",
            "rule-action": "map-record-to-record",
            "target-table-name": "tablename",
            "object-locator": {
                "schema-name": "",
                "table-name": ""
            },
            "mapping-parameters": {
                "timestream-dimensions": [
                    "column_name1",
                     "column_name2"
                ],
                "timestream-timestamp-name": "time_column_name",
```

```
                "timestream-multi-measure-name": "column_name1or2",
                "timestream-hash-measure-name":  true or false,
                "timestream-memory-duration": x,
                "timestream-magnetic-duration": y
            }
        }
    ]
}
```

AWS DMS는 현재 `map-record-to-record`와 `map-record-to-document`를 `rule-action` 파라미터의 유일한 유효값으로 지원합니다. `map-record-to-record` 및 `map-record-to-document` 값은 `exclude-columns` 속성 목록의 일부로 제외되지 않은 레코드에 대해 AWS DMS가 기본적으로 수행하는 작업을 지정합니다. 이러한 값은 어떤 식으로든 속성 매핑에 영향을 미치지 않습니다.

관계형 데이터베이스에서 Timestream 주제로 마이그레이션할 때 `map-record-to-record`를 사용합니다. 이러한 규칙 유형은 Timestream 주제의 파티션 키로 관계형 데이터베이스의 `taskResourceId.schemaName.tableName` 값을 사용하며 소스 데이터베이스의 각 열마다 속성을 하나씩 생성합니다. `map-record-to-record`를 사용할 때 `exclude-columns` 속성 목록에 소스 테이블의 열이 나열되지 않은 경우, AWS DMS는 대상 주제에 해당 속성을 생성합니다. 이러한 해당 속성은 원본 열이 속성 매핑에 사용되는지 여부와 상관없이 생성됩니다.

`map-record-to-record`를 이해하는 한 가지 방법은 작업 중일 때 관찰하는 것입니다. 이 예에서는 다음 구조와 데이터를 사용하여 관계형 데이터베이스 테이블 행에서 시작한다고 가정합니다.

FirstName	LastName	StoreId	HomeAddress	HomePhone	WorkAddress	WorkPhone	DateofBirth
Randy	Marsh	5	221B Baker Street	1234567890	31 Spooner Street, Quahog	9876543210	1988/02/29

이 정보를 `Test`라는 스키마에서 Timestream 주제로 마이그레이션하려면 데이터를 대상 주제로 매핑하는 규칙을 생성합니다. 다음 규칙은 그 매핑 과정을 보여 줍니다.

```
{
    "rules": [
        {
```

```
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "rule-action": "include",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            }
        },
        {
            "rule-type": "object-mapping",
            "rule-id": "2",
            "rule-name": "DefaultMapToTimestream",
            "rule-action": "map-record-to-record",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "Customers"
            }
        }
    ]
}
```

다음은 지정된 Timestream 주제와 파티션 키(이 경우 `taskResourceId.schemaName.tableName`)로 Timestream 대상 주제의 샘플 데이터를 사용하여 결과 레코드 형식을 보여줍니다.

```
  {
     "FirstName": "Randy",
     "LastName": "Marsh",
     "StoreId":  "5",
     "HomeAddress": "221B Baker Street",
     "HomePhone": "1234567890",
     "WorkAddress": "31 Spooner Street, Quahog",
     "WorkPhone": "9876543210",
     "DateOfBirth": "02/29/1988"
  }
```

AWS Database Migration Service의 대상으로 Amazon Timestream 사용 시 제한 사항

Amazon Timestream을 대상으로 사용할 때 적용되는 제한 사항은 다음과 같습니다.

- 차원 및 타임스탬프: Timestream은 소스 데이터의 차원 및 타임스탬프를 복합 기본 키처럼 사용하며 이러한 값을 업서트하는 것도 허용하지 않습니다. 즉, 소스 데이터베이스에서 레코드의 타임스탬프

또는 차원을 변경하면 Timestream 데이터베이스가 새 레코드를 생성하려고 합니다. 따라서 레코드의 차원 또는 타임스탬프를 다른 기존 레코드의 차원 또는 타임스탬프와 일치하도록 변경하는 경우 AWS DMS가 새 레코드를 생성하거나 이전의 해당 레코드를 업데이트하는 대신 다른 레코드의 값을 업데이트할 수 있습니다.

- DDL 명령: 현재 AWS DMS 릴리스에서는 `CREATE TABLE` 및 `DROP TABLE` DDL 명령만 지원합니다.
- 레코드 제한: Timestream에는 레코드 크기 및 측정값 크기와 같은 레코드에 대한 제한이 있습니다. 자세한 내용은 Amazon Timestream 개발자 안내서의 할당량을 참조하세요.
- 레코드 및 Null 값 삭제: Timestream은 레코드 삭제를 지원하지 않습니다. 소스에서 삭제된 레코드의 마이그레이션을 지원하기 위해 AWS DMS는 Timestream 대상 데이터베이스의 레코드에서 해당 필드를 지웁니다. AWS DMS는 해당 대상 레코드의 필드 값을 숫자 필드의 경우 0, 텍스트 필드의 경우 null, 부울 필드의 경우 false로 변경합니다.
- 대상으로 사용 시 Timestream은 관계형 데이터베이스(RDBMS)가 아닌 소스를 지원하지 않습니다.
- AWS DMS는 다음 리전에서만 Timestream을 대상으로 지원합니다.
  - 미국 동부(버지니아 북부)
  - 미국 동부(오하이오)
  - 미국 서부(오레곤)
  - 유럽(아일랜드)
  - 유럽(프랑크푸르트)
  - 아시아 태평양(시드니)
  - 아시아 태평양(도쿄)
- 대상으로 사용 시 Timestream은 `TargetTablePrepMode`를 `TRUNCATE_BEFORE_LOAD`로 설정하는 것을 지원하지 않습니다. 이 설정에는 `DROP_AND_CREATE`를 사용하는 것이 좋습니다.

Db2용 Amazon RDS와 IBM Db2 LUW를 AWS DMS의 대상으로 사용

AWS Database Migration Service(AWS DMS)를 사용하여 Db2 LUW 데이터베이스에서 Db2용 Amazon RDS 또는 온프레미스 Db2 데이터베이스로 데이터를 마이그레이션할 수 있습니다

AWS DMS가 대상으로 지원하는 Db2 LUW 버전에 관한 자세한 내용은 대상: AWS DMS 섹션을 참조하세요.

Secure Sockets Layer(SSL)를 사용하여 Db2 LUW 엔드포인트와 복제 인스턴스 간 연결을 암호화할 수 있습니다. Db2 LUW 엔드포인트에서 SSL을 사용하는 방법에 대한 자세한 내용은 AWS Database Migration Service에서 SSL 사용 섹션을 참조하십시오.

AWS DMS의 대상으로서 Db2 LUW 사용 시 제한 사항

Db2 LUW 데이터베이스를 AWS DMS의 대상으로 사용 시 다음과 같은 제한 사항이 적용됩니다. Db2 LUW를 소스로 사용 시 적용되는 제한 사항은 Db2 LUW를 원본으로 사용할 때의 제한 사항 AWS DMS 섹션을 참조하세요.

- AWS DMS는 소스가 Db2 LUW 또는 z/OS용 Db2인 경우에만 Db2 LUW를 대상으로 지원합니다.
- Db2 LUW를 대상으로 사용하는 경우 전체 LOB 모드에서의 복제가 지원되지 않습니다.
- Db2 LUW를 대상으로 사용하는 경우 전체 로드 단계에서 XML 데이터 유형이 지원되지 않습니다. 이는 IBM dbload 유틸리티의 제한 사항입니다. 자세한 내용은 IBM Informix 서버 설명서의 dbload 유틸리티를 참조하세요.
- AWS DMS는 큰따옴표 문자(")에 해당하는 값으로 BLOB 필드를 자릅니다. 이는 IBM dbload 유틸리티의 제한 사항입니다.

Db2 LUW를 AWS DMS 대상으로 사용 시 엔드포인트 설정

추가 연결 속성을 사용하는 것과 비슷하게 엔드포인트 설정을 사용하여 Db2 LUW 대상 데이터베이스를 구성할 수 있습니다. AWS DMS 콘솔을 사용하거나 AWS CLI의 `create-endpoint` 명령을 `--ibm-db2-settings '{"EndpointSetting": "value", ...}'` JSON 구문과 함께 사용하여 대상 엔드포인트를 생성할 때 설정을 지정합니다.

Db2 LUW를 대상으로 하여 사용할 수 있는 엔드포인트 설정은 다음 표에 나와 있습니다.

명칭	Description
`KeepCsvFiles`	true인 경우 AWS DMS는 데이터 복제에 사용된 모든 .csv 파일을 Db2 LUW 대상에 저장합니다. DMS는 이러한 파일을 분석 및 문제 해결에 사용합니다.
`LoadTimeout`	Db2 대상에서 DMS가 수행하는 작업이 AWS DMS 제한 시간을 초과하기까지 걸리는 시간(밀리초)입니다. 기본값은 1,200(20분)입니다.
`MaxFileSize`	데이터를 Db2 LUW로 전송할 때 사용되는 .csv 파일의 최대 크기(KB 단위)를 지정합니다.

명칭	Description
`WriteBufferSize`	DMS 복제 인스턴스의 로컬 디스크에 .csv 파일을 생성할 때 사용되는 메모리 내 파일 쓰기 버퍼의 크기(KB 단위)입니다. 기본값은 1,024(1MB)입니다.

VPC 엔드포인트를 AWS DMS 소스 및 대상 엔드포인트로 구성

AWS DMS는 Amazon Virtual Private Cloud(VPC) 엔드포인트를 소스 및 대상으로 지원합니다. AWS DMS는 모든 AWS 소스 및 대상 데이터베이스에 대해 명시적으로 정의된 경로가 AWS DMS VPC에 정의되어 있는 한, 이러한 소스 또는 대상 데이터베이스에 연결할 수 있습니다.

AWS DMS는 Amazon VPC 엔드포인트를 지원하므로, 추가 네트워킹 구성 및 설정 없이도 모든 복제 태스크에 대한 엔드 투 엔드 네트워크 보안을 더 쉽게 유지할 수 있습니다. 모든 소스 및 대상 엔드포인트에 VPC 엔드포인트를 사용하면 모든 트래픽을 VPC 내에서 유지하고 제어할 수 있습니다. AWS DMS 버전 3.4.7 이상으로 업그레이드하려면 AWS DMS를 구성하여 VPC 엔드포인트를 사용하도록 하거나, 다음과 같이 Amazon Web Services와 상호 작용하는 모든 소스 및 대상 엔드포인트에 대한 퍼블릭 경로를 사용하도록 해야 합니다.

- Amazon S3
- Amazon Kinesis
- AWS Secrets Manager
- Amazon DynamoDB
- Amazon Redshift
- Amazon OpenSearch Service

아래에 설명된 대로, 버전 3.4.7로 시작하는 AWS DMS를 지원하려면 VPC 엔드포인트가 필요할 수 있습니다.

AWS DMS 버전 3.4.7 이상으로 마이그레이션할 경우 영향을 받는 사용자

앞서 나열한 AWS DMS 엔드포인트 중 하나 이상을 사용 중이고 이러한 엔드포인트가 공개적으로 라우팅 불가능하거나 VPC 엔드포인트가 기존에 연결되어 있지 않은 경우, 사용자가 영향을 받습니다.

AWS DMS 버전 3.4.7 이상으로 마이그레이션할 경우 영향을 받지 않는 사용자

다음과 같은 경우에는 영향을 받지 않습니다.

- 앞서 나열한 AWS DMS 엔드포인트 중 하나 이상을 사용하고 있지 않은 경우.
- 앞서 나열한 엔드포인트 중 하나를 사용하고 있으며, 해당 엔드포인트가 공개적으로 라우팅 가능한 경우.
- 앞서 나열한 엔드포인트 중 하나를 사용하고 있으며, 해당 엔드포인트에 VPC 엔드포인트가 연결된 경우.

AWS DMS 버전 3.4.7 이상으로 마이그레이션 준비

앞서 설명한 엔드포인트를 사용할 때 AWS DMS 태스크 실패를 방지하려면 AWS DMS를 버전 3.4.7 이상으로 업그레이드하기 전에 다음 단계 중 하나를 수행하세요.

- 영향을 받는 AWS DMS 엔드포인트를 공개적으로 라우팅 가능하도록 설정합니다. 예를 들어, AWS DMS 복제 인스턴스에서 이미 사용하고 있는 VPC에 인터넷 게이트웨이(IGW) 경로를 추가하여 모든 소스 및 대상 엔드포인트를 공개적으로 라우팅 가능하도록 설정합니다.
- 아래에 설명된 대로, AWS DMS에서 사용하는 모든 소스 및 대상 엔드포인트에 액세스할 수 있는 VPC 엔드포인트를 생성합니다.

AWS DMS 소스 및 대상 엔드포인트에 사용하는 기존 VPC 엔드포인트의 경우, XML 정책 문서 `dms-vpc-role`을 준수하는 신뢰 정책을 사용해야 합니다. 이 XML 정책 문서에 대한 자세한 내용은 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성 섹션을 참조하세요.

또는 복제 인스턴스가 포함된 VPC에 VPC 엔드포인트를 추가하여 복제 인스턴스를 VPC 엔드포인트로 구성합니다. 퍼블릭 엔드포인트 없이 복제 인스턴스를 구성한 경우, 공개적으로 액세스 가능한 VPC 엔드포인트를 복제 인스턴스가 포함된 VPC에 추가하면 공개적으로 액세스가 가능해집니다. 복제 인스턴스를 VPC 엔드포인트와 명확히 연결하기 위해 추가 작업을 수행하지 않아도 됩니다.

> **Note**
>
> 서비스마다 고유한 VPC 엔드포인트 구성이 존재할 수 있습니다. 예를 들어, AWS Secrets Manager를 사용할 경우 일반적으로 라우팅 테이블을 조정할 필요가 없습니다. 항상 각 서비스의 특정한 요구 사항을 확인하세요.

복제 인스턴스가 포함된 VPC에서 VPC 엔드포인트를 생성합니다.

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/vpc/에서 Amazon VPC 콘솔을 엽니다.
2. VPC 콘솔 메뉴 표시줄에서 AWS DMS 복제 인스턴스와 동일한 AWS 리전을 선택합니다.
3. VPC 탐색 창에서 엔드포인트를 선택합니다.
4. 엔드포인트에서 엔드포인트 생성을 선택합니다.
5. 사용자가 이름 태그를 지정할 수도 있습니다. 예: **`my-endpoint-DynamoDB-01`**.
6. S3 또는 DynamoDB 전용 서비스 아래에서, 유형이 게이트웨이로 설정된 서비스 이름을 선택합니다.
7. VPC 아래에서 AWS DMS 복제 인스턴스와 동일한 VPC를 선택하여 엔드포인트를 생성합니다.
8. 라우팅 테이블 아래에서 사용 가능한 모든 라우팅 테이블 ID 값을 선택합니다.
9. 액세스 제어를 지정하려면 정책 아래에서 전체 액세스를 선택합니다. 정책 생성 도구를 사용하여 고유한 액세스 제어를 지정하려면 사용자 지정을 선택합니다. 어떤 경우든 JSON 정책 문서 `dms-vpc-role`을 준수하는 신뢰 정책을 사용합니다. 이 정책 문서에 대한 자세한 내용은 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성 섹션을 참조하세요.
10. 엔드포인트 아래에서 새로 생성된 VPC 엔드포인트의 상태가 사용 가능인지 확인합니다.

AWS DMS 복제 인스턴스의 VPC 엔드포인트 구성에 대한 자세한 내용은 데이터베이스 마이그레이션을 위한 네트워크 구성 섹션을 참조하세요. 일반적으로 AWS 서비스에 액세스하기 위한 인터페이스 VPC 엔드포인트를 생성하는 방법에 대한 자세한 내용은 AWS PrivateLink 가이드의 인터페이스 VPC 엔드포인트를 사용하여 AWS 서비스 액세스 섹션을 참조하세요. VPC 엔드포인트의 AWS DMS 리전별 가용성에 대한 자세한 내용은 AWS 리전 표를 참조하세요.

AWS DMS에 의해 지원되는 DDL 문

데이터 마이그레이션 프로세스 중에 원본 데이터베이스에서 데이터 정의 언어(DDL) 문을 실행할 수 있습니다. 이 문은 복제 서버를 통해 대상 데이터베이스로 복제됩니다.

지원되는 DDL 문은 다음과 같습니다.

- 테이블 생성
- 테이블 삭제
- 테이블 이름 바꾸기
- 테이블 자르기

- 열 추가
- 열 삭제
- 열 이름 바꾸기
- 열 데이터 유형 변경

DMS는 일부 소스 엔진 유형에서 지원되는 모든 DDL 문을 캡처하지 않습니다. 또한 DMS는 DDL 문을 특정 대상 엔진에 적용할 때 DDL 문을 다르게 처리합니다. 특정 소스에 지원되는 DDL 문 및 이러한 문이 대상에 어떻게 적용되는지에 대한 자세한 내용은 해당 소스 및 대상 엔드포인트에 대한 설명서의 관련 항목을 참조하세요.

태스크 설정을 사용하여 변경 데이터 캡처(CDC) 중에 DMS가 DDL 동작을 처리하는 방식을 구성할 수 있습니다. 자세한 내용은 변경 처리 DDL을 다루기 위한 작업 설정을(를) 참조하세요.

AWS DMS 작업 사용

AWS Database Migration Service(AWS DMS) 태스크는 모든 작업이 이루어지는 곳입니다. 마이그레이션 및 특수 처리(로깅 요구 사항, 제어 테이블 데이터 및 오류 처리 등)에 사용할 테이블(또는 뷰)과 스키마를 지정합니다.

작업은 세 가지 주요 단계로 구성되어 있습니다.

- 기존 데이터 마이그레이션(전체 로드)
- 캐시된 변경 사항 적용
- 지속적 복제(변경 데이터 캡처)

AWS DMS 마이그레이션 태스크에서 데이터를 마이그레이션하는 방법에 대한 자세한 내용 및 개요는 개괄적인 관점 AWS DMS 섹션을 참조하세요.

마이그레이션 작업 생성 시 몇 가지 사항에 대해 알고 있어야 합니다.

- 작업을 생성하려면 먼저 소스 엔드포인트, 대상 엔드포인트, 복제 인스턴스를 생성해야 합니다.
- 많은 작업 설정을 지정하여 마이그레이션 작업을 조정할 수 있습니다. 이러한 조정 작업은 AWS Management Console, AWS Command Line Interface(AWS CLI) 또는 AWS DMS API를 사용하여 수행합니다. 이러한 설정에는 마이그레이션 오류의 처리 방식 지정, 오류 로깅 및 제어 테이블 정보가 포함됩니다. 태스크 구성 파일을 사용하여 태스크 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제 섹션을 참조하세요.
- 작업을 생성하면 즉시 실행할 수 있습니다. 필수 메타데이터가 정의된 대상 테이블은 자동으로 생성되어 로드되고, 복제 프로세스가 지속됨을 지정할 수 있습니다.
- 기본적으로 AWS DMS는 작업 생성 즉시 작업을 시작합니다. 그렇지만 경우에 따라 작업 시작을 연기하기도 합니다. 예를 들어, AWS CLI를 사용할 경우 한 프로세스로는 작업을 생성하고 다른 프로세스로는 트리거 이벤트에 따라 해당 작업을 시작하게 할 수 있습니다. 필요에 따라 작업 시작을 연기할 수 있습니다.
- 콘솔, AWS CLI 또는 AWS DMS API를 사용하여 작업을 모니터링, 중지 또는 다시 시작할 수 있습니다. AWS DMS API를 사용하여 태스크를 중지하는 방법에 대한 내용은 AWS DMS API 참조의 StopReplicationTask를 참조하세요.

다음은 AWS DMS 작업으로 수행할 수 있는 작업입니다.

작업	관련 설명서
태스크 생성 작업 생성 시 마이그레이션 설정과 함께 소스, 대상 및 복제 인스턴스를 지정합니다.	작업 생성
진행 중인 복제 작업 생성 원본과 대상 간에 지속적인 복제를 수행하는 작업을 설정할 수 있습니다.	AWS DMS를 사용하여 지속 복제를 위한 작업 생성
작업 설정 적용 각 작업에는 데이터베이스 마이그레이션의 요구 사항에 따라 구성할 수 있는 설정이 있습니다. JSON 파일로 이 설정을 생성하거나 AWS DMS 콘솔을 사용하여 일부 설정과 함께 이 설정을 지정할 수 있습니다. 태스크 구성 파일을 사용하여 태스크 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제 섹션을 참조하세요.	AWS Database Migration Service 작업에 대한 작업 설정 지정
테이블 매핑 사용 테이블 매핑은 여러 가지 유형의 규칙을 사용하여 테이블에 대한 추가 태스크 설정을 지정합니다. 이러한 규칙을 사용하여 데이터 소스, 소스 스키마, 테이블 및 보기, 데이터, 태스크 중에 발생하는 테이블 변환 및 데이터 변환, 이러한 테이블과 열을 소스에서 대상으로 마이그레이션하는 방법에 대한 설정을 지정할 수 있습니다.	선택 규칙 선택 규칙 및 작업 변환 규칙 변환 규칙 및 작업 테이블 설정 규칙 테이블 및 컬렉션 설정 규칙과 작업
마이그레이션 전 태스크 평가 실행	작업에 대한 마이그레이션 전 평가 활성화 및 활용

작업	관련 설명서
마이그레이션 도중 문제를 일으킬 수 있는 소스 및 대상 데이터베이스와 관련된 문제를 보여주는 마이그레이션 전 태스크 평가를 활성화하고 실행할 수 있습니다. 여기에는 지원되지 않는 데이터 형식, 일치하지 않는 인덱스와 프라이머리 키, 기타 태스크 설정 충돌 등을 비롯한 문제가 포함될 수 있습니다. 태스크 실행 전에 이러한 마이그레이션 전 평가를 실행하여 마이그레이션 도중 문제가 발생하기 전에 잠재적 문제를 파악합니다.	
데이터 유효성 검사 데이터 검증은 AWS DMS가 원본 데이터 스토어의 데이터와 대상 데이터 스토어의 데이터 비교를 수행하도록 하는 데 사용할 수 있는 작업 설정입니다.	AWS DMS 데이터 검증.
태스크 수정 작업이 중지되면 작업 설정을 수정할 수 있습니다.	작업 수정
태스크 이동 태스크가 중지되면 태스크를 다른 복제 인스턴스로 이동할 수 있습니다.	태스크 이동
태스크 도중 테이블 다시 로드 작업 중에 오류가 발생하면 작업 중에 테이블을 다시 로드할 수 있습니다.	작업 중 테이블 다시 로드

작업	관련 설명서
필터 적용 소스 필터를 사용하여 소스에서 대상으로 전송되는 레코드의 수와 유형을 제한할 수 있습니다. 예를 들어, 본사 사업장에서 근무하는 직원만이 대상 데이터베이스로 이동하도록 지정할 수 있습니다. 데이터 열에서 필터를 적용합니다.	소스 필터 사용
작업 모니터링 작업에서 사용하는 테이블과 작업 성능에 관한 정보를 가져오는 방법에는 여러 가지가 있습니다.	AWS DMS 태스크 모니터링
작업 로그 관리 AWS DMS API 또는 AWS CLI를 사용하여 작업 로그를 보고 삭제할 수 있습니다.	AWS DMS 태스크 로그 보기 및 관리

주제

- 작업 생성
- AWS DMS를 사용하여 지속 복제를 위한 작업 생성
- 작업 수정
- 태스크 이동
- 작업 중 테이블 다시 로드
- 작업 설정을 지정하기 위한 테이블 매핑 사용
- 소스 필터 사용
- 작업에 대한 마이그레이션 전 평가 활성화 및 활용
- 작업 설정에 대한 보충 데이터 지정

작업 생성

AWS DMS 마이그레이션 작업을 생성하려면 다음과 같이 하십시오.

- 마이그레이션 작업을 생성하기 전에 먼저 소스 엔드포인트, 대상 엔드포인트 및 복제 인스턴스를 생성합니다.
- 마이그레이션 방법을 선택합니다.
  - 데이터를 대상 데이터베이스로 마이그레이션 – 이 프로세스는 대상 데이터베이스에 파일 또는 테이블을 생성하며 대상에서 필요한 메타데이터를 자동으로 정의합니다. 또한 소스의 데이터로 테이블을 채웁니다. 테이블의 데이터가 동시에 로드되므로 효율성이 향상됩니다. 이 프로세스는 의 기존 데이터 마이그레이션 AWS Management Console 옵션이며 `Full Load` API에서 호출됩니다.
  - 마이그레이션 중 변경 사항 캡처 – 이 프로세스는 데이터가 소스에서 대상으로 마이그레이션되는 동안 발생하는 소스 데이터베이스의 변경 사항을 캡처합니다. 원래 요청된 데이터의 마이그레이션이 완료되면 변경 데이터 캡처(CDC) 프로세스에서는 캡처된 변경 사항을 대상 데이터베이스에 적용합니다. 변경 사항이 커밋된 트랜잭션의 단위로서 캡처 및 적용되며 다양한 대상 테이블을 하나의 소스 커밋으로서 업데이트할 수 있습니다. 이 접근 방식은 대상 데이터베이스에서 트랜잭션 무결성을 보장합니다. 이 프로세스는 콘솔의 기존 데이터 마이그레이션 및 지속적인 변경 사항 복제 옵션이며 API에서 `full-load-and-cdc`라고 합니다.
  - 소스 데이터베이스에서 유일한 데이터 변경 사항 복제 – 이 프로세스는 원본 DBMS(데이터베이스 관리 시스템)의 복구 로그 파일을 읽고 트랜잭션별로 항목을 그룹화합니다. 경우에 따라 적절한 시간 내에 대상에 변경 내용을 적용할 AWS DMS 수 없는 경우가 있습니다 (예: 대상에 액세스할 수 없는 경우). 이러한 경우 는 필요한 기간 동안 복제 서버의 변경 내용을 AWS DMS 버퍼링합니다. 원본 DBMS 로그를 다시 읽지 않으므로 오랜 시간이 소요될 수 있습니다. 이 프로세스는 AWS DMS 콘솔의 데이터 변경 내용 복제 전용 옵션입니다.
- 작업에서 소스에 대해 대용량 이진 객체(LOB)를 처리하는 방식을 결정합니다. 자세한 정보는 태스크의 소스 데이터베이스에 대한 LOB 지원 설정 AWS DMS을 참조하세요.
- 마이그레이션 작업 설정을 지정합니다. 여기에는 로깅 설정, 마이그레이션 제어 테이블에 기록될 데이터 지정, 오류 처리 방식 및 기타 설정이 포함됩니다. 작업 설정에 관한 자세한 내용은 AWS Database Migration Service 작업에 대한 작업 설정 지정 섹션을 참조하십시오.
- 테이블 매핑을 설정하여 마이그레이션 중인 데이터를 선택하고 필터링하기 위한 규칙을 정의합니다. 테이블 매핑에 관한 자세한 내용은 작업 설정을 지정하기 위한 테이블 매핑 사용 섹션을 참조하십시오. 매핑을 지정하기 전에 원본 및 대상 데이터베이스의 데이터 형식 매핑에 대한 관련 설명서 단원을 참조하십시오.

- 작업을 실행하기 전에 마이그레이션 전 작업 평가를 활성화하고 실행하십시오. 마이그레이션 전 평가에 관한 자세한 내용은 작업에 대한 마이그레이션 전 평가 활성화 및 활용 단원을 참조하십시오.
- 데이터를 마이그레이션하는 작업에 필요한 보충 데이터를 지정합니다. 자세한 정보는 작업 설정에 대한 보충 데이터 지정을 참조하세요.

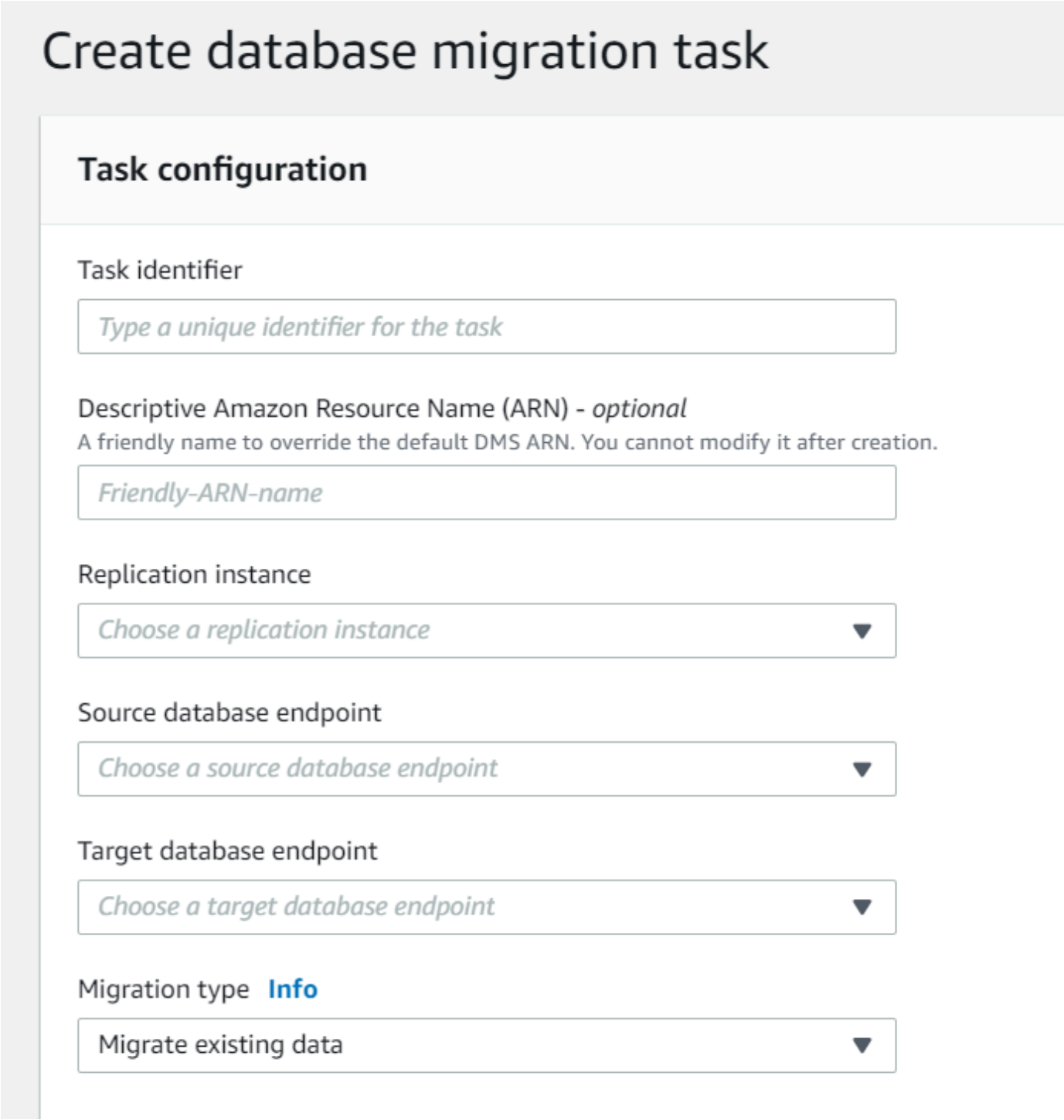
작업 생성 페이지에서 해당 작업에 관한 정보를 모두 지정하면 바로 작업을 시작할 수 있습니다. 또는 나중에 대시보드 페이지에서 작업을 시작할 수도 있습니다.

다음 절차에서는 복제 인스턴스 정보와 엔드포인트를 이미 지정했다고 가정합니다. 엔드포인트 설정에 관한 자세한 내용은 소스 및 대상 엔드포인트 생성 단원을 참조하십시오.

마이그레이션 작업을 생성하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.

   AWS Identity and Access Management (IAM) 사용자로 로그인한 경우 적절한 액세스 AWS DMS 권한이 있는지 확인하십시오. 필요한 권한에 관한 자세한 내용은 AWS DMS 사용에 필요한 IAM 권한 단원을 참조하십시오.
2. 탐색 창에서 데이터베이스 마이그레이션 작업을 선택한 후 작업 생성을 선택합니다.
3. 데이터베이스 마이그레이션 작업 생성 페이지의 작업 구성 섹션에서 작업 옵션을 지정합니다. 다음 표는 설정에 대한 설명입니다.



옵션	조치
작업 식별자	작업의 이름을 입력합니다.

옵션	조치
설명이 포함된 Amazon 리소스 이름(ARN) - 선택 사항	기본 AWS DMS ARN을 재정의하기 위한 친숙한 이름입니다. 작업을 생성한 후에는 이 이름을 변경할 수 없습니다.
복제 인스턴스	사용할 복제 인스턴스를 표시합니다.
소스 데이터베이스 엔드포인트	사용할 소스 엔드포인트를 표시합니다.
대상 데이터베이스 엔드포인트	사용할 대상 엔드포인트를 표시합니다.
[Migration type]	사용할 마이그레이션 방법을 선택합니다. 기존 데이터가 대상 데이터베이스에 마이그레이션되도록 선택하거나 마이그레이션된 데이터 외에 지속적 변경 사항이 대상 데이터베이스에 전송되도록 선택할 수 있습니다.

4. 작업 설정 섹션에서 작업 편집, 대상 테이블 준비 모드, 작업 중지, LOB 설정, 검증 및 로깅을 위한 값을 지정합니다.

옵션	조치
[Editing mode]	작업 설정을 지정하는 데 마법사를 사용할지 아니면 JSON 편집기를 사용할지 선택합니다. 마법사를 선택하면 다음 옵션이 표시됩니다.
소스 트랜잭션을 위한 CDC 시작 모드	이 설정은 이전 섹션의 마이그레이션 유형으로만 데이터 변경 복제를 선택한 경우에만 표시됩니다. 사용자 지정 CDC 시작 모드 비활성화 - 이 옵션을 선택하면 다음에 나오는 생성 시 자동으로 옵션을 사용하여 작업을 자동으로 시작하거나 콘솔을 사용하여 수동으로 작업을 시작할 수 있습니다. 사용자 지정 CDC 시작 모드 활성화 - 이 옵션을 선택하면 변경 처리를 시작할 사용자 지정 UTC 시작 시간을 지정할 수 있습니다.

옵션	조치
[Target table preparation mode]	이 설정은 이전 섹션의 마이그레이션 유형에 대해 기존 데이터 마이그레이션 또는 기존 데이터를 마이그레이션하고 진행 중인 변경 사항을 복제를 선택하는 경우에만 표시됩니다. Do nothing — Do nothing 모드에서는 대상 테이블이 타겟에 미리 생성되었다고 AWS DMS 가정합니다. 테이블이 비어 있지 않으면 데이터 마이그레이션 중에 충돌이 발생하여 DMS 작업 오류가 발생할 수 있습니다. 대상 테이블이 없는 경우 DMS에서는 테이블을 생성합니다. 테이블 구조는 그대로 유지되고 기존 데이터는 테이블에 남습니다. 아무 작업 안 함 모드는 소스에서 대상 테이블이 채워져 있는 경우 CDC 전용 작업에 적합하고 진행 중 복제는 소스 및 대상 간 동기화를 유지합니다. 테이블을 미리 생성하려면 AWS Schema Conversion Tool (AWS SCT)를 사용하면 됩니다. 자세한 내용은 설치를 참조하십시오. AWS SCT 대상에서 테이블 삭제 – 대상에서 테이블 삭제 모드에서는 AWS DMS 가 마이그레이션 시작 전에 대상 테이블을 삭제하고 다시 만듭니다. 이 방법을 사용하면 마이그레이션이 시작될 때 대상 테이블이 비어 있게 됩니다. AWS DMS 데이터를 효율적으로 마이그레이션하는 데 필요한 객체 (테이블, 기본 키, 경우에 따라 고유 인덱스) 만 생성합니다. AWS DMS 보조 인덱스, 기본이 아닌 키 제약 조건 또는 열 데이터 기본값을 만들지 않습니다. 전체 로드와 CDC 또는 CDC 전용 작업을 수행하는 경우, 이 시점에서 마이그레이션을 일시 중지하는 것이 좋습니다. 그런 다음, update 및 delete 문에 대한 필터링을 지원하는 보조 인덱스를 생성합니다. 대상에서 테이블 삭제 모드를 사용할 경우 대상 데이터베이스에서 몇 가지 구성을 실행해야 합니다. 예를 들어 Oracle 대상의 경우 보안상의 이유로 스키마 (데이터베이스 사용자) 를 만들 AWS DMS 수 없습니다. 이 경우

옵션	조치
	마이그레이션이 시작될 때 테이블을 만들 AWS DMS 수 있도록 스키마 사용자를 미리 생성합니다. 대부분의 다른 대상 유형의 경우 적절한 구성 매개 변수를 사용하여 스키마 및 모든 관련 테이블을 AWS DMS 생성합니다. 잘라내기 - 잘라내기 모드에서는 마이그레이션이 시작되기 전에 모든 대상 AWS DMS 테이블을 잘라냅니다. 대상 테이블이 없는 경우 DMS에서 테이블을 생성합니다. 테이블 구조는 그대로 유지되지만 대상에서 테이블이 잘립니다. 자르기 모드는 마이그레이션이 시작되기 전에 대상 스키마가 미리 생성된 전체 로드 또는 전체 로드와 CDC 마이그레이션에 적합합니다. 테이블을 미리 생성하려면 AWS SCT를 사용하면 됩니다. 자세한 내용은 설치를 참조하십시오. AWS SCT **Note** 대상이 MongoDB인 경우 자르기 모드에서는 대상의 테이블이 잘리지 않습니다. 그 대신 컬렉션이 삭제되고 모든 인덱스가 손실됩니다. 대상이 MongoDB인 경우 자르기 모드를 사용하지 마십시오.

옵션	조치
[Stop task after full load completes]	이 설정은 이전 섹션의 마이그레이션 유형에 대해 기존 데이터를 마이그레이션하고 진행 중인 변경 사항을 복제를 선택하는 경우에만 표시됩니다. 멈추지 않음 – 작업을 중지하지 않고, 캐시된 변경 사항을 즉시 적용하고 계속 진행합니다. 캐시된 변경 사항 적용 전 멈춤 - 캐시된 변경 사항을 적용하기 전에 작업을 중지합니다. 이 방법을 사용하면 보조 인덱스를 추가할 수 있으며 이에 따라 변경 사항을 적용하는 속도가 향상될 수 있습니다. [Stop after applying cached changes] - 캐시된 변경 사항 적용 후 작업을 중지합니다. 이 방법을 사용하면 트랜잭션 적용을 사용하는 경우 외래 키를 추가할 수 있습니다.
[Include LOB columns in replication]	LOB 열 포함 안 함 – LOB 열이 마이그레이션에서 제외됩니다. 전체 LOB 모드 - 크기에 관계없이 전체 LOB를 마이그레이션합니다. AWS DMS 최대 LOB 크기 매개변수로 제어되는 청크 단위로 LOB를 마이그레이션합니다. 이 모드는 제한적 LOB 모드를 사용하는 것보다 더 느립니다. 제한적 LOB 모드 – LOB를 최대 LOB 크기 파라미터 값으로 자릅니다. 이 모드는 전체 LOB 모드를 사용하는 것보다 더 빠릅니다.
최대 LOB 크기(kb)	제한적 LOB 모드에서는 최대 LOB 크기의 설정을 초과하는 LOB 열이 지정된 최대 LOB 크기 값으로 잘립니다.
검증 활성화	원본에서 대상으로 데이터가 정확히 마이그레이션되는지 확인하기 위해 데이터 검증을 활성화합니다. 자세한 정보는 AWS DMS 데이터 검증을 참조하세요.

옵션	조치
CloudWatch 로그 활성화	Amazon의 로깅을 CloudWatch 활성화합니다.

5. 마이그레이션 전 평가 섹션에서 마이그레이션 전 평가 실행 여부를 선택합니다. 마이그레이션 전 평가를 통해 데이터베이스 마이그레이션 작업을 시작하기 전에 잠재적인 마이그레이션 문제를 경고할 수 있습니다. 자세한 정보는 마이그레이션 전 평가 활성화 및 활용을 참조하세요.
6. 마이그레이션 작업 시작 구성 섹션에서 작업 생성 후 자동으로 시작할지 여부를 지정합니다.
7. 태그 섹션에서 작업을 정리하는 데 필요한 태그를 지정합니다. 태그를 사용하여 IAM 역할 및 정책을 관리하고 DMS 비용을 추적할 수 있습니다. 자세한 정보는 리소스에 태그 지정을 참조하세요.
8. 작업 설정을 완료했다면 작업 생성을 선택합니다.

AWS Database Migration Service 작업에 대한 작업 설정 지정

각 작업에는 데이터베이스 마이그레이션의 요구 사항에 따라 구성할 수 있는 설정이 있습니다. 이러한 설정은 JSON 파일에서 생성하거나 일부 설정의 경우 AWS DMS 콘솔을 사용하여 설정을 지정할 수 있습니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

기본적인 작업 설정 유형은 다음과 같습니다.

주제

- 작업 설정 예제
- 대상 메타데이터 작업 설정
- 전체 로드 작업 설정
- Time Travel 작업 설정
- 작업 설정 로깅
- 제어 테이블 작업 설정
- 스트림 버퍼 작업 설정
- 변경 처리 튜닝 설정
- 데이터 검증 작업 설정
- 변경 처리 DDL을 다루기 위한 작업 설정
- 문자 대체 작업 설정
- 이전 이미지 작업 설정

- 오류 처리 작업 설정
- 작업 설정 저장

작업 설정	관련 설명서
작업 평가 보고서 생성 마이그레이션 중에 문제를 유발할 수 있는 지원되지 않는 데이터 형식을 보여 주는 작업 평가 보고서를 생성할 수 있습니다. 작업을 실행하기 전에 잠재적인 문제를 알아보기 위해 작업에 대해 이 보고서를 실행할 수 있습니다.	작업에 대한 마이그레이션 전 평가 활성화 및 활용
작업 생성 작업 생성 시 마이그레이션 설정과 함께 소스, 대상 및 복제 인스턴스를 지정합니다.	작업 생성
지속적 복제 태스크 생성 원본과 대상 간에 지속적인 복제를 수행하는 작업을 설정할 수 있습니다.	AWS DMS를 사용하여 지속 복제를 위한 작업 생성
태스크 설정 적용 각 작업에는 데이터베이스 마이그레이션의 요구 사항에 따라 구성할 수 있는 설정이 있습니다. 이러한 설정은 JSON 파일에서 생성하거나 일부 설정의 경우 콘솔을 사용하여 설정을 지정할 수 있습니다. AWS DMS	AWS Database Migration Service 작업에 대한 작업 설정 지정
데이터 유효성 검사 데이터 검증을 사용하여 대상 데이터 저장소의 데이터를 원본 데이터 저장	AWS DMS 데이터 검증

작업 설정	관련 설명서
소의 데이터와 AWS DMS 비교합니다.	
작업 수정 작업이 중지되면 작업 설정을 수정할 수 있습니다.	작업 수정
작업 중 테이블 다시 로드 작업 중에 오류가 발생하면 작업 중에 테이블을 다시 로드할 수 있습니다.	작업 중 테이블 다시 로드
테이블 매핑 사용 테이블 매핑에서는 여러 유형의 규칙을 사용하여 데이터 소스, 원본 스키마, 데이터, 작업 중 발생하는 모든 변환 등에 대한 작업 설정을 지정합니다.	선택 규칙 선택 규칙 및 작업 변환 규칙 변환 규칙 및 작업
필터 적용 소스 필터를 사용하여 소스에서 대상으로 전송되는 레코드의 수와 유형을 제한할 수 있습니다. 예를 들어, 본사 사업장에서 근무하는 직원만이 대상 데이터베이스로 이동하도록 지정할 수 있습니다. 데이터 열에서 필터를 적용합니다.	소스 필터 사용
태스크 모니터링 작업에서 사용하는 테이블과 작업 성능에 관한 정보를 가져오는 방법에는 여러 가지가 있습니다.	AWS DMS 태스크 모니터링

작업 설정	관련 설명서
태스크 로그 관리 AWS DMS API 또는 를 사용하여 작업 로그를 보고 삭제할 수 AWS CLI있습니다.	AWS DMS 태스크 로그 보기 및 관리

작업 설정 예제

AWS Management Console 또는 를 AWS CLI 사용하여 복제 작업을 생성할 수 있습니다. 를 사용하는 경우 JSON 파일을 만든 다음 JSON 파일의 file://URI를 작업 ReplicationTaskSettings매개 변수로 지정하여 작업 설정을 지정합니다. AWS CLICreateReplicationTask

다음 예제는 를 사용하여 작업을 AWS CLI 호출하는 방법을 보여줍니다. `CreateReplicationTask`

```
aws dms create-replication-task \
--replication-task-identifier MyTask \
--source-endpoint-arn arn:aws:dms:us-
west-2:123456789012:endpoint:ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890ABC \
--target-endpoint-arn arn:aws:dms:us-
west-2:123456789012:endpoint:ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890ABC \
--replication-instance-arn arn:aws:dms:us-
west-2:123456789012:rep:ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890ABC \
--migration-type cdc \
--table-mappings file://tablemappings.json \
--replication-task-settings file://settings.json
```

위 예제에서는 `tablemappings.json`이라는 테이블 매핑 파일을 사용합니다. 테이블 매핑 예는 작업 설정을 지정하기 위한 테이블 매핑 사용 단원을 참조하십시오.

작업 설정 JSON 파일은 다음과 같습니다.

```
{
  "TargetMetadata": {
    "TargetSchema": "",
    "SupportLobs": true,
    "FullLobMode": false,
    "LobChunkSize": 64,
```

```
    "LimitedSizeLobMode": true,
    "LobMaxSize": 32,
    "InlineLobMaxSize": 0,
    "LoadMaxFileSize": 0,
    "ParallelLoadThreads": 0,
    "ParallelLoadBufferSize":0,
    "ParallelLoadQueuesPerThread": 1,
    "ParallelApplyThreads": 0,
    "ParallelApplyBufferSize": 100,
    "ParallelApplyQueuesPerThread": 1,
    "BatchApplyEnabled": false,
    "TaskRecoveryTableEnabled": false
  },
  "FullLoadSettings": {
    "TargetTablePrepMode": "DO_NOTHING",
    "CreatePkAfterFullLoad": false,
    "StopTaskCachedChangesApplied": false,
    "StopTaskCachedChangesNotApplied": false,
    "MaxFullLoadSubTasks": 8,
    "TransactionConsistencyTimeout": 600,
    "CommitRate": 10000
  },
    "TTSettings" : {
    "EnableTT" : true,
    "TTS3Settings": {
        "EncryptionMode": "SSE_KMS",
        "ServerSideEncryptionKmsKeyId": "arn:aws:kms:us-west-2:112233445566:key/
myKMSKey",
        "ServiceAccessRoleArn": "arn:aws:iam::112233445566:role/dms-tt-s3-access-role",
        "BucketName": "myttbucket",
        "BucketFolder": "myttfolder",
        "EnableDeletingFromS3OnTaskDelete": false
      },
    "TTRecordSettings": {
        "EnableRawData" : true,
        "OperationsToLog": "DELETE,UPDATE",
        "MaxRecordSize": 64
      }
  },
  "Logging": {
    "EnableLogging": false
  },
  "ControlTablesSettings": {
    "ControlSchema":"",
```

```
    "HistoryTimeslotInMinutes":5,
    "HistoryTableEnabled": false,
    "SuspendedTablesTableEnabled": false,
    "StatusTableEnabled": false
  },
  "StreamBufferSettings": {
    "StreamBufferCount": 3,
    "StreamBufferSizeInMB": 8
  },
  "ChangeProcessingTuning": {
    "BatchApplyPreserveTransaction": true,
    "BatchApplyTimeoutMin": 1,
    "BatchApplyTimeoutMax": 30,
    "BatchApplyMemoryLimit": 500,
    "BatchSplitSize": 0,
    "MinTransactionSize": 1000,
    "CommitTimeout": 1,
    "MemoryLimitTotal": 1024,
    "MemoryKeepTime": 60,
    "StatementCacheSize": 50
  },
  "ChangeProcessingDdlHandlingPolicy": {
    "HandleSourceTableDropped": true,
    "HandleSourceTableTruncated": true,
    "HandleSourceTableAltered": true
  },
  "LoopbackPreventionSettings": {
    "EnableLoopbackPrevention": true,
    "SourceSchema": "LOOP-DATA",
    "TargetSchema": "loop-data"
  },

  "CharacterSetSettings": {
    "CharacterReplacements": [ {
        "SourceCharacterCodePoint": 35,
        "TargetCharacterCodePoint": 52
      }, {
        "SourceCharacterCodePoint": 37,
        "TargetCharacterCodePoint": 103
      }
    ],
    "CharacterSetSupport": {
      "CharacterSet": "UTF16_PlatformEndian",
      "ReplaceWithCharacterCodePoint": 0
```

```
    }
  },
  "BeforeImageSettings": {
    "EnableBeforeImage": false,
    "FieldName": "",
    "ColumnFilter": "pk-only"
  },
  "ErrorBehavior": {
    "DataErrorPolicy": "LOG_ERROR",
    "DataTruncationErrorPolicy":"LOG_ERROR",
    "DataErrorEscalationPolicy":"SUSPEND_TABLE",
    "DataErrorEscalationCount": 50,
    "TableErrorPolicy":"SUSPEND_TABLE",
    "TableErrorEscalationPolicy":"STOP_TASK",
    "TableErrorEscalationCount": 50,
    "RecoverableErrorCount": 0,
    "RecoverableErrorInterval": 5,
    "RecoverableErrorThrottling": true,
    "RecoverableErrorThrottlingMax": 1800,
    "ApplyErrorDeletePolicy":"IGNORE_RECORD",
    "ApplyErrorInsertPolicy":"LOG_ERROR",
    "ApplyErrorUpdatePolicy":"LOG_ERROR",
    "ApplyErrorEscalationPolicy":"LOG_ERROR",
    "ApplyErrorEscalationCount": 0,
    "FullLoadIgnoreConflicts": true
  },
  "ValidationSettings": {
    "EnableValidation": false,
    "ValidationMode": "ROW_LEVEL",
    "ThreadCount": 5,
    "PartitionSize": 10000,
    "FailureMaxCount": 1000,
    "RecordFailureDelayInMinutes": 5,
    "RecordSuspendDelayInMinutes": 30,
    "MaxKeyColumnSize": 8096,
    "TableFailureMaxCount": 10000,
    "ValidationOnly": false,
    "HandleCollationDiff": false,
    "RecordFailureDelayLimitInMinutes": 1,
    "SkipLobColumns": false,
    "ValidationPartialLobSize": 0,
    "ValidationQueryCdcDelaySeconds": 0
  }
}
```

대상 메타데이터 작업 설정

대상 메타데이터 설정에는 다음이 포함됩니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

- TargetSchema – 대상 테이블 스키마 이름입니다. 이 메타데이터 옵션이 비어 있는 경우, 원본 테이블의 스키마가 사용됩니다. AWS DMS 는 원본 스키마가 정의되지 않은 경우 대상 데이터베이스의 소유자 접두사를 모든 테이블에 자동으로 추가합니다. 이 옵션은 MySQL 유형 대상 엔드포인트에서 비어두어야 합니다. 데이터 매핑에서 스키마 이름을 바꾸는 것이 이 설정보다 우선합니다.
- LOB 설정 – 대형 객체(LOB)가 관리되는 방법을 결정하는 설정입니다. SupportLobs=true를 설정하는 경우, 다음 중 하나를 true로 설정해야 합니다.
  - FullLobMode – 이 옵션을 true로 설정하는 경우, LobChunkSize 옵션 값을 입력해야 합니다. 데이터를 대상으로 복제할 때 사용할 LOB 청크의 크기(단위: 킬로바이트[KB])를 입력합니다. FullLobMode 옵션은 매우 큰 LOB 크기에서 최고 성능을 발휘하지만 로드가 느려지는 경향이 있습니다. LobChunkSize의 권장값은 64킬로바이트[KB]입니다. LobChunkSize의 값을 64KB 이상으로 늘리면 작업이 실패할 수 있습니다.
  - InlineLobMaxSize— 이 값은 전체 로드 중에 인라인으로 AWS DMS 전송할 LOB를 결정합니다. 작은 LOB를 전송하면 소스 테이블에서 조회하는 것보다 효율적입니다. 전체 로드 중에는 모든 LOB를 AWS DMS 검사하고 보다 작은 LOB에 대해 인라인 전송을 수행합니다. InlineLobMaxSize AWS DMS in보다 큰 모든 LOB를 전송합니다. InlineLobMaxSize FullLobMode InlineLobMaxSize의 기본값은 0이고 범위는 1~102400킬로바이트(100MB)입니다. 모든 LOB가 InlineLobMaxSize에 지정된 값보다 작으면 InlineLobMaxSize 값만 지정하십시오.
  - LimitedSizeLobMode – 이 옵션을 true로 설정하는 경우, LobMaxSize 옵션 값을 입력해야 합니다. 개별 LOB에서 최대 크기(단위: 킬로바이트[KB])를 입력합니다. LobMaxSize의 최대 권장 값은 102400킬로바이트(100MB)입니다.

  이러한 작업 LOB 설정을 사용하기 위한 기준에 관한 자세한 내용은 태스크의 소스 데이터베이스에 대한 LOB 지원 설정 AWS DMS 단원을 참조하십시오. 개별 테이블에 대한 LOB 관리를 제어할 수도 있습니다. 자세한 정보는 테이블 및 컬렉션 설정 규칙과 작업을 참조하세요.
- LoadMaxFileSize – 데이터 로드에 쉼표로 구분된 값(.csv) 파일을 사용할 수 있도록 지원하는 MySQL, PostgreSQL, Amazon Redshift와 같은 CSV 기반 대상 엔드포인트를 위한 옵션입니다. LoadMaxFileSize는 저장되었지만 로드되지 않은 데이터(예: .csv 파일)의 디스크 최대 크기를 정

의합니다. 이 옵션은 대상 엔드포인트 연결 속성 maxFileSize보다 우선합니다. 값은 0부터 제공할 수 있으며, 이는 이 옵션이 연결 속성을 100,000KB로 재정의하지 않음을 나타냅니다.

- BatchApplyEnabled – 각 트랜잭션이 개별적으로 적용되는지 또는 변경 사항이 배치 단위로 커밋되는지 여부를 결정합니다. 기본 값은 false입니다.

  BatchApplyEnabled가 true로 설정된 경우, DMS는 소스 테이블에 기본 키(PK) 또는 고유 키(UK)를 필요로 합니다. 소스 테이블에 PK 또는 UK가 없으면 배치 삽입만 적용되며 배치 업데이트 및 삭제는 적용되지 않습니다.

  BatchApplyEnabled를 true로 설정할 때 대상 테이블에 고유한 제약 조건과 기본 키가 있는 경우 AWS DMS 에서는 오류 메시지가 생성됩니다. BatchApplyEnabled가 true로 설정된 경우, 고유 제약조건과 기본 키가 모두 있는 대상 테이블은 지원되지 않습니다.

  BatchApplyEnabled가 true로 설정되고 기본 오류 처리 정책이 적용된 테이블에서 데이터 오류가 AWS DMS 발생하면 나머지 테이블의 AWS DMS 작업이 배치 모드에서 one-by-one 모드로 전환됩니다. 이러한 동작을 변경하려면 작업 설정 JSON 파일의 "ErrorBehavior" 그룹 속성에서 다음 정책에 대한 "SUSPEND_TABLE" 작업을 설정할 수 있습니다.

  - DataErrorPolicy
  - ApplyErrorDeletePolicy
  - ApplyErrorInsertPolicy
  - ApplyErrorUpdatePolicy

  이 "ErrorBehavior" 그룹 속성에 관한 자세한 내용은 AWS Database Migration Service 작업에 대한 작업 설정 지정의 예제 작업 설정 JSON 파일을 참조하십시오. 이러한 정책을 로 "SUSPEND_TABLE" 설정한 후에는 오류가 발생하는 모든 테이블에서 데이터 오류가 일시 중단되고 모든 테이블에 대해 일괄 처리 모드로 AWS DMS 작업이 계속됩니다.

  BatchApplyEnabled 파라미터를 BatchApplyPreserveTransaction 파라미터와 함께 사용할 수 있습니다. BatchApplyEnabled가 true로 설정되면 BatchApplyPreserveTransaction 파라미터는 트랜잭션 무결성을 결정합니다.

  BatchApplyPreserveTransaction이 true로 설정되면 트랜잭션 무결성이 유지되며 배치는 원본의 트랜잭션 내에 있는 모든 변경 사항을 포함하게 됩니다.

  BatchApplyPreserveTransaction이 false로 설정되면 트랜잭션 무결성이 일시적인 시간 경과가 있어 성능이 개선될 수 있습니다.

`BatchApplyPreserveTransaction` 파라미터는 Oracle 대상 엔드포인트에만 적용되고 `BatchApplyEnabled` 파라미터가 `true`로 설정되는 경우에만 관련됩니다.

복제에 LOB 열이 포함되면 `BatchApplyEnabled`를 제한된 LOB 모드에서만 사용할 수 있습니다.

CDC(Change Data Capture) 로드에 이 설정을 사용하는 방법에 관한 자세한 내용은 변경 처리 튜닝 설정 단원을 참조하십시오.

- `MaxFullLoadSubTasks` – 동시에 로드할 최대 테이블 수를 나타냅니다. 기본값은 8이며 최대값은 49입니다.
- `ParallelLoadThreads`— 각 테이블을 대상 데이터베이스에 로드하는 데 AWS DMS 사용하는 스레드 수를 지정합니다. 이 파라미터는 RDBMS가 아닌 대상에 대한 최대값을 가집니다. DynamoDB 대상의 최대값은 200입니다. Amazon Kinesis Data Streams, 아파치 카프카 또는 OpenSearch 아마존 서비스 대상의 최대값은 32입니다. 이 최대 한도를 늘리도록 요청할 수 있습니다. `ParallelLoadThreads`는 전체 로드 작업에 적용됩니다. 개별 테이블의 병렬 로드 설정에 관한 자세한 내용은 테이블 및 컬렉션 설정 규칙과 작업 단원을 참조하십시오.

  이 설정은 다음 엔드포인트 엔진 유형에 적용됩니다.

  - DynamoDB
  - Amazon Kinesis Data Streams
  - Amazon MSK
  - 아마존 OpenSearch 서비스
  - Amazon Redshift

  AWS DMS `ParallelLoadThreadsMySQL`을 추가 연결 속성으로 지원합니다. `ParallelLoadThreadsMySQL`에는 작업 설정으로 적용되지 않습니다.
- `ParallelLoadBufferSize`는 데이터를 대상에 로드하기 위해 병렬 로드 스레드에 사용되는 버퍼에 저장할 최대 레코드 수를 지정합니다. 기본값은 50입니다. 최대값은 1,000입니다. 이 설정은 현재 DynamoDB, Kinesis, Apache Kafka 또는 이 대상인 경우에만 유효합니다. OpenSearch 이 파라미터는 `ParallelLoadThreads`와 함께 사용하십시오. `ParallelLoadBufferSize`는 둘 이상의 스레드가 있는 경우에만 유효합니다. 개별 테이블의 병렬 로드 설정에 관한 자세한 내용은 테이블 및 컬렉션 설정 규칙과 작업 단원을 참조하십시오.
- `ParallelLoadQueuesPerThread` – 대기열에서 데이터 레코드를 가져오고 대상에 대한 배치 로드를 생성하기 위해 각 동시 스레드가 액세스하는 대기열의 수를 지정합니다. 기본 값은 1입니다. 현재 이 설정은 Kinesis 또는 Apache Kafka가 대상일 때만 유효합니다.

- `ParallelApplyThreads`— CDC를 로드하는 동안 Amazon DocumentDB, Kinesis, Amazon MSK 또는 Amazon Redshift 대상 엔드포인트로 데이터 레코드를 푸시하는 데 AWS DMS 사용하는 동시 스레드 수를 지정합니다. OpenSearch 기본값은 0입니다.

  이 설정은 CDC 전용입니다. 이 설정은 전체 로드에 적용되지 않습니다.

  이 설정은 다음 엔드포인트 엔진 유형에 적용됩니다.

  - Amazon DocumentDB(MongoDB 호환)
  - Amazon Kinesis Data Streams
  - Amazon Managed Streaming for Apache Kafka
  - 아마존 OpenSearch 서비스
  - Amazon Redshift
- `ParallelApplyBufferSize`— CDC 로드 중에 동시 스레드가 Amazon DocumentDB, Kinesis, Amazon OpenSearch MSK 또는 Amazon Redshift 대상 엔드포인트로 푸시할 수 있도록 각 버퍼 대기열에 저장할 최대 레코드 수를 지정합니다. 기본 값은 100입니다. 최대값은 1000입니다. `ParallelApplyThreads`가 둘 이상의 스레드를 지정할 때 이 옵션을 사용합니다.
- `ParallelApplyQueuesPerThread`— CDC 중에 각 스레드가 액세스하여 대기열에서 데이터 레코드를 제거하고 Amazon DocumentDB, Kinesis, Amazon MSK 또는 엔드포인트에 대한 배치 로드를 생성하는 대기열의 수를 지정합니다. OpenSearch 기본값은 1입니다.

전체 로드 작업 설정

전체 로드 설정에는 다음이 포함됩니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

- 전체 로드 시작 시 대상 로드 처리 방법을 표시하려면, `TargetTablePrepMode` 옵션에서 다음 값 중 하나를 지정합니다.
  - `DO_NOTHING` – 기존 대상 테이블의 데이터와 메타데이터가 영향을 받지 않습니다.
  - `DROP_AND_CREATE` – 기존 테이블은 삭제되고 새 테이블이 그 자리에 생성됩니다.
  - `TRUNCATE_BEFORE_LOAD` – 테이블 메타데이터에 영향을 주지 않고 데이터가 잘립니다.
- 전체 로드가 완료될 때까지 기본 키 또는 고유 인덱스 생성을 지연시키려면 `CreatePkAfterFullLoad` 옵션을 `true`로 설정합니다.
- 전체 로드 및 CDC 활성 작업에서 `Stop task after full load completes`에 대해 다음과 같은 옵션을 설정할 수 있습니다.

  - `StopTaskCachedChangesApplied` – 이 옵션을 `true`로 설정하면 전체 로드가 완료되고 캐시된 변경 사항이 적용된 후 작업이 중지됩니다.
  - `StopTaskCachedChangesNotApplied` – 이 옵션을 `true`로 설정하면 작업을 중지한 후 캐시된 변경 사항이 적용됩니다.
- 동시에 로드할 테이블의 최대 개수를 나타내려면 `MaxFullLoadSubTasks` 옵션을 설정합니다. 기본값은 8이며 최대값은 49입니다.
- 데이터 레코드를 대상 엔드포인트로 푸시하기 위해 전체 로드 프로세스 중에 DMS가 사용할 동시 스레드 수를 나타내도록 `ParallelLoadThreads` 옵션을 설정합니다. 기본값은 0입니다.

> **Important**
>
> `MaxFullLoadSubTasks`는 동시에 로드할 테이블 또는 테이블 세그먼트의 수를 제어합니다. `ParallelLoadThreads`는 마이그레이션 작업에서 로드를 동시에 실행하는 데 사용하는 스레드의 수를 제어합니다. 이러한 설정은 배수로 적용됩니다. 따라서 전체 로드 작업 중에 사용된 스레드의 총 개수는 대략 `ParallelLoadThreads`의 값에 `MaxFullLoadSubTasks`의 값을 곱한 결과(즉, `ParallelLoadThreads * MaxFullLoadSubtasks`))입니다.
> 전체 로드 하위 작업의 수가 많고 병렬 로드 스레드의 수가 많은 작업을 만들면 작업이 너무 많은 메모리를 소비하여 실패할 수 있습니다.

- 전체 로드 작업을 시작하기 전에 트랜잭션이 종료될 AWS DMS 때까지 대기하는 시간(초) 을 설정할 수 있습니다. 이렇게 하려면 작업을 시작할 때 트랜잭션이 열려 있는 경우 `TransactionConsistencyTimeout` 옵션을 설정합니다. 기본값은 600 (10분) 입니다. AWS DMS 열려 있는 트랜잭션이 있더라도 제한 시간 값에 도달하면 전체 로드를 시작합니다. full-load-only 작업은 10분을 기다리지 않고 즉시 시작됩니다.
- 함께 전송할 수 있는 레코드의 최대 수를 나타내려면 `CommitRate` 옵션을 설정합니다. 기본값은 10,000이고 최대값은 50,000입니다.

Time Travel 작업 설정

AWS DMS Time Travel을 사용하여 복제 작업을 기록하고 디버깅할 수 있습니다. 이 방식에서는 Amazon S3를 사용하여 로그를 저장하고 암호화 키를 사용하여 로그를 암호화합니다. Time Travel S3 버킷에 액세스할 경우에만 날짜-시간 필터를 사용하여 S3 로그를 검색한 다음, 필요에 따라 로그를 보고 다운로드하며 난독화할 수 있습니다. 이렇게 하면 안전하게 '과거로 이동'하여 데이터베이스 활동을 조사할 수 있습니다. 시간 여행은 CloudWatch 로깅과 독립적으로 작동합니다. CloudWatch 로깅에 대한 자세한 내용은 을 참조하십시오작업 설정 로깅.

AWS DMS지원되는 오라클, Microsoft SQL Server 및 PostgreSQL 소스 엔드포인트와 지원되는 PostgreSQL 및 MySQL 대상 엔드포인트가 있는 모든 AWS 지역에서 타임 트래블을 사용할 수 있습니다. AWS DMS Time Travel은 전체 로드 및 변경 데이터 캡처(CDC) 작업과 CDC 전용 작업에만 사용할 수 있습니다. Time Travel을 켜거나 기존의 Time Travel 설정을 수정하려면 복제 작업을 중지해야 합니다.

Time Travel 설정에는 다음과 같은 `TTSettings` 속성이 포함됩니다.

- `EnableTT` – 이 옵션을 `true`로 설정하면 작업에 대한 Time Travel 로깅이 켜집니다. 기본 값은 `false`입니다.

  타입: 부울

  필수 항목 여부: 아니요
- `EncryptionMode` – S3 버킷에서 데이터 및 로그를 저장하는 데 사용되는 서버 측 암호화 유형입니다. `"SSE_S3"`(기본값) 또는 `"SSE_KMS"`를 지정할 수 있습니다.

  `EncryptionMode`를 `"SSE_KMS"`에서 `"SSE_S3"`으로 변경할 수 있지만 그 반대로 변경할 수는 없습니다.

  타입: 문자열

  필수 항목 여부: 아니요
- `ServerSideEncryptionKmsKeyId`— 를 지정하는 `"SSE_KMS"` 경우 사용자 지정 관리 키의 ID를 제공하십시오. `EncryptionMode` AWS KMS 사용하는 키에 AWS Identity and Access Management (IAM) 사용자 권한을 활성화하고 키 사용을 허용하는 정책이 첨부되어 있는지 확인하십시오.

  이 `"SSE_KMS"` 옵션에서는 사용자 지정 관리형 대칭 KMS 키만 지원됩니다.

  타입: 문자열

  필수: `EncryptionMode`이 `"SSE_KMS"`로 설정된 경우에만 해당됨
- `ServiceAccessRoleArn` – 서비스에서 IAM 역할에 액세스하는 데 사용되는 Amazon 리소스 이름(ARN)입니다. 역할 이름을 `dms-tt-s3-access-role`로 설정합니다. 이는 S3 버킷에서 객체를 쓰고 읽을 수 AWS DMS 있도록 허용하는 필수 설정입니다.

  타입: 문자열

  필수: Time Travel이 켜진 경k우

다음은 이 역할에 대한 정책 예제입니다.

```
{
 "Version": "2012-10-17",
 "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "kms:GenerateDataKey",
                "kms:Decrypt",
                "s3:ListBucket",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::S3bucketName*",
                "arn:aws:kms:us-east-1:112233445566:key/1234a1a1-1m2m-1z2z-
d1d2-12dmstt1234"
            ]
        }
    ]
}
```

다음은 이 역할에 대한 신뢰 정책 예제입니다.

```
{
 "Version": "2012-10-17",
 "Statement": [
         {
             "Effect": "Allow",
             "Principal": {
                 "Service": [
                     "dms.amazonaws.com"
                 ]
             },
             "Action": "sts:AssumeRole"
        }
    ]
}
```

- `BucketName` – Time Travel 로그를 저장할 S3 버킷의 이름입니다. Time Travel 로그를 켜기 전에 이 S3 버킷을 생성해야 합니다.

  타입: 문자열

  필수: Time Travel이 켜진 경우

- `BucketFolder` – S3 버킷의 폴더 이름을 설정하는 선택적 파라미터입니다. 이 파라미터를 지정하면 DMS는 경로 "/*BucketName*/*BucketFolder*/*taskARN*/*YYYY*/*MM*/*DD*/*hh*"에 Time Travel 로그를 생성합니다. 이 파라미터를 지정하지 않는 경우 는 기본 경로를 로 AWS DMS 생성합니다"/*BucketName*/dms-time-travel-logs/*taskARN*/*YYYY*/*MM*/*DD*/*hh*.

  타입: 문자열

  필수 항목 여부: 아니요

- `EnableDeletingFromS3OnTaskDelete`— 이 옵션을 로 설정하면 작업이 `true` AWS DMS 삭제되면 S3에서 Time Travel 로그가 삭제됩니다. 기본 값은 `false`입니다.

  타입: 문자열

  필수 항목 여부: 아니요

- `EnableRawData` – 이 옵션을 `true`로 설정하면 Time Travel 로그의 DML(데이터 조작 언어) 원시 데이터가 Time Travel 로그의 `raw_data` 열 아래에 나타납니다. 자세한 내용은 Time Travel 로그 사용을 참조하세요. 기본 값은 `false`입니다. 이 옵션을 `false`로 설정하면 DML 유형만 캡처됩니다.

  타입: 문자열

  필수 항목 여부: 아니요

- `RawDataFormat`— AWS DMS 버전 3.5.0 이상에서는 `EnableRawData` 시기가 로 설정됩니다. `true` 이 속성은 Time Travel 로그의 DML 원시 데이터 형식을 지정하며 다음과 같이 표시할 수 있습니다.
  - `"TEXT"` – CDC 중에 캡처한 DML 이벤트의 구문 분석된 읽을 수 있는 열 이름과 값을 Raw 필드로 표시합니다.
  - `"HEX"` – CDC 중에 DML 이벤트에 대해 캡처한 열 이름 및 값의 원래 16진수입니다.

  이 속성은 Oracle 및 Microsoft SQL Server 데이터베이스 원본에 적용됩니다.

  타입: 문자열

  필수 항목 여부: 아니요

- `OperationsToLog` – Time Travel 로그에 로그인하기 위한 DML 작업의 유형을 지정합니다. 다음 중 한 가지를 지정할 수 있습니다.
  - `"INSERT"`
  - `"UPDATE"`
  - `"DELETE"`
  - `"COMMIT"`
  - `"ROLLBACK"`
  - `"ALL"`

  기본값은 `"ALL"`입니다.

  타입: 문자열

  필수 항목 여부: 아니요
- `MaxRecordSize` - 각 행에 기록되는 Time Travel 로그 레코드의 최대 크기를 지정합니다. 이 속성을 사용하여 특히 사용량이 많은 테이블의 Time Travel 로그 증가를 제어할 수 있습니다. 기본값은 64KB입니다.

  유형: 정수

  필수 항목 여부: 아니요

Time Travel 로그를 켜고 사용하는 방법에 관한 자세한 내용은 다음 주제를 참조하세요.

주제

- 작업의 Time Travel 로그 켜기
- Time Travel 로그 사용
- Time Travel 로그를 S3에 AWS DMS 업로드하는 빈도

작업의 Time Travel 로그 켜기

앞에서 설명한 작업 설정을 사용하여 AWS DMS 작업에 대해 시간 여행을 켤 수 있습니다. Time Travel을 켜기 전에 복제 작업이 중지되었는지 확인하세요.

를 사용하여 시간 여행을 켜려면 AWS CLI

1. DMS 작업 구성 JSON 파일을 만들고 다음과 같은 TTSettings 섹션을 추가합니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

```
 .
 .
 .
    },
"TTSettings" : {
  "EnableTT" : true,
  "TTS3Settings": {
      "EncryptionMode": "SSE_KMS",
      "ServerSideEncryptionKmsKeyId": "arn:aws:kms:us-west-2:112233445566:key/
myKMSKey",
      "ServiceAccessRoleArn": "arn:aws:iam::112233445566:role/dms-tt-s3-access-
role",
      "BucketName": "myttbucket",
      "BucketFolder": "myttfolder",
      "EnableDeletingFromS3OnTaskDelete": false
    },
  "TTRecordSettings": {
      "EnableRawData" : true,
      "OperationsToLog": "DELETE,UPDATE",
      "MaxRecordSize": 64
    },
 .
 .
 .
```

2. 적절한 작업 동작에서 --replication-task-settings 옵션을 사용하여 이 JSON 파일을 지정합니다. 예를 들어, 다음 CLI 코드 조각은 이 Time Travel 설정 파일을 create-replication-task의 일부로 지정합니다.

```
aws dms create-replication-task
--target-endpoint-arn arn:aws:dms:us-
east-1:112233445566:endpoint:ELS5O7YTYV452CAZR2EYBNQGILFHQIFVPWFRQAY \
--source-endpoint-arn arn:aws:dms:us-
east-1:112233445566:endpoint:HNX2BWIIN5ZYFF7F6UFFZVWTDFFSMTNOV2FTXZA \
--replication-instance-arn arn:aws:dms:us-
east-1:112233445566:rep:ERLHG2UA52EEJJKFYNYWRPCG6T7EPUAB5AWBUJQ \
```

```
--migration-type full-load-and-cdc --table-mappings 'file:///FilePath/
mappings.json' \
--replication-task-settings 'file:///FilePath/task-settings-tt-enabled.json' \
--replication-task-identifier test-task
                              .
                              .
                              .
```

여기서 이 Time Travel 설정 파일의 이름은 `task-settings-tt-enabled.json`입니다.

마찬가지로 이 파일을 `modify-replication-task` 동작의 일부로 지정할 수 있습니다.

다음과 같은 작업 동작에 대한 Time Travel 로그의 특수 처리에 유의하십시오.

- `start-replication-task` – 복제 작업을 실행할 때 Time Travel에 사용되는 S3 버킷에 액세스할 수 없는 경우 작업은 `FAILED`로 표시됩니다.
- `stop-replication-task`— 작업이 중지되면 현재 복제 인스턴스에 사용할 수 있는 모든 Time Travel 로그를 Time Travel에 사용되는 S3 버킷으로 AWS DMS 즉시 푸시합니다.

복제 작업이 실행되는 동안 `EncryptionMode` 값을 `"SSE_KMS"`에서 `"SSE_S3"`로 변경할 수 있지만 그 반대로 변경할 수는 없습니다.

진행 중인 작업의 Time Travel 로그 크기가 1GB를 초과하는 경우, DMS는 해당 크기에 도달한 후 5분 이내에 로그를 S3로 푸시합니다. 작업이 실행된 후 S3 버킷 또는 KMS 키에 액세스할 수 없게 되면 DMS는 이 버킷으로의 로그 푸시를 중단합니다. 로그가 S3 버킷으로 푸시되지 않는 경우 S3와 AWS KMS 권한을 확인하십시오. DMS가 이러한 로그를 S3로 푸시하는 빈도에 관한 자세한 내용은 Time Travel 로그를 S3에 AWS DMS 업로드하는 빈도을 참조하십시오.

콘솔에서 기존 작업에 대한 Time Travel을 켜려면 작업 설정에서 JSON 편집기 옵션을 사용하여 `TTSettings` 섹션을 추가하십시오.

Time Travel 로그 사용

Time Travel 로그 파일은 다음과 같은 필드가 있는 쉼표로 구분된 값(CSV) 파일입니다.

```
log_timestamp
component
dms_source_code_location
transaction_id
```

```
event_id
event_timestamp
lsn/scn
primary_key
record_type
event_type
schema_name
table_name
statement
action
result
raw_data
```

S3에서 Time Travel 로그를 사용할 수 있게 되면 Amazon Athena와 같은 도구를 사용하여 이 로그에 직접 액세스하고 쿼리할 수 있습니다. 또는 S3에서 다른 파일처럼 로그를 다운로드할 수 있습니다.

다음 예제는 `mytable`이라는 테이블의 트랜잭션이 기록되는 Time Travel 로그를 보여줍니다. 가독성을 위해 다음 로그의 줄 끝을 추가했습니다.

```
"log_timestamp ","tt_record_type","dms_source_code_location ","transaction_id",
"event_id","event_timestamp","scn_lsn","primary_key","record_type","event_type",
"schema_name","table_name","statement","action","result","raw_data"
"2021-09-23T01:03:00:778230","SOURCE_CAPTURE","postgres_endpoint_wal_engine.c:00819",
"609284109","565612992","2021-09-23 01:03:00.765321+00","00000E9C/D53AB518","","DML",
"UPDATE (3)","dmstest","mytable","","Migrate","","table dmstest.mytable:
UPDATE: id[bigint]:2244937 phone_number[character varying]:'phone-number-482'
age[integer]:82 gender[character]:'f' isactive[character]:'true '
date_of_travel[timestamp without time zone]:'2021-09-23 01:03:00.76593'
description[text]:'TEST DATA TEST DATA TEST DATA TEST DATA'"
```

Time Travel 로그를 S3에 AWS DMS 업로드하는 빈도

복제 인스턴스의 스토리지 사용량을 최소화하려면 주기적으로 Time Travel 로그를 AWS DMS 오프로드하십시오.

다음과 같은 경우 Time Travel 로그가 Amazon S3 버킷으로 푸시됩니다.

- 현재 로그 크기가 1GB를 초과하는 경우 5분 내에 S3에 로그를 AWS DMS 업로드합니다. 따라서 S3와 실행 중인 각 AWS KMS 작업에 대해 시간당 최대 12건의 호출을 할 AWS DMS 수 있습니다.
- AWS DMS 로그 크기에 관계없이 1시간마다 S3에 로그를 업로드합니다.
- 작업이 중지되면 시간 여행 로그를 S3에 AWS DMS 즉시 업로드합니다.

작업 설정 로깅

로깅은 마이그레이션 프로세스 중에 CloudWatch Amazon을 사용하여 정보를 기록합니다. 작업 설정 로깅을 사용하면 로깅할 구성 요소 활동과 로그에 작성할 정보량을 지정할 수 있습니다. 작업 설정 로깅은 JSON 파일에 작성됩니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

여러 가지 방법으로 CloudWatch 로깅을 활성화할 수 있습니다. 마이그레이션 작업을 생성할 AWS Management Console 때 에서 `EnableLogging` 옵션을 선택할 수 있습니다. 또는 AWS DMS API를 사용하여 작업을 생성할 `true` 때 `EnableLogging` 옵션을 로 설정할 수 있습니다. 작업 설정의 JSON 로깅 섹션에서 `"EnableLogging": true`를 지정할 수도 있습니다.

`EnableLogging`로 `true` 설정하면 다음과 같이 CloudWatch 그룹 이름과 스트림 이름을 AWS DMS 할당합니다. 이러한 값을 직접 설정할 수 없습니다.

- CloudWatchLogGroup: `dms-tasks-<REPLICATION_INSTANCE_IDENTIFIER>`
- CloudWatchLogStream: `dms-task-<REPLICATION_TASK_EXTERNAL_RESOURCE_ID>`

`<REPLICATION_INSTANCE_IDENTIFIER>`는 복제 인스턴스의 식별자입니다. `<REPLICATION_TASK_EXTERNAL_RESOURCE_ID>`는 태스크 ARN의 `<resourcename>` 섹션의 값입니다. 리소스 ARN AWS DMS 생성 방법에 대한 자세한 내용은 을 참조하십시오. 에 대한 아마존 리소스 이름 (ARN) 생성 AWS DMS

CloudWatch AWS Identity and Access Management (IAM) 과 통합되며 AWS 계정의 사용자가 수행할 수 있는 CloudWatch 작업을 지정할 수 있습니다. 에서 CloudWatch IAM을 사용하는 방법에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon의 자격 증명 CloudWatch 및 액세스 관리 및 Amazon CloudWatch API 호출 로깅을 참조하십시오.

작업 로그를 삭제하려면 작업 설정의 로깅 섹션 중 JSON에서 `DeleteTaskLogs`를 true로 설정하면 됩니다.

다음 유형의 이벤트에 대한 로깅을 지정할 수 있습니다.

- `FILE_FACTORY` – 파일 팩토리는 배치 적용 및 배치 로드에 사용되는 파일을 관리하고 Amazon S3 엔드포인트를 관리합니다.
- `METADATA_MANAGER` – 메타데이터 관리자는 복제 중에 소스 및 대상 메타데이터, 파티셔닝, 테이블 상태를 관리합니다.
- `SORTER` – `SORTER`는 `SOURCE_CAPTURE` 프로세스에서 들어오는 이벤트를 수신합니다. 이벤트는 트랜잭션으로 일괄 처리되어 `TARGET_APPLY` 서비스 구성 요소로 전달됩니다. `SOURCE_CAPTURE` 프

로세스에서 TARGET_APPLY 구성 요소가 이벤트를 소비하는 속도보다 더 빠르게 이벤트를 생성하는 경우, SORTER 구성 요소는 백로그된 이벤트를 디스크나 스왑 파일에 캐시합니다. 캐시된 이벤트는 복제 인스턴스에서 스토리지 부족 문제를 일으키는 공통적 원인에 속합니다.

SORTER 서비스 구성 요소는 캐시된 이벤트를 관리하고 CDC 통계를 수집하며 작업 지연 시간을 보고합니다.

- SOURCE_CAPTURE – 지속적 복제(CDC) 데이터는 소스 데이터베이스 또는 서비스에서 캡처되어 SORTER 서비스 구성 요소로 전달됩니다.
- SOURCE_UNLOAD – 전체 로드 중에 소스 데이터베이스 또는 서비스에서 데이터가 언로드됩니다.
- TABLES_MANAGER – 테이블 관리자는 캡처한 테이블을 추적하고 테이블 마이그레이션 순서를 관리하며 테이블 통계를 수집합니다.
- TARGET_APPLY – 데이터와 데이터 정의 언어(DDL) 문을 대상 데이터베이스에 적용합니다.
- TARGET_LOAD – 데이터는 대상 데이터베이스로 로드됩니다.
- TASK_MANAGER – 작업 관리자는 실행 중인 작업을 관리하고 병렬 데이터 처리를 위해 작업을 하위 작업으로 분류합니다.
- TRANSFORMATION – 테이블 매핑 변환 이벤트. 자세한 정보는 작업 설정을 지정하기 위한 테이블 매핑 사용을 참조하세요.
- VALIDATOR/ VALIDATOR_EXT – VALIDATOR 서비스 구성 요소는 데이터가 소스에서 타겟으로 정확하게 마이그레이션되었는지 확인합니다. 자세한 정보는 데이터 유효성 검사을 참조하세요.

다음 로깅 구성 요소는 LOGGER_SEVERITY_DETAILED_DEBUG 로그 심각도 수준을 사용할 때 대량의 로그를 생성합니다.

- COMMON
- ADDONS
- DATA_STRUCTURE
- COMMUNICATION
- FILE_TRANSFER
- FILE_FACTORY

문제 해결 중에 이러한 구성 요소에 대해 DEFAULT 이외의 로깅 수준은 거의 필요하지 않습니다. AWS Support에서 DEFAULT 특별히 요청하지 않는 한 이러한 구성 요소의 로깅 수준을 변경하지 않는 것이 좋습니다.

위의 항목 중 하나를 지정한 후 다음 목록과 같이 기록되는 정보의 양을 지정할 수 있습니다.

심각도 수준이 가장 낮은 정보에서 가장 높은 수준의 정보 순으로 표시됩니다. 높은 수준은 항상 낮은 수준의 정보를 포함합니다.

- LOGGER_SEVERITY_ERROR – 오류 메시지가 로그에 작성됩니다.
- LOGGER_SEVERITY_WARNING – 경고와 오류 메시지가 로그에 작성됩니다.
- LOGGER_SEVERITY_INFO – 정보 메시지, 경고 및 오류 메시지가 로그에 작성됩니다.
- LOGGER_SEVERITY_DEFAULT – 정보 메시지, 경고 및 오류 메시지가 로그에 작성됩니다.
- LOGGER_SEVERITY_DEBUG – 디버그 메시지, 정보 메시지, 경고 및 오류 메시지가 로그에 작성됩니다.
- LOGGER_SEVERITY_DETAILED_DEBUG – 모든 정보가 로그에 작성됩니다.

다음 JSON 예제에서는 모든 작업 및 심각도 수준을 로깅하기 위한 작업 설정을 보여줍니다.

```
…
  "Logging": {
    "EnableLogging": true,
    "LogComponents": [
      {
        "Id": "FILE_FACTORY",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
      },{
        "Id": "METADATA_MANAGER",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
      },{
        "Id": "SORTER",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
      },{
        "Id": "SOURCE_CAPTURE",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
      },{
        "Id": "SOURCE_UNLOAD",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
      },{
        "Id": "TABLES_MANAGER",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
      },{
        "Id": "TARGET_APPLY",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
```

```
      },{
        "Id": "TARGET_LOAD",
        "Severity": "LOGGER_SEVERITY_INFO"
      },{
        "Id": "TASK_MANAGER",
        "Severity": "LOGGER_SEVERITY_DEBUG"
      },{
        "Id": "TRANSFORMATION",
        "Severity": "LOGGER_SEVERITY_DEBUG"
      },{
        "Id": "VALIDATOR",
        "Severity": "LOGGER_SEVERITY_DEFAULT"
      }
    ],
    "CloudWatchLogGroup": null,
    "CloudWatchLogStream": null
  },
…
```

제어 테이블 작업 설정

컨트롤 테이블은 AWS DMS 작업에 대한 정보를 제공합니다. 또한 현재 마이그레이션 작업과 향후 작업 모두를 계획하고 관리하는 데 사용할 수 있는 유용한 통계가 나와 있습니다. JSON 파일에 이러한 작업 설정을 적용하거나 AWS DMS 콘솔의 작업 생성 페이지에서 고급 설정을 선택하여 적용할 수 있습니다. 예외 적용 테이블(`dmslogs.awsdms_apply_exceptions`)은 데이터베이스 대상에 항상 생성됩니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

AWS DMS Full Load + CDC 또는 CDC 전용 작업 중에만 제어 테이블을 생성하고 전체 로드 전용 작업 중에는 생성하지 않습니다.

전체 로드 및 CDC(기존 데이터 마이그레이션 및 진행 중인 변경 사항 복제) 및 CDC 전용(데이터 변경 사항만 복제) 작업의 경우, 다음을 비롯한 추가 테이블을 만들 수도 있습니다.

- 복제 상태(dmslogs.awsdms_status) – 이 테이블은 현재 작업에 대한 세부 정보를 제공합니다. 작업 상태, 작업에서 사용하는 메모리 크기, 대상에 아직 적용되지 않은 변경 수 등이 표시됩니다. 또한 이 표에는 원본 데이터베이스에서 AWS DMS 현재 읽고 있는 위치도 표시됩니다. 또한 이것은 작업이 전체 로드 단계이거나 변경 데이터 캡처(CDC)인지 여부를 나타냅니다.
- 일시 중지된 테이블(dmslogs.awsdms_suspended_tables) – 이 테이블에는 일시 중지된 테이블뿐 아니라 일시 중지된 이유가 나와 있습니다.

- 복제 기록(dmslogs.awsdms_history) – 이 테이블은 복제 기록에 관한 정보를 제공합니다. 작업 중에 처리된 레코드의 수와 볼륨, CDC 작업 종료 시 지연 시간, 기타 통계 등이 표시됩니다.

예외 적용 테이블(`dmslogs.awsdms_apply_exceptions`)에는 다음 파라미터가 포함되어 있습니다.

열	유형	설명
TASK_NAME	nvchar	AWS DMS 작업의 리소스 ID. 리소스 ID는 태스크 ARN에서 찾을 수 있습니다.
TABLE_OWNER	nvchar	테이블 소유자입니다.
TABLE_NAME	nvchar	테이블 이름.
ERROR_TIME	타임스탬프	예외(오류)가 발생한 시간입니다.
STATEMENT	nvchar	오류가 발생했을 때 실행되고 있었던 문입니다.
ERROR	nvchar	오류 이름 및 설명입니다.

복제 상태 테이블(`dmslogs.awsdms_status`)에는 작업과 대상 데이터베이스의 현재 상태가 포함되어 있습니다. 설정은 다음과 같습니다.

열	유형	설명
SERVER_NAME	nvchar	복제 작업이 실행되고 있는 시스템의 이름입니다.
TASK_NAME	nvchar	AWS DMS 작업의 리소스 ID. 리소스 ID는 태스크 ARN에서 찾을 수 있습니다.
TASK_STATUS	varchar	다음 값 중 하나입니다. • FULL LOAD

열	유형	설명
		• CHANGE PROCESSING (CDC) • 실행 중이 아님 전체 로드에 최소 1개 이상의 테이블이 있는 한 작업 상태가 FULL LOAD로 설정됩니다. 모든 테이블이 로드된 후 CDC가 활성화되어 있으면 작업 상태가 CHANGE PROCESSING으로 바뀝니다. 작업을 시작하기 전이나 작업이 완료된 후에 작업이 NOT RUNNING으로 설정됩니다.
STATUS_TIME	타임스탬프	작업 상태의 타임스탬프.
PENDING_CHANGES	int	소스 데이터베이스에서 커밋되고 복제 인스턴스의 메모리와 디스크에 캐시된 변경 레코드의 수입니다.
DISK_SWAP_SIZE	int	이전 또는 오프로드된 트랜잭션에서 사용하는 디스크 공간 크기입니다.
TASK_MEMORY	int	사용된 현재 메모리(단위: MB).
SOURCE_CURRENT_POSITION	varchar	현재 읽고 있는 원본 데이터베이스에서의 위치. AWS DMS
SOURCE_CURRENT_TIMESTAMP	타임스탬프	현재 읽고 있는 원본 데이터베이스의 AWS DMS 타임스탬프입니다.
SOURCE_TAIL_POSITION	varchar	커밋되지 않은 가장 오래된 시작 트랜잭션의 위치입니다. 이 값은 변경 사항을 잃지 않고도 되돌릴 수 있는 가장 최근 위치입니다.

열	유형	설명
SOURCE_TAIL _TIMESTAMP	타임스탬프	커밋되지 않은 가장 오래된 시작 트랜잭션의 타임스탬프입니다. 이 값은 변경 사항을 잃지 않고도 되돌릴 수 있는 가장 최근 타임스탬프입니다.
SOURCE_TI MESTAMP _APPLIED	타임스탬프	마지막 트랜잭션의 타임스탬프가 커밋됩니다. 대량 적용 프로세스에서 이 값은 대량으로 마지막 트랙잭션을 커밋하기 위한 타임스탬프입니다.

일시 중단된 테이블(`dmslogs.awsdms_suspended_tables`)에는 다음과 같은 파라미터가 포함되어 있습니다.

열	유형	설명
SERVER_NAME	nvchar	복제 작업이 실행되고 있는 시스템의 이름입니다.
TASK_NAME	nvchar	태스크의 AWS DMS 이름
TABLE_OWNER	nvchar	테이블 소유자입니다.
TABLE_NAME	nvchar	테이블 이름.
SUSPEND_REASON	nvchar	일시 중지 사유.
SUSPEND_T IMESTAMP	타임스탬프	일시 중단이 발생한 시간입니다.

복제 기록 테이블(`dmslogs.awsdms_history`)에는 다음 파라미터가 포함되어 있습니다.

열	유형	설명
SERVER_NAME	nvchar	복제 작업이 실행되고 있는 시스템의 이름입니다.
TASK_NAME	nvchar	AWS DMS 작업의 리소스 ID. 리소스 ID는 태스크 ARN에서 찾을 수 있습니다.
TIMESLOT_TYPE	varchar	다음 값 중 하나입니다. • FULL LOAD • CHANGE PROCESSING (CDC) 작업이 전체 로드와 CDC를 실행하고 있는 경우, 두 이력 레코드는 타임 슬롯에 작성됩니다.
TIMESLOT	타임스탬프	시간 슬롯의 종료 타임스탬프입니다.
TIMESLOT_DURATION	int	시간 슬롯의 지속 시간(분)입니다.
TIMESLOT_LATENCY	int	시간 슬롯이 끝날 때의 대상 지연 시간(초)입니다. 이 값은 CDC 시간 슬롯에만 적용됩니다.
RECORDS	int	시간 슬롯 중에 처리되는 레코드의 수입니다.
TIMESLOT_VOLUME	int	처리되는 데이터의 볼륨(단위: MB)입니다.

검증 실패 테이블(`awsdms_validation_failures_v1`)에는 작업에 대한 모든 데이터 검증 실패가 포함되어 있습니다. 자세한 내용은 데이터 검증 문제 해결을 참조하십시오.

추가 제어 테이블 설정에는 다음이 포함됩니다.

- `HistoryTimeslotInMinutes` – 이 옵션을 사용하여 복제 이력 테이블에서 각 시간 슬롯의 길이를 나타냅니다. 기본값은 5분입니다.
- `ControlSchema`— 이 옵션을 사용하면 AWS DMS 대상의 제어 테이블에 대한 데이터베이스 스키마 이름을 지정할 수 있습니다. 이 옵션에 대한 정보를 입력하지 않으면 테이블이 다음과 같이 데이터베이스의 기본 위치로 복사됩니다.
  - PostgreSQL, 퍼블릭
  - Oracle, 대상 스키마
  - Microsoft SQL 서버, 대상 데이터베이스에 dbo
  - MySQL, awsdms_control
  - MariaDB, awsdms_control
  - Amazon Redshift, 퍼블릭
  - DynamoDB, 데이터베이스에 개별 테이블로 생성됨
  - IBM Db2 LUW, awsdms_control

스트림 버퍼 작업 설정

를 사용하여 다음을 AWS CLI포함하여 스트림 버퍼 설정을 지정할 수 있습니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

- `StreamBufferCount` – 이 옵션을 사용하여 마이그레이션 작업을 위한 데이터 스트림 버퍼의 수를 지정합니다. 기본 스트림 버퍼 개수는 3개입니다. 이 설정의 값이 증가하면 데이터 추출 속도가 증가할 수 있습니다. 그렇지만 이 성능 증가는 원본 시스템과 복제 서버의 인스턴스 클래스를 포함한 마이그레이션 환경에 따라 크게 달라집니다. 대부분의 상황에서는 기본값으로 충분합니다.
- `StreamBufferSizeInMB` – 이 옵션을 사용하여 각 데이터 스트림 버퍼의 최대 크기를 나타냅니다. 기본 크기는 8MB입니다. 매우 큰 LOB를 사용하는 경우 이 옵션의 값을 늘려야 할 수 있습니다. 로그 파일에 스트림 버퍼 크기가 부족하다는 메시지가 표시될 경우 이 값을 증가시켜야 할 수도 있습니다. 이 옵션의 크기를 계산하려면 다음 수식을 사용합니다. `[Max LOB size (or LOB chunk size)]*[number of LOB columns]*[number of stream buffers]*[number of tables loading in parallel per task(MaxFullLoadSubTasks)]*3`
- `CtrlStreamBufferSizeInMB` – 이 옵션을 사용하여 제어 스트림 버퍼의 크기를 설정합니다. 값의 단위는 메가바이트이고 범위는 1~8입니다. 기본값은 5입니다. 예를 들어, 수천 개에 이르는 많은 테이블을 사용할 경우 이 값을 증가시켜야 할 수도 있습니다.

변경 처리 튜닝 설정

다음 설정은 변경 데이터 캡처 (CDC) 중에 대상 테이블의 변경을 AWS DMS 처리하는 방법을 결정합니다. 이 설정 중 몇 개는 대상 메타데이터 파라미터 `BatchApplyEnabled`의 값에 따라 달라집니다. `BatchApplyEnabled` 파라미터에 관한 자세한 내용은 대상 메타데이터 작업 설정를 참조하십시오. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

변경 처리 튜닝 설정에는 다음이 포함됩니다.

다음 설정은 대상 메타데이터 파라미터 `BatchApplyEnabled`가 `true`로 설정되어 있는 경우에만 적용됩니다.

- `BatchApplyPreserveTransaction` – `true`로 설정하면 트랜잭션 무결성이 유지되고 배치에서는 원본의 트랜잭션 내에 있는 모든 변경 사항이 포함됩니다. 기본 값은 `true`입니다. 이 설정은 Oracle 대상 엔드포인트에만 적용됩니다.

  `false`로 설정되면 트랜잭션 무결성이 일시적인 시간 경과가 있어 성능이 개선될 수 있습니다. 원본 트랜잭션의 모든 변경 사항이 단일한 배치(batch)의 대상에 적용된다는 보장은 없습니다.

  기본적으로 은 트랜잭션 모드에서 변경을 AWS DMS 처리하므로 트랜잭션 무결성이 유지됩니다. 트랜잭션 무결성이 일시적으로 저하될 여지가 있다면 배치 최적화 적용 옵션을 대신 사용할 수 있습니다. 이 옵션은 효율성을 위해 트랜잭션을 효율적으로 그룹화하고 일괄적으로 적용합니다. 배치 최적화 적용 옵션을 사용하면 거의 항상 참조 무결성 제약 조건을 위반합니다. 따라서 마이그레이션 프로세스 중에 이러한 제약 조건을 해제한 다음, 전환 프로세스의 일부로 다시 설정하는 것이 좋습니다.
- `BatchApplyTimeoutMin`— 각 일괄 변경 적용 사이에 AWS DMS 대기하는 최소 시간 (초) 을 설정합니다. 기본값은 1입니다.
- `BatchApplyTimeoutMax`— 각 일괄 변경 적용 사이에 시간이 초과되기 전에 AWS DMS 대기하는 최대 시간 (초) 을 설정합니다. 기본값은 30입니다.
- `BatchApplyMemoryLimit` – 배치 최적화 적용 모드에서 사전 처리를 위해 사용하는 최대 메모리 양(MB)을 설정합니다. 기본값은 500입니다.
- `BatchSplitSize` – 단일 배치에서 적용되는 변경 사항의 최대 수를 설정합니다. 기본값이 0이면 적용되는 제한이 없음을 뜻합니다.

다음 설정은 대상 메타데이터 파라미터 `BatchApplyEnabled`가 `false`로 설정되어 있는 경우에만 적용됩니다.

- `MinTransactionSize` – 각 트랜잭션에 포함할 변경 사항의 최소 수를 설정합니다. 기본값은 1000입니다.
- `CommitTimeout`— 제한 시간을 선언하기 전에 트랜잭션을 일괄적으로 수집하는 최대 시간을 초 단위로 설정합니다. AWS DMS 기본값은 1입니다.

양방향 복제의 경우, 다음 설정은 대상 메타데이터 파라미터 `BatchApplyEnabled`가 `false`로 설정되어 있는 경우에만 적용됩니다.

- `LoopbackPreventionSettings` – 이러한 설정은 양방향 복제와 관련된 작업 쌍으로 된 지속적 복제 작업마다 루프백 방지를 제공합니다. 루프백 방지는 동일한 변경 사항이 양방향 복제의 두 방향 모두에 적용되어 데이터가 손상되는 문제를 방지합니다. 양방향 복제에 관한 자세한 내용은 양방향 복제 수행 단원을 참조하십시오.

AWS DMS 트랜잭션이 소스, 타겟 또는 둘 다에 완전히 커밋될 때까지 트랜잭션 데이터를 메모리에 보관하려고 시도합니다. 그렇지만 할당된 메모리보다 크거나 지정된 시간 한도 내에서 커밋되지 않은 트랜잭션은 디스크에 기록됩니다.

다음 설정은 변경 처리 모드에 상관 없이 변경 처리 튜닝에 적용됩니다.

- `MemoryLimitTotal` – 디스크에 기록되기 전에 모든 트랜잭션이 메모리에서 점유할 수 있는 최대 크기(MB 단위)를 설정합니다. 기본값은 1,024입니다.
- `MemoryKeepTime` – 디스크에 기록되기 전에 각 트랜잭션이 메모리에 유지될 수 있는 최대 시간(초)을 설정합니다. 기간은 트랜잭션 캡처를 AWS DMS 시작한 시간부터 계산됩니다. 기본값은 60입니다.
- `StatementCacheSize` – 변경 사항을 대상에 적용할 때 나중에 실행할 수 있도록 서버에서 저장하는 준비된 문의 최대 수를 설정합니다. 기본값은 50입니다. 최대값은 200입니다.

다음은 변경 처리 튜닝을 처리하는 작업 설정이 작업 설정 JSON 파일에 표시되는 방식의 한 가지 예에 속합니다.

```
"ChangeProcessingTuning": {
        "BatchApplyPreserveTransaction": true,
        "BatchApplyTimeoutMin": 1,
        "BatchApplyTimeoutMax": 30,
        "BatchApplyMemoryLimit": 500,
        "BatchSplitSize": 0,
        "MinTransactionSize": 1000,
```

```
        "CommitTimeout": 1,
        "MemoryLimitTotal": 1024,
        "MemoryKeepTime": 60,
        "StatementCacheSize": 50
}
```

데이터 복제 작업 중에 Amazon S3 대상에 대한 쓰기 빈도를 제어하기 위해 `cdcMaxBatchInterval` 및 `cdcMinFileSize` 추가 연결 속성을 구성할 수 있습니다. 따라서 추가 오버헤드 작업 없이 데이터를 분석할 때 성능이 향상될 수 있습니다. 자세한 정보는 Amazon S3를 AWS DMS의 대상으로 사용 시 엔드포인트 설정을 참조하세요.

데이터 검증 작업 설정

원본에서 대상으로 데이터가 정확히 마이그레이션되었는지 확인할 수 있습니다. 작업에 대한 검증을 활성화하면 테이블에 대해 전체 로드가 수행된 직후에 소스와 대상 데이터를 AWS DMS 비교하기 시작합니다. 작업 데이터 검증, 요구 사항, 데이터베이스 지원 범위, 보고 대상 지표에 관한 자세한 내용은 AWS DMS 데이터 검증 단원을 참조하십시오. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

데이터 검증 설정과 그 값에는 다음 사항이 포함됩니다.

- `EnableValidation` – true로 설정한 경우 데이터 검증을 활성화합니다. 그렇지 않으면 작업에 대한 검증은 비활성화됩니다. 기본값은 false입니다.
- `ValidationMode` – DMS가 대상 테이블의 데이터를 원본 테이블과 비교하여 검증하는 방법을 제어합니다. AWS DMS 는 향후 확장성을 위해 이 설정을 제공합니다. 현재 유효한 기본값은 뿐입니다`ROW_LEVEL`. AWS DMS 소스 테이블과 대상 테이블 사이의 모든 행을 검증합니다.
- `FailureMaxCount` – 작업에 대한 검증이 일시 중지되기 전에 검증에 실패할 수 있는 레코드의 최대 수를 지정합니다. 기본값은 10,000입니다. 검증에 실패하는 레코드 수와 상관없이 검증을 계속하려면 이 값을 원본의 레코드 수보다 더 높게 설정합니다.
- `HandleCollationDiff` – 이 옵션이 `true`로 설정되면 비교할 소스 및 대상 레코드 식별 시 검증은 PostgreSQL 및 Microsoft SQL Server 엔드포인트의 열 기준 데이터 정렬 차이를 고려합니다. 그렇지 않은 경우 열 기준 데이터의 이러한 차이는 검증에서 무시됩니다. 열 기준 데이터 정렬은 데이터 검증에 중요한 행의 순서를 지정할 수 있습니다. `HandleCollationDiff`를 true로 설정하면 이러한 데이터 정렬 차이가 자동으로 해결되고 데이터 검증에서 거짓 양성이 방지됩니다. 기본 값은 `false`입니다.
- `RecordFailureDelayInMinutes` – 검증 실패 세부 정보를 보고하기 전에 지연 시간을 분 단위로 지정합니다.

- `RecordFailureDelayLimitInMinutes` – 검증 실패 세부 정보를 보고하기 전에 지연 시간을 지정합니다. 일반적으로 AWS DMS 는 작업 지연 시간을 사용하여 변경 사항이 대상에 이르는 실제 지연 시간을 인식함으로써 거짓 양성을 방지합니다. 이러한 설정을 통해 실제 지연 값이 재정의되고, 검증 지표를 보고하기 전에 지연을 더 높게 설정할 수 있습니다. 기본값은 0입니다.
- `RecordSuspendDelayInMinutes` – `FailureMaxCount`에 설정된 오류 임계값으로 인해 테이블 검증이 일시 중단될 때까지의 지연 시간(분)을 지정합니다.
- `SkipLobColumns`— 이 옵션을 로 설정하면 작업 `true` AWS DMS 검증의 테이블 부분에 있는 모든 LOB 열에 대한 데이터 검증을 건너뛰습니다. 기본 값은 `false`입니다.
- `TableFailureMaxCount` – 테이블에 대한 검증이 일시 중지되기 전에 하나의 테이블에서 검증에 실패할 수 있는 행의 최대 수를 지정합니다. 기본값은 1,000입니다.
- `ThreadCount`— 검증 중에 AWS DMS 사용하는 실행 스레드 수를 지정합니다. 각 스레드는 소스와 타겟에서 not-yet-validated 데이터를 선택하여 비교 및 검증합니다. 기본값은 5입니다. 더 큰 수로 설정하면 `ThreadCount` 검증을 더 빨리 AWS DMS 완료할 수 있습니다. 하지만 AWS DMS 는 동시 쿼리를 더 많이 실행하여 소스 및 대상에서 더 많은 리소스를 사용합니다.
- `ValidationOnly` – 이 옵션이 `true`로 설정되면 작업을 실행하는 경우 데이터 마이그레이션이나 복제를 수행하지 않고 데이터 검증을 미리 봅니다. 기본 값은 `false`입니다. 작업이 생성된 후에는 이 `ValidationOnly` 설정을 수정할 수 없습니다.

  `DO_NOTHING`(검증 전용 작업의 기본값) 로 설정하고 마이그레이션 유형을 다음 중 하나로 설정해야 합니다. TargetTablePrepMode

  - 전체 로드 - AWS DMS 콘솔에서 기존 데이터를 마이그레이션하도록 작업 마이그레이션 유형을 설정합니다. 또는 AWS DMS API에서 마이그레이션 유형을 FULL-LOAD로 설정합니다.
  - CDC - AWS DMS 콘솔에서 마이그레이션 유형 작업을 데이터 변경만 복제로 설정합니다. 또는 AWS DMS API에서 마이그레이션 유형을 CDC로 설정합니다.

  선택한 마이그레이션 유형에 관계없이 검증 전용 작업 중에는 데이터가 실제로 마이그레이션되거나 복제되지 않습니다.

  자세한 정보는 검증 전용 태스크을 참조하세요.

  > **Important**
  >
  > `ValidationOnly` 설정은 변경할 수 없습니다. 작업이 생성된 후에는 해당 작업에 대해 수정할 수 없습니다.

- `ValidationPartialLobSize` – 열에 저장된 모든 데이터를 검증하는 대신 LOB 열에 대한 부분 검증을 수행할지 여부를 지정합니다. 이는 전체 LOB 데이터 집합이 아닌 일부 LOB 데이터만 마이그레이션할 때 유용할 수 있습니다. 값의 단위는 KB입니다. 기본값은 0이며, 이는 AWS DMS 가 모든 LOB 열 데이터를 검증함을 의미합니다. 예를 들어 원본과 대상 모두에서 열 데이터의 처음 AWS DMS 32KB만 `"ValidationPartialLobSize": 32` 검증한다는 의미입니다.
- `PartitionSize` – 원본과 대상 모두에서 비교하기 위해 읽을 레코드의 배치 크기를 지정합니다. 기본값은 10,000입니다.
- `ValidationQueryCdcDelaySeconds` – 각 CDC 업데이트에 대해 원본과 대상 모두에서 첫 번째 유효성 검사 쿼리가 지연되는 시간입니다. 이렇게 하면 마이그레이션 지연 시간이 길어질 때 리소스 경합을 줄이는 데 도움이 될 수 있습니다. 검증 전용 작업에서는 이 옵션이 180초로 자동 설정됩니다. 기본값은 0입니다.

예를 들어 다음 JSON은 스레드의 기본 수를 두 배로 늘려 데이터 검증을 활성합니다. 또한 PostgreSQL 엔드포인트에서 열 기준 데이터 정렬 차이로 야기된 레코드 순서의 차이도 고려합니다. 또한 검증 실패를 처리할 추가 시간을 고려하도록 검증 보고 지연을 제공합니다.

```
"ValidationSettings": {
     "EnableValidation": true,
     "ThreadCount": 10,
     "HandleCollationDiff": true,
     "RecordFailureDelayLimitInMinutes": 30
  }
```

> **Note**
>
> 오라클 엔드포인트의 경우 DBMS_CRYPTO AWS DMS 를 사용하여 BLOB의 유효성을 검사합니다. Oracle 엔드포인트에서 BLOB이 사용되는 경우, Oracle 엔드포인트에 액세스하는 사용자 계정에 DBMS_CRYPTO에 대한 `execute` 권한을 부여해야 합니다. 이 작업을 수행하려면 다음 문을 실행하세요.
>
> ```
> grant execute on sys.dbms_crypto to dms_endpoint_user;
> ```

변경 처리 DDL을 다루기 위한 작업 설정

다음 설정은 변경 데이터 캡처 (CDC) 중에 대상 테이블에 대한 데이터 정의어 (DDL) 변경을 AWS DMS 처리하는 방법을 결정합니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

변경 처리 DDL을 처리하는 작업 설정에는 다음이 포함됩니다.

- `HandleSourceTableDropped` – 이 옵션을 `true`로 설정하면 원본 테이블을 삭제할 때 대상 테이블을 삭제합니다.
- `HandleSourceTableTruncated` – 이 옵션을 `true`로 설정하면 원본 테이블을 자를 때 대상 테이블을 자릅니다.
- `HandleSourceTableAltered` – 원본 테이블을 변경할 때 대상 테이블을 변경하려면 이 옵션을 `true`로 설정합니다.

다음은 변경 처리 DDL을 처리하는 작업 설정이 작업 설정 JSON 파일에 표시되는 방식의 예입니다.

```
"ChangeProcessingDdlHandlingPolicy": {
   "HandleSourceTableDropped": true,
   "HandleSourceTableTruncated": true,
   "HandleSourceTableAltered": true
},
```

> **Note**
>
> 특정 엔드포인트에서 지원되는 DDL 문에 관한 자세한 내용은 이 엔드포인트에 대한 설명이 나오는 주제를 참조하십시오.

문자 대체 작업 설정

AWS DMS `STRING`또는 `WSTRING` 데이터 유형의 모든 원본 데이터베이스 열에 대해 복제 작업이 대상 데이터베이스에서 문자 대체를 수행하도록 지정할 수 있습니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

다음 원본 및 대상 데이터베이스에서 엔드포인트가 있는 모든 작업에 대한 문자 대체를 구성할 수 있습니다.

- 소스 데이터베이스:

  - Oracle
  - Microsoft SQL Server
  - MySQL
  - PostgreSQL
  - SAP Adaptive Server Enterprise(ASE)
  - IBM Db2 LUW
- 대상 데이터베이스:
  - Oracle
  - Microsoft SQL Server
  - MySQL
  - PostgreSQL
  - SAP Adaptive Server Enterprise(ASE)
  - Amazon Redshift

작업 설정에서 CharacterSetSettings 파라미터를 사용하여 문자 대체를 지정할 수 있습니다. 이 문자 대체는 16진수 표기법으로 유니코드 코드 포인트 값을 사용하여 지정된 문자에 대해 발생합니다. 두 단계 모두 지정된 경우 두 단계를 다음 순서로 사용하여 문자 대체를 구현할 수 있습니다.

1. 개별 문자 교체 — 소스에서 선택한 문자 값을 대상에 있는 해당 문자의 지정된 대체 값으로 바꿀 AWS DMS 수 있습니다. CharacterSetSettings에서 CharacterReplacements 배열을 사용하여 지정한 유니코드 코드 포인트가 있는 모든 소스 문자를 선택합니다. 또한 이 배열을 사용하여 대상의 해당 문자에 대한 대체 코드 포인트를 지정합니다.

   지정된 코드 포인트가 있는 소스의 모든 문자를 선택하려면 CharacterReplacements 배열에서 SourceCharacterCodePoint의 인스턴스를 해당 코드 포인트로 설정합니다. 그런 다음 이 배열에서 TargetCharacterCodePoint의 해당 인스턴스를 설정하여 동등한 모든 대상 문자에 대한 대체 코드 포인트를 지정합니다. 대상 문자를 바꾸지 않고 삭제하려면 TargetCharacterCodePoint의 해당 인스턴스를 0으로 설정합니다. CharacterReplacements 배열에서 SourceCharacterCodePoint 및 TargetCharacterCodePoint 설정의 추가 쌍을 지정하여 원하는 만큼 서로 다른 값의 대상 문자를 바꾸거나 삭제할 수 있습니다. SourceCharacterCodePoint의 여러 인스턴스에 동일한 값을 지정하는 경우, TargetCharacterCodePoint의 마지막 해당 설정 값이 대상에 적용됩니다.

   예를 들어 CharacterReplacements에 다음과 같은 값을 지정한다고 가정하겠습니다.

```
"CharacterSetSettings": {
    "CharacterReplacements": [ {
        "SourceCharacterCodePoint": 62,
        "TargetCharacterCodePoint": 61
        }, {
        "SourceCharacterCodePoint": 42,
        "TargetCharacterCodePoint": 41
        }
    ]
}
```

이 예제에서는 대상의 소스 코드 포인트 16진수 값 62인 모든 문자를 코드 포인트 값 61의 문자로 AWS DMS 바꿉니다. 또한 대상의 소스 코드 포인트 42인 모든 문자를 코드 포인트 값이 41인 문자로 AWS DMS 바꿉니다. 다시 말해, AWS DMS 는 대상에 있는 문자 `'b'`의 모든 인스턴스를 문자 `'a'`로 바꿉니다. 마찬가지로 대상에 있는 문자의 모든 인스턴스를 문자로 AWS DMS `'B'` 바꿉니다. `'A'`

2. 문자 집합 유효성 검사 및 교체 — 개별 문자 교체가 완료된 후 지정한 단일 문자 집합에서 모든 대상 문자에 유효한 유니코드 코드 포인트가 있는지 확인할 AWS DMS 수 있습니다. `CharacterSetSettings`에서 `CharacterSetSupport`를 사용하여 이 대상 문자 검증 및 수정을 구성할 수 있습니다. 검증 문자 집합을 지정하려면 `CharacterSetSupport`의 `CharacterSet`를 문자 집합의 문자열 값으로 설정합니다. (`CharacterSet`에 대해 가능한 값은 다음과 같습니다.) 다음 방법 중 하나로 잘못된 대상 문자를 AWS DMS 수정할 수 있습니다.

   - 현재 코드 포인트에 관계없이 유효하지 않은 모든 대상 문자에 대해 단일 대체 유니코드 코드 포인트를 지정합니다. 이 대체 코드 포인트를 구성하려면 `CharacterSetSupport`의 `ReplaceWithCharacterCodePoint`를 지정된 값으로 설정합니다.
   - `ReplaceWithCharacterCodePoint`를 0으로 설정하여 유효하지 않은 모든 대상 문자의 삭제를 구성합니다.

   예를 들어 `CharacterSetSupport`에 다음과 같은 값을 지정한다고 가정하겠습니다.

```
"CharacterSetSettings": {
    "CharacterSetSupport": {
        "CharacterSet": "UTF16_PlatformEndian",
        "ReplaceWithCharacterCodePoint": 0
    }
}
```

이 예제에서는 대상에서 발견된 문자 중 "UTF16_PlatformEndian" 문자 집합에서 유효하지 않은 문자를 모두 AWS DMS 삭제합니다. 따라서 16진수 값 2FB6으로 지정된 모든 문자가 삭제됩니다. 이 값은 4바이트 유니코드 코드 포인트이고 UTF16 문자 집합은 2바이트 코드 포인트 문자만 수락하므로 유효하지 않습니다.

> **Note**
>
> 복제 작업은 테이블 매핑을 통해 지정한 전역 또는 테이블 수준 변환을 시작하기 전에 지정된 문자 대체를 모두 완료합니다. 테이블 매핑에 관한 자세한 내용은 작업 설정을 지정하기 위한 테이블 매핑 사용 섹션을 참조하십시오.
>
> 문자 대체는 LOB 데이터 유형을 지원하지 않습니다. 여기에는 DMS가 LOB 데이터 유형으로 간주하는 모든 데이터 유형이 포함됩니다. 예를 들어, Oracle의 Extended 데이터 유형은 LOB로 간주됩니다. 소스 데이터 유형에 대한 자세한 내용은 Oracle용 소스 데이터 형식 섹션을 참조하세요.

for를 AWS DMS 지원하는 값은 다음 표에 CharacterSet 나와 있습니다.

UTF-8	ibm-860_P100-1995	ibm-280_P100-1995
UTF-16	ibm-861_P100-1995	ibm-284_P100-1995
UTF-16BE	ibm-862_P100-1995	ibm-285_P100-1995
UTF-16LE	ibm-863_P100-1995	ibm-290_P100-1995
UTF-32	ibm-864_X110-1999	ibm-297_P100-1995
UTF-32BE	ibm-865_P100-1995	ibm-420_X120-1999
UTF-32LE	ibm-866_P100-1995	ibm-424_P100-1995
UTF16_PlatformEndian	ibm-867_P100-1998	ibm-500_P100-1995
UTF16_OppositeEndian	ibm-868_P100-1995	ibm-803_P100-1999
UTF32_PlatformEndian	ibm-869_P100-1995	ibm-838_P100-1995

UTF32_OppositeEndian	ibm-878_P100-1996	ibm-870_P100-1995
UTF-16BE,version=1	ibm-901_P100-1999	ibm-871_P100-1995
UTF-16LE,version=1	ibm-902_P100-1999	ibm-875_P100-1995
UTF-16,version=1	ibm-922_P100-1999	ibm-918_P100-1995
UTF-16,version=2	ibm-1168_P100-2002	ibm-930_P120-1999
UTF-7	ibm-4909_P100-1999	ibm-933_P110-1995
IMAP-mailbox-name	ibm-5346_P100-1998	ibm-935_P110-1999
SCSU	ibm-5347_P100-1998	ibm-937_P110-1999
BOCU-1	ibm-5348_P100-1997	ibm-939_P120-1999
CESU-8	ibm-5349_P100-1998	ibm-1025_P100-1995
ISO-8859-1	ibm-5350_P100-1998	ibm-1026_P100-1995
US-ASCII	ibm-9447_P100-2002	ibm-1047_P100-1995
gb18030	ibm-9448_X100-2005	ibm-1097_P100-1995
ibm-912_P100-1995	ibm-9449_P100-2002	ibm-1112_P100-1995
ibm-913_P100-2000	ibm-5354_P100-1998	ibm-1122_P100-1999
ibm-914_P100-1995	ibm-1250_P100-1995	ibm-1123_P100-1995
ibm-915_P100-1995	ibm-1251_P100-1995	ibm-1130_P100-1997
ibm-1089_P100-1995	ibm-1252_P100-2000	ibm-1132_P100-1998
ibm-9005_X110-2007	ibm-1253_P100-1995	ibm-1137_P100-1999
ibm-813_P100-1995	ibm-1254_P100-1995	ibm-4517_P100-2005
ibm-5012_P100-1999	ibm-1255_P100-1995	ibm-1140_P100-1997

ibm-916_P100-1995	ibm-5351_P100-1998	ibm-1141_P100-1997
ibm-920_P100-1995	ibm-1256_P110-1997	ibm-1142_P100-1997
iso-8859_10-1998	ibm-5352_P100-1998	ibm-1143_P100-1997
iso-8859_11-2001	ibm-1257_P100-1995	ibm-1144_P100-1997
ibm-921_P100-1995	ibm-5353_P100-1998	ibm-1145_P100-1997
iso-8859_14-1998	ibm-1258_P100-1997	ibm-1146_P100-1997
ibm-923_P100-1998	macos-0_2-10.2	ibm-1147_P100-1997
ibm-942_P12A-1999	macos-6_2-10.4	ibm-1148_P100-1997
ibm-943_P15A-2003	macos-7_3-10.2	ibm-1149_P100-1997
ibm-943_P130-1999	macos-29-10.2	ibm-1153_P100-1999
ibm-33722_P12A_P12A-2009_U2	macos-35-10.2	ibm-1154_P100-1999
ibm-33722_P120-1999	ibm-1051_P100-1995	ibm-1155_P100-1999
ibm-954_P101-2007	ibm-1276_P100-1995	ibm-1156_P100-1999
euc-jp-2007	ibm-1006_P100-1995	ibm-1157_P100-1999
ibm-1373_P100-2002	ibm-1098_P100-1995	ibm-1158_P100-1999
windows-950-2000	ibm-1124_P100-1996	ibm-1160_P100-1999
ibm-950_P110-1999	ibm-1125_P100-1997	ibm-1164_P100-1999
ibm-1375_P100-2008	ibm-1129_P100-1997	ibm-1364_P110-2007
ibm-5471_P100-2006	ibm-1131_P100-1997	ibm-1371_P100-1999
ibm-1386_P100-2001	ibm-1133_P100-1997	ibm-1388_P103-2001

windows-936-2000	ISO_2022,locale=ja,version=0	ibm-1390_P110-2003
ibm-1383_P110-1999	ISO_2022,locale=ja,version=1	ibm-1399_P110-2003
ibm-5478_P100-1995	ISO_2022,locale=ja,version=2	ibm-5123_P100-1999
euc-tw-2014	ISO_2022,locale=ja,version=3	ibm-8482_P100-1999
ibm-964_P110-1999	ISO_2022,locale=ja,version=4	ibm-16684_P110-2003
ibm-949_P110-1999	ISO_2022,locale=ko,version=0	ibm-4899_P100-1998
ibm-949_P11A-1999	ISO_2022,locale=ko,version=1	ibm-4971_P100-1999
ibm-970_P110_P110-2006_U2	ISO_2022,locale=zh,version=0	ibm-9067_X100-2005
ibm-971_P100-1995	ISO_2022,locale=zh,version=1	ibm-12712_P100-1998
ibm-1363_P11B-1998	ISO_2022,locale=zh,version=2	ibm-16804_X110-1999
ibm-1363_P110-1997	HZ	ibm-37_P100-1995,swaplfnl
windows-949-2000	x11-compound-text	ibm-1047_P100-1995,swaplfnl
windows-874-2000	ISCII,version=0	ibm-1140_P100-1997,swaplfnl

ibm-874_P100-1995	ISCII,version=1	ibm-1141_P100-1997,swaplfnl
ibm-1162_P100-1999	ISCII,version=2	ibm-1142_P100-1997,swaplfnl
ibm-437_P100-1995	ISCII,version=3	ibm-1143_P100-1997,swaplfnl
ibm-720_P100-1997	ISCII,version=4	ibm-1144_P100-1997,swaplfnl
ibm-737_P100-1997	ISCII,version=5	ibm-1145_P100-1997,swaplfnl
ibm-775_P100-1996	ISCII,version=6	ibm-1146_P100-1997,swaplfnl
ibm-850_P100-1995	ISCII,version=7	ibm-1147_P100-1997,swaplfnl
ibm-851_P100-1995	ISCII,version=8	ibm-1148_P100-1997,swaplfnl
ibm-852_P100-1995	LMBCS-1	ibm-1149_P100-1997,swaplfnl
ibm-855_P100-1995	ibm-37_P100-1995	ibm-1153_P100-1999,swaplfnl
ibm-856_P100-1995	ibm-273_P100-1995	ibm-12712_P100-1998,swaplfnl
ibm-857_P100-1995	ibm-277_P100-1995	ibm-16804_X110-1999,swaplfnl
ibm-858_P100-1997	ibm-278_P100-1995	ebcdic-xml-us

이전 이미지 작업 설정

Kinesis 또는 Kafka 같은 데이터 스트리밍 대상에 CDC 업데이트를 작성하는 경우 업데이트를 통한 변경 전에 소스 데이터베이스 행의 원래 값을 볼 수 있습니다. 이를 가능하게 하기 위해 원본 데이터베이스 엔진에서 제공하는 데이터를 기반으로 업데이트 이벤트의 이전 이미지를 AWS DMS 채웁니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

이를 위해 `BeforeImageSettings` 파라미터를 사용하면 소스 데이터베이스 시스템에서 수집된 값을 사용하여 모든 업데이트 작업에 새 JSON 속성이 추가됩니다.

`BeforeImageSettings`는 전체 로드와 CDC 작업 또는 CDC 전용 작업에만 적용해야 합니다. 전체 로드와 CDC 작업은 기존 데이터를 마이그레이션하고 지속적인 변경 사항을 복제합니다. CDC 전용 작업은 데이터 변경 사항만 복제합니다.

전체 로드 전용 작업에는 `BeforeImageSettings`를 적용하지 마십시오.

`BeforeImageSettings`의 옵션은 다음과 같습니다.

- `EnableBeforeImage` – `true`로 설정하면 이전 이미징이 활성화됩니다. 기본값은 `false`입니다.
- `FieldName` – 새 JSON 속성에 이름을 지정합니다. `EnableBeforeImage`가 `true`인 경우 `FieldName`은 필수이며 비워 둘 수 없습니다.
- `ColumnFilter` – 이전 이미징을 사용하여 추가할 열을 지정합니다. 테이블 기본 키의 일부에 속하는 열만 추가하려면 기본값 `pk-only`를 사용하고, 이전 이미지 값이 있는 모든 열을 추가하려면 `all`을 사용합니다. 참고로 이전 이미지는 CLOB 및 BLOB와 같은 대용량 이진 객체(LOB) 데이터 유형을 지원하지 않습니다.

다음은 `BeforeImageSettings`의 용례를 보여줍니다.

```
"BeforeImageSettings": {
    "EnableBeforeImage": true,
    "FieldName": "before-image",
    "ColumnFilter": "pk-only"
  }
```

추가 테이블 매핑 설정을 비롯해 Kinesis 사용 시 이전 이미지 설정에 관한 자세한 내용은 Kinesis 데이터 스트림 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기 단원을 참조하십시오.

추가 테이블 매핑 설정을 비롯해 Kafka 사용 시 이전 이미지 설정에 관한 자세한 내용은 Apache Kafka 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기 단원을 참조하십시오.

오류 처리 작업 설정

다음 설정을 사용하여 복제 작업의 오류 처리 동작을 설정할 수 있습니다. 작업 구성 파일을 사용하여 작업 설정을 지정하는 방법에 관한 자세한 내용은 작업 설정 예제을 참조하십시오.

- DataErrorPolicy— 레코드 수준에서 데이터를 처리하는 것과 관련된 오류가 발생할 경우 AWS DMS가 취하는 조치를 결정합니다. 일부 데이터 처리 오류 예제에는 전환 오류, 변환 오류, 잘못된 데이터가 포함됩니다. 기본값은 LOG_ERROR입니다.
  - IGNORE_RECORD – 작업은 계속되고 해당 레코드의 데이터는 무시됩니다. DataErrorEscalationCount 속성의 오류 카운터가 증가합니다. 따라서 테이블에 대한 오류 한도를 설정하면 이 오류는 한도를 향해 증가합니다.
  - LOG_ERROR – 작업은 계속되고 오류가 작업 로그에 기록됩니다.
  - SUSPEND_TABLE – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
  - STOP_TASK – 작업은 중지되고 수동 개입이 필요합니다.
- DataTruncationErrorPolicy – 데이터를 자를 때 AWS 에서 수행하는 작업을 결정합니다. 기본값은 LOG_ERROR입니다.
  - IGNORE_RECORD – 작업은 계속되고 해당 레코드의 데이터는 무시됩니다. DataErrorEscalationCount 속성의 오류 카운터가 증가합니다. 따라서 테이블에 대한 오류 한도를 설정하면 이 오류는 한도를 향해 증가합니다.
  - LOG_ERROR – 작업은 계속되고 오류가 작업 로그에 기록됩니다.
  - SUSPEND_TABLE – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
  - STOP_TASK – 작업은 중지되고 수동 개입이 필요합니다.
- DataErrorEscalationPolicy – 최대 오류 수(DataErrorEscalationCount 파라미터에서 설정됨)에 도달할 때 AWS DMS에서 수행하는 작업을 결정합니다. 기본값은 SUSPEND_TABLE입니다.
  - SUSPEND_TABLE – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
  - STOP_TASK – 작업은 중지되고 수동 개입이 필요합니다.
- DataErrorEscalationCount – 특정 레코드의 데이터에 발생할 수 있는 최대 오류 수를 설정합니다. 이 수치에 도달하면 오류 레코드를 포함하는 테이블의 데이터는 DataErrorEscalationPolicy에 설정된 정책에 따라 처리됩니다. 기본값은 0입니다.
- EventErrorPolicy— 작업 관련 이벤트를 보내는 동안 오류가 발생할 경우 AWS DMS에서 취하는 조치를 결정합니다. 가능한 값은 다음과 같습니다.

  - IGNORE – 작업은 계속되며 해당 이벤트와 관련된 모든 데이터는 무시됩니다.
  - STOP_TASK – 작업은 중지되고 수동 개입이 필요합니다.
- TableErrorPolicy – 특정 테이블의 데이터 또는 메타데이터를 처리할 때 오류가 발생하는 경우 AWS DMS에서 수행할 작업을 결정합니다. 이 오류는 일반 테이블 데이터에만 적용되고 특정 레코드와 관련된 오류가 아닙니다. 기본값은 SUSPEND_TABLE입니다.
  - SUSPEND_TABLE – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
  - STOP_TASK – 작업은 중지되고 수동 개입이 필요합니다.
- TableErrorEscalationPolicy – 최대 오류 수(TableErrorEscalationCount 파라미터를 사용하여 설정됨)에 도달할 때 AWS 에서 수행하는 작업을 결정합니다. 기본값과 유일한 사용자 설정은 STOP_TASK이며 여기에서 작업은 중지되고 수동 개입이 필요합니다.
- TableErrorEscalationCount – 특정 테이블의 일반 데이터 또는 메타데이터에 발생할 수 있는 오류의 최대 수입니다. 이 수치에 도달하면 테이블의 데이터는 TableErrorEscalationPolicy에 설정된 정책에 따라 처리됩니다. 기본값은 0입니다.
- RecoverableErrorCount – 환경 오류가 발생할 때 작업을 다시 시작하기 위한 시도의 최대 횟수입니다. 시스템이 작업을 지정된 횟수만큼 다시 시작하려고 시도하고 나면, 이 작업은 중지되고 수동 개입이 필요합니다. 기본값은 -1이며, 이 값은 작업을 무기한 다시 AWS DMS 시작하도록 지시합니다. 이 값을 0으로 설정하면 작업을 다시 시작하기 위한 시도를 하지 않습니다. 치명적 오류가 발생하는 경우, 6회 시도 후 작업 재시작 시도를 AWS DMS 중지합니다.
- RecoverableErrorInterval— AWS DMS가 작업 재시작을 시도할 때까지 대기하는 시간 (초) 기본값은 5입니다.
- RecoverableErrorThrottling – 활성화되면 작업 재시작 시도 간격이 RecoverableErrorInterval 값을 기준으로 연속적으로 늘어납니다. 예를 들어 RecoverableErrorInterval을 5초로 설정하면 다음 재시도는 10초 후, 20초 후, 40초 후 등등의 시간 경과 후에 다시 시도됩니다. 기본값은 true입니다.
- RecoverableErrorThrottlingMax— 활성화된 경우 AWS DMS가 작업 재시작 시도 사이에 대기하는 최대 시간 (초) RecoverableErrorThrottling 기본값은 1800입니다.
- RecoverableErrorStopRetryAfterThrottlingMax— 로 설정하면 복구 시도 사이에 AWS DMS 대기하는 true 최대 시간 (초) 에 도달한 후 작업 재시작을 중지합니다. RecoverableErrorThrottlingMax
- ApplyErrorDeletePolicy – DELETE 작업과 충돌이 있을 때 AWS 에서 수행하는 작업을 결정합니다. 기본값은 IGNORE_RECORD입니다. 가능한 값은 다음과 같습니다.

  - IGNORE_RECORD – 작업은 계속되고 해당 레코드의 데이터는 무시됩니다. ApplyErrorEscalationCount 속성의 오류 카운터가 증가합니다. 따라서 테이블에 대한 오류 한도를 설정하면 이 오류는 한도를 향해 증가합니다.
  - LOG_ERROR – 작업은 계속되고 오류가 작업 로그에 기록됩니다.
  - SUSPEND_TABLE – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
  - STOP_TASK – 작업은 중지되고 수동 개입이 필요합니다.
- ApplyErrorInsertPolicy – INSERT 작업과 충돌이 있을 때 AWS DMS에서 수행하는 작업을 결정합니다. 기본값은 LOG_ERROR입니다. 가능한 값은 다음과 같습니다.
  - IGNORE_RECORD – 작업은 계속되고 해당 레코드의 데이터는 무시됩니다. ApplyErrorEscalationCount 속성의 오류 카운터가 증가합니다. 따라서 테이블에 대한 오류 한도를 설정하면 이 오류는 한도를 향해 증가합니다.
  - LOG_ERROR – 작업은 계속되고 오류가 작업 로그에 기록됩니다.
  - SUSPEND_TABLE – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
  - STOP_TASK – 작업은 중지되고 수동 개입이 필요합니다.
  - INSERT_RECORD – 삽입한 원본 레코드와 동일한 기본 키를 사용하는 기존 대상 레코드가 있으면 대상 레코드가 업데이트됩니다.
- ApplyErrorUpdatePolicy – UPDATE 작업과 누락 데이터 충돌이 있을 때 AWS DMS에서 수행하는 작업을 결정합니다. 기본값은 LOG_ERROR입니다. 가능한 값은 다음과 같습니다.
  - IGNORE_RECORD – 작업은 계속되고 해당 레코드의 데이터는 무시됩니다. ApplyErrorEscalationCount 속성의 오류 카운터가 증가합니다. 따라서 테이블에 대한 오류 한도를 설정하면 이 오류는 한도를 향해 증가합니다.
  - LOG_ERROR – 작업은 계속되고 오류가 작업 로그에 기록됩니다.
  - SUSPEND_TABLE – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
  - STOP_TASK – 작업은 중지되고 수동 개입이 필요합니다.
  - UPDATE_RECORD— 대상 레코드가 누락된 경우 누락된 대상 레코드가 대상 테이블에 삽입됩니다. AWS DMS 작업에 대한 LOB 열 지원을 완전히 비활성화합니다. 이 옵션을 선택하는 경우, Oracle 이 소스 데이터베이스이면 모든 원본 테이블 열에서 전체 보충 로깅을 활성화해야 합니다.
- ApplyErrorEscalationPolicy— 최대 오류 수 (ApplyErrorEscalationCount매개 변수를 사용하여 설정) 에 도달했을 때 AWS DMS에서 수행하는 작업을 결정합니다. 기본값은 LOG_ERROR입니다.

  - LOG_ERROR – 작업은 계속되고 오류가 작업 로그에 기록됩니다.
  - SUSPEND_TABLE – 작업은 계속되지만 오류 레코드가 있는 테이블의 데이터는 오류 상태로 이동하고 데이터가 복제되지 않습니다.
  - STOP_TASK – 작업은 중지되고 수동 개입이 필요합니다.
- ApplyErrorEscalationCount – 이 옵션은 변경 프로세스 작업 중에 특정 테이블에서 발생할 수 있는 최대 APPLY 충돌 개수를 설정합니다. 이 개수에 도달하면 테이블 데이터는 ApplyErrorEscalationPolicy 파라미터에 설정된 정책에 따라 처리됩니다. 기본값은 0입니다.
- ApplyErrorFailOnTruncationDdl – CDC 중에 추적되는 테이블에 대해 잘림이 수행될 때 작업이 실패하도록 하려면 이 옵션을 true로 설정합니다. 기본값은 false입니다.

  이 방법은 DDL 테이블 잘림을 복제하지 않는 PostgreSQL 버전 11.x나 이전 버전 또는 다른 소스 엔드포인트에서는 사용할 수 없습니다.
- FailOnNoTablesCaptured – 작업이 시작될 때 작업에 대해 정의된 테이블 매핑이 테이블을 찾지 못하는 경우 작업이 실패하도록 하려면 이 옵션을 true로 설정합니다. 기본값은 false입니다.
- FailOnTransactionConsistencyBreached – 이 옵션은 CDC와 함께 사용하기 위한 원본으로 Oracle을 사용하는 작업에 적용됩니다. 기본값은 false입니다. 트랜잭션이 지정된 제한 시간보다 더 오랫동안 열려 있고 삭제될 수 있는 경우 작업이 실패하도록 하려면 true로 설정합니다.

  CDC 작업이 Oracle과 함께 시작되면 가장 오래된 미해결 트랜잭션이 종료될 때까지 제한된 시간 동안 AWS DMS 기다린 후 CDC를 시작합니다. 가장 오래된 미해결 거래가 제한 시간에 도달할 때까지 종료되지 않으면 대부분의 경우 해당 거래를 무시하고 CDC를 AWS DMS 시작합니다. 이 옵션을 true로 설정하면 작업이 실패합니다.
- FullLoadIgnoreConflicts— 캐시된 이벤트를 적용할 때 “영향을 받은 행이 0개” 및 “중복됨” 오류를 true AWS DMS 무시하도록 이 옵션을 설정하십시오. 로 설정하면 오류를 무시하는 false 대신 모든 오류를 AWS DMS 보고합니다. 기본값은 true입니다.

참고로 Redshift의 테이블 로드 오류는 STL_LOAD_ERRORS에서 보고됩니다. 자세한 내용은 Amazon Redshift 데이터베이스 개발자 안내서에서 STL_LOAD_ERRORS를 참조하세요.

작업 설정 저장

다른 작업의 설정을 다시 사용해야 하는 경우에 작업 설정을 JSON 파일로 저장할 수 있습니다. 작업의 개요 세부 정보 섹션에서 JSON 파일에 복사할 작업 설정을 찾을 수 있습니다.

> **Note**
>
> 작업 설정을 다른 작업에 재사용하는 동안에는 모든 `CloudWatchLogGroup` 및 `CloudWatchLogStream` 속성을 제거하세요. 그렇지 않으면 다음 오류가 발생합니다. 시스템 오류 메시지:작업 설정 CloudWatchLogGroup 또는 생성 시 설정할 CloudWatchLogStream 수 없습니다.

예를 들어, 다음 JSON 파일에는 작업에 대해 저장된 설정이 포함됩니다.

```
{
    "TargetMetadata": {
        "TargetSchema": "",
        "SupportLobs": true,
        "FullLobMode": false,
        "LobChunkSize": 0,
        "LimitedSizeLobMode": true,
        "LobMaxSize": 32,
        "InlineLobMaxSize": 0,
        "LoadMaxFileSize": 0,
        "ParallelLoadThreads": 0,
        "ParallelLoadBufferSize": 0,
        "BatchApplyEnabled": false,
        "TaskRecoveryTableEnabled": false,
        "ParallelLoadQueuesPerThread": 0,
        "ParallelApplyThreads": 0,
        "ParallelApplyBufferSize": 0,
        "ParallelApplyQueuesPerThread": 0
    },
    "FullLoadSettings": {
        "TargetTablePrepMode": "DO_NOTHING",
        "CreatePkAfterFullLoad": false,
        "StopTaskCachedChangesApplied": false,
        "StopTaskCachedChangesNotApplied": false,
        "MaxFullLoadSubTasks": 8,
        "TransactionConsistencyTimeout": 600,
        "CommitRate": 10000
    },
    "Logging": {
        "EnableLogging": true,
        "LogComponents": [
```

```
            {
                "Id": "TRANSFORMATION",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "SOURCE_UNLOAD",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "IO",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "TARGET_LOAD",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "PERFORMANCE",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "SOURCE_CAPTURE",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "SORTER",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "REST_SERVER",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "VALIDATOR_EXT",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "TARGET_APPLY",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "TASK_MANAGER",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
```

```
            {
                "Id": "TABLES_MANAGER",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "METADATA_MANAGER",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "FILE_FACTORY",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "COMMON",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "ADDONS",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "DATA_STRUCTURE",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "COMMUNICATION",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            },
            {
                "Id": "FILE_TRANSFER",
                "Severity": "LOGGER_SEVERITY_DEFAULT"
            }
        ]
    },
    "ControlTablesSettings": {
        "ControlSchema": "",
        "HistoryTimeslotInMinutes": 5,
        "HistoryTableEnabled": false,
        "SuspendedTablesTableEnabled": false,
        "StatusTableEnabled": false,
        "FullLoadExceptionTableEnabled": false
    },
    "StreamBufferSettings": {
        "StreamBufferCount": 3,
```

```
        "StreamBufferSizeInMB": 8,
        "CtrlStreamBufferSizeInMB": 5
    },
    "ChangeProcessingDdlHandlingPolicy": {
        "HandleSourceTableDropped": true,
        "HandleSourceTableTruncated": true,
        "HandleSourceTableAltered": true
    },
    "ErrorBehavior": {
        "DataErrorPolicy": "LOG_ERROR",
        "DataTruncationErrorPolicy": "LOG_ERROR",
        "DataErrorEscalationPolicy": "SUSPEND_TABLE",
        "DataErrorEscalationCount": 0,
        "TableErrorPolicy": "SUSPEND_TABLE",
        "TableErrorEscalationPolicy": "STOP_TASK",
        "TableErrorEscalationCount": 0,
        "RecoverableErrorCount": -1,
        "RecoverableErrorInterval": 5,
        "RecoverableErrorThrottling": true,
        "RecoverableErrorThrottlingMax": 1800,
        "RecoverableErrorStopRetryAfterThrottlingMax": true,
        "ApplyErrorDeletePolicy": "IGNORE_RECORD",
        "ApplyErrorInsertPolicy": "LOG_ERROR",
        "ApplyErrorUpdatePolicy": "LOG_ERROR",
        "ApplyErrorEscalationPolicy": "LOG_ERROR",
        "ApplyErrorEscalationCount": 0,
        "ApplyErrorFailOnTruncationDdl": false,
        "FullLoadIgnoreConflicts": true,
        "FailOnTransactionConsistencyBreached": false,
        "FailOnNoTablesCaptured": true
    },
    "ChangeProcessingTuning": {
        "BatchApplyPreserveTransaction": true,
        "BatchApplyTimeoutMin": 1,
        "BatchApplyTimeoutMax": 30,
        "BatchApplyMemoryLimit": 500,
        "BatchSplitSize": 0,
        "MinTransactionSize": 1000,
        "CommitTimeout": 1,
        "MemoryLimitTotal": 1024,
        "MemoryKeepTime": 60,
        "StatementCacheSize": 50
    },
    "PostProcessingRules": null,
```

```
    "CharacterSetSettings": null,
    "LoopbackPreventionSettings": null,
    "BeforeImageSettings": null,
    "FailTaskWhenCleanTaskResourceFailed": false
}
```

태스크의 소스 데이터베이스에 대한 LOB 지원 설정 AWS DMS

경우에 따라 대용량 이진 객체(LOB)는 시스템 간에 마이그레이션하기 어려울 수 있습니다. AWS DMS에는 LOB 열을 튜닝할 때 도움이 되는 다양한 옵션이 있습니다. 어떤 데이터 유형이 언제 LOB로 간주되는지 AWS DMS알아보려면 설명서를 참조하십시오. AWS DMS

다른 데이터베이스로 데이터를 마이그레이션할 때에는 LOB 저장 방법을 재고해 볼 수 있으며, 특히 이종 마이그레이션에서 그러합니다. 이 작업을 수행할 경우, LOB 데이터를 마이그레이션할 필요가 없습니다.

LOB를 포함하기로 결정하는 경우 다른 LOB 설정을 결정할 수 있습니다.

- LOB 모드는 LOB가 처리되는 방식을 결정합니다.
  - 전체 LOB 모드 - 전체 LOB 모드에서는 크기에 관계없이 모든 LOB를 소스에서 타겟으로 AWS DMS 마이그레이션합니다. 이 구성에서는 예상할 수 AWS DMS 있는 최대 LOB 크기에 대한 정보가 없습니다. 또한 LOB는 한 번에 하나씩, 조각별로 마이그레이션됩니다. 전체 LOB 모드는 매우 느릴 수 있습니다.
  - 제한적 LOB 모드 - 제한적 LOB 모드에서는 DMS가 수용해야 하는 최대 LOB 크기를 설정합니다. 이를 통해 DMS는 메모리를 사전 할당하고 LOB 데이터를 대량으로 로드할 수 있습니다. 최대 LOB 크기를 초과하는 LOB는 잘리고 경고가 로그 파일에 기록됩니다. 제한적 LOB 모드는 전체 LOB 모드에 비해 상당한 성능 이점이 있습니다. 가능하다면 제한적 LOB 모드 사용을 권장합니다. 최대 권장값은 102,400KB(100MB)입니다.

    > **Note**
    >
    > 63KB보다 큰 값의 최대 LOB 크기(K) 옵션을 사용하면 제한된 LOB 모드에서 실행하도록 구성된 전체 로드의 성능에 영향을 줍니다. 전체 로드 시 DMS는 최대 LOB 크기(k) 값에 커밋 비율을 곱하여 메모리를 할당하며, 해당 제품에는 LOB 열 수를 곱합니다. DMS에서 해당 메모리를 미리 할당할 수 없는 경우, DMS는 스왑 메모리를 사용하기 시작하며 이는 전체 로드 성능에 영향을 미칩니다. 따라서 제한된 LOB 모드를 사용할 때 성능 문제가 발생하는 경우, 적정한 수준의 성능에 도달할 때까지 커밋 비율을 낮추는 것이 좋습니다. 테

이블의 LOB 배포를 이해한 후에는 지원되는 엔드포인트에 인라인 LOB 모드 사용을 고려할 수도 있습니다.
제한된 LOB 크기를 확인하려면 `ValidationPartialLobSize`를 `LobMaxSize`(K)와 같은 값으로 설정해야 합니다.

- 인라인 LOB 모드 - 인라인 LOB 모드에서는 DMS가 인라인으로 전송하는 최대 LOB 크기를 설정합니다. 지정된 크기보다 작은 LOB는 인라인으로 전송됩니다. 지정된 크기보다 큰 LOB는 전체 LOB 모드를 사용하여 복제됩니다. 대부분의 LOB가 작은 경우, 이 옵션을 선택하여 작은 LOB와 큰 LOB를 모두 복제할 수 있습니다. DMS는 S3 및 Redshift와 같이 전체 LOB 모드를 지원하지 않는 엔드포인트에 대해서는 인라인 LOB 모드를 지원하지 않습니다.

> **Note**
>
> Oracle을 통해 LOB는 가능할 때마다 VARCHAR 데이터 형식으로 처리됩니다. 이 방법을 사용하면 데이터베이스에서 대량으로 AWS DMS 가져올 수 있어 다른 방법보다 훨씬 빠릅니다. Oracle에서 VARCHAR의 최대 크기는 32K이므로 Oracle이 소스 데이터베이스일 때 32K 미만의 제한적인 LOB 크기가 최적입니다.

- 작업이 제한적 LOB 모드에서 실행되도록 구성된 경우 최대 LOB 크기(K) 옵션은 AWS DMS 가 허용하는 최대 크기의 LOB를 설정합니다. 이 값보다 큰 LOB는 이 값으로 잘립니다.
- 작업이 전체 LOB 모드를 사용하도록 구성된 경우 LOB를 분할하여 AWS DMS 검색합니다. LOB 청크 크기(K) 옵션은 각 조각의 크기를 결정합니다. 이 옵션을 설정하면 네트워크 구성에서 허용된 최대 패킷 크기에 특히 유의해야 합니다. LOB 청크 크기가 최대 허용 패킷 크기를 초과하면 연결 해제 오류가 나타날 수 있습니다. `LobChunkSize`의 권장값은 64킬로바이트[KB]입니다. `LobChunkSize`의 값을 64KB 이상으로 늘리면 작업이 실패할 수 있습니다.
- 작업이 인라인 LOB 모드에서 실행되도록 구성된 경우, `InlineLobMaxSize` 설정에 따라 DMS가 인라인으로 전송할 LOB가 결정됩니다.

이 옵션을 지정하는 작업 설정에 관한 자세한 내용은 대상 메타데이터 작업 설정 단원을 참조하십시오.

여러 작업 생성

일부 마이그레이션 시나리오에서는 여러 마이그레이션 작업을 생성해야 할 수도 있습니다. 작업은 독립적으로 수행되며 동시에 실행할 수 있습니다. 각 작업에는 자체 초기 로드, CDC, 로그 읽기 프로세스가 있습니다. 데이터 조작 언어(DML)를 통해 관련된 테이블은 동일한 작업의 일부여야 합니다.

마이그레이션에서 여러 작업을 생성해야 하는 이유는 주로 다음과 같습니다.

- 작업용 대상 테이블은 시스템을 팬아웃하거나 여러 시스템으로 나눌 경우 등 각기 다른 데이터베이스에 상주합니다.
- 대규모 테이블의 마이그레이션을 여러 작업으로 나누려면 필터링을 사용해야 합니다.

> **Note**
>
> 각 작업에는 자체 변경 캡처와 로그 읽기 프로세스가 있기 때문에 변경 사항은 여러 작업에서 조정되지 않습니다. 따라서 여러 작업을 사용하여 마이그레이션을 수행할 때에는 개별 원본 트랜잭션이 단일 작업 내에 완전히 포함되도록 해야 합니다. 개별 트랜잭션이 여러 작업으로 분할되지 않은 경우, 여러 작업을 사용하여 마이그레이션을 수행할 수 있습니다.

AWS DMS를 사용하여 지속 복제를 위한 작업 생성

원본 데이터 스토어에 대한 지속적 변경 사항을 캡처하는 AWS DMS 작업을 생성할 수 있습니다. 데이터를 마이그레이션하는 동안에도 이 변경 사항을 캡처할 수 있습니다. 작업을 생성하여 지원된 대상 데이터 스토어로 초기(전체 로드) 마이그레이션을 완료한 후 지속적 변경 사항을 캡처할 수도 있습니다. 이 프로세스를 진행 중인 복제 또는 변경 데이터 캡처(CDC)라고 합니다. AWS DMS에서는 원본 데이터 스토어에서 지속적 변경 사항을 복제할 때 이 프로세스를 사용합니다. 이 프로세스는 데이터베이스 엔진의 기본 API를 사용하여 데이터베이스 로그에 대한 변경 사항을 수집합니다.

> **Note**
>
> 뷰는 전체 로드 작업만 사용하여 마이그레이션할 수 있습니다. 작업이 CDC 전용 작업이거나 완료된 후 CDC를 시작하는 전체 로드 작업인 경우, 소스의 테이블만 마이그레이션에 포함됩니다. 전체 로드 전용 작업을 사용하면 뷰 또는 테이블 및 뷰 조합을 마이그레이션할 수 있습니다. 자세한 내용은 JSON을 사용하여 테이블 선택 및 변환 지정 섹션을 참조하세요.

각 소스 엔진에는 이 변경 스트림을 지정된 사용자 계정에 노출하기 위한 특정 구성 요구 사항이 있습니다. 대다수 엔진에서는 데이터 손실 없이 캡처 프로세스를 지원할 수 있도록 유의미한 방식으로 변경 데이터를 소모하도록 해주는 일부 추가 구성이 필요합니다. 예를 들어, Oracle은 보충 로깅 추가를 필요로 하고, MySQL은 행 수준의 2진법 로깅(빈 로깅)을 필요로 합니다.

AWS DMS에서는 소스 데이터베이스의 지속적 변경 사항을 읽기 위해 엔진별 API 작업을 사용하여 소스 엔진의 트랜잭션 로그에서 변경 사항을 읽습니다. 다음은 AWS DMS 작동 방식에 대한 몇 가지 예입니다.

- Oracle의 경우 AWS DMS는 Oracle LogMiner API나 bfile API(binary reader API) 중 하나를 사용하여 지속적 변경 사항을 읽습니다. AWS DMS는 시스템 변경 수(SCN)에 따라 온라인 또는 아카이브 redo 로그에서 지속적 변경 사항을 읽습니다.
- Microsoft SQL Server의 경우 AWS DMS는 MS-Replication 또는 MS-CDC를 사용해 SQL 서버 트랜잭션 로그에 정보를 기록합니다. 그러면 SQL 서버에서 `fn_dblog()` 또는 `fn_dump_dblog()` 함수를 사용해 로그 시퀀스 번호(LSN)에 토대를 둔 트랜잭션 로그의 변경 사항을 읽습니다.
- MySQL의 경우, AWS DMS는 행 기반 바이너리 로그(binlogs)의 변경 사항을 읽고, 이 변경 사항을 대상으로 마이그레이션합니다.
- PostgreSQL의 경우, AWS DMS는 논리적 복제 슬롯을 설정하고, test_decoding 플러그인을 사용해 원본의 변경 사항을 읽고, 이를 대상으로 마이그레이션합니다.
- Amazon RDS를 소스로 할 경우, CDC를 설정할 수 있는 백업을 권장합니다. 충분한 시간 동안(대개 24시간이면 충분) 소스 데이터베이스가 변경 로그를 유지하도록 구성하는 것이 좋습니다. 엔드포인트에 대한 특정 설정은 다음을 참조하세요.
  - Amazon RDS for Oracle: AWS관리형 Oracle 소스 구성 대상 AWS DMS.
  - Amazon RDS for MySQL 및 Aurora MySQL: AWS DMS 소스로 AWS 관리형 MySQL 호환 데이터베이스 사용.
  - Amazon RDS for SQL Server: 클라우드 SQL Server DB 인스턴스에서 지속적 복제 설정.
  - Amazon RDS for PostgreSQL 및 Aurora PostgreSQL: PostgreSQL은 필요한 WAL을 자동으로 유지합니다.

다음은 진행 중인 복제 작업의 두 가지 유형입니다.

- 전체 로드+CDC – 이 작업은 기존 데이터를 마이그레이션한 다음, 소스 데이터베이스의 변경 내용을 기반으로 대상 데이터베이스를 업데이트합니다.
- CDC만 해당 – 대상 데이터베이스에 데이터를 확보한 후 진행 중인 변경 내용을 마이그레이션하는 작업입니다.

CDC 시작 지점에서 복제를 시작

몇몇 지점에서 AWS DMS 지속적 복제 작업(변경 데이터 캡처만 해당)을 시작할 수 있습니다. 여기에는 다음이 포함됩니다.

- 사용자 지정 CDC 시작 시간부터 – AWS Management Console 또는 AWS CLI를 사용하여 복제를 시작하고 싶은 지점에 타임스탬프로 AWS DMS를 제공합니다. AWS DMS는 사용자 지정 CDC 시작 시간부터 지속적 복제 작업을 시작합니다. AWS DMS는 주어진 타임스탬프(UTC 시간)를 SQL 서버

용 LSN 또는 Oracle용 SCN과 같은 기본 시작점으로 변환합니다. AWS DMS는 엔진별 방법을 사용하여 소스 엔진의 변경 스트림에 따른 마이그레이션 작업을 어디서 시작할지 결정합니다.

> **Note**
>
> `StartFromContext` 연결 속성을 필수 타임스탬프로 설정해야만 원본인 Db2는 사용자 지정된 CDC 시작 시간을 제공할 수 있습니다.
> PostgreSQL을 소스로 사용할 경우 사용자 지정 CDC 시작 시간을 지원하지 않습니다. PostgreSQL 데이터베이스 엔진에는 타임스탬프를 LSN 또는 SCN으로 매핑하는 방법이 없기 때문입니다. Oracle 및 SQL Server에는 그러한 방법이 있습니다.

- CDC 원래 시작 지점부터 – 소스 엔진 트랜잭션 로그의 원래 포인트에서 시작할 수도 있습니다. 타임스탬프는 트랜잭션 로그에 여러 네이티브 지점을 나타낼 수 있기 때문에 경우에 따라 이 접근 방식이 좋을 수 있습니다. AWS DMS는 다음 소스 엔드포인트에 대해 이 기능을 지원합니다.
  - SQL Server
  - PostgreSQL
  - Oracle
  - MySQL
  - MariaDB

작업이 생성되면 AWS DMS는 CDC 시작 지점을 표시하며 이 시작 지점은 변경할 수 없습니다. 다른 CDC 시작 지점을 사용하려면 새 작업을 만드세요.

CDC 기본 시작점 결정

CDC 기본 시작점은 CDC를 시작할 수 있는 시간을 정의하는 데이터베이스 엔진 로그의 지점입니다. 예를 들어, 대량 데이터 덤프가 이미 대상에 적용된 경우를 생각해볼 수 있습니다. 진행 중인 복제 전용 작업의 기본 시작점을 찾아볼 수 있습니다. 데이터 불일치를 방지하려면 복제 전용 작업의 시작 지점을 신중하게 선택하십시오. DMS는 선택한 CDC 시작 지점 이후에 시작된 트랜잭션을 캡처합니다.

다음은 지원되는 소스 엔진에서 CDC 기본 시작점을 찾는 방법에 대한 예입니다.

SQL Server

SQL Server의 로그 시퀀스 번호(LSN)는 세 부분으로 구성되어 있습니다.

- 가상 로그 파일(VLF) 시퀀스 번호
- 로그 블록의 시작 오프셋

- 슬롯 번호

LSN의 예는 다음과 같습니다. `00000014:00000061:0001`

트랜잭션 로그 백업 설정에 따라 SQL Server 마이그레이션 작업의 시작 지점을 확인하려면 SQL Server의 `fn_dblog()` 또는 `fn_dump_dblog()` 함수를 사용합니다.

SQL Server에서 CDC 기본 시작점을 사용하려면 진행 중인 복제에 참여하는 테이블에 게시를 만드십시오. AWS DMS는 CDC 기본 시작점을 사용하지 않고 CDC를 사용하면 자동으로 간행물을 만듭니다.

PostgreSQL

PostgreSQL 소스 데이터베이스에 CDC 복구 체크포인트를 사용할 수 있습니다. 이 체크포인트 값은 다양한 지점에서 소스 데이터베이스에 대해 지속적 복제 작업(상위 작업)으로 생성됩니다. 일반적인 체크포인트에 관한 자세한 내용은 체크포인트를 CDC 시작 지점으로 사용 단원을 참조하십시오.

기본 시작점으로 사용할 체크포인트를 식별하려면 AWS Management Console에서 데이터베이스 `pg_replication_slots` 뷰 또는 상위 작업의 개요 세부 정보를 사용합니다.

콘솔에서 상위 작업에 대한 개요 세부 정보를 찾으려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/dms/v2/에서 AWS DMS 콘솔을 엽니다.

   IAM 사용자로 로그인한 경우, AWS DMS에 액세스하려면 적절한 권한이 있어야 합니다. 필요한 권한에 관한 자세한 내용은 AWS DMS 사용에 필요한 IAM 권한 단원을 참조하십시오.
2. 탐색 창에서 데이터베이스 마이그레이션 작업을 선택합니다.
3. 데이터베이스 마이그레이션 작업 페이지의 목록에서 상위 작업을 선택합니다. 그러면 개요 세부 정보가 표시된 상위 작업 페이지가 열립니다.
4. 변경 데이터 캡처(CDC), 변경 데이터 캡처(CDC) 시작 위치 및 변경 데이터 캡처(CDC) 복구 체크포인트 아래에서 체크포인트 값을 찾습니다.

   이 값은 다음과 유사합니다.

   ```
   checkpoint:V1#1#000004AF/B00000D0#0#0#*#0#0
   ```

   여기서 `4AF/B00000D0` 구성 요소는 이 기본 CDC 시작점을 지정하는 데 필요한 것입니다. PostgreSQL 원본의 이 시작 지점에서 복제를 시작하는 CDC 작업을 만들 때 DMS API

`CdcStartPosition` 파라미터를 이 값으로 설정합니다. AWS CLI를 사용하여 이 CDC 작업을 만드는 방법에 관한 자세한 내용은 AWS DMS를 사용하여 AWS 관리형 PostgreSQL DB 인스턴스에서 CDC 활성화 단원을 참조하십시오.

Oracle

시스템 변경 번호(SCN)는 Oracle 데이터베이스에서 사용하는 논리적 내부 타임스탬프입니다. SCN은 트랜잭션의 ACID 속성을 충족하는 데 필요한 (데이터베이스 내부에서 발생하는) 이벤트를 배열합니다. Oracle 데이터베이스는 SCN을 사용하여 디스크에 작성된 모든 변화가 진행된 위치를 표시합니다. 그리하여 복원 작업이 이미 작성된 변경 사항을 적용하지 않습니다. 또한 Oracle은 SCN을 사용하여 복구가 중지되지 않도록 데이터 세트에 다시 실행이 없는 지점을 표시합니다.

다음 명령을 실행해 Oracle 데이터베이스의 현재 SCN을 가져옵니다.

```
SELECT CURRENT_SCN FROM V$DATABASE
```

SCN 또는 타임스탬프를 사용하여 CDC 작업을 시작하면 열린 트랜잭션의 결과를 놓치고 해당 결과를 마이그레이션하지 못하게 됩니다. 열린 트랜잭션은 작업 시작 지점 이전에 시작되고 작업 시작 지점 이후에 커밋된 트랜잭션입니다. SCN과 타임스탬프를 식별하여 모든 열린 트랜잭션이 포함된 시점에서 CDC 작업을 시작할 수 있습니다. 자세한 내용은 Oracle 온라인 설명서의 트랜잭션을 참조하세요. 버전 3.5.1 및 이후 버전에서는 SCN 또는 타임스탬프를 사용하여 작업을 시작하는 경우 AWS DMS가 `openTransactionWindow` 엔드포인트 설정을 사용하여 CDC 전용 작업에 대한 열린 트랜잭션을 지원합니다.

`openTransactionWindow` 설정을 사용할 때는 열린 트랜잭션을 처리할 수 있는 창을 분 단위로 제공해야 합니다. AWS DMS는 캡처 위치를 이동하고 새 위치를 찾아 데이터 캡처를 시작합니다. AWS DMS는 새 시작 위치를 사용하여 필수 Oracle redo 로그 또는 보관된 redo 로그에서 열린 트랜잭션을 모두 스캔합니다.

MySQL

MySQL 버전 5.6.3이 출시되기 전에는 MySQL 로그 시퀀스 번호(LSN)는 4바이트의 부호 없는 정수였습니다. redo 로그 파일 크기 한도가 4GB에서 512GB로 증가한 MySQL 5.6.3부터 LSN은 8바이트의 부호 없는 정수가 되었습니다. 추가 사이즈 정보를 저장하기 위해 증가되면 바이트 추가가 요청됩니다. LSN 값을 사용하는 MySQL 5.6.3 또는 이후 버전에 빌드된 애플리케이션은 LSN 값을 저장해 비교하기 위해 32비트 대신 64비트 변수를 사용해야 합니다. MySQL LSN에 관한 자세한 내용은 MySQL 설명서를 참조하십시오.

다음 명령을 실행해 MySQL 데이터베이스의 현재 LSN을 가져옵니다.

```
mysql> show master status;
```

쿼리는 binlog 파일 이름, 위치 및 기타 값을 반환합니다. CDC 기본 시작점은 binlogs 파일 이름과 위치의 조합으로 표시됩니다. 예를 들어, `mysql-bin-changelog.000024:373`입니다. 이 예제에서 `mysql-bin-changelog.000024`는 binlogs 파일 이름이고, `373`은 AWS DMS가 변경 캡처를 시작하는 데 필요한 위치입니다.

체크포인트를 CDC 시작 지점으로 사용

지속적 복제 작업이 변경 사항을 마이그레이션하고 AWS DMS는 AWS DMS에 대한 체크포인트 정보를 캐싱하는 경우가 있습니다. AWS DMS가 생성한 체크포인트에는 정보가 포함되어 있으므로 복제 엔진이 변경 스트림의 복구 지점을 파악할 수 있습니다. 체크포인트를 사용하여 변경 타임라인으로 되돌아가거나 실패한 마이그레이션 작업을 복구합니다. 또한 이 체크포인트를 사용해 특정 시간에 다른 대상으로 다른 지속적 복제 작업을 시작할 수 있습니다.

다음 세 가지 방법 중 하나로 체크포인트 정보를 얻을 수 있습니다.

- API 작업 `DescribeReplicationTasks`를 실행하고 결과를 검토합니다. 작업 별로 정보를 필터링하고 체크포인트를 검색할 수 있습니다. 작업이 중지/실패 상태일 때 마지막 체크포인트를 검색할 수 있습니다. 작업이 삭제되면 이 정보는 없어집니다.
- 대상 인스턴스에서 `awsdms_txn_state`라는 메타데이터를 봅니다. 테이블을 쿼리하여 체크포인트 정보를 얻을 수 있습니다. 메타데이터 테이블을 생성하기 위해서 작업을 생성할 때 `TaskRecoveryTableEnabled` 파라미터를 `Yes`로 설정합니다. 이 설정을 적용하면 AWS DMS가 대상 메타데이터 테이블에 체크포인트 정보를 계속 기록합니다. 작업이 삭제되면 이 정보는 없어집니다.

  예를 들어, 다음은 메타데이터 테이블의 체크포인트 예입니다.`checkpoint:V1#34#00000132/0F000E48#0#0#*#0#121`
- 탐색 창에서 데이터베이스 마이그레이션 작업을 선택하고 데이터베이스 마이그레이션 작업 페이지에 나타나는 목록에서 상위 작업을 선택합니다. 개요 세부 정보가 표시된 상위 작업 페이지가 열립니다. 변경 데이터 캡처(CDC), 변경 데이터 캡처(CDC) 시작 위치 및 변경 데이터 캡처(CDC) 복구 체크포인트 아래에서 체크포인트 값을 찾습니다. 체크포인트 값은 다음과 유사합니다.

  `checkpoint:V1#1#000004AF/B00000D0#0#0#*#0#0`

커밋 또는 서버 시점에서 작업 중지

CDC 기본 시작점의 소개와 함께 AWS DMS는 다음 지점에서 작업을 중지할 수도 있습니다.

- 소스의 약정 시간
- 복제 인스턴스의 서버 시간

필요에 따라 작업을 중지할 수 있도록 작업을 수정하고 시간을 UTC로 설정할 수 있습니다. 설정한 약정 또는 서버 시간에 따라 작업은 자동으로 중지됩니다. 또는 작업 생성 시 마이그레이션 작업을 중지할 적절한 시점을 알고 있다면 작업을 생성할 때 중지 시간을 설정할 수 있습니다.

> **Note**
>
> 새 AWS DMS 서버리스 복제를 처음 시작할 때 모든 리소스를 초기화하는 데 최대 40분이 걸릴 수 있습니다. 단, 이 `server_time` 옵션은 리소스 초기화가 완료된 후에만 적용할 수 있습니다.

양방향 복제 수행

AWS DMS 작업을 사용하여 두 시스템 간에 양방향 복제를 수행할 수 있습니다. 양방향 복제에서는 두 시스템 간에 동일한 테이블(또는 테이블 집합)의 데이터를 두 방향으로 모두 복제할 수 있습니다.

예를 들어 데이터베이스 A에서 데이터베이스 B로 EMPLOYEE 테이블을 복사하고 데이터베이스 A에서 데이터베이스 B로 테이블에 변경 사항을 복제할 수 있습니다. 또한 데이터베이스 B에서 A로 다시 EMPLOYEE 테이블에 변경 사항을 복제할 수 있습니다. 즉, 양방향 복제를 수행할 수 있습니다.

> **Note**
>
> AWS DMS 양방향 복제는 프라이머리 노드, 충돌 해결 등을 포함해 완전한 멀티 마스터 솔루션으로 사용할 수 없습니다.

서로 다른 노드의 데이터가 운영상 분리되는 경우에 양방향 복제를 사용합니다. 즉, 노드 A에서 작동하는 애플리케이션에 의해 변경된 데이터 요소가 있고 노드 A가 노드 B와 양방향 복제를 수행하는 경우, 노드 A의 데이터 요소는 노드 B에서 작동하는 애플리케이션에 의해 변경되지 않습니다.

AWS DMS는 다음과 같은 데이터베이스 엔진에서 양방향 복제를 지원합니다.

- Oracle
- SQL Server
- MySQL
- PostgreSQL
- Amazon Aurora MySQL 호환 버전
- Aurora PostgreSQL 호환 버전

양방향 복제 작업 생성

AWS DMS 양방향 복제를 활성화하려면 두 데이터베이스(A 및 B)에 대해 소스 엔드포인트 및 대상 엔드포인트를 구성합니다. 예를 들어 데이터베이스 A의 소스 엔드포인트, 데이터베이스 B의 소스 엔드포인트, 데이터베이스 A의 대상 엔드포인트 및 데이터베이스 B의 대상 엔드포인트을 구성합니다.

그런 다음 두 가지 작업(데이터를 대상 B로 이동하기 위한 소스 A에 대한 작업과 데이터를 대상 A로 이동하기 위한 소스 B에 대한 작업)을 만듭니다. 또한 각 작업이 루프백 방지로 구성되어 있는지 확인합니다. 작업이 루프백 방지가 설정되어 있으면 동일한 변경 사항이 두 작업의 대상에 적용되지 않으므로 하나 이상의 작업에 대한 데이터가 손상됩니다. 자세한 내용은 루프백 방지 섹션을 참조하세요.

가장 간단한 접근 방식은 데이터베이스 A와 데이터베이스 B 모두에서 동일한 데이터 세트로 시작합니다. 그런 다음 두 개의 CDC 전용 작업(A에서 B로 데이터를 복제하기 위한 작업과 B에서 A로 데이터를 복제하기 위한 작업)을 만듭니다.

노드 A에서 노드 B의 새 데이터 세트(데이터베이스)를 인스턴스화하는 데 AWS DMS를 사용하려면 다음을 수행합니다.

1. 전체 로드 및 CDC 작업을 사용하여 데이터베이스 A에서 B로 데이터를 이동합니다. 이 때 데이터베이스 B의 데이터를 수정하는 애플리케이션이 없는지 확인합니다.
2. 전체 로드가 완료되고 애플리케이션이 데이터베이스 B의 데이터를 수정할 수 있도록 허용되기 전에 데이터베이스 B의 시작 시간 또는 CDC 시작 위치를 기록해 둡니다. 자세한 내용은 CDC 시작 지점에서 복제를 시작 단원을 참조하십시오.
3. 이 시작 시간 또는 CDC 시작 위치를 사용하여 데이터베이스 B에서 A로 데이터를 다시 이동하는 CDC 전용 작업을 만듭니다.

> **Note**
>
> 양방향 쌍에서 하나의 작업만 전체 로드 및 CDC일 수 있습니다.

루프백 방지

루프백 방지를 표시하기 위해, 작업 T1에서 AWS DMS가 소스 데이터베이스 A에서 변경 로그를 읽고 변경 사항을 대상 데이터베이스 B에 적용한다고 가정하겠습니다.

그런 다음 두 번째 작업 T2는 소스 데이터베이스 B에서 변경 로그를 읽고 변경 사항을 대상 데이터베이스 A에 다시 적용합니다. T2가 이를 수행하기 전에 DMS는 소스 데이터베이스 A의 대상 데이터베이스 B에 적용된 동일한 변경 사항이 소스 데이터베이스 A에 적용되지 않았는지 확인해야 합니다. 즉, DMS는 이러한 변경 사항이 대상 데이터베이스 A에 반영(순환)되지 않는지 확인해야 합니다. 그렇지 않으면 데이터베이스 A의 데이터가 손상될 수 있습니다.

변경 사항의 루프백을 방지하려면 각 양방향 복제 작업에 다음 작업 설정을 추가합니다. 이렇게 하면 루프백 데이터 손상이 어느 방향으로도 발생하지 않습니다.

```
{
. . .

  "LoopbackPreventionSettings": {
    "EnableLoopbackPrevention": Boolean,
    "SourceSchema": String,
    "TargetSchema": String
  },

. . .
}
```

LoopbackPreventionSettings 작업 설정은 트랜잭션이 새 트랜잭션인지 또는 반대 복제 작업의 에코인지를 결정합니다. AWS DMS가 대상 데이터베이스에 트랜잭션을 적용하면 변경 표시와 함께 DMS 테이블(awsdms_loopback_prevention)을 갱신됩니다. 각 트랜잭션을 대상에 적용하기 전에 DMS는 이 awsdms_loopback_prevention 테이블에 대한 참조를 포함하는 모든 트랜잭션을 무시합니다. 따라서 변경 사항이 적용되지 않습니다.

양방향 쌍으로 된 각 복제 작업에 이러한 작업 설정을 포함합니다. 이러한 설정은 루프백 방지를 활성화합니다. 또한 각 엔드포인트에 대한 awsdms_loopback_prevention 테이블을 포함하는 작업의 각 소스 데이터베이스 및 대상 데이터베이스에 대한 스키마를 지정합니다.

각 작업에서 이러한 에코를 식별하고 무시하도록 하려면 EnableLoopbackPrevention을 true로 설정합니다. awsdms_loopback_prevention을 포함하는 소스에서 스키마를 지정하려면 SourceSchema를 소스 데이터베이스의 해당 스키마 이름으로 설정합니다. 동일한 테이블을 포함하는 대상에서 스키마를 지정하려면 TargetSchema를 대상 데이터베이스의 해당 스키마 이름으로 설정합니다.

다음 예제에서는 복제 작업 T1 및 해당 반대 복제 작업 T2에 대한 SourceSchema 및 TargetSchema 설정이 반대 설정으로 지정됩니다.

작업 T1에 대한 설정은 다음과 같습니다.

```
{
. . .

  "LoopbackPreventionSettings": {
    "EnableLoopbackPrevention": true,
    "SourceSchema": "LOOP-DATA",
    "TargetSchema": "loop-data"
  },

. . .
}
```

반대 작업 T2에 대한 설정은 다음과 같습니다.

```
{
. . .

  "LoopbackPreventionSettings": {
    "EnableLoopbackPrevention": true,
    "SourceSchema": "loop-data",
    "TargetSchema": "LOOP-DATA"
  },

. . .
}
```

> **Note**
>
> AWS CLI를 사용할 때는 create-replication-task 또는 modify-replication-task 명령만 사용하여 양방향 복제 작업에서 LoopbackPreventionSettings를 구성합니다.

양방향 복제의 제한 사항

AWS DMS에 대한 양방향 복제에는 다음과 같은 제한 사항이 있습니다.

- 루프백 방지는 데이터 조작 언어(DML) 문만 추적합니다. AWS DMS는 데이터 정의 언어(DDL) 루프백 방지를 지원하지 않습니다. 이렇게 하려면 양방향 쌍의 작업 중 하나를 구성하여 DDL 문을 필터링합니다.
- 루프백 방지를 사용하는 작업은 배치로 변경 사항을 커밋하는 것을 지원하지 않습니다. 루프백 방지로 작업을 구성하려면 `BatchApplyEnabled`를 `false`로 설정해야 합니다.
- DMS 양방향 복제에는 충돌 감지 또는 충돌 해결이 포함되지 않습니다. 데이터 불일치를 감지하려면 두 작업 모두에서 데이터 검증을 사용합니다.

작업 수정

작업 설정, 테이블 매핑 또는 기타 설정을 변경해야 하는 경우 작업을 수정할 수 있습니다. 수정된 태스크를 실행하기 전에 마이그레이션 전 태스크 평가를 활성화하고 실행할 수 있습니다. 콘솔에서 태스크를 수정하려면 태스크를 선택하고 수정을 선택합니다. CLI 명령 또는 API 작업 ModifyReplicationTask를 사용할 수도 있습니다.

작업을 수정하는 몇 가지 제한 사항이 있습니다. 여기에는 다음이 포함됩니다.

- 작업의 원본 또는 대상 엔드포인트를 수정할 수 없습니다.
- 작업의 마이그레이션 유형은 변경할 수 없습니다.
- 실행한 태스크의 상태가 중지됨 또는 실패여야 수정할 수 있습니다.

태스크 이동

사용 사례에 다음과 같은 상황이 적용될 경우 태스크를 다른 복제 인스턴스로 이동할 수 있습니다.

- 현재 특정 유형의 인스턴스를 사용 중이고 다른 인스턴스 유형으로 전환하려고 하는 경우.
- 현재 인스턴스에 많은 복제 태스크로 인해 과부하가 걸린 상태이고, 부하를 여러 인스턴스로 분산하려고 하는 경우.
- 인스턴스 스토리지가 꽉 차 있고, 스토리지나 컴퓨팅이 크기를 조정하는 대신 해당 인스턴스에서 더 강력한 인스턴스로 태스크를 이동하려는 경우.
- AWS DMS의 새로 출시된 기능을 사용하고 싶지만, 새 태스크를 생성하고 마이그레이션을 재시작하고 싶지는 않은 경우. 그 대신 해당 기능을 지원하는 새 AWS DMS 버전으로 복제 인스턴스를 실행하고, 기존 태스크를 해당 인스턴스로 이동하려는 경우.

콘솔에서 태스크를 이동하려면 태스크를 선택하고 이동을 선택합니다. CLI 명령 또는 API 작업 `MoveReplicationTask`를 실행하여 태스크를 이동할 수도 있습니다. 데이터베이스 엔진을 대상 엔드포인트로 보유한 태스크를 이동할 수 있습니다.

대상 복제 인스턴스에 이동 중인 태스크를 수용할 수 있는 충분한 스토리지 공간이 있는지 확인합니다. 그렇지 않을 경우, 태스크를 이동하기 전에 스토리지의 크기를 조정하여 대상 복제 인스턴스를 위한 공간을 확보합니다.

또한 대상 복제 인스턴스가 현재의 복제 인스턴스와 동일하거나 더 높은 AWS DMS 엔진 버전을 사용하여 생성되었는지 확인합니다.

> **Note**
>
> - 태스크는 현재 상주하고 있는 동일한 복제 인스턴스로 이동할 수 없습니다.
> - 이동 중에는 태스크 설정을 수정할 수 없습니다.
> - 태스크를 이동하려면 실행한 태스크의 상태가 중지됨, 실패 또는 이동 실패 상태여야 합니다.

DMS 태스크 이동과 관련된 태스크 상태로는 이동 중과 이동 실패라는 두 가지 상태가 있습니다. 이러한 태스크 상태에 대한 자세한 내용은 작업 상태 섹션을 참조하세요.

태스크를 이동한 후에는 이동된 태스크를 실행하기 전에 마이그레이션 전 평가를 활성화하고 실행하여 차단 문제가 있는지 확인할 수 있습니다.

작업 중 테이블 다시 로드

작업을 실행하는 동안 원본의 데이터를 사용하여 대상 데이터베이스 테이블을 다시 로드할 수 있습니다. 작업 중에 파티션 작업으로 인해 오류가 발생하거나 데이터가 변경될 경우(예: Oracle을 사용하는 경우) 테이블을 다시 로드해야 할 수도 있습니다. 작업에서 최대 10개 테이블을 다시 로드할 수 있습니다.

테이블을 다시 로드해도 태스크가 중단되지는 않습니다.

테이블을 다시 로드하려면, 다음 조건을 적용해야 합니다.

- 작업이 실행 중이어야 합니다.
- 작업의 마이그레이션 방법이 [Full Load] 또는 [Full Load with CDC]여야 합니다.

- 중복 테이블은 허용되지 않습니다.
- AWS DMS는 이전에 읽은 테이블 정의를 보관하며 재로드 작업 중에 이를 다시 생성하지 않습니다. ALTER TABLE ADD COLUMN 또는 DROP COLUMN 등 테이블이 다시 로드되기 전에 테이블에 만들어진 DDL 문으로 인해 다시 로드 작업이 실패할 수 있습니다.

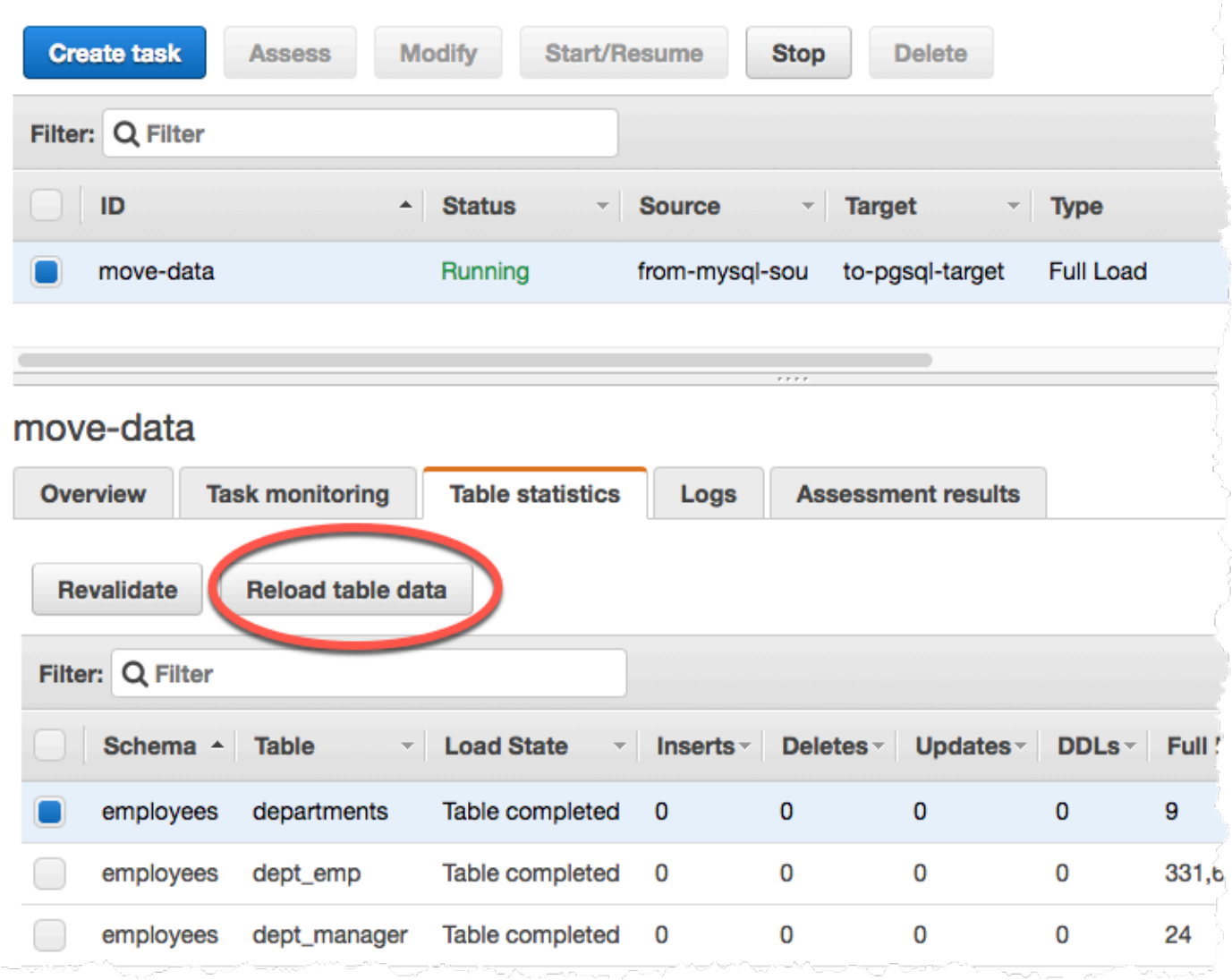
> **Note**
>
> DMS는 테이블을 다시 로드하기 전에 `TargetTablePrepMode` 설정을 적용합니다. `TargetTablePrepMode`를 `DO_NOTHING`으로 설정하는 경우 먼저 테이블을 수동으로 잘라야 합니다.

AWS Management Console

AWS DMS 콘솔을 사용하여 테이블을 다시 로드하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/dms/v2/에서 AWS DMS 콘솔을 엽니다.

   IAM 사용자로 로그인한 경우, AWS DMS에 액세스하려면 적절한 권한이 있어야 합니다. 필요한 권한에 관한 자세한 내용은 AWS DMS 사용에 필요한 IAM 권한 단원을 참조하십시오.
2. 탐색 창에서 [작업]을 선택합니다.
3. 다시 로드할 테이블이 있는 실행 중인 작업을 선택합니다.
4. 테이블 통계 탭을 선택합니다.



5. 다시 로드할 테이블을 선택합니다. 작업이 더 이상 실행되고 있지 않으면 테이블을 다시 로드할 수 없습니다.
6. Reload table data(테이블 데이터 다시 로드)를 선택합니다.

AWS DMS가 테이블을 다시 로드하려고 준비하고 있는 경우 콘솔은 테이블 상태를 Table is being reloaded(테이블 다시 로드 중)로 변경합니다.

작업 설정을 지정하기 위한 테이블 매핑 사용

테이블 매핑에서는 여러 유형의 규칙을 사용하여 데이터 소스, 소스 스키마, 데이터, 작업 중 발생하는 모든 변환을 지정합니다. 테이블 매핑을 사용하여 마이그레이션할 데이터베이스의 개별 테이블과 마이그레이션에 사용할 스키마를 지정할 수 있습니다.

테이블 매핑 작업 시 필터를 사용하여 테이블 열에서 복제할 데이터를 지정할 수 있습니다. 또한 변환을 사용하여 선택한 스키마, 테이블 또는 뷰를 대상 데이터베이스에 기록하기 전에 수정할 수 있습니다.

주제

- 콘솔에서 테이블 선택 및 변환 규칙 지정
- JSON을 사용하여 테이블 선택 및 변환 지정
- 선택 규칙 및 작업
- 테이블 매핑의 와일드카드
- 변환 규칙 및 작업
- 변환 규칙 표현식을 사용하여 열 내용 정의
- 테이블 및 컬렉션 설정 규칙과 작업

> **Note**
>
> MongoDB 소스 엔드포인트에 대한 테이블 매핑 작업을 수행하는 경우, 필터를 사용하여 복제할 데이터를 지정하고 schema_name 대신 데이터베이스 이름을 지정할 수 있습니다. 아니면 기본 "%"를 사용할 수 있습니다.

콘솔에서 테이블 선택 및 변환 규칙 지정

를 사용하여 테이블 선택 및 AWS Management Console 변환 지정을 비롯한 테이블 매핑을 수행할 수 있습니다. 콘솔에서 위치 섹션을 사용하여 스키마, 테이블, 작업을 지정합니다(포함 또는 제외). 필터 섹션을 사용하여 테이블에서 열 이름과 복제 작업에 적용할 조건을 지정합니다. 이와 함께 이 두 작업은 선택 규칙을 생성합니다.

1개 이상의 선택 규칙을 지정한 후 테이블 매핑에 변환을 포함시킬 수 있습니다. 변환을 사용하여 스키마나 테이블 이름을 변경하거나, 스키마나 테이블에 접두사나 접미사를 추가하거나, 또는 테이블 열을 제거할 수 있습니다.

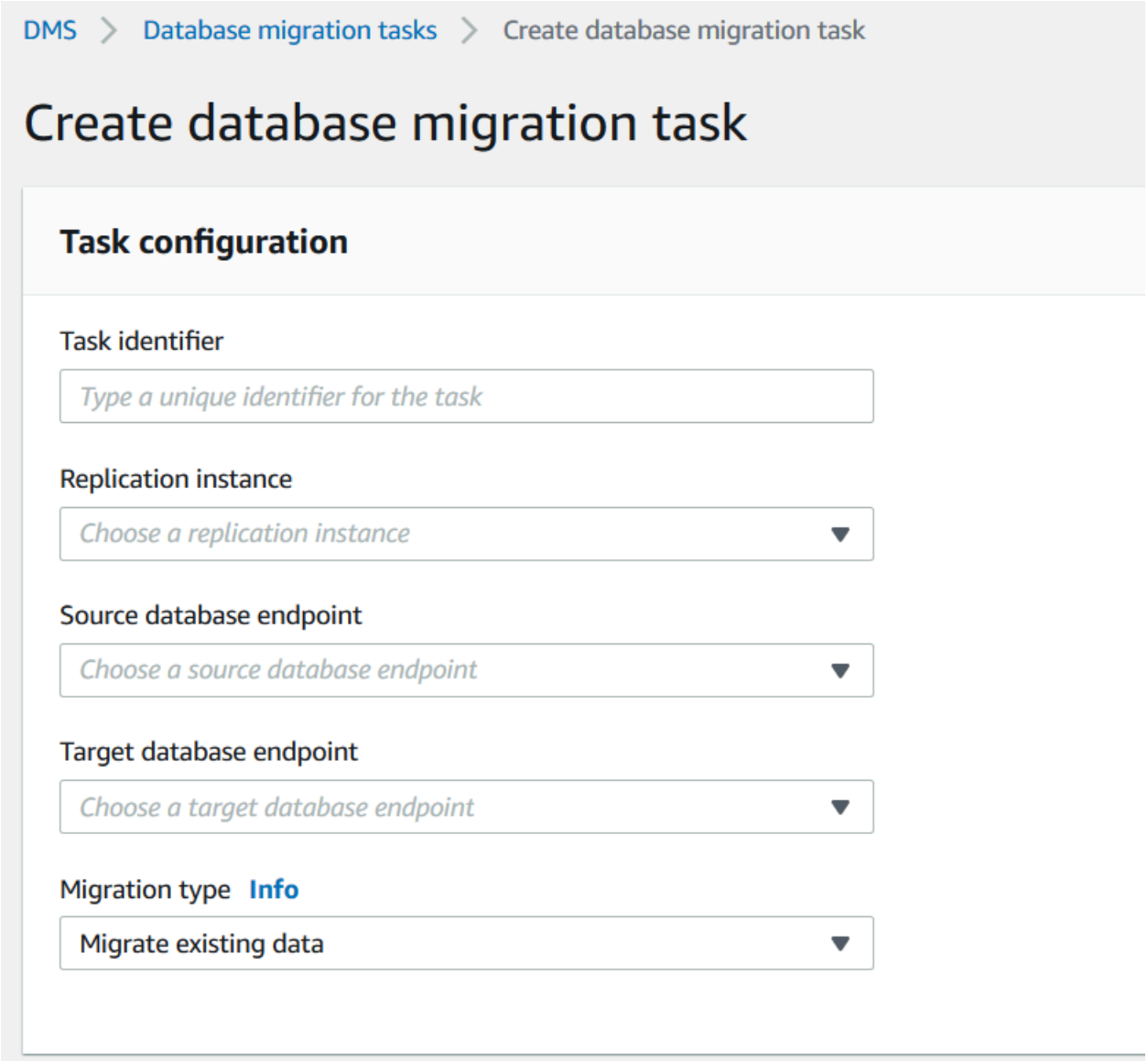
> **Note**
>
> AWS DMS 스키마 수준, 테이블 수준 또는 열 수준당 둘 이상의 변환 규칙을 지원하지 않습니다.

다음 절차에서는 **EntertainmentAgencySample**이라는 스키마에서 호출된 **Customers**라는 테이블을 기반으로 선택 규칙을 설정하는 방법을 보여줍니다.

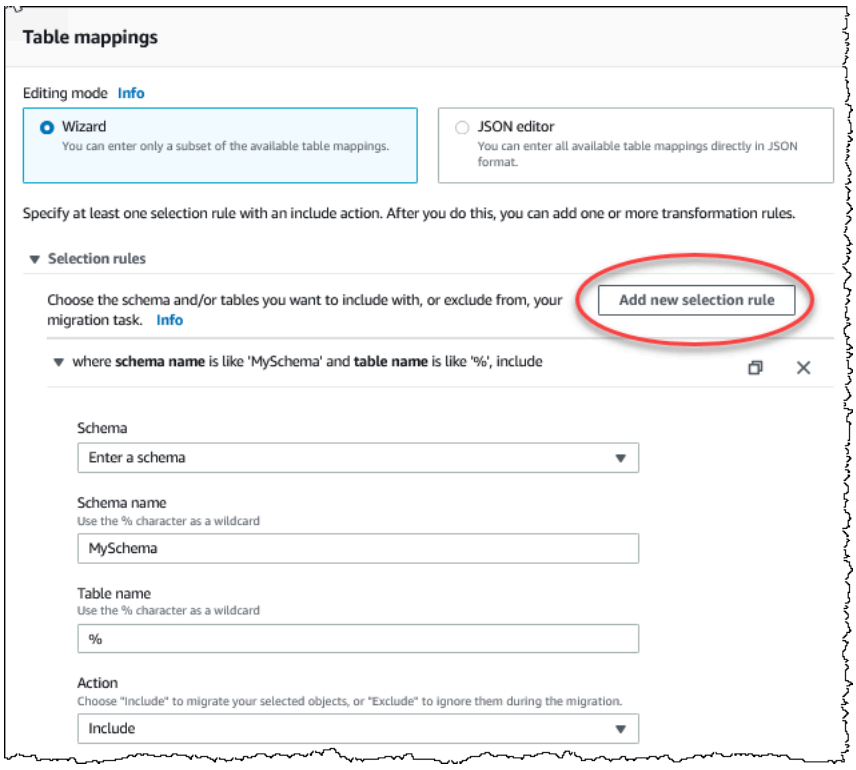
콘솔을 사용하여 테이블 선택, 필터 기준, 변환을 지정하려면

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.

   IAM 사용자로 로그인한 경우, AWS DMS에 액세스하려면 적절한 권한이 있어야 합니다. 필요한 권한에 관한 자세한 내용은 AWS DMS 사용에 필요한 IAM 권한 단원을 참조하십시오.
2. 대시보드 페이지에서 데이터베이스 마이그레이션 작업을 선택합니다.
3. 작업 생성을 선택합니다.
4. 작업 구성 섹션에서 작업 식별자, 복제 인스턴스, 소스 데이터베이스 엔드포인트, 대상 데이터베이스 엔드포인트, 마이그레이션 유형 등의 작업 정보를 입력합니다.



5. 테이블 매핑 섹션에서 스키마 이름과 테이블 이름을 입력합니다. 스키마 이름 또는 테이블 이름을 지정할 때 "%"를 와일드카드 값으로 사용할 수 있습니다. 사용할 수 있는 다른 와일드카드에 관한 자세한 내용은 the section called “테이블 매핑의 와일드카드”을 참조하십시오. 수행할 작업을 지정하여 필터에서 정의한 데이터를 포함하거나 제외합니다.

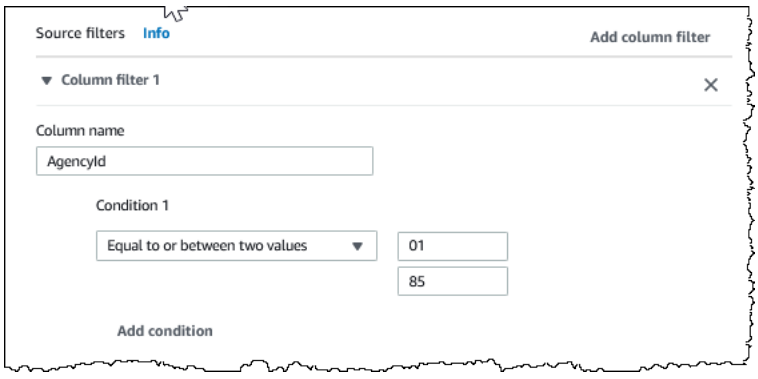


6. 열 필터 추가와 조건 추가 링크를 사용하여 필터 정보를 지정합니다.

   a. 열 필터 추가를 선택하여 열과 조건을 지정합니다.

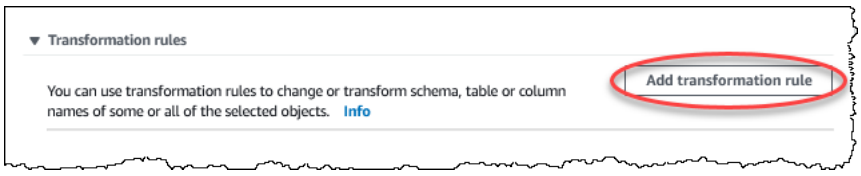
   b. 조건 추가를 선택하여 조건을 추가할 수 있습니다.

   다음은 **01** 및 **85** 사이에 **AgencyIDs**를 포함하는 **Customers** 테이블에 대한 필터를 보여주는 예입니다.

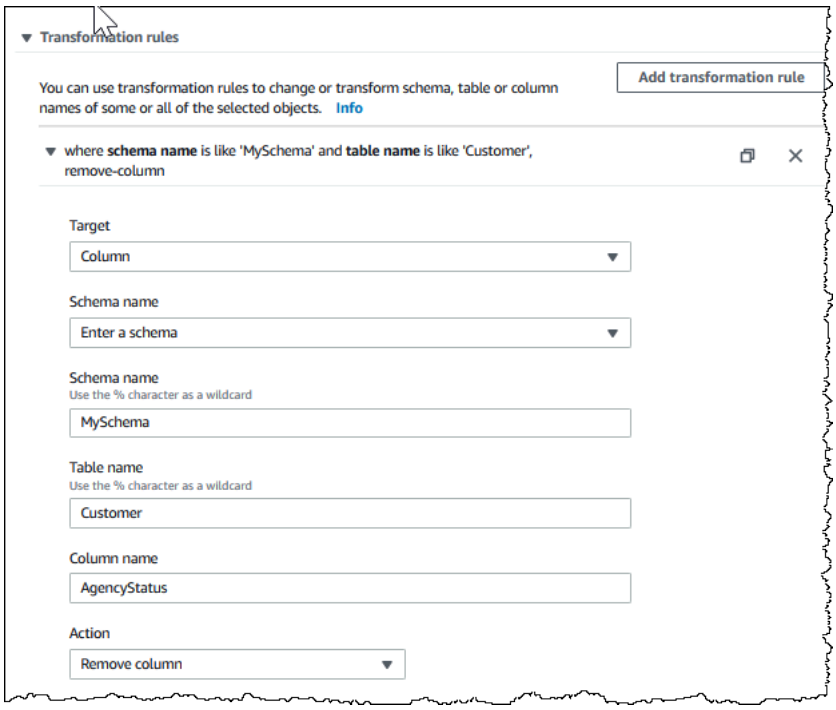


7. 원하는 대로 선택하고 새 선택 규칙 추가를 선택합니다.

8. 최소 1개 이상의 선택 규칙을 생성한 후 작업에 변환을 추가할 수 있습니다. 변환 규칙 추가를 선택합니다.



9. 변환할 대상을 선택하고 요청된 추가 정보를 입력합니다. 다음은 **`Customer`** 테이블에서 **`AgencyStatus`** 열을 삭제하는 변환을 보여주는 예입니다.



10. 변환 규칙 추가를 선택합니다.
11. 작업 생성을 선택합니다.

> **Note**
>
> AWS DMS 스키마 수준 또는 테이블 수준당 둘 이상의 변환 규칙을 지원하지 않습니다.

JSON을 사용하여 테이블 선택 및 변환 지정

마이그레이션 중에 적용할 테이블 매핑을 지정하려면 JSON 파일을 만들 수 있습니다. 콘솔을 사용하여 마이그레이션 작업을 만드는 경우, 이 JSON 파일을 찾아보거나 JSON을 테이블 매핑 상자에 직접 입력할 수 있습니다. CLI 또는 API를 사용하여 마이그레이션을 수행하는 경우, `CreateReplicationTask` 또는 `ModifyReplicationTask` API 작업의 `TableMappings` 파라미터를 사용하여 이 파일을 지정할 수 있습니다.

AWS DMS 최대 2MB 크기의 테이블 매핑 JSON 파일만 처리할 수 있습니다. DMS 작업을 수행하는 동안에는 매핑 규칙 JSON 파일 크기를 2MB 미만으로 유지하는 것이 좋습니다. 이렇게 하면 작업 생성 또는 수정 중에 예상치 못한 오류가 발생하지 않습니다. 매핑 규칙 파일이 2MB 제한을 초과하는 경우, 테이블을 여러 작업으로 분할하여 매핑 규칙 파일이 이 한도 이하로 유지되도록 크기를 줄이는 것이 좋습니다.

사용할 테이블, 뷰 및 스키마를 지정할 수 있습니다. 또한 테이블, 뷰 및 스키마 변환을 수행하고 AWS DMS 에서 개별 테이블 및 뷰를 로드하는 방법에 대한 설정을 지정할 수 있습니다. 다음 규칙 유형을 사용하여 이러한 옵션에 대한 테이블 매핑 규칙을 만듭니다.

- `selection` 규칙 – 로드할 소스 테이블, 뷰 및 스키마의 유형과 이름을 식별합니다. 자세한 정보는 선택 규칙 및 작업을 참조하세요.
- `transformation` 규칙 – 대상에 로드되기 전에 소스의 특정 소스 테이블 및 스키마에 대한 특정 변경 사항 또는 추가 사항을 지정합니다. 자세한 정보는 변환 규칙 및 작업을 참조하세요.

  또한 변환 규칙 내에서 표현식을 사용하여 새 열과 기존 열의 내용을 정의할 수 있습니다. 자세한 정보는 변환 규칙 표현식을 사용하여 열 내용 정의을 참조하세요.
- `table-settings` 규칙 – DMS 작업이 개별 테이블에 대한 데이터를 로드하는 방법을 지정합니다. 자세한 정보는 테이블 및 컬렉션 설정 규칙과 작업을 참조하세요.

> **Note**
>
> Amazon S3 대상의 경우, `post-processing` 규칙 유형과 `add-tag` 규칙 작업을 사용하여 선택된 테이블 및 스키마로 매핑된 S3 객체에 태그를 지정할 수도 있습니다. 자세한 정보는 Amazon S3 객체 태그 지정을 참조하세요.
> 다음 대상의 경우 `object-mapping` 규칙 유형을 사용하여 선택한 스키마와 테이블을 대상으로 마이그레이션하는 방법과 위치를 지정할 수 있습니다.
>
> - Amazon DynamoDB – 자세한 내용은 객체 매핑을 사용하여 데이터를 DynamoDB로 마이그레이션을 참조하세요.
> - Amazon Kinesis – 자세한 내용은 객체 매핑을 사용하여 데이터를 Kinesis 데이터 스트림으로 마이그레이션을 참조하세요.
> - Apache Kafka – 자세한 내용은 객체 매핑을 사용하여 데이터를 Kafka 주제로 마이그레이션을 참조하세요.

선택 규칙 및 작업

테이블 매핑을 사용하여 사용할 테이블, 뷰 및 스키마를 지정하기 위해 선택 규칙과 작업을 이용할 수 있습니다. 선택 규칙 유형을 사용하는 테이블 매핑 규칙의 경우, 다음 값을 적용할 수 있습니다.

파라미터	가능한 값	설명
rule-type	selection	선택 규칙입니다. 테이블 매핑을 지정할 때에는 1개 이상의 선택 규칙을 정의합니다.
rule-id	숫자 값.	규칙을 식별하기 위한 고유한 숫자 값입니다.
rule-name	영숫자 값입니다.	규칙을 식별하기 위한 고유한 이름입니다.
rule-action	include, exclude, explicit	규칙에서 선택한 특정 객체(단일 또는 다수)를 포함하거나 제외하는 값입니다. explicit이 지정되면 명시적으로 지정된 테이블 및 스키마에 해당하는 객체 하나만을 선택 및 포함할 수 있습니다.
object-locator	객체는 다음 파라미터를 사용합니다. • schema-name – 스키마의 이름입니다. • table-name – 테이블의 이름입니다. • (선택 사항) table-type – table | view | all은 table-name 이 테이블, 뷰 또는 테이블과 뷰 모두를 지칭하는지를 나타냅니다. 기본값은 table입니다. AWS DMS 전체 로드 작업에서만 뷰를 로드합니다. 전체 로드 및 변경	규칙이 적용되는 각 스키마와 테이블 또는 뷰의 이름입니다. 또한 규칙에 테이블만 포함되는지, 뷰만 포함되는지, 또는 테이블과 뷰가 모두 포함되는지 여부를 지정할 수도 있습니다. rule-action 이 include이거나 exclude인 경우 각 schema-name 및 table-name 파라미터의 값의 전부 또는 일부에 대한 와일드카드로 '%' 퍼센트 기호를 사용할 수 있습니다. 사용할 수 있는 다른 와일드카드에 관한 자세한 내용은 the section called "테이블 매핑의 와일드카드"을 참조하

파라미터	가능한 값	설명
	데이터 캡처 (CDC) 작업만 있는 경우 뷰를 로드하도록 하나 full-load-only 이상의 작업을 구성하십시오. 모든 대상 엔드포인트가 전체 부하 상태에서도 뷰를 복제 소스로 받아들이는 것은 아닙니다 (예: Amazon OpenSearch Service). 대상 엔드포인트의 한계를 확인하십시오.	십시오. 따라서 다음 항목들을 일치시킬 수 있습니다. • 단일 스키마의 단일한 테이블, 뷰 또는 컬렉션 • 일부 또는 전체 스키마의 단일한 테이블, 뷰 또는 컬렉션 • 단일 스키마의 일부 또는 모든 테이블과 뷰 또는 단일 데이터베이스의 컬렉션 • 일부 또는 모든 스키마의 일부 또는 전체 테이블과 뷰 또는 일부 또는 전체 데이터베이스의 컬렉션 `rule-action` 이 `explicit`인 경우, 단일 테이블 또는 뷰 및 스키마의 정확한 이름만을 지정할 수 있습니다 (와일드카드 없음). 뷰에 지원되는 소스에는 다음이 포함됩니다. • Oracle • Microsoft SQL Server • PostgreSQL • IBM Db2 LUW • IBM Db2 z/OS • SAP Adaptive Server Enterprise(ASE) • MySQL • AURORA • AURORA Serverless • MariaDB

파라미터	가능한 값	설명
		Note AWS DMS 소스 뷰를 타겟 뷰에 로드하지 마십시오. 소스 뷰는 소스의 뷰와 이름이 동일한 대상의 동등한 테이블에 로드됩니다. 컬렉션이 포함된 데이터베이스에 대해 지원되는 소스는 다음과 같습니다. • MongoDB • Amazon DocumentDB
`load-order`	양의 정수입니다. 최대값은 2,147,483,647입니다.	테이블 및 뷰를 로드하기 위한 우선순위입니다. 우선순위 값이 높은 테이블 및 뷰가 먼저 로드됩니다.
`filters`	객체의 배열입니다.	소스를 필터링하기 위한 하나 이상의 객체입니다. 소스의 단일 열에서 필터링할 객체 파라미터를 지정합니다. 여러 열에서 필터링할 여러 객체를 지정합니다. 자세한 정보는 소스 필터 사용을 참조하세요.

Example 스키마에서 모든 테이블 마이그레이션

다음은 소스에서 이름이 `Test`인 스키마의 모든 테이블을 대상 엔드포인트로 마이그레이션하는 예입니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
```

```
            "rule-name": "1",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            },
            "rule-action": "include"
        }
    ]
}
```

Example 스키마에서 일부 테이블 마이그레이션

다음은 소스에서 이름이 Test인 스키마에서 DMS로 시작하는 테이블을 제외한 모든 테이블을 대상 엔드포인트로 마이그레이션하는 예입니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "selection",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "DMS%"
            },
            "rule-action": "exclude"
        }
    ]
}
```

Example 단일 스키마에서 지정된 단일 테이블 마이그레이션

다음은 소스의 NewCust 스키마에서 Customer 테이블을 대상 엔드포인트로 마이그레이션하는 예입니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "NewCust",
                "table-name": "Customer"
            },
            "rule-action": "explicit"
        }
    ]
}
```

> **Note**
>
> 여러 선택 규칙을 지정하여 여러 테이블 및 스키마에서 명시적으로 선택할 수 있습니다.

Example 설정된 순서대로 테이블 마이그레이션

다음은 두 테이블을 마이그레이션하는 예제입니다. 테이블 loadfirst (우선순위 1) 이 loadsecond 테이블보다 먼저 초기화됩니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "loadsecond"
            },
            "rule-action": "include",
```

```
            "load-order": "2"
        },
        {
            "rule-type": "selection",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "loadfirst"
            },
            "rule-action": "include",
            "load-order": "1"
        }
    ]
}
```

> **Note**
>
> `load-order`는 테이블 초기화에 적용됩니다. `MaxFullLoadSubTasks`가 1보다 큰 경우 연속 테이블 로드는 이전 테이블 로드가 완료될 때까지 기다리지 않습니다.

Example 스키마에서 일부 뷰 마이그레이션

다음은 소스에서 이름이 `Test`인 스키마의 일부 뷰를 대상의 동등한 테이블로 마이그레이션하는 예입니다.

```
{
   "rules": [
        {
           "rule-type": "selection",
           "rule-id": "2",
           "rule-name": "2",
           "object-locator": {
               "schema-name": "Test",
               "table-name": "view_DMS%",
               "table-type": "view"
            },
           "rule-action": "include"
        }
    ]
}
```

Example 스키마에서 모든 테이블 및 뷰 마이그레이션

다음은 소스에서 이름이 `report`인 스키마의 모든 테이블 및 뷰를 대상의 동등한 테이블로 마이그레이션하는 예입니다.

```
{
   "rules": [
        {
           "rule-type": "selection",
           "rule-id": "3",
           "rule-name": "3",
           "object-locator": {
               "schema-name": "report",
               "table-name": "%",
               "table-type": "all"
            },
           "rule-action": "include"
        }
    ]
}
```

테이블 매핑의 와일드카드

이 섹션에서는 테이블 매핑을 위한 스키마와 테이블 이름을 지정할 때 사용할 수 있는 와일드카드에 대해 설명합니다.

와일드카드	일치 항목
%	Zero or more characters
_	A single character
[_]	A literal underscore character
[ab]	A set of characters. For example, [ab] matches either 'a' or 'b'.
[a-d]	A range of characters. For example,[a-d] matches either 'a', 'b', 'c', or 'd'.

Oracle 소스 및 대상 엔드포인트의 경우, escapeCharacter 추가 연결 속성을 사용하여 이스케이프 문자를 지정할 수 있습니다. 이스케이프 문자를 사용하면 와일드카드 문자를 와일드카드가 아닌 것처럼 표현식에 사용할 수 있습니다. 예를 들어, escapeCharacter=#를 사용하면 이 샘플 코드에서처럼 표현식에 '#'을 사용하여 와일드카드 문자가 일반 문자로 작동하도록 할 수 있습니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "542485267",
            "rule-name": "542485267",
            "object-locator": { "schema-name": "ROOT", "table-name": "TEST#_T%" },
            "rule-action": "include",
            "filters": []
        }
    ]
}
```

여기서 '#' 이스케이프 문자는 '_' 와일드카드 문자가 일반 문자처럼 작동하도록 합니다. AWS DMS 이름이 지정된 ROOT 스키마에서 테이블을 선택합니다. 여기서 각 테이블에는 TEST_T 접두사로 이 이름이 지정됩니다.

변환 규칙 및 작업

변환 작업을 사용하여 선택한 스키마, 테이블 또는 뷰에 적용할 변환을 지정합니다. 변환 규칙은 선택적입니다.

제한 사항

- 동일한 객체(스키마, 테이블, 열, 테이블-테이블스페이스 또는 인덱스-테이블스페이스)에 대해 둘 이상의 변환 규칙 작업을 적용할 수 없습니다. 각 변형 작업이 서로 다른 객체에 적용되는 한, 모든 레벨에 여러 변환 규칙 작업을 적용할 수 있습니다.
- 변환 규칙의 테이블 이름과 열 이름은 대소문자를 구분합니다. 예를 들어, Oracle 또는 Db2 데이터베이스의 테이블 이름과 열 이름을 대문자로 제공해야 합니다.
- 오른쪽에서 왼쪽으로 쓰는 언어를 사용하는 열 이름에는 변환이 지원되지 않습니다.
- 이름에 특수 문자(예: #, \, /, -)가 포함된 열에서는 변환을 수행할 수 없습니다.
- BLOB/CLOB 데이터 형식에 매핑된 열에 대해 지원되는 유일한 변환은 대상에서 열을 삭제하는 것입니다.

- AWS DMS 두 개의 원본 테이블을 단일 대상 테이블에 복제하는 것은 지원하지 않습니다. AWS DMS 복제 작업의 변환 규칙에 따라 테이블에서 테이블로, 열에서 열로 레코드를 복제합니다. 객체 이름은 중복되지 않도록 고유한 이름이어야 합니다.

  예를 들어, 소스 테이블에는 이름이 `ID`인 열이 있고 그에 상응하는 대상 테이블에는 `id`라는 기존 열이 있습니다. 하나의 규칙에서 `ADD-COLUMN` 문을 사용하여 `id`라는 새 열을 추가하고 SQLite 문을 사용하여 열을 사용자 지정 값으로 채우는 경우, 이름이 `id`인 모호한 중복 객체가 생성되며 이 객체는 지원되지 않습니다.

값

변환 규칙 유형을 사용하는 테이블 매핑 규칙의 경우, 다음 값을 적용할 수 있습니다.

파라미터	가능한 값	설명
`rule-type`	`transformation`	선택 규칙에서 지정한 각 객체에 규칙을 적용하는 값입니다. 달리 지정되지 않는 한, `transformation` 을 사용합니다.
`rule-id`	숫자 값.	규칙을 식별하기 위한 고유한 숫자 값입니다. 동일한 객체 (스키마, 테이블, 열, 테이블 간 공간 또는 인덱스 테이블 공간) 에 대해 여러 변환 규칙을 지정하는 경우 는 하위 rule-id를 사용하여 변환 규칙을 AWS DMS 적용합니다.
`rule-name`	영숫자 값입니다.	규칙을 식별하기 위한 고유한 이름입니다.
`object-locator`	객체는 다음 파라미터를 사용합니다. • `schema-name` – 스키마의 이름입니다. MongoDB 및 Amazon DocumentDB 엔드포인트의 경우, 컬렉션 세트를 보관하는 데이터베이스의 이름입니다.	규칙이 적용되는 각 스키마, 테이블 또는 뷰, 테이블 테이블스페이스, 인덱스 테이블스페이스 및 열의 이름입니다. 각 `object-locator` 파라미터 (`data-type` 제외) 값의 전부 또는 일부에 '%'(퍼센트 기호)를 와일드카

파라미터	가능한 값	설명
	• `table-name` – 테이블, 뷰 또는 컬렉션의 이름입니다. • `table-tablespace-name` – 기존 테이블 테이블스페이스의 이름입니다. • `index-tablespace-name` – 기존 인덱스 테이블스페이스의 이름입니다. • `column-name` – 기존 열의 이름입니다. • `data-type` – 기존 열 데이터 형식의 이름입니다.	드로 사용할 수 있습니다. 따라서 다음 항목들을 일치시킬 수 있습니다. • 단일 스키마의 단일 테이블 또는 뷰 • 일부 또는 전체 스키마의 단일 테이블 또는 뷰 • 단일 스키마의 일부 또는 전체 테이블 및 뷰 • 일부 또는 전체 스키마의 일부 또는 전체 테이블 및 뷰 • (하나 또는 둘 이상의) 지정된 테이블, 뷰 및 스키마에 있는 하나 이상의 열 • 여러 열이 지정된 경우 지정된 `data-type` 이 있는 열. `data-type` 의 가능한 값은 이 표 다음에 설명된 `data-type` 을 참조하십시오. 또한 `table-tablespace-name` 또는 `index-tablespace-name` 파라미터는 Oracle 소스 엔드포인트와 일치시키는 데만 사용할 수 있습니다. 단일 규칙에서 `table-tablespace-name` 또는 `index-tablespace-name` 를 지정할 수 있지만 둘 다 지정할 수는 없습니다. 따라서 다음 항목 중 하나와 일치시킬 수 있습니다. • 하나의, 일부 또는 모든 테이블 테이블스페이스

파라미터	가능한 값	설명
		• 하나의, 일부 또는 모든 인덱스 테이블스페이스

파라미터	가능한 값	설명
rule-action	add-column , include-column , remove-column rename convert-lowercase , convert-uppercase add-prefix , remove-prefix , replace-prefix add-suffix , remove-suffix , replace-suffix define-primary-key change-data-type add-before-image-columns	객체에 적용할 변환. 모든 변환 규칙 작업은 대소문자를 구분합니다. rule-action 파라미터의 add-column 값은 테이블에 열을 추가합니다. 하지만 같은 테이블의 기존 열과 이름이 같은 새 열을 추가할 수는 없습니다. expression 및 data-type 파라미터와 함께 사용할 경우 add-column 은 새 열 데이터의 값을 지정합니다. rule-action 에 대한 change-data-type 값은 column 규칙 대상에만 사용할 수 있습니다. rule-action 파라미터의 include-column 값은 기본적으로 모든 열을 삭제하고 지정된 열을 포함하도록 테이블 모드를 변경합니다. include-column 규칙을 여러 번 호출하면 여러 열이 대상에 포함됩니다. define-primary-key 규칙의 스키마 또는 테이블 이름에 와일드카드(%)가 있으면 이 규칙을 사용할 수 없습니다. 기존 작업의 경우 remove-column , rename 또는 add-prefix 같은 대상 테이블 스키마를 변경하는 변환 규칙 작업은 작업을 다시 시작할 때까지 적용되지 않습니다. 변환 규칙을 추가

파라미터	가능한 값	설명
		한 후 작업을 재개하면 변경된 열에 예기치 않은 동작이 발생할 수 있으며, 여기에는 열 데이터 누락이 포함될 수 있습니다. 변환 규칙이 제대로 작동하려면 작업을 다시 시작해야 합니다.
`rule-target`	`schema`, `table`, `column`, `table-tablespace` , `index-tablespace`	변환할 객체의 유형입니다. `table-tablespace` 및 `index-tablespace` 값은 Oracle 대상 엔드포인트에만 사용할 수 있습니다. `object-locator` : `table-tablespace-name` 또는 `index-tablespace-name` 이름의 일부로 지정하는 파라미터의 값을 지정해야 합니다.
`value`	대상 유형에서 명명 규칙을 따르는 영숫자 값입니다.	`rename`과 같이 입력이 필요한 작업의 새 값입니다.
`old-value`	대상 유형에서 명명 규칙을 따르는 영숫자 값입니다.	`replace-prefix` 와 같이 대체가 필요한 작업의 이전 값입니다.

파라미터	가능한 값	설명

data-type	type – rule-action 이 add-column 인 경우 사용할 데이터 형식 또는 rule-action 이 change-data-type 인 경우 대체 데이터 형식입니다.
	
	또는 rule-action 이 change-data-type 이고, column-name 의 값이 "%"이고, 기존 데이터 형식을 식별하기 위한 추가 data-type 파라미터가 object-locator 에 포함된 경우 대체 데이터 형식의 이름입니다.
	
	AWS DMS 다음 DMS 데이터 유형에 대한 열 데이터 유형 변환을 지원합니다. "bytes", "date", "time", "datetime", "int1", "int2", "int4", "int8", "numeric", "real4", "real8", "string", "uint1", "uint2", "uint4", "uint8", "wstring", "blob", "nclob", "clob", "boolean", "set", "list" "map", "tuple"
	
	precision – 추가된 열 또는 대체 데이터 형식에 정밀도가 있는 경우 정밀도를 지정하는 정수 값.
	
	scale – 추가된 열 또는 대체 데이터 형식에 척도가 있는 경우 척도를 지정하는 정수 값 또는 날짜 시간 값.
	
	length – 새 열 데이터의 길이(add-column 과 함께 사용하는 경우).
		다음은 대체할 기존 데이터 형식을 지정하는 data-type 파라미터의 예입니다.
		
		{
		"rules": [{
		"rule-type": "selection",
		"rule-id": "1",
		"rule-name": "1",
		"object-locator": {
		"schema-name": "%",
		"table-name": "%"
		},
		"rule-action": "include"
		},
		{
		"rule-type": "transformation",
		"rule-id": "2",
		"rule-name": "2",
		"rule-target": "column",
		"object-locator": {
		"schema-name": "test",
		"table-name": "table_t",
		"column-name": "col10"
		},
		"rule-action": "change-data-type",
		"data-type": {
		"type": "string",
		"length": "4092",
		"scale": ""
		}
		}
		]
		}

파라미터	가능한 값	설명
		여기서는 `table_t` 테이블의 `col10` 열이 `string` 데이터 유형으로 변경됩니다.
`expression`	SQLite 구문을 따르는 영숫자 값입니다.	`rename-schema` 로 설정된 `rule-action` 과 함께 사용되는 경우 `expression` 파라미터는 새 스키마를 지정합니다. `rename-table` 로 설정된 `rule-action` 과 함께 사용되는 경우 `expression` 은 새 테이블을 지정합니다. `rename-column` 으로 설정된 `rule-action` 과 함께 사용되는 경우 `expression` 은 새 열 이름을 지정합니다. `add-column` 으로 설정된 `rule-action` 과 함께 사용되는 경우 `expression` 은 새 열을 구성하는 데이터를 지정합니다. 이 파라미터에는 표현식만 지원된다는 점에 유의하십시오. 연산자와 명령은 지원되지 않습니다. 변환 규칙에 표현식을 사용하는 방법에 관한 자세한 내용은 변환 규칙 표현식을 사용하여 열 내용 정의 단원을 참조하십시오. SQLite 표현식에 관한 자세한 내용은 SQLite 함수를 사용하여 표현식 작성 섹션을 참조하세요.

파라미터	가능한 값	설명
primary-key-def	객체는 다음 파라미터를 사용합니다. • name – 테이블 또는 뷰에 대한 새 기본 키 또는 고유 인덱스의 이름입니다. • (선택 사항) origin – 정의할 고유 키의 유형: primary-key (기본값) 또는 unique-index . • columns – 기본 키 또는 고유 인덱스에서 나타나는 순서대로 열의 이름을 나열하는 문자열 배열입니다.	이 파라미터는 변환된 테이블 또는 뷰에서 고유 키의 이름, 유형 및 콘텐츠를 정의할 수 있습니다. 이는 rule-action 이 define-primary-key 로 설정되고 rule-target 이 table로 설정된 경우에 해당됩니다. 기본적으로 고유 키는 기본 키로 정의됩니다.

파라미터	가능한 값	설명
before-image-def	객체는 다음 파라미터를 사용합니다. • column-prefix – 열 이름 앞에 추가되는 값입니다. 기본 값은 BI_입니다. • column-suffix – 열 이름에 뒤에 추가되는 값입니다. 기본값은 비어 있음입니다. • column-filter – 다음 값 중 하나가 필요합니다. pk-only(기본값), non-lob(선택 사항) 및 all(선택 사항).	이 파라미터는 이전 이미지 열을 식별하는 명명 규칙을 정의하고 대상에 이전 이미지 열을 만들 수 있는 소스 열을 식별하는 필터를 지정합니다. rule-action 이 add-before-image-columns 로 설정되고 rule-target 이 column으로 설정된 경우 이 파라미터를 지정할 수 있습니다. column-prefix 와 column-suffix 를 모두 빈 문자열로 설정하지 마십시오. column-filter 에서 다음을 선택합니다. • pk-only – 테이블 기본 키의 일부인 열만 추가합니다. • non-lob – LOB 유형이 아닌 열만 추가합니다. • all – 이전 이미지 값이 있는 열을 추가합니다. AWS DMS 대상 엔드포인트에서 이전 이미지 지원에 관한 자세한 내용은 다음을 참조하십시오. • Kinesis 데이터 스트림 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기 • Apache Kafka 대상의 경우 이전 이미지를 사용하여 CDC 행의 원래 값 보기

예제

Example 스키마 이름 바꾸기

다음은 소스의 스키마 이름 `Test`를 대상의 `Test1`로 변경하는 예입니다.

```
{

    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "transformation",
            "rule-id": "2",
            "rule-name": "2",
            "rule-action": "rename",
            "rule-target": "schema",
            "object-locator": {
                "schema-name": "Test"
            },
            "value": "Test1"
        }
    ]
}
```

Example 테이블 이름 바꾸기

다음은 소스의 테이블 이름 `Actor`를 대상의 `Actor1`로 변경하는 예입니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
```

```
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "transformation",
            "rule-id": "2",
            "rule-name": "2",
            "rule-action": "rename",
            "rule-target": "table",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "Actor"
            },
            "value": "Actor1"
        }
    ]
}
```

Example 열 이름 바꾸기

다음은 `Actor` 테이블에서 소스의 열 이름 `first_name`을 대상의 `fname`으로 변경하는 예입니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
         {
            "rule-type": "transformation",
            "rule-id": "4",
            "rule-name": "4",
            "rule-action": "rename",
            "rule-target": "column",
            "object-locator": {
```

```
                "schema-name": "test",
                "table-name": "Actor",
                "column-name" : "first_name"
            },
            "value": "fname"
        }
    ]
}
```

Example Oracle 테이블 테이블스페이스의 이름 바꾸기

다음은 Oracle 소스에서 Actor라는 테이블에 대해 SetSpace라는 테이블 테이블스페이스의 이름을 Oracle 대상 엔드포인트의 SceneTblSpace로 바꾸는 예입니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "Play",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "transformation",
            "rule-id": "2",
            "rule-name": "2",
            "rule-action": "rename",
            "rule-target": "table-tablespace",
            "object-locator": {
                "schema-name": "Play",
                "table-name": "Actor",
                "table-tablespace-name": "SetSpace"
            },
            "value": "SceneTblSpace"
        }
    ]
}
```

Example Oracle 인덱스 테이블스페이스의 이름 바꾸기

다음은 Oracle 소스에서 Actor라는 테이블에 대해 SetISpace라는 인덱스 테이블스페이스의 이름을 Oracle 대상 엔드포인트의 SceneIdxSpace로 바꾸는 예입니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "Play",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "transformation",
            "rule-id": "2",
            "rule-name": "2",
            "rule-action": "rename",
            "rule-target": "table-tablespace",
            "object-locator": {
                "schema-name": "Play",
                "table-name": "Actor",
                "table-tablespace-name": "SetISpace"
            },
            "value": "SceneIdxSpace"
        }
    ]
}
```

Example 열 추가

다음 예에서는 스키마 test의 테이블 Actor에 datetime 열을 추가합니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
```

```
            "object-locator": {
                "schema-name": "test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "transformation",
            "rule-id": "2",
            "rule-name": "2",
            "rule-action": "add-column",
            "rule-target": "column",
            "object-locator": {
                "schema-name": "test",
                "table-name": "actor"
            },
            "value": "last_updated",
            "data-type": {
                "type": "datetime",
                "precision": 6
            }
        }
    ]
}
```

Example 열 제거

다음은 소스의 `Actor`라는 테이블을 변환하여 `col` 문자로 시작하는 모든 열을 대상에서 제거하는 예입니다.

```
{
  "rules": [{
  "rule-type": "selection",
  "rule-id": "1",
  "rule-name": "1",
  "object-locator": {
   "schema-name": "test",
   "table-name": "%"
  },
  "rule-action": "include"
 }, {
  "rule-type": "transformation",
  "rule-id": "2",
```

```
  "rule-name": "2",
  "rule-action": "remove-column",
  "rule-target": "column",
  "object-locator": {
   "schema-name": "test",
   "table-name": "Actor",
   "column-name": "col%"
  }
 }]
 }
```

Example 소문자로 변환

다음은 소스의 테이블 이름 ACTOR를 대상의 actor로 변환하는 예입니다.

```
{
 "rules": [{
  "rule-type": "selection",
  "rule-id": "1",
  "rule-name": "1",
  "object-locator": {
   "schema-name": "test",
   "table-name": "%"
  },
  "rule-action": "include"
 }, {
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "convert-lowercase",
  "rule-target": "table",
  "object-locator": {
   "schema-name": "test",
   "table-name": "ACTOR"
  }
 }]
}
```

Example 대문자로 변환

다음은 모든 테이블의 전체 열과 모든 스키마를 소스의 소문자에서 대상의 대문자로 변환하는 예제입니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "transformation",
            "rule-id": "2",
            "rule-name": "2",
            "rule-action": "convert-uppercase",
            "rule-target": "column",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%",
                "column-name": "%"
            }
        }
    ]
}
```

Example 접두사 추가

다음은 소스의 모든 테이블을 변환하여 대상에서 테이블에 DMS_ 접두사를 추가하는 예입니다.

```
{
  "rules": [{
  "rule-type": "selection",
  "rule-id": "1",
  "rule-name": "1",
  "object-locator": {
   "schema-name": "test",
   "table-name": "%"
  },
  "rule-action": "include"
 }, {
  "rule-type": "transformation",
```

```
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "add-prefix",
  "rule-target": "table",
  "object-locator": {
   "schema-name": "test",
   "table-name": "%"
  },
  "value": "DMS_"
 }]

}
```

Example 접두사 바꾸기

다음은 소스에서 Pre_ 접두사를 포함하는 모든 열을 변환하여 이 접두사를 대상의 NewPre_ 접두사로 대체하는 예입니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "transformation",
            "rule-id": "2",
            "rule-name": "2",
            "rule-action": "replace-prefix",
            "rule-target": "column",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%",
                "column-name": "%"
            },
            "value": "NewPre_",
            "old-value": "Pre_"
```

```
        }
    ]
}
```

Example 접미사 제거

다음은 소스의 모든 테이블을 변환하여 대상의 테이블에서 _DMS 접미사를 제거하는 예입니다.

```
{
 "rules": [{
  "rule-type": "selection",
  "rule-id": "1",
  "rule-name": "1",
  "object-locator": {
   "schema-name": "test",
   "table-name": "%"
  },
  "rule-action": "include"
 }, {
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "remove-suffix",
  "rule-target": "table",
  "object-locator": {
   "schema-name": "test",
   "table-name": "%"
  },
  "value": "_DMS"
 }]
}
```

Example 기본 키 정의

대상 엔드포인트로 마이그레이션된 ITEM 테이블의 세 열에서 ITEM-primary-key라는 기본 키를 정의하는 예입니다.

```
{
 "rules": [{
  "rule-type": "selection",
  "rule-id": "1",
  "rule-name": "1",
  "object-locator": {
```

```
   "schema-name": "inventory",
   "table-name": "%"
  },
  "rule-action": "include"
 }, {
  "rule-type": "transformation",
  "rule-id": "2",
  "rule-name": "2",
  "rule-action": "define-primary-key",
  "rule-target": "table",
  "object-locator": {
   "schema-name": "inventory",
   "table-name": "ITEM"
  },
  "primary-key-def": {
   "name": "ITEM-primary-key",
   "columns": [
    "ITEM-NAME",
    "BOM-MODEL-NUM",
    "BOM-PART-NUM"
   ]
              }
 }]
}
```

Example 고유 인덱스 정의

대상 엔드포인트로 마이그레이션된 `ITEM` 테이블의 세 열에서 `ITEM-unique-idx`라는 고유 인덱스를 정의하는 예입니다.

```
{
 "rules": [{
  "rule-type": "selection",
  "rule-id": "1",
  "rule-name": "1",
  "object-locator": {
   "schema-name": "inventory",
   "table-name": "%"
  },
  "rule-action": "include"
 }, {
  "rule-type": "transformation",
  "rule-id": "2",
```

```
  "rule-name": "2",
  "rule-action": "define-primary-key",
  "rule-target": "table",
  "object-locator": {
   "schema-name": "inventory",
   "table-name": "ITEM"
  },
  "primary-key-def": {
   "name": "ITEM-unique-idx",
   "origin": "unique-index",
   "columns": [
    "ITEM-NAME",
    "BOM-MODEL-NUM",
    "BOM-PART-NUM"
   ]
              }
 }]
}
```

Example 대상 열의 데이터 형식 변경

다음 예제에서는 이름이 SALE_AMOUNT인 대상 열의 데이터 형식을 기존 데이터 형식에서 int8로 변경합니다.

```
{
    "rule-type": "transformation",
    "rule-id": "1",
    "rule-name": "RuleName 1",
    "rule-action": "change-data-type",
    "rule-target": "column",
    "object-locator": {
        "schema-name": "dbo",
        "table-name": "dms",
        "column-name": "SALE_AMOUNT"
    },
    "data-type": {
        "type": "int8"
    }
}
```

Example 이전 이미지 열 추가

emp_no라는 소스 열에 대해 다음 예의 변환 규칙은 대상에 BI_emp_no라는 새 열을 추가합니다.

```
{
 "rules": [{
   "rule-type": "selection",
   "rule-id": "1",
   "rule-name": "1",
   "object-locator": {
    "schema-name": "%",
    "table-name": "%"
   },
   "rule-action": "include"
  },
  {
   "rule-type": "transformation",
   "rule-id": "2",
   "rule-name": "2",
   "rule-target": "column",
   "object-locator": {
    "schema-name": "%",
    "table-name": "employees"
   },
   "rule-action": "add-before-image-columns",
   "before-image-def": {
    "column-prefix": "BI_",
    "column-suffix": "",
    "column-filter": "pk-only"
   }
  }
 ]
}
```

여기서 다음 문은 해당 행의 BI_emp_no 열을 1로 채웁니다.

```
UPDATE employees SET emp_no = 3 WHERE BI_emp_no = 1;
```

지원되는 AWS DMS 대상에 CDC 업데이트를 기록할 때 BI_emp_no 열을 통해 열의 값이 업데이트된 행을 확인할 수 있습니다. emp_no

변환 규칙 표현식을 사용하여 열 내용 정의

변환 규칙 내에서 표현식을 사용하여 새 열과 기존 열의 내용을 정의할 수 있습니다. 예를 들어, 표현식을 사용하여 열을 추가하거나 소스 테이블 헤더를 대상에 복제할 수 있습니다. 표현식을 사용하여 대상 테이블의 레코드를 소스에서 삽입, 업데이트 또는 삭제된 것으로 플래그를 지정할 수도 있습니다.

주제

- 표현식을 사용하여 열 추가
- 표현식을 사용하여 대상 레코드에 플래그 지정
- 표현식을 사용하여 소스 테이블 헤더 복제
- SQLite 함수를 사용하여 표현식 작성
- 표현식을 사용하여 대상 테이블에 메타데이터 추가

표현식을 사용하여 열 추가

변환 규칙에서 표현식을 사용하여 테이블에 열을 추가하려면 add-column 규칙 작업 및 column 규칙 대상을 사용합니다.

다음 예에서는 ITEM 테이블에 새 열을 추가합니다. 50자 길이의 string 데이터 형식을 사용하여 새 열 이름을 FULL_NAME으로 설정합니다. 표현식은 기존의 두 열 FIRST_NAME 및 LAST_NAME의 값을 결합하여 FULL_NAME으로 평가합니다. schema-name, table-name 및 표현식 파라미터는 소스 데이터베이스 테이블의 객체를 참조합니다. Value 및 data-type블록은 대상 데이터베이스 테이블의 객체를 참조합니다.

```
{
    "rules": [
        {
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "transformation",
            "rule-id": "2",
            "rule-name": "2",
            "rule-action": "add-column",
            "rule-target": "column",
            "object-locator": {
                "schema-name": "Test",
                "table-name": "ITEM"
            },
```

```
            "value": "FULL_NAME",
            "expression": "$FIRST_NAME||'_'||$LAST_NAME",
            "data-type": {
                 "type": "string",
                 "length": 50
            }
        }
    ]
}
```

표현식을 사용하여 대상 레코드에 플래그 지정

변환 규칙에서 표현식을 사용하여 대상 테이블의 레코드를 소스 테이블에서 삽입, 업데이트 또는 삭제된 것으로 플래그를 지정할 수 있습니다. 표현식은 `operation_indicator` 함수를 사용하여 레코드에 플래그를 지정합니다. 소스에서 삭제된 레코드는 대상에서 삭제되지 않습니다. 그 대신, 대상 레코드가 소스에서 삭제되었음을 나타내는 사용자 제공 값으로 플래그가 지정됩니다.

> **Note**
>
> 이 `operation_indicator` 함수는 원본 및 대상 데이터베이스에 기본 키가 있는 테이블에서만 작동합니다.

예를 들어, 다음 변환 규칙은 먼저 대상 테이블에 새 `Operation` 열을 추가합니다. 그런 다음, 레코드가 소스 테이블에서 삭제될 때마다 `D` 값으로 열을 업데이트합니다.

```
{
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-target": "column",
      "object-locator": {
        "schema-name": "%",
        "table-name": "%"
      },
      "rule-action": "add-column",
      "value": "Operation",
      "expression": "operation_indicator('D', 'U', 'I')",
      "data-type": {
        "type": "string",
        "length": 50
      }
```

```
}
```

표현식을 사용하여 소스 테이블 헤더 복제

기본적으로 소스 테이블의 헤더는 대상에 복제되지 않습니다. 복제할 헤더를 나타내려면 테이블 열 헤더를 포함하는 표현식과 함께 변환 규칙을 사용합니다.

표현식에서 다음 열 헤더를 사용할 수 있습니다.

헤더	지속적 복제의 값	전체 로드의 값	데이터 유형
AR_H_STREAM_POSITION	소스의 스트림 위치 값. 이 값은 소스 엔드포인트에 따라 시스템 변경 번호(SCN) 또는 로그 시퀀스 번호(LSN)일 수 있습니다.	빈 문자열.	STRING
AR_H_TIMESTAMP	변경 시간을 나타내는 타임스탬프.	데이터가 대상에 도달한 현재 시간을 나타내는 타임스탬프입니다.	DATETIME(크기 = 7)
AR_H_COMMIT_TIMESTAMP	커밋 시간을 나타내는 타임 스탬프.	현재 시간을 나타내는 타임스탬프.	DATETIME(크기 = 7)
AR_H_OPERATION	INSERT, UPDATE 또는 DELETE	INSERT	STRING
AR_H_USER	변경한 사용자에 대해 소스에서 제공되는 사용자 이름, ID 또는 기타 정보. 이 헤더는 SQL Server 및 Oracle(버전 11.2.0.3 이상) 소스 엔드포인트에서만 지원됩니다.	객체에 적용할 변환. 변환 규칙 작업은 대소문자를 구분합니다.	STRING

헤더	지속적 복제의 값	전체 로드의 값	데이터 유형
AR_H_CHANGE_SEQ	타임스탬프와 자동 증분 숫자로 구성된 소스 데이터베이스의 고유한 증분 숫자입니다. 값은 소스 데이터베이스 시스템에 따라 달라집니다.	빈 문자열.	STRING

다음 예에서는 소스의 스트림 위치 값을 사용하여 대상에 새 열을 추가합니다. SQL Server의 경우, 스트림 위치 값은 소스 엔드포인트의 LSN입니다. Oracle의 경우, 스트림 위치 값은 소스 엔드포인트의 SCN입니다.

```
{
      "rule-type": "transformation",
     "rule-id": "2",
      "rule-name": "2",
      "rule-target": "column",
      "object-locator": {
        "schema-name": "%",
        "table-name": "%"
      },
      "rule-action": "add-column",
      "value": "transact_id",
      "expression": "$AR_H_STREAM_POSITION",
      "data-type": {
        "type": "string",
        "length": 50
      }
    }
```

다음 예제에서는 원본의 고유한 증분 번호가 있는 새 열을 대상에 추가합니다. 이 값은 작업 수준에서 35자리 고유 숫자를 나타냅니다. 처음 16자리는 타임스탬프의 일부이고, 마지막 19자리는 DBMS에서 증분한 record_id 번호입니다.

```
{
"rule-type": "transformation",
"rule-id": "2",
```

```
"rule-name": "2",
"rule-target": "column",
"object-locator": {
"schema-name": "%",
"table-name": "%"
},
"rule-action": "add-column",
"value": "transact_id",
"expression": "$AR_H_CHANGE_SEQ",
"data-type": {
"type": "string",
"length": 50
}
}
```

SQLite 함수를 사용하여 표현식 작성

테이블 설정을 사용하여 지정된 작업에서 선택한 테이블 또는 뷰에 적용할 설정을 지정합니다. 테이블 설정 규칙은 선택 사항입니다.

> **Note**
>
> MongoDB 및 DocumentDB 데이터베이스는 테이블과 뷰의 개념 대신 데이터 레코드를 컬렉션에 함께 수집된 문서로 저장합니다. 따라서 MongoDB 또는 DocumentDB 소스에서 마이그레이션할 때는 테이블과 뷰보다는 선택한 컬렉션에 대한 병렬 로드 설정의 범위 세그먼트화 유형을 고려하십시오.

주제

- CASE 표현식 사용
- 예제:

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 문자열 함수를 찾을 수 있습니다.

문자열 함수	설명
`lower(x)`	이 `lower(x)` 함수는 모든 문자가 소문자로 변환된 *x* 문자열의 복사본을 반환합니다. 기본 내장 `lower` 함수는 ASCII 문자에만 사용할 수 있습니다.
`upper(x)`	이 `upper(x)` 함수는 모든 문자가 소문자로 변환된 *x* 문자열의 복사본을 반환합니다. 기본 내장 `upper` 함수는 ASCII 문자에만 사용할 수 있습니다.
`ltrim(x,y)`	이 `ltrim(x,y)` 함수는 x의 왼쪽에서부터 y에 표시되는 모든 문자를 제거하여 구성된 문자열을 반환합니다. y에 값이 없는 경우, `ltrim(x)`는 x의 왼쪽에서 공백을 제거합니다.
`replace(x,y,z)`	이 `replace(x,y,z)` 함수는 문자열 x에서 나타나는 모든 문자열 y를 문자열 z로 대체하여 구성된 문자열을 반환합니다.
`rtrim(x,y)`	이 `rtrim(x,y)` 함수는 x의 오른쪽에서부터 y에 표시되는 모든 문자를 제거하여 구성된 문자열을 반환합니다. y에 값이 없는 경우, `rtrim(x)`는 x의 오른쪽에서 공백을 제거합니다.
`substr(x,y,z)`	이 `substr(x,y,z)` 함수는 입력 문자열의 첫 *y*번째 문자로 시작하는 하위 문자열 *x*를 반환합니다. 이 문자열의 길이는 *z*자입니다. *z*를 생략하면 `substr(x,y)`은 *y*번째 문자로 시작하는 *x* 문자열의 끝까지 모든 문자를 반환합니다. *x*의 가장 왼쪽 문자는 숫자 1입니다. *y*가 음수인 경우, 왼쪽이 아닌 오른쪽부터 계수하여 하위 문자열의 첫 번째 문자를 찾습니다. *z*가 음수인 경우, 첫 *y* 번째 문자 앞의 `abs(z)` 문자가 반환됩니다. *x*가 문자열인 경우, 문자의 인덱스는 실제 UTF-8 문자를 참조합니다. *x*가 BLOB인 경우, 인덱스는 바이트를 나타냅니다.
`trim(x,y)`	이 `trim(x,y)` 함수는 *x*의 양쪽에서부터 *y*에 표시되는 모든 문자를 제거하여 구성된 문자열을 반환합니다. *y*에 값이 없는 경우, `trim(x)`는 *x*의 오른쪽에서 공백을 제거합니다.

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 LOB 함수를 찾을 수 있습니다.

LOB 함수	설명
`hex(x)`	이 hex 함수는 BLOB를 인수로 받고 BLOB 내용의 대문자 16진수 문자열 버전을 반환합니다.
`randomblob ( N)`	이 `randomblob( N)` 함수는 의사 랜덤 바이트를 포함하는 *N*바이트 BLOB를 반환합니다. *N*이 1보다 작으면 1바이트 임의 BLOB가 반환됩니다.
`zeroblob(N)`	이 `zeroblob( N)` 함수는 *N*바이트의 0x00으로 구성된 BLOB를 반환합니다.

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 numeric 함수를 찾을 수 있습니다.

Numeric 함수	설명
`abs(x)`	이 `abs(x)` 함수는 숫자 인수 *x*의 절대값을 반환합니다. *x*가 NULL인 경우 `abs(x)` 함수는 NULL을 반환합니다. *x*가 숫자 값으로 변환할 수 없는 문자열 또는 BLOB인 경우 `abs(x)` 함수는 0.0을 반환합니다.
`random()`	이 random 함수는 -9,223,372,036,854,775,808 ~ +9,223,372,036,854,775,807 사이의 의사 랜덤 정수를 반환합니다.
`round (x,y)`	이 `round (x,y)` 함수는 소수점 오른쪽의 *y*자리 숫자로 반올림된 부동 소수점 값 *x*를 반환합니다. *y*에 값이 없으면 0으로 간주됩니다.
`max (x,y...)`	다중 인수 max 함수는 최대값이 포함된 인수를 반환하며 인수가 NULL이면 NULL을 반환합니다. max 함수는 인수의 왼쪽에서 오른쪽으로 데이터 정렬 함수를 정의하는 인수를 검색합니다. 검색으로 찾은 인수는 모든 문자열 비교에 해당 데이터 정렬 함수를 사용합니다. 데이터 정렬 함수를 정의하는 인수가 max에 하나도 없다면 BINARY 데이터 정렬 함수

Numeric 함수	설명
	가 사용됩니다. max 함수는 인수가 두 개 이상이면 단순 함수이지만 인수가 하나뿐이면 집계 함수로 작동합니다.
min (*x*,*y*...)	다중 인수 min 함수는 최소값이 포함된 인수를 반환합니다. min 함수는 인수의 왼쪽에서 오른쪽으로 데이터 정렬 함수를 정의하는 인수를 검색합니다. 검색으로 찾은 인수는 모든 문자열 비교에 해당 데이터 정렬 함수를 사용합니다. 데이터 정렬 함수를 정의하는 인수가 min에 하나도 없다면 BINARY 데이터 정렬 함수가 사용됩니다. min 함수는 인수가 두 개 이상이면 단순 함수이지만 인수가 하나뿐이면 집계 함수로 작동합니다.

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 NULL 체크 함수를 찾을 수 있습니다.

NULL 체크 함수	설명
coalesce (*x*,*y*...)	이 coalesce 함수는 NULL이 아닌 첫 번째 인수의 복사본을 반환하지만 모든 인수가 NULL이면 NULL을 반환합니다. coalesce 함수에는 인수가 두 개 이상 있습니다.
ifnull(*x*,*y*)	이 ifnull 함수는 NULL이 아닌 첫 번째 인수의 복사본을 반환하지만 두 인수가 모두 NULL이면 NULL을 반환합니다. 이 ifnull 함수에는 정확히 두 개의 인수가 있습니다. ifnull 함수는 두 개의 인수가 있는 coalesce와 같은 함수입니다.
nullif(*x*,*y*)	nullif(*x*,*y*) 함수는 인수가 다르면 첫 번째 인수의 복사본을 반환하지만 인수가 같으면 NULL을 반환합니다. nullif(*x*,*y*) 함수는 인수의 왼쪽에서 오른쪽으로 데이터 정렬 함수를 정의하는 인수를 검색합니다. 검색으로 찾은 인수는 모든 문자열 비교에 해당 데이터 정렬 함수를 사용합니다. 데이터 정렬 함수를 정의하는 인수가 nullif에 하나도 없다면 BINARY 데이터 정렬 함수가 사용됩니다.

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 날짜 및 시간 함수를 찾을 수 있습니다.

날짜 및 시간 함수	설명
`date(timestring , modifier, modifier...)`	이 date 함수는 YYYY-MM-DD 형식으로 날짜를 반환합니다.
`time(timestring , modifier, modifier...)`	이 time 함수는 HH:MM:SS 형식으로 시간을 반환합니다.
`datetime( timestring , modifier, modifier...)`	이 datetime 함수는 YYYY-MM-DD HH:MM:SS 형식으로 날짜 및 시간을 반환합니다.
`julianday( timestring , modifier, modifier...)`	이 julianday 함수는 기원전 4714년 11월 24일 그리니치에서 정오 이후 경과한 일수를 반환합니다.
`strftime( format, timestring , modifier, modifier...)`	이 strftime 함수는 다음 변수 중 하나를 사용하여 첫 번째 인수로 지정된 형식 문자열에 따라 날짜를 반환합니다. %d: 날짜(월, 일) %H: 00~24시 %f: ** 소수 초(SS.SSS) %j: 연중 일(001~366) %J: ** 율리우스력 날수 %m: 01~12개월 차 %M: 00~59분 %s: 1970-01-01 이후 경과 시간(초) %S: 00~59초 %w: 주중 일(0~6, 일요일 =0) %W: 연중 주(00~53) %Y: 0000~9999년도

날짜 및 시간 함수	설명
	%%: %

다음은 변환 규칙 표현식을 작성하는 데 사용할 수 있는 해시 함수를 찾을 수 있습니다.

해시 함수	설명
hash_sha256(*x*)	이 hash 함수는 (SHA-256 알고리즘을 사용하여) 입력 열에 대한 해시 값을 생성하고 생성된 해시 값의 16진수 값을 반환합니다. 표현식에서 hash 함수를 사용하려면 표현식에 hash_sha256(*x*) 함수를 추가하고 *x*를 소스 열 이름으로 바꾸십시오.

CASE 표현식 사용

SQLite CASE 표현식은 조건 목록을 평가하고 결과를 기반으로 표현식을 반환합니다. 다음 구문이 나와 있습니다.

```
    CASE case_expression
     WHEN when_expression_1 THEN result_1
     WHEN when_expression_2 THEN result_2
     ...
     [ ELSE result_else ]
    END

# Or

     CASE
     WHEN case_expression THEN result_1
     WHEN case_expression THEN result_2
     ...
     [ ELSE result_else ]
    END
```

예제:

Example CASE 조건을 사용하여 대상 테이블에 새 문자열 열을 추가하는 방법

예를 들어, 다음 변환 규칙은 먼저 대상 테이블 employee에 새 문자열 열 emp_seniority를 추가합니다. 급여 열에 SQLite round 함수를 사용하고 이 때 급여가 20,000과 같거나 이를 초과하는지 확인하는 CASE 조건을 함께 적용합니다. 값이 맞으면 해당 열에 값 SENIOR이 적용되고 다른 모든 항목은 해당 값 JUNIOR를 갖습니다.

```
 {
     "rule-type": "transformation",
     "rule-id": "2",
     "rule-name": "2",
     "rule-action": "add-column",
     "rule-target": "column",
     "object-locator": {
       "schema-name": "public",
       "table-name": "employee"
     },
     "value": "emp_seniority",
     "expression": " CASE WHEN round($emp_salary)>=20000 THEN 'SENIOR' ELSE 'JUNIOR'
END",
     "data-type": {
       "type": "string",
       "length": 50
     }

 }
```

Example 대상 테이블에 새 날짜 열 추가

다음 예에서는 대상 테이블 employee에 새 날짜 열 createdate를 추가합니다. SQLite 날짜 함수 datetime을 사용하면 삽입된 각 행에 대해 새로 생성된 테이블에 날짜가 추가됩니다.

```
 {
     "rule-type": "transformation",
     "rule-id": "2",
     "rule-name": "2",
     "rule-action": "add-column",
     "rule-target": "column",
     "object-locator": {
       "schema-name": "public",
```

```
        "table-name": "employee"
      },
      "value": "createdate",
      "expression": "datetime ()",
      "data-type": {
        "type": "datetime",
        "precision": 6
      }
  }
```

Example 대상 테이블에 새 숫자 열 추가

다음 예에서는 대상 테이블 employee에 새 숫자 열 rounded_emp_salary를 추가합니다. SQLite round 함수를 사용하여 반올림된 급여를 추가합니다.

```
  {
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-action": "add-column",
      "rule-target": "column",
      "object-locator": {
        "schema-name": "public",
        "table-name": "employee"
      },
      "value": "rounded_emp_salary",
      "expression": "round($emp_salary)",
      "data-type": {
        "type": "int8"
      }
  }
```

Example 해시 함수를 사용하여 대상 테이블에 새 문자열 열을 추가하는 방법

다음 예에서는 대상 테이블 employee에 새 문자열 열 hashed_emp_number를 추가합니다. SQLite hash_sha256(*x*) 함수는 소스 열 emp_number의 대상에 해시된 값을 생성합니다.

```
  {
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-action": "add-column",
      "rule-target": "column",
```

```
      "object-locator": {
        "schema-name": "public",
        "table-name": "employee"
      },
      "value": "hashed_emp_number",
      "expression": "hash_sha256($emp_number)",
      "data-type": {
        "type": "string",
        "length": 64
      }
  }
```

표현식을 사용하여 대상 테이블에 메타데이터 추가

다음 표현식을 사용하여 대상 테이블에 메타데이터 정보를 추가할 수 있습니다.

- $AR_M_SOURCE_SCHEMA – 소스 스키마의 이름입니다.
- $AR_M_SOURCE_TABLE_NAME – 소스 테이블의 이름입니다.
- $AR_M_SOURCE_COLUMN_NAME – 소스 테이블의 열 이름입니다.
- $AR_M_SOURCE_COLUMN_DATATYPE – 소스 테이블 내 열의 데이터 형식입니다.

Example 소스의 스키마 이름을 사용하여 스키마 이름에 대한 열을 추가하는 방법

다음 예에서는 소스의 스키마 이름을 사용하여 대상에 schema_name이라는 새 열을 추가합니다.

```
  {
      "rule-type": "transformation",
      "rule-id": "2",
      "rule-name": "2",
      "rule-action": "add-column",
      "rule-target": "column",
      "object-locator": {
        "schema-name": "%",
        "table-name": "%"
      },
      "rule-action": "add-column",
      "value":"schema_name",
      "expression": "$AR_M_SOURCE_SCHEMA",
      "data-type": {
         "type": "string",
         "length": 50
```

```
    }
}
```

테이블 및 컬렉션 설정 규칙과 작업

테이블 설정을 사용하여 지정된 작업에서 선택한 테이블 또는 뷰에 적용할 설정을 지정합니다. 테이블 설정 규칙은 엔드포인트 및 마이그레이션 요구 사항에 따라 선택 사항입니다.

MongoDB 및 Amazon DocumentDB 데이터베이스는 테이블과 뷰의 개념 대신 데이터 레코드를 컬렉션에 함께 수집된 문서로 저장합니다. MongoDB 또는 Amazon DocumentDB 엔드포인트에 대한 단일 데이터베이스는 데이터베이스 이름으로 식별되는 특정 컬렉션 집합입니다.

MongoDB 또는 Amazon DocumentDB 소스에서 마이그레이션할 때는 병렬 로드 설정을 사용하는 방식이 약간 다릅니다. 이 경우, 테이블과 뷰보다는 선택한 컬렉션에 대한 자동 세분화 또는 범위 세그먼트화 유형의 병렬 로드 설정을 고려해 보십시오.

주제

- 테이블 설정의 와일드카드는 제한됨
- 선택한 테이블, 뷰 및 컬렉션에 병렬 로드 사용
- 선택한 테이블 또는 뷰의 LOB 설정 지정
- 테이블 설정 예제

테이블 설정 규칙 유형을 사용하는 테이블 매핑 규칙의 경우, 다음 파라미터를 적용할 수 있습니다.

파라미터	가능한 값	설명
`rule-type`	`table-settings`	선택 규칙에서 지정한 테이블, 뷰 또는 컬렉션에 규칙을 적용하는 값입니다.
`rule-id`	숫자 값.	규칙을 식별하기 위한 고유한 숫자 값입니다.
`rule-name`	영숫자 값입니다.	규칙을 식별하기 위한 고유한 이름입니다.
`object-locator`	객체는 다음 파라미터를 사용합니다.	특정 스키마와 테이블 또는 뷰의 이름 또는 특정 데이터

파라미터	가능한 값	설명
	• `schema-name` – 스키마의 이름입니다. MongoDB 및 Amazon DocumentDB 엔드포인트의 경우, 컬렉션 세트를 보관하는 데이터베이스의 이름입니다. • `table-name` – 테이블, 뷰 또는 컬렉션의 이름입니다.	베이스 및 컬렉션의 이름(와일드카드 없음)입니다.

파라미터	가능한 값	설명
`parallel-load`	객체는 다음 파라미터를 사용합니다. • `type` – 병렬 로드를 활성화할지 여부를 지정합니다. 사용할 경우 이 파라미터는 동시에 로드할 테이블 또는 뷰 파티션, 하위 파티션 또는 기타 세그먼트를 식별하는 메커니즘을 지정합니다. 파티션은 소스 테이블 또는 뷰에서 이미 이름별로 정의되고 식별된 세그먼트입니다. MongoDB 및 Amazon DocumentDB 엔드포인트의 경우 파티션은 세그먼트입니다. AWS DMS 자동으로 지정된 관련 자동 세그멘테이션 파라미터를 계산할 수 있습니다. 또는 범위 세그먼트화 파라미터를 사용하여 이러한 파라미터를 수동으로 지정할 수도 있습니다. Oracle 엔드포인트의 경우에 한해 하위 파티션은 이미 소스 테이블 또는 뷰에서 이름별로 정의되고 식별된 추가 수준의 세그먼트입니다. 하나 이상의 테이블 또는 뷰 열에 대한 값 범위의 경계를 지정하여 `table-settings` 규칙에서 다른 세그먼트를 식별할 수 있습니다. • `partitions` – `type`이 `partitions-list` 일 때 이 값은 동시에 로드할 모든 파티션을 지정합니다.	`object-locator` 옵션으로 식별되는 테이블 또는 뷰에서 병렬 로드(다중 스레드) 작업을 지정하는 값입니다. 이 경우, 다음과 같은 방법으로 동시에 로드할 수 있습니다. • 사용 가능한 모든 파티션이나 하위 파티션으로 지정된 세그먼트별로 로드합니다. • 선택한 파티션 및 하위 파티션별로 로드합니다. • 자동 세분화로 로드하거나 지정한 범위 기반 세그먼트별로 로드합니다. 병렬 로드에 관한 자세한 내용은 선택한 테이블, 뷰 및 컬렉션에 병렬 로드 사용 단원을 참조하십시오.

파라미터	가능한 값	설명
	• `subpartitions` – Oracle 엔드포인트의 경우에 한해 `type`이 `partitions-list` 일 때 이 값은 동시에 로드할 모든 하위 파티션을 지정합니다. • `columns` – `type`이 `ranges`일 때 이 값은 동시에 로드할 범위 기반 세그먼트를 식별하는 데 사용되는 열의 이름을 지정합니다. • `boundaries` – `type`이 `ranges`일 때 이 값은 동시에 로드할 범위 기반 세그먼트를 식별하는 데 사용되는 `columns`의 값을 지정합니다.	

파라미터	가능한 값	설명
type	parallel-load 에 다음 중 하나를 수행하십시오. • partitions-auto – 테이블 또는 뷰의 모든 파티션이 동시에 로드됩니다. 모든 파티션은 자체 스레드에 할당됩니다. 이는 MongoDB 및 Amazon DocumentDB 소스 엔드포인트가 병렬 전체 로드의 자동 세분화 옵션을 사용하기 위한 필수 설정입니다. • subpartitions-auto – (Oracle 엔드포인트 전용) 테이블 또는 뷰의 모든 하위 파티션이 동시에 로드됩니다. 모든 하위 파티션은 자체 스레드에 할당됩니다. • partitions-list – 테이블 또는 뷰에서 지정된 모든 파티션이 동시에 로드됩니다. Oracle 엔드포인트의 경우에 한해 테이블 또는 뷰에서 지정된 모든 하위 파티션이 동시에 로드됩니다. 지정한 각 파티션 및 하위 파티션은 자체 스레드에 할당됩니다. 파티션 이름(partitions) 및 하위 파티션 이름(subpartitions)에 따라 동시에 로드할 파티션과 하위 파티션을 지정합니다. • ranges – 테이블, 뷰 또는 컬렉션에서 범위가 지정된 모든 세그먼트가 동시에 로드됩니다. 식별하는 각 테이블, 뷰 또는 컬렉션 세그먼트는 자체 스레드에 할당됩니다. 이러한	동시에 로드할 테이블, 뷰 또는 컬렉션 파티션, 하위 파티션 또는 세그먼트를 식별하는 메커니즘입니다.

파라미터	가능한 값	설명
	세그먼트는 열 이름(columns) 및 열 값(boundaries)으로 지정합니다. PostgreSQL 엔드포인트는 이러한 유형의 병렬 로드만 지원합니다. MongoDB와 Amazon DocumentDB는 소스 엔드포인트로 이 범위 세그먼트화 유형과 병렬 전체 로드(partitions-auto)의 자동 세분화 유형을 모두 지원합니다. • none – 테이블, 뷰 또는 컬렉션은 파티션 또는 하위 파티션에 관계없이 단일 스레드 작업(기본값)으로 로드됩니다. 자세한 정보는 작업 생성을 참조하세요.	
number-of-partitions	(선택 사항) type이 MongoDB 또는 Amazon DocumentDB 엔드포인트의 지정된 컬렉션에 대한 partitions-auto 인 경우, 이 파라미터는 마이그레이션에 사용되는 총 파티션(세그먼트) 수를 지정합니다. 기본값은 16입니다.	동시에 로드할 파티션의 정확한 이름을 지정합니다.
collection-count-from-metadata	(선택 사항) MongoDB 또는 Amazon DocumentDB 엔드포인트의 지정된 컬렉션용이고 이 파라미터가 true 로 AWS DMS 설정된 경우 예상 수집 수를 사용하여 파티션 수를 결정합니다. type partitions-auto 이 매개변수가 로 설정된 경우 실제 false 수집 수를 AWS DMS 사용합니다. 기본값은 true입니다.	동시에 로드할 파티션 수를 계산할 때 예상 컬렉션 수를 사용할지 아니면 실제 컬렉션 수를 사용할지를 지정합니다.

파라미터	가능한 값	설명
max-records-skip-per-page	(선택 사항) MongoDB 또는 Amazon DocumentDB 엔드포인트의 지정된 컬렉션에 대한 type이 partitions-auto 인 경우, 이것은 각 파티션의 경계를 결정할 때 한 번에 건너뛰어야 하는 레코드 수를 나타냅니다. AWS DMS 는 페이징된 건너뛰기 방식을 사용하여 파티션의 최소 경계를 결정합니다. 기본값은 10,000입니다.	각 파티션의 경계를 결정할 때 한 번에 건너뛸 레코드 수를 지정합니다. 기본값보다 비교적 큰 값을 설정하면 커서 제한 시간이 초과되고 작업이 실패할 수 있습니다. 기본값보다 비교적 작은 값을 설정하면 페이지당 작업 수가 늘어나고 전체 로드 속도가 느려집니다.
batch-size	(선택 사항) MongoDB 또는 Amazon DocumentDB 엔드포인트의 지정된 컬렉션에 대한 type이 partitions-auto 인 경우, 이 정수 값은 한 번의 왕복 배치에서 반환되는 문서 수를 제한합니다. 배치 크기가 0인 경우, 커서는 서버에서 정의한 최대 배치 크기를 사용합니다. 기본값은 0입니다.	하나의 배치에 반환된 최대 문서 수를 지정합니다. 배치마다 서버를 왕복해야 합니다.
partitions	type이 partitions-list 일 때 동시에 로드할 파티션의 이름을 지정하는 문자열 배열입니다.	동시에 로드할 파티션의 이름입니다.
subpartitions	(Oracle 엔드포인트 전용) type이 partitions-list 일 때 동시에 로드할 하위 파티션의 이름을 지정하는 문자열 배열입니다.	동시에 로드할 하위 파티션의 이름입니다.
columns	type이 ranges일 때 동시에 로드할 범위 기반 테이블, 뷰 또는 컬렉션 세그먼트를 식별하는 열 이름으로 설정된 문자열 배열입니다.	동시에 로드할 범위 기반 테이블, 뷰 또는 컬렉션 세그먼트를 식별하는 열 이름입니다.

파라미터	가능한 값	설명
boundaries	type이 ranges일 때 열-값 배열의 배열입니다. 각 열-값 배열은 columns에 의해 지정된 수량과 순서로 열 값을 포함합니다. 열-값 배열은 테이블, 뷰 또는 컬렉션 세그먼트의 상한을 지정합니다. 각 추가 열-값 배열은 하나의 추가 테이블, 뷰 또는 컬렉션 세그먼트에 대한 상한을 추가합니다. 이러한 모든 범위 기반 테이블, 뷰 또는 컬렉션 세그먼트는 동시에 로드됩니다.	동시에 로드할 범위 기반 테이블, 뷰 또는 컬렉션 파티션을 식별하는 열 값입니다.
lob-settings	객체는 다음 파라미터를 사용합니다. • mode – LOB에 대한 마이그레이션 처리 모드를 지정합니다. • bulk-max-size – mode 설정에 따라 LOB의 최대 크기를 지정합니다.	object-locator 옵션으로 식별된 테이블 또는 뷰에 대한 LOB 처리를 지정하는 값입니다. 지정된 LOB 처리는 이 테이블 또는 뷰에만 지정된 작업 LOB 설정을 재정의합니다. LOB 설정 파라미터 사용에 관한 자세한 내용은 선택한 테이블 또는 뷰의 LOB 설정 지정 단원을 참조하십시오.

파라미터	가능한 값	설명
mode	다음 값을 사용하여 지정한 테이블 또는 뷰의 LOB에 대한 마이그레이션 처리를 지정합니다. • limited – (기본값) 이 값은 모든 LOB가 테이블 또는 뷰의 다른 모든 열 데이터 형식과 함께 인라인으로 마이그레이션되는 제한적 LOB 모드로 마이그레이션을 설정합니다. 이 값은 주로 작은 LOB(100MB 이하)를 복제하는 경우에 사용합니다. 또한 bulk-max-size 값을 지정합니다(0은 유효하지 않음). 마이그레이션된 모든 LOB가 bulk-max-size 보다 클 경우 사용자가 설정한 크기로 잘립니다. • unlimited – 이 값은 마이그레이션을 전체 LOB 모드로 설정합니다. 이 값은 복제하려는 LOB의 전체 또는 대부분이 1GB보다 큰 경우 사용합니다. bulk-max-size 값을 0으로 지정하면 모든 LOB가 표준 전체 LOB 모드로 마이그레이션됩니다. 이 형식의 unlimited 모드에서 모든 LOB는 소스 테이블 또는 뷰의 조회를 사용하여 다른 열 데이터 형식과 별도로 마이그레이션됩니다. bulk-max-size 값을 0보다 큰 값으로 지정하면 모든 LOB가 조합 전체 LOB 모드로 마이그레이션됩니다. 이 형식의 unlimited 모드에서는 bulk-max-size 보다 큰 LOB가 표준 전체 LOB 모드와 비슷한 방식으로 소스 테이블 또	LOB를 마이그레이션하는 데 사용되는 메커니즘입니다.

파라미터	가능한 값	설명
	는 뷰 조회를 사용하여 마이그레이션됩니다. 그렇지 않으면 제한적 LOB 모드와 유사하게 이 크기 이하의 LOB가 인라인으로 마이그레이션됩니다. 사용하는 형식에 상관없이 `unlimited` 모드에서는 LOB가 잘리지 않습니다. • none – 모든 테이블 또는 뷰 LOB는 작업 LOB 설정에 따라 마이그레이션됩니다. 작업 LOB 설정에 관한 자세한 내용은 대상 메타데이터 작업 설정 단원을 참조하십시오. LOB를 마이그레이션하는 방법 및 이러한 작업 LOB 설정을 지정하는 방법에 관한 자세한 내용은 태스크의 소스 데이터베이스에 대한 LOB 지원 설정 AWS DMS 단원을 참조하십시오.	
`bulk-max-size`	이 값의 효과는 `mode`에 따라 다릅니다.	LOB의 최대 크기입니다(킬로바이트 단위). 작은 LOB를 복제해야 하거나 대상 엔드포인트가 무제한 LOB 크기를 지원하지 않는 경우에만 이 옵션을 지정하십시오.

테이블 설정의 와일드카드는 제한됨

`"table-settings"` 규칙에 퍼센트 와일드카드(`"%"`)를 사용하는 것은 다음과 같이 소스 데이터베이스에서 지원되지 않습니다.

```
{
    "rule-type": "table-settings",
```

```
    "rule-id": "8",
    "rule-name": "8",
    "object-locator": {
        "schema-name": "ipipeline-prod",
        "table-name": "%"
    },
    "parallel-load": {
        "type": "partitions-auto",
        "number-of-partitions": 16,
        "collection-count-from-metadata": "true",
        "max-records-skip-per-page": 1000000,
        "batch-size": 50000
    }
  }
```

그림과 같이 "%" "table-settings" 규칙에서 사용하면 다음과 같은 예외가 AWS DMS 반환됩니다.

```
Error in mapping rules. Rule with ruleId = x failed validation. Exact
schema and table name required when using table settings rule.
```

또한 를 사용하여 단일 작업을 사용하여 많은 수의 대규모 컬렉션을 로드하지 않는 것이 좋습니다parallel-load. AWS 단, AWS DMS 는 MaxFullLoadSubTasks 작업 설정 파라미터의 값에 따라 동시에 로드되는 세그먼트 수는 물론, 리소스 경합도 제한하며 최대값은 49입니다.

그 대신 각 "schema-name" 및 "table-name"을 개별적으로 지정하여 가장 큰 컬렉션에 대한 소스 데이터베이스의 모든 컬렉션을 지정하십시오. 또한 마이그레이션 규모를 적절하게 확장하세요. 예를 들어, 충분한 수의 복제 인스턴스에서 여러 작업을 실행하여 데이터베이스의 수많은 대규모 컬렉션을 처리할 수 있습니다.

선택한 테이블, 뷰 및 컬렉션에 병렬 로드 사용

선택한 관계형 테이블, 뷰 및 컬렉션에 병렬 로드를 사용하여 마이그레이션 속도와 효율성을 향상시킬 수 있습니다. 즉, 여러 스레드를 동시에 사용하여 단일한 세그먼트화 테이블, 뷰 또는 컬렉션을 마이그레이션할 수 있습니다. 이를 위해 AWS DMS 는 전체 로드 작업을 스레드로 분할하고 각 테이블 세그먼트를 자체 스레드에 할당합니다.

이 병렬 로드 프로세스를 사용하면 먼저 다중 스레드가 소스 엔드포인트에서 여러 테이블, 뷰 및 컬렉션을 동시에 언로드할 수 있습니다. 그런 다음, 여러 스레드가 동일한 테이블, 뷰 및 컬렉션을 대상 엔드포인트에 동시에 마이그레이션하고 로드할 수 있습니다. 일부 데이터베이스 엔진의 경우, 기존 파티션

이나 하위 파티션별로 테이블 및 뷰를 세그먼트화할 수 있습니다. 다른 데이터베이스 엔진의 경우 특정 매개 변수에 따라 컬렉션을 AWS DMS 자동으로 분할할 수 있습니다 (자동 세그멘테이션). 그렇지 않으면 지정한 열 값의 범위로 테이블, 뷰 또는 컬렉션을 세그먼트화할 수 있습니다.

병렬 로드가 지원되는 소스 엔드포인트는 다음과 같습니다.

- Oracle
- Microsoft SQL Server
- MySQL
- PostgreSQL
- IBM Db2
- SAP Adaptive Server Enterprise(ASE)
- MongoDB(병렬 전체 로드의 자동 세분화 및 범위 세그먼트화 옵션만 지원)
- Amazon DocumentDB(병렬 전체 로드의 자동 세분화 및 범위 세그먼트화 옵션만 지원)
- Amazon DynamoDB

MongoDB 및 Amazon DocumentDB 엔드포인트의 AWS DMS 경우 병렬 전체 로드의 범위 분할 옵션에 대한 파티션 키인 열에 대해 다음 데이터 유형을 지원합니다.

- Double
- String
- ObjectId
- 32비트 정수
- 64비트 정수

테이블 설정 규칙과 함께 사용할 병렬 로드는 다음 대상 엔드포인트에 대해 지원됩니다.

- Oracle
- Microsoft SQL Server
- MySQL
- PostgreSQL
- Amazon S3
- SAP Adaptive Server Enterprise(ASE)

- Amazon Redshift
- MongoDB(병렬 전체 로드의 자동 세분화 및 범위 세그먼트화 옵션만 지원)
- Amazon DocumentDB(병렬 전체 로드의 자동 세분화 및 범위 세그먼트화 옵션만 지원)

동시에 로드할 최대 테이블 및 뷰 수를 지정하려면 `MaxFullLoadSubTasks` 작업 설정을 사용합니다.

병렬 로드 작업의 지원되는 대상에 대해 테이블 또는 뷰당 최대 스레드 수를 지정하려면 열-값 경계를 사용하여 더 많은 세그먼트를 정의하십시오.

> **Important**
>
> `MaxFullLoadSubTasks`는 동시에 로드할 테이블 또는 테이블 세그먼트의 수를 제어합니다. `ParallelLoadThreads`는 마이그레이션 작업에서 로드를 동시에 실행하는 데 사용하는 스레드의 수를 제어합니다. 이러한 설정은 배수로 적용됩니다. 따라서 전체 로드 작업 중에 사용된 스레드의 총 개수는 대략 `ParallelLoadThreads` 의 값에 `MaxFullLoadSubTasks`의 값을 곱한 결과(즉, `ParallelLoadThreads` * `MaxFullLoadSubtasks`))입니다.
> 전체 로드 하위 작업의 수가 많고 병렬 로드 스레드의 수가 많은 작업을 만들면 작업이 너무 많은 메모리를 소비하여 실패할 수 있습니다.

Amazon DynamoDB, Amazon Kinesis Data Streams, Apache Kafka 또는 Amazon Elasticsearch Service 대상의 테이블당 최대 스레드 수를 지정하려면 `ParallelLoadThreads` 대상 메타데이터 작업 설정을 사용하십시오.

`ParallelLoadThreads` 사용 시 병렬 로드 작업의 버퍼 크기를 지정하려면 `ParallelLoadBufferSize` 대상 메타데이터 작업 설정을 사용합니다.

`ParallelLoadThreads` 및 `ParallelLoadBufferSize`의 가용성과 설정은 대상 엔드포인트에 따라 다릅니다.

`ParallelLoadThreads` 및 `ParallelLoadBufferSize` 설정에 관한 자세한 내용은 대상 메타데이터 작업 설정을 참조하세요. `MaxFullLoadSubTasks` 설정에 관한 자세한 내용은 전체 로드 작업 설정를 참조하십시오. 대상 엔드포인트에 관한 자세한 내용은 관련 주제를 참조하십시오.

병렬 로드를 사용하려면 `parallel-load` 옵션을 사용하여 `table-settings` 유형의 테이블 매핑 규칙을 만듭니다. `table-settings` 규칙 내에서 동시에 로드하려는 단일한 테이블, 뷰 또는 컬렉션의 세그먼트화 기준을 지정할 수 있습니다. 이렇게 하려면 `parallel-load` 옵션의 `type` 파라미터를 여러 옵션 중 하나로 설정합니다.

이러한 설정을 수행하는 방법은 병렬 로드를 위해 테이블, 뷰 또는 컬렉션을 세그먼트화하려는 방법에 따라 다릅니다.

- 파티션별(세그먼트별) – `partitions-auto` 유형을 사용하여 기존 테이블 또는 뷰 파티션(세그먼트)을 모두 로드합니다. 또는 지정된 파티션 배열이 있는 `partitions-list` 유형을 사용하여 선택한 파티션만 로드합니다.

  MongoDB 및 Amazon DocumentDB 엔드포인트의 경우에만 유형 및 추가 선택적 파라미터를 사용하여 자동으로 AWS DMS 계산되는 세그먼트별로 전체 또는 지정된 컬렉션을 로드하십시오. `partitions-auto table-settings`
- (Oracle 엔드포인트 전용) 하위 파티션별 – `subpartitions-auto` 유형을 사용하여 기존 테이블 또는 뷰 하위 파티션을 모두 로드합니다. 또는 지정된 `subpartitions` 배열이 있는 `partitions-list` 유형을 사용하여 선택한 하위 파티션만 로드합니다.
- 정의한 세그먼트별 – 열-값 경계를 사용하여 정의한 테이블, 뷰 또는 컬렉션 세그먼트를 로드합니다. 이렇게 하려면 지정된 `columns` 및 `boundaries` 배열이 있는 `ranges` 유형을 사용하십시오.

> **Note**
>
> PostgreSQL 엔드포인트는 이러한 유형의 병렬 로드만 지원합니다. MongoDB와 Amazon DocumentDB는 소스 엔드포인트로 이 범위 세그먼트화 유형과 병렬 전체 로드(`partitions-auto`)의 자동 세분화 유형을 모두 지원합니다.

동시에 로드할 추가 테이블, 뷰 또는 컬렉션을 식별하려면 `parallel-load` 옵션을 사용하여 추가 `table-settings` 객체를 지정합니다.

다음 절차에서는 가장 간단한 것부터 가장 복잡한 것까지 각 병렬 로드 유형별로 JSON을 코드화하는 방법을 확인할 수 있습니다.

모든 테이블, 뷰 또는 컬렉션 파티션이나 모든 테이블 또는 뷰 하위 파티션을 지정하려면

- `parallel-load`를 `partitions-auto` 유형 또는 `subpartitions-auto` 유형으로 지정하십시오(동시 지정은 불가).

  그러면 모든 테이블, 뷰 또는 컬렉션 파티션(또는 세그먼트) 또는 하위 파티션이 자동으로 자체 스레드에 할당됩니다.

  일부 엔드포인트의 경우, 파티션 또는 하위 파티션은 테이블 또는 뷰에 이미 정의된 경우에만 병렬 로드에 포함됩니다. MongoDB 및 Amazon DocumentDB 소스 엔드포인트의 경우 선택적 추가 파

라미터를 기반으로 파티션 (또는 세그먼트) 을 자동으로 계산할 수 있습니다 AWS DMS . 여기에는 `number-of-partitions`, `collection-count-from-metadata`, `max-records-skip-per-page`, `batch-size`가 포함됩니다.

선택한 테이블 또는 뷰 파티션, 하위 파티션 또는 둘 다 지정하려면

1. `partitions-list` 유형으로 `parallel-load`를 지정합니다.
2. (선택 사항) 파티션 이름의 배열을 `partitions` 값으로 지정하여 파티션을 포함합니다.

   그러면 지정된 각 파티션이 자체 스레드에 할당됩니다.

   > **Important**
   >
   > Oracle 엔드포인트의 경우, 병렬 로드를 위해 파티션과 하위 파티션을 선택할 때 파티션과 하위 파티션이 겹치지 않는지 확인하십시오. 겹치는 파티션과 하위 파티션을 사용하여 데이터를 동시에 로드하면 항목이 중복되거나 기본 키 중복 위반으로 인해 오류가 발생합니다.

3. (선택 사항) Oracle 엔드포인트의 경우 한해 하위 파티션 이름의 배열을 `subpartitions` 값으로 지정하여 하위 파티션을 포함합니다.

   그러면 지정된 각 하위 파티션이 자체 스레드에 할당됩니다.

   > **Note**
   >
   > 파티션 또는 하위 파티션은 테이블 또는 뷰에 이미 정의된 경우에만 병렬 로드에 포함됩니다.

테이블 또는 뷰 세그먼트를 열 값의 범위로 지정할 수 있습니다. 이렇게 하려면 다음 열 특성을 알고 있어야 합니다.

- 인덱싱된 열을 지정하면 성능이 크게 향상됩니다.
- 최대 10개의 열을 지정할 수 있습니다.
- DOUBLE, FLOAT, BLOB, CLOB, NCLOB 등의 AWS DMS 데이터 유형으로는 열을 사용하여 세그먼트 경계를 정의할 수 없습니다.
- null 값의 레코드는 복제되지 않습니다.

테이블, 뷰 또는 컬렉션 세그먼트를 열 값의 범위로 지정하려면

1. `ranges` 유형으로 `parallel-load`를 지정합니다.
2. 열 이름의 배열을 `columns` 값으로 지정하여 테이블 또는 뷰 세그먼트 사이의 경계를 정의합니다. 테이블 또는 뷰 세그먼트 사이의 경계를 정의하려는 모든 열에 대해 이 작업을 수행합니다.

   열의 순서는 중요합니다. 다음 설명과 같이 각 경계를 정의하는 데 있어 첫 번째 열이 가장 중요하고 마지막 열은 가장 덜 중요합니다.
3. 모든 테이블 또는 뷰 세그먼트의 데이터 범위를 정의하려면 경계 배열의 값을 `boundaries`로 지정하십시오. 경계 배열은 열-값 배열의 배열입니다. 이 작업을 수행하려면 다음 단계를 수행하십시오.

   a. 열-값 배열의 각 요소를 각 열에 맞는 값으로 지정합니다. 열-값 배열은 정의하려는 각 테이블 또는 뷰 세그먼트의 상한을 나타냅니다. `columns` 배열에서 해당 열을 지정한 순서대로 각 열을 지정합니다.

      소스에서 지원하는 형식으로 DATE 열의 값을 입력합니다.

   b. 각 열-값 배열을 테이블 또는 뷰의 맨 아래부터 세그먼트까지 순서대로 각 세그먼트의 상한으로 지정합니다. next-to-top 지정한 맨 위 상한선 위에 행이 있으면 이 행은 테이블 또는 뷰의 맨 위 세그먼트를 완료합니다. 따라서 범위 기반 세그먼트의 수가 경계 배열의 세그먼트 경계 수보다 하나 더 많을 수 있습니다. 이러한 범위 기반 세그먼트는 각각 자체 스레드에 할당됩니다.

      테이블 또는 뷰의 모든 열에 데이터 범위를 정의하지 않더라도 null이 아닌 모든 데이터가 복제됩니다.

   예를 들어, 다음과 같이 열 COL1, COL2 및 COL3에 세 개의 열-값 배열을 정의한다고 가정합니다.

COL1	COL2	COL3
10	30	105
20	20	120
100	12	99

   가능한 총 4개 세그먼트에 세 개의 세그먼트 경계를 정의했습니다.

각 세그먼트별로 복제할 행의 범위를 식별하기 위해 복제 인스턴스는 각각의 4개 세그먼트에서 이 3개 열에 검색을 적용합니다. 검색은 다음과 같습니다.

세그먼트 1

처음 두 열의 값이 해당 세그먼트 1 상한 값 이하이고 세 번째 열의 값이 세그먼트 1 상한 값 미만인 경우 모든 행을 복제합니다.

세그먼트 2

세그먼트 1 행을 제외한 모든 행을 복제하는 경우는 처음 두 열의 값이 해당 세그먼트 2 상한 값 이하이고 세 번째 열의 값이 세그먼트 2 상한 값 미만일 때 해당합니다.

세그먼트 3

세그먼트 2 행을 제외한 모든 행을 복제하는 경우는 처음 두 열의 값이 해당 세그먼트 3 상한 값 이하이고 세 번째 열의 값이 세그먼트 3 상한 값 미만일 때 해당합니다.

세그먼트 4

세그먼트 1, 2 및 3 행을 제외한 나머지 모든 행을 복제합니다.

이 경우, 복제 인스턴스는 다음과 같이 WHERE 절을 작성하여 각 세그먼트를 로드합니다.

세그먼트 1

```
((COL1 < 10) OR ((COL1 = 10) AND (COL2 < 30)) OR ((COL1 = 10) AND
(COL2 = 30) AND (COL3 < 105)))
```

세그먼트 2

```
NOT ((COL1 < 10) OR ((COL1 = 10) AND (COL2 < 30)) OR ((COL1 = 10) AND
(COL2 = 30) AND (COL3 < 105))) AND ((COL1 < 20) OR ((COL1 = 20) AND
(COL2 < 20)) OR ((COL1 = 20) AND (COL2 = 20) AND (COL3 < 120)))
```

세그먼트 3

```
NOT ((COL1 < 20) OR ((COL1 = 20) AND (COL2 < 20)) OR ((COL1 = 20) AND
(COL2 = 20) AND (COL3 < 120))) AND ((COL1 < 100) OR ((COL1 = 100) AND
(COL2 < 12)) OR ((COL1 = 100) AND (COL2 = 12) AND (COL3 < 99)))
```

세그먼트 4

```
NOT ((COL1 < 100) OR ((COL1 = 100) AND (COL2 < 12)) OR ((COL1 = 100)
AND (COL2 = 12) AND (COL3 < 99)))
```

선택한 테이블 또는 뷰의 LOB 설정 지정

하나 이상의 `table-settings` 객체에 대해 `lob-settings` 옵션으로 `table-settings` 유형의 테이블 매핑 규칙을 생성하여 하나 이상의 테이블에 대한 작업 LOB 설정을 지정할 수 있습니다.

선택한 테이블 또는 뷰에 대한 LOB 설정 지정은 다음 소스 엔드포인트에서 지원됩니다.

- Oracle
- Microsoft SQL Server
- MySQL
- PostgreSQL
- IBM Db2는 `mode` 및 `bulk-max-size` 설정에 따라 다음과 같이 설명됩니다.
- SAP Adaptive Server Enterprise(ASE)는 다음과 같이 `mode` 및 `bulk-max-size` 설정에 따라 다릅니다.

선택한 테이블 또는 뷰에 대한 LOB 설정 지정은 다음 대상 엔드포인트에서 지원됩니다.

- Oracle
- Microsoft SQL Server
- MySQL
- PostgreSQL
- SAP ASE는 `mode` 및 `bulk-max-size` 설정에 따라 다음과 같이 설명됩니다.

> **Note**
>
> 기본 키를 포함하는 테이블 및 뷰에서만 LOB 데이터 형식을 사용할 수 있습니다.

선택한 테이블 또는 뷰의 LOB 설정을 사용하려면 `lob-settings` 옵션을 사용하여 `table-settings` 유형의 테이블 매핑 규칙을 생성하십시오. 이렇게 하면 `object-locator` 옵션으로 식별

된 테이블 또는 뷰에 대한 LOB 처리가 지정됩니다. `table-settings` 규칙 내에서 다음 파라미터를 사용하여 `lob-settings` 객체를 지정할 수 있습니다.

- `mode` – 다음과 같이 선택한 테이블 또는 뷰에 LOB 마이그레이션 처리 메커니즘을 지정합니다.
  - `limited` – 기본 모드인 제한적 LOB 모드는 가장 빠르고 효율적입니다. 모든 LOB의 크기가 작거나(100MB 이내) 대상 엔드포인트가 무제한 LOB 크기를 지원하지 않는 경우에만 이 모드를 사용하십시오. 또한 `limited`를 사용하는 경우 모든 LOB는 `bulk-max-size`에 설정된 크기 내에 있어야 합니다.

    전체 로드 작업의 경우, 이 모드에서 복제 인스턴스는 기본 테이블 또는 뷰 스토리지의 일부로 다른 열 데이터 형식과 함께 모든 LOB를 인라인으로 마이그레이션합니다. 그러나 인스턴스는 `bulk-max-size` 값보다 큰 마이그레이션된 LOB를 모두 지정된 크기로 자릅니다. CDC(Change Data Capture) 로드 작업의 경우, 인스턴스는 표준 전체 LOB 모드(다음 참조)에서처럼 소스 테이블 조회를 사용하여 모든 LOB를 마이그레이션합니다. LOB 크기와는 관계없습니다.

    > **Note**
    >
    > 뷰는 전체 로드 작업에 대해서만 마이그레이션할 수 있습니다.

  - `unlimited` – 전체 LOB 모드의 마이그레이션 메커니즘은 다음과 같이 `bulk-max-size`에 설정한 값에 따라 다릅니다.
    - 표준 전체 LOB 모드 – `bulk-max-size`를 0으로 설정하면 복제 인스턴스가 표준 전체 LOB 모드를 사용하여 모든 LOB를 마이그레이션합니다. 이 모드에서는 크기에 관계없이 모든 LOB를 마이그레이션하려면 소스 테이블 또는 뷰를 조회해야 합니다. 이 경우, 일반적으로 제한적 LOB 모드보다 마이그레이션 속도가 훨씬 느립니다. 대부분 또는 모든 LOB가 큰 경우에만(1GB 이상) 이 모드를 사용하십시오.
    - 조합 전체 LOB 모드 – `bulk-max-size`를 0이 아닌 값으로 설정하면 이 전체 LOB 모드는 제한적 LOB 모드와 표준 전체 LOB 모드를 조합하여 사용합니다. 전체 로드 작업에서 LOB 크기가 `bulk-max-size` 값 이내에 있는 경우, 인스턴스는 제한적 LOB 모드에서처럼 LOB를 인라인으로 마이그레이션합니다. LOB 크기가 이 값보다 큰 경우, 인스턴스는 표준 전체 LOB 모드에서와 같이 소스 테이블 또는 뷰 조회를 사용하여 LOB를 마이그레이션합니다. CDC(Change Data Capture) 로드 작업의 경우, 인스턴스는 표준 전체 LOB 모드(다음 참조)에서처럼 소스 테이블 조회를 사용하여 모든 LOB를 마이그레이션합니다. LOB 크기와는 관계없습니다.

      > **Note**
      >
      > 뷰는 전체 로드 작업에 대해서만 마이그레이션할 수 있습니다.

이 모드의 마이그레이션 속도는 빠른 제한적 LOB 모드와 느린 표준 전체 LOB 모드 사이에서 절충한 수준이 됩니다. 작고 큰 LOB가 혼합되어 있지만 대부분의 LOB가 작은 경우에만 이 모드를 사용하십시오.

이 조합된 전체 LOB 모드는 다음 엔드포인트에서만 사용 가능합니다.

- IBM Db2 소스로만 사용
- SAP ASE 소스 또는 대상으로 사용

`unlimited` 모드를 지정하는 메커니즘에 관계없이, 인스턴스는 모든 LOB를 자르지 않고 온전히 마이그레이션합니다.

- none – 복제 인스턴스는 작업 LOB 설정을 사용하여 선택한 테이블 또는 뷰의 LOB를 마이그레이션합니다. 이 옵션을 사용하면 선택한 테이블 또는 뷰의 LOB 설정 유무와 관계없이 마이그레이션 결과를 쉽게 비교할 수 있습니다.

복제에 포함된 LOB가 있는 테이블 또는 뷰를 지정한 경우 `limited` LOB 모드 사용 시 `BatchApplyEnabled` 작업 설정을 `true`로만 설정할 수 있습니다.

경우에 따라 `BatchApplyEnabled`를 `true`로, `BatchApplyPreserveTransaction`을 `false`로 설정해야 할 수 있습니다. 이러한 경우, 테이블 또는 뷰에 LOB가 있고 소스 및 대상 엔드포인트가 Oracle이면 인스턴스가 `BatchApplyPreserveTransaction`을 `true`로 설정합니다.

- `bulk-max-size` – 이전 항목에서 설명한 대로, `mode`에 따라 이 값을 0 또는 0이 아닌 값(킬로바이트)으로 설정합니다. `limited` 모드에서는 이 파라미터에 0이 아닌 값을 설정해야 합니다.

  인스턴스는 LOB를 이진 형식으로 변환합니다. 따라서 복제해야 하는 가장 큰 LOB를 지정하려면 크기에 3을 곱하십시오. 예를 들어, 가장 큰 LOB가 2MB라면 `bulk-max-size`를 6,000(6MB)으로 설정합니다.

테이블 설정 예제

다음은 테이블 설정 사용법을 보여주는 몇 가지 예제입니다.

Example 파티션별로 세그먼트화된 테이블 로드

다음 예제는 모든 파티션을 기반으로 소스의 `SALES` 테이블을 동시에 로드하기 때문에 보다 효율적입니다.

```
{
```

```
   "rules": [{
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "table-settings",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "HR",
                "table-name": "SALES"
            },
            "parallel-load": {
                "type": "partitions-auto"
            }
        }
     ]
}
```

Example 하위 파티션별로 세그먼트화된 테이블 로드

다음 예제는 모든 하위 파티션을 기반으로 Oracle 소스의 SALES 테이블을 동시에 로드하기 때문에 보다 효율적입니다.

```
{
   "rules": [{
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "table-settings",
```

```
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "HR",
                "table-name": "SALES"
            },
            "parallel-load": {
                "type": "subpartitions-auto"
            }
        }
     ]
}
```

Example 파티션 목록별로 세그먼트화된 테이블 로드

다음 예제는 특정 파티션 목록별로 소스의 SALES 테이블을 동시에 로드합니다. 여기서 지정된 파티션은 알파벳의 일부로 시작하는 값을 따라 이름이 지정됩니다(예: ABCD, EFGH 등).

```
{
    "rules": [{
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "table-settings",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "HR",
                "table-name": "SALES"
            },
            "parallel-load": {
                "type": "partitions-list",
                "partitions": [
                    "ABCD",
                    "EFGH",
                    "IJKL",
```

```
                    "MNOP",
                    "QRST",
                    "UVWXYZ"
                ]
            }
        }
    ]
}
```

Example 선택한 파티션 및 하위 파티션 목록으로 세그먼트화된 Oracle 테이블 로드

다음 예제는 선택한 파티션 및 하위 파티션 목록별로 Oracle 소스의 SALES 테이블을 동시에 로드합니다. 여기서 지정된 파티션은 알파벳의 일부로 시작하는 값을 따라 이름이 지정됩니다(예: ABCD, EFGH 등). 지정된 하위 파티션의 이름은 숫자로 시작하는 값을 따라 지정됩니다(예: 01234 및 56789).

```
{
    "rules": [{
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "table-settings",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "HR",
                "table-name": "SALES"
            },
            "parallel-load": {
                "type": "partitions-list",
                "partitions": [
                    "ABCD",
                    "EFGH",
                    "IJKL",
                    "MNOP",
                    "QRST",
                    "UVWXYZ"
```

```
                ],
                "subpartitions": [
                    "01234",
                    "56789"
                ]
            }
        }
    ]
}
```

Example 열 값 범위별로 세그먼트화된 테이블 로드

다음 예에서는 SALES_NO 및 REGION 열 값의 범위로 지정된 세그먼트별로 소스의 SALES 테이블을 동시에 로드합니다.

```
{
    "rules": [{
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "table-settings",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "HR",
                "table-name": "SALES"
            },
            "parallel-load": {
                "type": "ranges",
                "columns": [
                    "SALES_NO",
                    "REGION"
                ],
                "boundaries": [
                    [
                        "1000",
```

```
                        "NORTH"
                    ],
                    [
                        "3000",
                        "WEST"
                    ]
                ]
            }
        }
    ]
}
```

여기에서는 이름이 SALES_NO 및 REGION인 세그먼트 범위에 두 개의 열이 지정됩니다. 두 경계는 두 열 값 집합(["1000","NORTH"] 및 ["3000","WEST"])으로 지정됩니다.

따라서 이 두 경계는 동시에 로드할 다음 세 개 테이블 세그먼트를 식별합니다.

세그먼트 1

SALES_NO가 1,000 이하이고 REGION이 'NORTH'보다 작은 행. 즉, EAST 리전에서 1,000 이하의 매출 수치입니다.

세그먼트 2

세그먼트 1 이외의 행이 3,000개 SALES_NO 이하이고 'WEST' REGION 미만입니다. 즉, NORTH 및 SOUTH 리전에서 1000보다 크고 3,000 이하인 매출 수치입니다.

세그먼트 3

세그먼트 1 및 세그먼트 2 이외의 나머지 모든 행. 즉, 'WEST' 리전에서 3,000을 초과하는 매출 수치입니다.

Example 두 개의 테이블 로드: 하나는 범위별로 세그먼트화, 다른 하나는 파티션별로 세그먼트화됨

다음 예제에서는 식별하는 세그먼트 경계로 SALES 테이블을 동시에 로드합니다. 또한 이전 예제와 같이 모든 파티션으로 ORDERS 테이블을 동시에 로드합니다.

```
{
    "rules": [{
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
```

```
                "schema-name": "%",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "table-settings",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "HR",
                "table-name": "SALES"
            },
            "parallel-load": {
                "type": "ranges",
                "columns": [
                    "SALES_NO",
                    "REGION"
                ],
                "boundaries": [
                    [
                        "1000",
                        "NORTH"
                    ],
                    [
                        "3000",
                        "WEST"
                    ]
                ]
            }
        },
        {
            "rule-type": "table-settings",
            "rule-id": "3",
            "rule-name": "3",
            "object-locator": {
                "schema-name": "HR",
                "table-name": "ORDERS"
            },
            "parallel-load": {
                "type": "partitions-auto"
            }
        }
    ]
```

```
}
```

Example 제한적 LOB 모드를 사용하여 LOB 포함 테이블 로드

다음 예제는 제한적 LOB 모드(기본값)를 사용하여 소스의 LOB를 포함하는 ITEMS 테이블을 로드하며, 잘리지 않는 최대 크기로 100MB를 지원합니다. 이 크기보다 큰 LOB는 100MB로 잘립니다. 모든 LOB는 다른 모든 열 데이터 형식과 인라인으로 로드됩니다.

```
{
   "rules": [{
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "table-settings",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "INV",
                "table-name": "ITEMS"
            },
            "lob-settings": {
                "bulk-max-size": "100000"
            }
        }
     ]
}
```

Example 표준 전체 LOB 모드를 사용하여 LOB 포함 테이블 로드

다음 예제는 표준 전체 LOB 모드를 사용하여 절단하지 않은 모든 LOB를 포함하여 소스의 ITEMS 테이블을 로드합니다. 크기에 관계없이 모든 LOB는 소스 테이블의 각 LOB 조회를 사용하여 다른 데이터 형식과 별도로 로드됩니다.

```
{
   "rules": [{
```

```
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "table-settings",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "INV",
                "table-name": "ITEMS"
            },
            "lob-settings": {
                "mode": "unlimited",
                "bulk-max-size": "0"
            }
        }
    ]
}
```

Example 조합 전체 LOB 모드를 사용하여 LOB 포함 테이블 로드

다음 예제는 조합 전체 LOB 모드를 사용하여 절단하지 않은 모든 LOB를 포함하여 소스의 ITEMS 테이블을 로드합니다. 제한적 LOB 모드에서와 같이, 크기 100MB 이내의 모든 LOB는 다른 데이터 형식과 함께 인라인으로 로드됩니다. 크기가 100MB를 초과하는 모든 LOB는 다른 데이터 형식과 별도로 로드됩니다. 이 별도의 로드는 표준 전체 LOB 모드에서와 같이 소스 테이블의 각 LOB에 대해 조회를 사용합니다.

```
{
   "rules": [{
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%"
            },
            "rule-action": "include"
```

```
        },
        {
            "rule-type": "table-settings",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "INV",
                "table-name": "ITEMS"
            },
            "lob-settings": {
                "mode": "unlimited",
                "bulk-max-size": "100000"
            }
        }
    ]
}
```

Example 작업 LOB 설정을 사용하여 LOB 포함 테이블 로드

다음 예제는 작업 LOB 설정을 사용하여 모든 LOB를 포함하여 소스의 ITEMS 테이블을 로드합니다. 100MB의 bulk-max-size 설정은 무시되며 limited 또는 unlimited 모드로 신속하게 재설정하는 경우에만 유지됩니다.

```
{
   "rules": [{
            "rule-type": "selection",
            "rule-id": "1",
            "rule-name": "1",
            "object-locator": {
                "schema-name": "%",
                "table-name": "%"
            },
            "rule-action": "include"
        },
        {
            "rule-type": "table-settings",
            "rule-id": "2",
            "rule-name": "2",
            "object-locator": {
                "schema-name": "INV",
                "table-name": "ITEMS"
            },
            "lob-settings": {
```

```
                "mode": "none",
                "bulk-max-size": "100000"
            }
        }
    ]
}
```

소스 필터 사용

소스 필터를 사용하여 소스에서 대상으로 전송되는 레코드의 수와 유형을 제한할 수 있습니다. 예를 들어, 본사 사업장에서 근무하는 직원만이 대상 데이터베이스로 이동하도록 지정할 수 있습니다. 필터는 선택 규칙의 일부입니다. 데이터 열에서 필터를 적용합니다.

소스 필터는 다음 제약 조건을 준수해야 합니다.

- 선택 규칙에는 필터가 전혀 없거나 필터가 하나 이상 있을 수 있습니다.
- 모든 필터에는 하나 이상의 필터 조건이 있을 수 있습니다.
- 필터를 2개 이상 사용하는 경우, 필터 사이에서 AND 연산자를 사용하는 것처럼 필터 목록이 결합됩니다.
- 단일 필터 내에서 필터 조건을 2개 이상 사용하면 필터 조건 사이에서 OR 연산자를 사용하는 것처럼 필터 조건 목록이 결합됩니다.
- `rule-action = 'include'`인 경우 필터만이 적용됩니다.
- 필터에는 열 이름과 필터 조건 목록이 필요합니다. 필터 조건에는 연산자에 따라 값이 한 개, 두 개 또는 아무 값에도 연결되지 않는 필터 연산자가 있어야 합니다.
- 열 이름, 테이블 이름, 조회 이름 및 스키마 이름은 대소문자를 구분합니다. Oracle 및 Db2는 항상 대문자를 사용해야 합니다.
- 필터는 정확한 이름을 가진 테이블만 지원합니다. 필터는 와일드카드를 지원하지 않습니다.

소스 필터 사용 시 다음과 같은 제한 사항이 적용됩니다.

- 필터는 오른쪽에서 왼쪽으로 쓰는 언어의 열을 계산하지 않습니다.
- LOB 열에 필터를 적용하지 마십시오.
- 생성 후 업데이트되지 않는 변경 불가능 열에만 필터를 적용합니다. 생성 후 업데이트할 수 있는 변경 가능 열에 소스 필터가 적용되면 불리한 동작이 발생할 수 있습니다.

  예를 들어, 열의 특정 행을 제외하거나 포함하는 필터는 나중에 행이 변경되더라도 항상 지정된 행을 제외하거나 포함합니다. A열에 1~10 행을 제외하거나 포함시키고 나중에 11~20 행이 되도록 변

경한다고 가정합니다. 이 경우, 데이터가 더 이상 동일하지 않은 경우에도 계속 제외되거나 포함됩니다.

마찬가지로 필터 범위 외부의 행이 나중에 업데이트(또는 업데이트 및 삭제)되고, 필터에 의해 정의된 대로 제외되거나 포함되어야 한다고 가정합니다. 이 경우, 대상에서 복제됩니다.

JSON에서 소스 필터 규칙 생성

선택 규칙의 JSON `filters` 파라미터를 사용하여 소스 필터를 생성할 수 있습니다. `filters` 파라미터는 1개 이상의 JSON 객체 배열을 지정합니다. 각 객체는 소스 필터 유형, 열 이름 및 필터 조건을 지정하는 파라미터를 가지고 있습니다. 이러한 필터 조건에는 하나 이상의 필터 연산자 및 필터 값이 포함됩니다.

다음 표에는 `filters` 객체에서 소스 필터링을 지정하기 위한 파라미터가 나와 있습니다.

파라미터	값
`filter-type`	`source`
`column-name`	필터를 적용할 원본 열의 이름이 있는 파라미터입니다. 이름은 대/소문자를 구분합니다.
`filter-conditions`	`filter-operator` 값에 따라 `filter-operator` 파라미터와 0개 이상의 관련 값 파라미터를 포함하는 하나 이상의 객체로 구성된 배열입니다.
`filter-operator`	다음 값 중 하나를 가진 파라미터: • `lte` – 하나의 값보다 작거나 같음 • `ste` – 하나의 값보다 작거나 같음(`lte` 별칭) • `gte` – 하나의 값보다 크거나 같음 • `eq` – 하나의 값과 같음 • `noteq` – 하나의 값과 같지 않음 • `between` – 두 값과 같음 또는 두 값 사이의 값임 • `notbetween` – 두 값과 같지 않음 또는 두 값 사이의 값이 아님 • `null` – NULL 값

파라미터	값
	• notnull – NULL 값 없음
value 또는 start-value 및 end-value 또는 값 없음	0개 이상의 값 파라미터가 filter-operator 과 연관되어 있음: • filter-operator 가 lte, ste, gte, eq 또는 noteq인 경우, 값 파라미터 하나를 지정하는 데 value를 사용합니다. • filter-operator 가 between 또는 notbetween 인 경우, start-value 및 end-value 를 사용하여 두 개의 값 파라미터를 지정합니다. • filter-operator 가 null 또는 notnull인 경우, 값 없음 파라미터를 지정하십시오.

다음 예제에서는 소스 필터를 사용하는 일반적인 방법에 대해 다룹니다.

Example 단일 필터

다음 필터는 empid >= 100인 모든 직원을 대상 데이터베이스로 복제합니다.

```
{
    "rules": [{
        "rule-type": "selection",
        "rule-id": "1",
        "rule-name": "1",
        "object-locator": {
            "schema-name": "test",
            "table-name": "employee"
        },
        "rule-action": "include",
        "filters": [{
            "filter-type": "source",
            "column-name": "empid",
            "filter-conditions": [{
                "filter-operator": "gte",
                "value": "50"
            },{
                "filter-operator": "noteq",
                "value": "100"
            }]
        }]
```

```
    }]
}
```

Example 여러 필터 연산자

다음 필터는 여러 필터 연산자를 단일 데이터 열에 적용합니다. 이 필터는 `(empid <= 10)` OR `(empid is between 50 and 75)` OR `(empid >= 100)`인 모든 직원을 대상 데이터베이스로 복제합니다.

```
{
    "rules": [{
        "rule-type": "selection",
        "rule-id": "1",
        "rule-name": "1",
        "object-locator": {
            "schema-name": "test",
            "table-name": "employee"
        },
        "rule-action": "include",
        "filters": [{
            "filter-type": "source",
            "column-name": "empid",
            "filter-conditions": [{
                "filter-operator": "lte",
                "value": "10"
            }, {
                "filter-operator": "between",
                "start-value": "50",
                "end-value": "75"
            }, {
                "filter-operator": "gte",
                "value": "100"
            }]
        }]
    }]
}
```

Example 여러 필터

다음 필터는 여러 필터를 테이블의 열 2개에 적용합니다. 이 필터는 `(empid <= 100)` AND `(dept = tech)`인 모든 직원을 대상 데이터베이스로 복제합니다.

```
{
    "rules": [{
        "rule-type": "selection",
        "rule-id": "1",
        "rule-name": "1",
        "object-locator": {
            "schema-name": "test",
            "table-name": "employee"
        },
        "rule-action": "include",
        "filters": [{
            "filter-type": "source",
            "column-name": "empid",
            "filter-conditions": [{
                "filter-operator": "lte",
                "value": "100"
            }]
        }, {
            "filter-type": "source",
            "column-name": "dept",
            "filter-conditions": [{
                "filter-operator": "eq",
                "value": "tech"
            }]
        }]
    }]
}
```

Example NULL 값 필터링

다음 필터는 빈 값을 필터링하는 방법을 보여줍니다. 이 필터는 `dept = NULL`인 모든 직원을 대상 데이터베이스로 복제합니다.

```
{
    "rules": [{
        "rule-type": "selection",
        "rule-id": "1",
        "rule-name": "1",
        "object-locator": {
            "schema-name": "test",
```

```
            "table-name": "employee"
        },
        "rule-action": "include",
        "filters": [{
            "filter-type": "source",
            "column-name": "dept",
            "filter-conditions": [{
                "filter-operator": "null"
            }]
        }]
    }]
}
```

Example NOT 연산자를 사용한 필터링

일부 연산자는 음수 형식으로 사용할 수 있습니다. 다음 필터는 `(empid is < 50) OR (empid is > 75)`인 모든 직원을 대상 데이터베이스로 복제합니다.

```
{
    "rules": [{
        "rule-type": "selection",
        "rule-id": "1",
        "rule-name": "1",
        "object-locator": {
            "schema-name": "test",
            "table-name": "employee"
        },
        "rule-action": "include",
        "filters": [{
            "filter-type": "source",
            "column-name": "empid",
            "filter-conditions": [{
                "filter-operator": "notbetween",
                "start-value": "50",
                "end-value": "75"
            }]
        }]
    }]
}
```

Example 혼합 필터 연산자 사용

AWS DMS 버전 3.5.0부터 포함 연산자와 음수 연산자를 혼합하여 사용할 수 있습니다.

다음 필터는 `(empid != 50) AND (dept is not NULL)`인 모든 직원을 대상 데이터베이스로 복제합니다.

```
{
    "rules": [{
        "rule-type": "selection",
        "rule-id": "1",
        "rule-name": "1",
        "object-locator": {
            "schema-name": "test",
            "table-name": "employee"
        },
        "rule-action": "include",
        "filters": [{
            "filter-type": "source",
            "column-name": "empid",
            "filter-conditions": [{
                "filter-operator": "noteq",
                "value": "50"
            }]
        }, {
            "filter-type": "source",
            "column-name": "dept",
            "filter-conditions": [{
                "filter-operator": "notnull"
            }]
        }]
    }]
}
```

null을 다른 필터 연산자와 함께 사용할 때는 다음 사항에 유의하십시오.

- 동일한 필터 내에서 포함, 음수 및 null 필터 조건을 함께 사용하면 NULL 값이 있는 레코드가 복제되지 않습니다.
- 동일한 필터 내에서 포함 필터 조건 없이 음수 필터 조건과 null 필터 조건을 함께 사용하면 데이터가 복제되지 않습니다.

- null 필터 조건을 명시적으로 설정하지 않고 음수 필터 조건을 사용하면 NULL 값이 있는 레코드가 복제되지 않습니다.

시간과 날짜를 기준으로 필터링

가져올 데이터를 선택할 때 필터 기준의 일부로 날짜 또는 시간을 지정할 수 있습니다. AWS DMS에서는 필터링에 YYYY-MM-DD 날짜 형식과 YYYY-MM-DD HH:MM:SS 시간 형식을 사용합니다. AWS DMS 비교 함수는 SQLite 규칙을 준수합니다. SQLite 데이터 형식과 날짜 비교에 관한 자세한 내용은 SQLite 설명서의 SQLite 버전 3의 데이터 형식을 참조하십시오.

다음 필터는 날짜에서 필터링하는 방법을 보여줍니다. 이 필터는 empstartdate >= January 1, 2002인 모든 직원을 대상 데이터베이스로 복제합니다.

Example 단일 날짜 필터

```
{
    "rules": [{
        "rule-type": "selection",
        "rule-id": "1",
        "rule-name": "1",
        "object-locator": {
            "schema-name": "test",
            "table-name": "employee"
        },
        "rule-action": "include",
        "filters": [{
            "filter-type": "source",
            "column-name": "empstartdate",
            "filter-conditions": [{
                "filter-operator": "gte",
                "value": "2002-01-01"
            }]
        }]
    }]
}
```

작업에 대한 마이그레이션 전 평가 활성화 및 활용

마이그레이션 전 평가에서는 데이터베이스 마이그레이션 작업의 특정 구성 요소를 평가하여 마이그레이션 작업이 예상대로 실행되는 데 방해가 될 수 있는 문제를 식별하는 데 도움이 됩니다. 이 평가를 통해 새 작업이나 수정된 작업을 실행하기 전에 문제를 식별하고 수정할 수 있습니다. 이를 통해 요구 사항 누락 또는 알려진 제한 사항으로 인한 작업 실패와 관련된 지연을 피할 수 있습니다.

AWS DMS 프리미어그레이션 평가를 위한 두 가지 옵션에 대한 액세스를 제공합니다.

- 데이터 유형 평가: 제한된 평가 범위를 제공하는 기존 보고서입니다.
- 업그레이드 평가 실행: 데이터 유형 평가 결과를 비롯한 다양한 유형의 개별 평가를 포함합니다.

> **Note**
>
> 프리미어레이션 평가를 선택하는 경우 데이터형 평가를 따로 선택할 필요가 없습니다.

이러한 옵션은 다음 항목에 설명되어 있습니다.

- 마이그레이션 전 평가 실행 지정, 시작, 보기: 프리미어 (권장) 평가 실행은 신규 또는 기존 마이그레이션 작업 구성을 기반으로 실행할 하나 이상의 개별 평가를 지정합니다. 각 개별 평가는 마이그레이션 유형, 지원되는 객체, 인덱스 구성 및 마이그레이션할 스키마와 테이블을 식별하는 테이블 매핑과 같은 기타 작업 설정과 같은 기준의 관점에서 지원되는 원본 및/또는 대상 데이터베이스의 특정 요소를 평가합니다.

  예를 들어, 개별 평가를 통해 엔진 버전을 기반으로 마이그레이션할 수 있거나 마이그레이션할 수 없는 원본 데이터 유형이나 기본 키 형식을 평가할 수 있습니다. AWS DMS AWS DMS 관리 콘솔을 사용하거나 AWS CLI 및 SDK를 사용하여 API에 액세스하여 최신 평가 실행을 시작하고 결과를 확인하고 작업에 대한 이전의 모든 평가 실행 결과를 볼 수 있습니다. AWS DMS 또한 결과를 AWS DMS 저장하기 위해 선택한 Amazon S3 버킷의 작업에 대한 이전 평가 실행 결과를 볼 수 있습니다.

  > **Note**
  >
  > 사용 가능한 개별 평가의 수와 유형은 시간이 지남에 따라 증가할 수 있습니다. 주기적 업데이트에 관한 자세한 내용은 개별 평가 지정을 참조하십시오.

- 데이터 유형 평가 시작 및 보기 (레거시): 데이터 유형 (레거시) 평가는 단일 JSON 구조, 즉 지원되는 관계형 소스 데이터베이스 인스턴스에서 올바르게 마이그레이션되지 않을 수 있는 데이터 유형에서

단일 유형의 프리미그레이션 평가 결과를 반환합니다. 이 보고서는 마이그레이션을 위해 선택한 원본 데이터베이스의 모든 스키마와 테이블에서 발견된 문제가 있는 모든 데이터 유형에 대한 결과를 반환합니다.

프리그레이션 평가를 위한 사전 요구 사항 만들기

이 섹션에서는 프리미어 평가를 생성하는 데 필요한 Amazon S3 및 IAM 리소스에 대해 설명합니다.

S3 버킷 생성

AWS DMS S3 버킷에 프리미어 평가 보고서를 저장합니다. S3 버킷을 생성하려면 다음과 같이 하십시오.

1. AWS Management Console 로그인하고 https://console.aws.amazon.com/s3/ 에서 Amazon S3 콘솔을 엽니다.
2. 버킷 생성을 선택합니다.
3. 버킷 생성 페이지에서 버킷의 *### ### ### ##### ### ## (#: dms-bucket -yoursignin) # #####.*
4. DMS 마이그레이션 작업에 AWS 리전 사용할 를 선택합니다.
5. 나머지 설정은 그대로 두고 버킷 생성을 선택합니다.

IAM 리소스 생성

DMS는 IAM 역할 및 정책을 사용하여 S3 버킷에 액세스하여 프리미어레이션 평가 결과를 저장합니다.

IAM 정책을 생성하려면 다음과 같이 하십시오.

1. https://console.aws.amazon.com/iam/ 에서 AWS Management Console 로그인하고 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 생성 페이지에서 JSON 탭을 선택합니다.
5. 다음 JSON 코드를 편집기에 붙여 넣으면서 예제 코드를 바꿉니다. *my-bucket#* 이전 섹션에서 생성한 Amazon S3 버킷의 이름으로 바꾸십시오.

```
{
```

```
    "Version":"2012-10-17",
    "Statement":[
       {
          "Effect":"Allow",
          "Action":[
             "s3:PutObject",
             "s3:DeleteObject",
             "s3:GetObject",
             "s3:PutObjectTagging"
          ],
          "Resource":[
             "arn:aws:s3:::my-bucket/*"
          ]
       },
       {
          "Effect":"Allow",
          "Action":[
             "s3:ListBucket",
             "s3:GetBucketLocation"
          ],
          "Resource":[
             "arn:aws:s3:::my-bucket"
          ]
       }
    ]
}
```

6. [다음: 태그] 를 선택한 다음 [다음: 검토] 를 선택합니다.
7. 이름*의 경우 **DMSPremigrationAssessmentS3Policy**를 입력한 후 정책 생성을 선택합니다.

IAM 역할을 생성하려면 다음과 같이 하십시오.

1. IAM 콘솔의 탐색 창에서 역할(Roles)을 선택합니다.
2. 역할 생성을 선택합니다.
3. 신뢰할 수 있는 엔터티 선택 페이지의 신뢰할 수 있는 엔터티 유형 아래에서 AWS 서비스를 선택합니다. 다른 AWS 서비스의 사용 사례에서는 DMS를 선택합니다.
4. DMS 확인란을 선택한 후 다음을 선택합니다.
5. 권한 추가 페이지에서 DMS PremigrationAssessment S3정책을 선택합니다. 다음을 선택합니다.
6. 이름 지정, 검토 및 생성 페이지에서 역할 이름에 **DMSPremigrationAssessmentS3Role**을 입력하고 역할 생성을 선택합니다.

7. 역할 페이지에서 역할 이름에 **DMSPremigrationAssessmentS3Role**을 입력합니다. DMS S3 역할을 선택합니다. PremigrationAssessment
8. DMS PremigrationAssessment S3Role 페이지에서 신뢰 관계 탭을 선택합니다. 신뢰 정책 편집을 선택합니다.
9. 신뢰 정책 편집 페이지에서 기존 텍스트를 대체하여 다음 JSON을 편집기에 붙여넣습니다.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"",
         "Effect":"Allow",
         "Principal":{
            "Service":"dms.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

   이 정책은 DMS에 프리미어레이션 평가 실행 결과를 S3 버킷에 넣을 수 있는 `sts:AssumeRole` 권한을 부여합니다.
10. 정책 업데이트를 선택합니다.

마이그레이션 전 평가 실행 지정, 시작, 보기

프리미그레이션 평가는 신규 또는 기존 마이그레이션 작업 구성을 기반으로 실행할 하나 이상의 개별 평가를 지정합니다. 개별 평가에서는 마이그레이션 유형, 지원되는 객체, 인덱스 구성, 기타 작업 설정(예: 마이그레이션할 스키마와 테이블을 식별하는 테이블 매핑) 등의 고려 사항에 따라 소스 또는 대상 데이터베이스의 특정 요소를 평가합니다. 예를 들어 개별 평가를 통해 마이그레이션할 수 있는 소스 데이터 유형이나 기본 키 형식을 평가할 수 있습니다.

개별 평가 지정

새 평가 실행을 생성할 때 작업 구성에 적용할 수 있는 개별 평가 중 일부 또는 전체를 실행하도록 선택할 수 있습니다.

AWS DMS 다음과 같은 관계형 원본 및 대상 데이터베이스 엔진에 대한 프리미어화 평가 실행을 지원합니다.

- 오라클 평가
- SQL 서버 평가
- MySQL 평가(MariaDB 및 아마존 Aurora MySQL 호환 에디션 포함)
- PostgreSQL 평가(아마존 Aurora 포스트그레SQL 호환 에디션 포함)

마이그레이션 전 평가 실행 시작 및 보기

AWS DMS 관리 콘솔, 및 API를 사용하여 신규 또는 기존 마이그레이션 작업에 대한 프리미어 평가 실행을 시작할 수 있습니다. AWS CLI AWS DMS

마이그레이션 전 평가를 시작하려면 새 작업이나 기존 작업에 대해 평가를 실행하십시오.

1. AWS DMS 관리 콘솔의 데이터베이스 마이그레이션 작업 페이지에서 다음 중 하나를 수행하십시오.

   - 새 작업을 생성하고 평가하려면 작업 생성을 선택합니다. 데이터베이스 마이그레이션 작업 생성 페이지가 열립니다.

     1. 테이블 매핑을 포함하여 작업을 생성하는 데 필요한 작업 설정을 입력합니다.
     2. 업그레이드 평가 섹션에서 업그레이드 평가 실행 확인란이 선택되어 있습니다. 이 페이지에는 새 작업에 대한 평가 실행을 지정하는 옵션이 있습니다.

     > **Note**
     >
     > 새 작업을 생성할 때 마이그레이션 전 평가 실행을 활성화하면 작업 생성 시 작업을 자동으로 시작하는 옵션이 비활성화됩니다. 평가 실행이 완료된 후 수동으로 작업을 시작할 수 있습니다.

   - 기존 작업을 평가하려면 데이터베이스 마이그레이션 작업 페이지에서 기존 작업의 식별자를 선택합니다. 선택한 기존 작업의 작업 페이지가 열립니다.

     1. 작업을 선택하고 마이그레이션 전 평가 생성을 선택합니다. 기존 작업에 대한 평가 실행을 지정하는 옵션이 포함된 프리미엄 평가 생성 페이지가 열립니다.

2. 평가 실행에 사용할 고유한 이름을 입력하거나 기본값을 그대로 두십시오.
3. 이 평가 실행에 포함하려는 사용 가능한 개별 평가를 선택합니다. 현재 작업 설정을 기반으로 사용 가능한 개별 평가만 선택할 수 있습니다. 기본적으로 사용 가능한 모든 개별 평가가 활성화되고 선택됩니다.

4. 계정에서 평가 결과 보고서를 저장할 Amazon S3 버킷과 폴더를 검색하고 선택합니다. 평가 실행을 위한 리소스 설정에 대한 자세한 내용은 을 참조하십시오프리그레이션 평가를 위한 사전 요구 사항 만들기.
5. 선택한 Amazon S3 버킷 및 폴더에 대한 전체 계정 액세스 권한이 있는 IAM 역할을 선택하거나 입력합니다. 평가 실행을 위한 리소스 설정에 대한 자세한 내용은 을 참조하십시오프리그레이션 평가를 위한 사전 요구 사항 만들기.
6. Amazon S3 버킷에 있는 평가 결과 보고서를 암호화하는 설정을 선택할 수 있습니다. S3 버킷 암호화에 대한 자세한 내용은 Amazon S3 버킷의 기본 서버 측 암호화 동작 설정을 참조하십시오.
7. 새 작업에 대해 작업 생성을 선택하거나 기존 작업에 대해 생성을 선택합니다.

   데이터베이스 마이그레이션 작업 페이지가 열리고 새 작업이나 수정된 작업이 나열되며 상태는 생성 중...입니다. 그리고 해당 작업이 생성되면 마이그레이션 전 평가 실행이 시작된다는 배너 메시지도 함께 표시됩니다.

AWS DMS AWS DMS 관리 콘솔, 또는 API를 사용하여 최신 및 이전의 모든 프리미어 평가 실행에 액세스할 수 있습니다. AWS CLI AWS DMS

평가 실행 결과를 보려면

1. AWS DMS Management Console의 데이터베이스 마이그레이션 작업 페이지에서 기존 작업의 식별자를 선택합니다. 기존 작업의 작업 페이지가 열립니다.
2. 기존 작업 페이지에서 마이그레이션 전 평가 탭을 선택합니다. 그러면 해당 페이지에 Premigration 평가 섹션이 열리고 평가 실행 결과가 시간순으로 이름별로 나열되어 있습니다. 최신 결과가 목록 상단에 표시됩니다. 결과를 보려는 평가 실행의 이름을 선택합니다.

이러한 평가 실행 결과는 최근 평가 실행의 이름과 그 상태에 관한 개요를 시작으로 하여 지정된 개별 평가 및 그 상태를 일련의 목록으로 보여줍니다. 그런 다음, 목록에서 평가 이름을 선택하여 개별 평가의 상태 세부 정보를 탐색할 수 있으며 평가 결과는 표의 열 수준까지 확인할 수 있습니다.

평가 실행의 상태 개요와 개별 평가에서는 상태 값이 모두 표시됩니다. 이 값은 평가 실행의 전체 상태와 개별 평가의 유사한 상태를 나타냅니다. 평가 실행의 상태 값을 목록으로 열거하면 다음과 같습니다.

- `"cancelling"` – 평가 실행이 취소되었습니다.
- `"deleting"` – 평가 실행이 삭제되었습니다.
- `"failed"` – `failed` 상태가 지정된 개별 평가가 하나 이상 완료되었습니다.

- "error-provisioning" – 리소스가 프로비저닝되는 동안 (provisioning 상태 도중) 내부 오류가 발생했습니다.
- "error-executing" – 개별 평가를 실행하는 동안 (running 상태 도중) 내부 오류가 발생했습니다.
- "invalid state" – 평가 실행이 알 수 없는 상태입니다.
- "passed" – 모든 개별 평가가 완료되었으며 failed 상태가 있는 평가는 없습니다.
- "provisioning" – 개별 평가를 실행하는 데 필요한 리소스가 프로비저닝되고 있습니다.
- "running" – 개별 평가가 실행 중입니다.
- "starting" – 평가 실행이 시작되었지만 개별 평가를 위한 리소스가 아직 프로비저닝되지 않았습니다.

평가 실행의 개별 평가에 대한 상태 값을 목록으로 열거하면 다음과 같습니다.

- "cancelled" – 평가 실행 취소의 일환으로 개별 평가가 취소되었습니다.
- "error" – 개별 평가가 성공적으로 완료되지 않았습니다.
- "failed" – 개별 평가가 성공적으로 완료되었지만 검증에 실패했다는 결과: 자세한 내용은 해당 결과의 세부 정보를 참조하십시오.
- "invalid state" – 개별 평가의 상태가 알 수 없는 상태입니다.
- "passed" – 개별 평가가 완료되었고 성공적인 검증 결과를 얻었습니다.
- "pending" – 개별 평가가 실행 대기 중입니다.
- "running" – 개별 평가가 실행 중입니다.
- "warning" – 개별 평가를 성공적으로 완료하고 경고 검증 결과를 얻음: 자세한 내용은 해당 결과의 세부 정보를 참조하십시오.

Amazon S3의 평가 실행 결과에 대한 JSON 파일도 볼 수 있습니다.

Amazon S3의 평가 실행에 대한 JSON 파일을 보려면

1. AWS DMS 관리 콘솔에서 평가 실행 상태 개요에 표시된 Amazon S3 버킷 링크를 선택합니다. 그러면 버킷에 저장된 버킷 폴더 및 기타 Amazon S3 객체 목록이 표시됩니다. 결과가 버킷 폴더에 저장되어 있다면 폴더를 엽니다.
2. 여러 JSON 파일에서 평가 실행 결과를 찾을 수 있습니다. summary.json 파일에는 평가 실행의 전체 결과가 포함됩니다. 나머지 파일에는 평가 실행을 위해 지정된 개별 평가의 이름(예:

unsupported-data-types-in-source.json)이 각각 지정됩니다. 각 파일에는 선택한 평가 실행의 해당 개별 평가 결과가 포함되어 있습니다.

다음 CLI 명령과 API 작업을 실행하여 기존 마이그레이션 작업에 대한 업그레이드 평가 실행 결과를 보고 시작할 수 있습니다. AWS DMS

- CLI: describe-applicable-individual-assessments, API: DescribeApplicableIndividualAssessments – 하나 이상의 작업 구성 파라미터가 주어지면 신규 마이그레이션 전 평가 실행에 대해 지정할 수 있는 개별 평가를 목록으로 보여줍니다.
- CLI: start-replication-task-assessment-run, API: StartReplicationTaskAssessmentRun – 기존 마이그레이션 작업에 대한 하나 이상의 개별 평가에 대해 신규 마이그레이션 전 평가 실행을 시작합니다.
- CLI: describe-replication-task-assessment-runs, API: DescribeReplicationTaskAssessmentRuns - 필터 설정에 따라 마이그레이션 전 실행의 페이지가 매겨진 목록을 반환합니다.
- CLI: describe-replication-task-individual-assessments, API: DescribeReplicationTaskIndividualAssessments - 필터 설정에 따라 개별 평가의 페이지가 매겨진 목록을 반환합니다.
- CLI: cancel-replication-task-assessment-run, API: CancelReplicationTaskAssessmentRun – 단일 마이그레이션 전 평가 실행을 취소하지만 삭제하지는 않습니다.
- CLI: delete-replication-task-assessment-run, API: DeleteReplicationTaskAssessmentRun – 단일 마이그레이션 전 평가 실행의 기록을 삭제합니다.

개별 평가

이 섹션에서는 개별 프리미어 평가에 대해 설명합니다.

API를 사용하여 개별 업그레이드 평가를 생성하려면 나열된 AWS DMS API 키를 작업 매개변수로 사용하십시오. `IncludeOnly` StartReplicationTaskAssessmentRun

주제

- 모든 엔드포인트 유형에 대한 평가
- 오라클 평가

- SQL 서버 평가
- MySQL 평가
- MariaDB 평가
- PostgreSQL 평가

모든 엔드포인트 유형에 대한 평가

이 섹션에서는 모든 엔드포인트 유형에 대한 개별 프리미어 평가에 대해 설명합니다.

주제

- 지원되지 않는 데이터 형식
- 대형 객체 (LOB) 가 사용되지만 대상 LOB 열은 Null로 지정할 수 없습니다.
- 대형 객체 (LOB) 가 포함되지만 기본 키나 고유한 제약 조건은 없는 소스 테이블
- 기본 키가 없는 소스 테이블 (CDC 또는 전체 로드 및 CDC 작업에만 해당)
- 기본 키가 없는 대상 테이블 (CDC 작업 전용)
- 지원되지 않는 소스 기본 키 유형 - 복합 기본 키

지원되지 않는 데이터 형식

API 키: `unsupported-data-types-in-source`

소스 엔드포인트에서 DMS가 지원하지 않는 데이터 유형이 있는지 확인합니다. 모든 데이터 형식을 엔진 간에 마이그레이션할 수 있는 것은 아닙니다.

대형 객체 (LOB) 가 사용되지만 대상 LOB 열은 Null로 지정할 수 없습니다.

API 키: `full-lob-not-nullable-at-target`

복제에 전체 LOB 모드 또는 인라인 LOB 모드를 사용하는 경우 대상의 LOB 열이 무효화되는지 확인합니다. DMS에서 이러한 LOB 모드를 사용할 경우 LOB 열이 null이어야 합니다. 이 평가를 위해서는 원본 및 대상 데이터베이스가 관계형이어야 합니다.

대형 객체 (LOB) 가 포함되지만 기본 키나 고유한 제약 조건은 없는 소스 테이블

API 키: `table-with-lob-but-without-primary-key-or-unique-constraint`

LOB는 있지만 기본 키나 고유 키는 없는 소스 테이블이 있는지 확인합니다. DMS가 LOB를 마이그레이션하려면 테이블에 기본 키 또는 고유 키가 있어야 합니다. 이 평가를 위해서는 원본 데이터베이스가 관계형이어야 합니다.

기본 키가 없는 소스 테이블 (CDC 또는 전체 로드 및 CDC 작업에만 해당)

API 키: `table-with-no-primary-key-or-unique-constraint`

전체 로드 및 변경 데이터 캡처 (CDC) 마이그레이션 또는 CDC 전용 마이그레이션을 위해 소스 테이블에 기본 키나 고유 키가 있는지 확인합니다. 기본 키나 고유 키가 없으면 CDC 마이그레이션 중에 성능 문제가 발생할 수 있습니다. 이 평가를 위해서는 원본 데이터베이스가 관계형이어야 하고 마이그레이션 유형에는 CDC가 포함되어야 합니다.

기본 키가 없는 대상 테이블 (CDC 작업 전용)

API 키: `target-table-has-unique-key-or-primary-key-for-cdc`

CDC 전용 마이그레이션을 위해 이미 생성된 대상 테이블에 기본 키 또는 고유 키가 있는지 확인합니다. 기본 키나 고유 키가 없으면 DMS에서 업데이트를 적용하고 삭제할 때 대상에서 전체 테이블을 스캔할 수 있습니다. 이로 인해 CDC 마이그레이션 중에 성능 문제가 발생할 수 있습니다. 이 평가를 위해서는 대상 데이터베이스가 관계형이어야 하고 마이그레이션 유형에는 CDC가 포함되어야 합니다.

지원되지 않는 소스 기본 키 유형 - 복합 기본 키

API 키: `unsupported-source-pk-type-for-elasticsearch-target`

Amazon OpenSearch Service로 마이그레이션할 때 원본 테이블에 복합 기본 키가 있는지 확인합니다. 소스 테이블의 기본 키는 단일 열로 구성되어야 합니다. 이 평가를 위해서는 원본 데이터베이스가 관계형이고 대상 데이터베이스가 DynamoDB여야 합니다.

> **Note**
>
> DMS는 소스 기본 키가 여러 열로 구성된 OpenSearch 서비스 대상으로 소스 데이터베이스를 마이그레이션하는 것을 지원합니다.

오라클 평가

이 섹션에서는 Oracle 소스 엔드포인트를 사용하는 마이그레이션 작업에 대한 개별 마이그레이션 전 평가에 대해 설명합니다.

> **Note**
>
> 이 섹션의 마이그레이션 전 평가 기능을 사용하려면 다음 권한을 dms_user에 추가해야 합니다.
>
> ```
> grant select on gv_$parameter to dms_user;
> grant select on v_$instance to dms_user;
> grant select on v_$version to dms_user;
> grant select on gv_$ASM_DISKGROUP to dms_user;
> grant select on gv_$database to dms_user;
> grant select on DBA_DB_LINKS to to dms_user;
> grant select on gv_$log_History to dms_user;
> grant select on gv_$log to dms_user;
> grant select on dba_types to dms_user;
> ```
>
> Oracle을 소스로 사용할 때의 권한에 관한 자세한 내용은 자체 관리형 Oracle 소스에 필요한 사용자 계정 권한은 다음과 같습니다. AWS DMS을 참조하십시오.

주제

- 데이터베이스 수준 보충 로깅 확인
- 대기에 필요한 DB 링크가 생성되었는지 확인합니다.
- LOB 데이터 형식 및 바이너리 리더 구성 여부에 대한 Oracle 검증
- 데이터베이스가 CDB인지 확인하십시오.
- Oracle Database Edition을 확인하세요.
- DMS에 대한 Oracle CDC 방법을 검증하십시오.
- DMS용 Oracle RAC 구성을 검증하십시오.

데이터베이스 수준 보충 로깅 확인

API 키: `oracle-supplemental-db-level`

이 마이그레이션 전 평가는 데이터베이스 수준에서 최소 보충 로깅이 활성화되는지 여부를 검증합니다. Oracle 데이터베이스를 마이그레이션 소스로 사용하려면 보충 로깅을 활성화해야 합니다.

보충 로깅을 활성화하려면 다음 쿼리를 사용하십시오.

```
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA
```

자세한 정보는 보충 로깅 설정을 참조하세요.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

대기에 필요한 DB 링크가 생성되었는지 확인합니다.

API 키: `oracle-validate-standby-dblink`

이 프리미어그레이션 평가는 Oracle 대기 데이터베이스 소스에 대해 Dblink가 생성되었는지 여부를 검증합니다. AWSDMS_DBLINK 대기 데이터베이스를 소스로 사용하기 위한 전제 조건입니다. Oracle Standby를 소스로 사용하는 경우, AWS DMS 는 기본적으로 열린 트랜잭션의 유효성을 검사하지 않습니다.

자세한 정보는 자체 관리형 Oracle 데이터베이스를 원본으로 사용 AWS DMS을 참조하세요.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

LOB 데이터 형식 및 바이너리 리더 구성 여부에 대한 Oracle 검증

API 키: `oracle-binary-lob-source-validation`

이 프리미어그레이션 LogMiner 평가는 Oracle 데이터베이스 엔드포인트 버전 12c 이상에 Oracle을 사용하는지 여부를 검증합니다. AWS DMS Oracle 데이터베이스 버전 12c에서 LOB 열을 마이그레이션하는 LogMiner 경우 Oracle을 지원하지 않습니다. 또한 이 평가에서는 LOB 열이 있는지 확인하고 적절한 권장 사항을 제공합니다.

Oracle을 사용하지 않도록 마이그레이션을 구성하려면 LogMiner 원본 엔드포인트에 다음 구성을 추가하십시오.

```
useLogMinerReader=N;useBfile=Y;
```

자세한 정보는 CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용을 참조하세요.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

데이터베이스가 CDB인지 확인하십시오.

API 키: `oracle-validate-cdb`

이 마이그레이션 전 평가는 데이터베이스가 컨테이너 데이터베이스인지 검증합니다. AWS DMS 는 멀티테넌트 컨테이너 루트 데이터베이스(CDB$ROOT)는 지원하지 않습니다.

> **Note**
>
> 이 평가는 Oracle 12.1.0.1 또는 이후 버전에만 필요합니다. 이 평가는 12.1.0.1 이전의 Oracle 버전에는 적용되지 않습니다.

자세한 정보는 Oracle을 원본으로 사용하는 데 따른 제한 사항 AWS DMS을 참조하세요.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

Oracle Database Edition을 확인하세요.

API 키: `oracle-check-cdc-support-express-edition`

이 마이그레이션 전 평가는 Oracle 소스 데이터베이스가 Express Edition인지 검증합니다. AWS DMS 는 Oracle Express Edition(Oracle Database XE) 버전 18.0 및 이후 버전에 대한 CDC는 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

DMS에 대한 Oracle CDC 방법을 검증하십시오.

API 키: `oracle-recommendation-cdc-method`

이 업그레이드 평가는 지난 7일간의 리두 로그 생성을 검증하고 CDC용 AWS DMS Binary Reader를 사용할지 아니면 Oracle을 사용할지를 권장합니다. LogMiner

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

사용할 CDC 방법을 결정하는 방법에 관한 자세한 내용은 CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용 단원을 참조하세요.

DMS용 Oracle RAC 구성을 검증하십시오.

API 키: `oracle-check-rac`

이 마이그레이션 전 평가는 Oracle 데이터베이스가 Real Application Cluster인지 여부를 검증합니다. Real Application Cluster 데이터베이스를 올바르게 구성해야 합니다. 데이터베이스가 RAC를 기반으로 하는 경우 Oracle 대신 CDC용 AWS DMS 바이너리 리더를 사용하는 것이 좋습니다. LogMiner

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 정보는 CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용을 참조하세요.

SQL 서버 평가

이 섹션에서는 Microsoft SQL Server 원본 엔드포인트를 사용하는 마이그레이션 작업에 대한 개별 마이그레이션 전 평가에 대해 설명합니다.

주제

- 데이터베이스 복구 모델이 단순한지 확인합니다.
- 작업 범위의 테이블에 계산된 열이 포함되어 있는지 확인합니다.
- 작업 범위의 테이블에 열 저장소 인덱스가 있는지 확인합니다.
- 메모리 최적화 테이블이 작업 범위에 속하는지 확인합니다.
- 임시 테이블이 작업 범위에 속하는지 확인합니다.
- 데이터베이스 수준에서 지연된 내구성이 활성화되어 있는지 확인합니다.
- 데이터베이스 수준에서 가속화된 데이터 복구가 활성화되어 있는지 확인하십시오.
- 테이블 매핑에 기본 키가 포함된 1만 개 이상의 테이블이 있는지 확인하십시오.

데이터베이스 복구 모델이 단순한지 확인합니다.

API 키: `sqlserver-check-for-recovery-model`

이 업그레이드 평가는 소스 엔드포인트 복구 모델을 검증합니다. AWS DMS 복구 모델을 지속적인 복제로 `Bulk logged` 설정하거나 계속 복제하도록 설정해야 합니다`Full`.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 정보는 SQL Server 소스에서 지속적 복제(CDC) 사용을 위한 사전 요구 사항을 참조하세요.

작업 범위의 테이블에 계산된 열이 포함되어 있는지 확인합니다.

API 키: `sqlserver-check-for-computed-fields`

이 업그레이드 평가에서는 계산된 열이 있는지 확인합니다. AWS DMS SQL Server 계산 열의 변경 내용 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 정보는 SQL Server를 원본으로 사용할 때의 제한 사항 AWS DMS을 참조하세요.

작업 범위의 테이블에 열 저장소 인덱스가 있는지 확인합니다.

API 키: `sqlserver-check-for-columnstore-indexes`

이 업그레이드 평가에서는 열 저장소 인덱스가 있는 테이블이 있는지 확인합니다. AWS DMS 열 저장소 인덱스가 있는 SQL Server 테이블의 변경 내용 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 정보는 SQL Server를 원본으로 사용할 때의 제한 사항 AWS DMS을 참조하세요.

메모리 최적화 테이블이 작업 범위에 속하는지 확인합니다.

API 키: `sqlserver-check-for-memory-optimized-tables`

이 업그레이드 평가에서는 메모리가 최적화된 테이블이 있는지 확인합니다. AWS DMS 메모리가 최적화된 테이블의 변경 내용 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 정보는 SQL Server를 원본으로 사용할 때의 제한 사항 AWS DMS을 참조하세요.

임시 테이블이 작업 범위에 속하는지 확인합니다.

API 키: `sqlserver-check-for-temporal-tables`

이 프리미어그레이션 평가에서는 임시 테이블이 있는지 확인합니다. AWS DMS 임시 테이블의 변경 내용 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 정보는 SQL Server를 원본으로 사용할 때의 제한 사항 AWS DMS을 참조하세요.

데이터베이스 수준에서 지연된 내구성이 활성화되어 있는지 확인합니다.

API 키: `sqlserver-check-for-delayed-durability`

이 프리미어그레이션 평가에서는 지연된 내구도가 있는지 확인합니다. AWS DMS 지연된 내구성을 사용하는 트랜잭션의 변경 사항 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 정보는 SQL Server를 원본으로 사용할 때의 제한 사항 AWS DMS을 참조하세요.

데이터베이스 수준에서 가속화된 데이터 복구가 활성화되어 있는지 확인하십시오.

API 키: `sqlserver-check-for-accelerated-data-recovery`

이 업그레이드 평가는 가속화된 데이터 복구가 있는지 확인합니다. AWS DMS 데이터 복구 속도가 빠른 데이터베이스의 변경 사항 복제를 지원하지 않습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자세한 정보는 SQL Server를 원본으로 사용할 때의 제한 사항 AWS DMS을 참조하세요.

테이블 매핑에 기본 키가 포함된 1만 개 이상의 테이블이 있는지 확인하십시오.

API 키: `sqlserver-large-number-of-tables`

이 마이그레이션 전 평가에서는 기본 키가 포함된 10,000개 이상의 테이블이 있는지 확인합니다. MS-Replication으로 구성된 데이터베이스는 기본 키가 포함된 테이블이 너무 많으면 작업 실패가 발생할 수 있습니다.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

MS-Replication 구성에 관한 자세한 내용은 온프레미스 또는 Amazon EC2에서 자체 관리형 SQL Server의 데이터 변경 캡처 섹션을 참조하세요.

MySQL 평가

이 섹션에서는 MySQL 소스 엔드포인트를 사용하는 마이그레이션 작업에 대한 개별 업그레이드 평가에 대해 설명합니다.

주제

- 테이블이 Innodb 이외의 스토리지 엔진을 사용하는지 검증하십시오.
- 마이그레이션에 사용되는 모든 테이블에서 자동 증분이 활성화되어 있는지 확인합니다.
- 데이터베이스 binlog 이미지가 DMS CDC를 FULL 지원하도록 설정되었는지 확인합니다.
- 원본 데이터베이스가 MySQL 읽기 전용 복제본인지 확인
- 테이블에 파티션이 있는지 확인하고 전체 로드 작업 설정을 권장합니다target_table_prep_mode.
- DMS가 데이터베이스 버전을 지원하는지 확인하십시오.
- 대상 데이터베이스가 1로 local_infile 설정되도록 구성되어 있는지 확인합니다.
- 대상 데이터베이스에 외래 키가 있는 테이블이 있는지 확인합니다.
- 작업 범위의 원본 테이블에 계단식 제약 조건이 있는지 검증하십시오.
- 타임아웃 값이 MySQL 소스 또는 타겟에 적합한지 검증

테이블이 Innodb 이외의 스토리지 엔진을 사용하는지 검증하십시오.

API 키: `mysql-check-table-storage-engine`

이 업그레이드 평가는 Source MySQL 데이터베이스의 테이블에 사용되는 스토리지 엔진이 Innodb가 아닌 다른 엔진인지 여부를 검증합니다. DMS는 기본적으로 InnoDB 스토리지 엔진으로 대상 테이블을 생성합니다. InnoDB 이외의 스토리지 엔진을 사용해야 하는 경우 대상 데이터베이스에 테이블을 수동으로 생성하고 전체 로드 작업 설정으로 `TRUNCATE_BEFORE_LOAD` 사용하거나 `DO_NOTHING` 사용할 DMS 작업을 구성해야 합니다. 전체 로드 작업 설정에 대한 자세한 내용은 을 참조하십시오. 전체 로드 작업 설정

MySQL 엔드포인트 제한에 대한 자세한 내용은 을 참조하십시오. AWS DMS에서 MySQL 데이터베이스를 소스로 사용 시 적용되는 제한 사항

마이그레이션에 사용되는 모든 테이블에서 자동 증분이 활성화되어 있는지 확인합니다.

API 키: `mysql-check-auto-increment`

이 업그레이드 평가는 작업에 사용되는 원본 테이블에 자동 증분이 활성화되어 있는지 여부를 검증합니다. DMS는 열의 AUTO_INCREMENT 속성을 대상 데이터베이스로 마이그레이션하지 않습니다.

MySQL 엔드포인트 제한에 대한 자세한 내용은 을 참조하십시오. AWS DMS에서 MySQL 데이터베이스를 소스로 사용 시 적용되는 제한 사항 MySQL의 ID 열 처리에 대한 자세한 내용은 2부의 ID 열 처리를 참조하십시오 AWS DMS.

데이터베이스 binlog 이미지가 DMS CDC를 **FULL** 지원하도록 설정되었는지 확인합니다.

API 키: `mysql-check-binlog-image`

이 업그레이드 평가는 원본 데이터베이스의 binlog 이미지가 로 설정되었는지 여부를 확인합니다. FULL MySQL에서 `binlog_row_image` 변수는 형식을 사용할 때 이진 로그 이벤트가 작성되는 방식을 결정합니다. ROW DMS와의 호환성을 보장하고 CDC를 지원하려면 변수를 로 설정하십시오. `binlog_row_image` FULL 이 설정을 통해 DMS는 마이그레이션 중에 대상 데이터베이스의 전체 데이터 조작 언어 (DML) 를 구성하기에 충분한 정보를 수신할 수 있습니다.

binlog 이미지를 로 FULL 설정하려면 다음과 같이 하십시오.

- Amazon RDS의 경우 FULL 기본적으로 이 값이 사용됩니다.
- 온프레미스 또는 Amazon EC2에 호스팅되는 데이터베이스의 경우 `my.ini` (Microsoft Windows) `my.cnf` 또는 (UNIX) 에서 값을 설정합니다`binlog_row_image`.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

원본 데이터베이스가 MySQL 읽기 전용 복제본인지 확인

API 키: `mysql-check-database-role`

이 업그레이드 평가는 원본 데이터베이스가 읽기 전용 복제본인지 여부를 확인합니다. 읽기 전용 복제본에 연결된 상태에서 DMS에 대한 CDC 지원을 활성화하려면 파라미터를 로 설정합니다. `log_slave_updates` True 자체 관리형 MySQL 데이터베이스 사용에 대한 자세한 내용은 을 참조하십시오. AWS DMS용 소스로 자체 관리형 MySQL 호환 데이터베이스 사용

`log_slave_updates`값을 로 `True` 설정하려면 다음과 같이 하십시오.

- Amazon RDS의 경우 데이터베이스의 파라미터 그룹을 사용합니다. RDS 데이터베이스 파라미터 그룹 사용에 대한 자세한 내용은 Amazon RDS 사용 설명서의 파라미터 그룹 사용을 참조하십시오.
- 온프레미스 또는 Amazon EC2에 호스팅되는 데이터베이스의 경우 `my.ini` (Microsoft Windows) `my.cnf` 또는 (UNIX) 에서 값을 설정합니다`log_slave_updates`.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

테이블에 파티션이 있는지 확인하고 전체 로드 작업 설정을 권장합니다**target_table_prep_mode**.

API 키: `mysql-check-table-partition`

이 업그레이드 평가는 원본 데이터베이스에 파티션이 있는 테이블이 있는지 확인합니다. DMS는 MySQL 타겟에 파티션 없이 테이블을 생성합니다. 파티션을 나눈 테이블을 타겟의 파티션을 나눈 테이블로 마이그레이션하려면 다음 작업을 수행해야 합니다.

- 대상 MySQL 데이터베이스에서 분할된 테이블을 미리 생성합니다.
- `TRUNCATE_BEFORE_LOAD`사용하거나 `DO_NOTHING` 전체 로드 작업 설정으로 사용하도록 DMS 작업을 구성하십시오.

MySQL 엔드포인트 제한에 대한 자세한 내용은 을 참조하십시오. AWS DMS에서 MySQL 데이터베이스를 소스로 사용 시 적용되는 제한 사항

DMS가 데이터베이스 버전을 지원하는지 확인하십시오.

API 키: `mysql-check-supported-version`

이 프리미그레이션 평가는 원본 데이터베이스 버전이 DMS와 호환되는지 여부를 확인합니다. CDC는 Amazon RDS MySQL 버전 5.5 이하 또는 MySQL 8.0.x 이상의 버전에서는 지원되지 않습니다. CDC는 MySQL 버전 5.6, 5.7 또는 8.0에서만 지원됩니다. 지원되는 MySQL 버전에 대한 자세한 내용은 을 참조하십시오. 데이터 마이그레이션에 사용할 수 있는 소스 엔드포인트

대상 데이터베이스가 1로 **local_infile** 설정되도록 구성되어 있는지 확인합니다.

API 키: `mysql-check-target-localinfile-set`

이 업그레이드 평가는 대상 데이터베이스의 `local_infile` 파라미터가 1로 설정되었는지 여부를 확인합니다. DMS를 사용하려면 대상 데이터베이스가 완전히 로드되는 동안 'local_infile' 매개변수를 1로 설정해야 합니다. 자세한 정보는 AWS DMS를 사용하여 MySQL에서 MySQL로 마이그레이션.을 참조하세요.

이 평가는 전체 로드 또는 전체 로드 및 CDC 작업에만 유효합니다.

대상 데이터베이스에 외래 키가 있는 테이블이 있는지 확인합니다.

API 키: `mysql-check-fk-target`

이 업그레이드 평가는 MySQL 데이터베이스로 마이그레이션하는 전체 로드 또는 전체 및 CDC 작업에 외부 키가 있는 테이블이 있는지 확인합니다. DMS의 기본 설정은 테이블을 알파벳 순으로 로드하는

것입니다. 외래 키와 참조 무결성 제약 조건이 있는 테이블은 상위 테이블과 하위 테이블을 동시에 로드하지 못할 수 있으므로 로드가 실패할 수 있습니다.

DMS의 참조 무결성에 대한 자세한 내용은 해당 항목의 인덱스, 트리거 및 참조 무결성 제약 조건 사용을 참조하십시오. AWS DMS 마이그레이션 성능 개선

작업 범위의 원본 테이블에 계단식 제약 조건이 있는지 검증하십시오.

API 키: `mysql-check-cascade-constraints`

이 업그레이드 평가는 MySQL 소스 테이블에 캐스케이드 제약 조건이 있는지 확인합니다. MySQL은 binlog에 이러한 이벤트의 변경 사항을 기록하지 않기 때문에 계단식 제약 조건은 DMS 작업에 의해 마이그레이션되거나 복제되지 않습니다. 는 이러한 제약 조건을 AWS DMS 지원하지 않지만 관계형 데이터베이스 대상에 대한 해결 방법을 사용할 수 있습니다.

캐스케이스 제약 조건 및 기타 제약 조건 지원에 대한 자세한 내용은 마이그레이션 작업 문제 해결 항목을 참조하십시오인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음. AWS DMS

타임아웃 값이 MySQL 소스 또는 타겟에 적합한지 검증

API 키: `mysql-check-network-parameter`

이 업그레이드 평가는 작업의 MySQL 엔드포인트에 `net_wait_timeout` 및 `wait_timeout` 설정이 300초 `net_read_timeout` 이상으로 설정되어 있는지 확인합니다. 이는 마이그레이션 중에 연결이 끊기는 것을 방지하는 데 필요합니다.

자세한 정보는 작업 중에 대상 MySQL 인스턴스 연결이 끊김을 참조하세요.

MariaDB 평가

이 섹션에서는 MariaDB 소스 엔드포인트를 사용하는 마이그레이션 작업에 대한 개별 업그레이드 평가에 대해 설명합니다.

AWS DMS API를 사용하여 개별 업그레이드 평가를 생성하려면 작업 매개변수에 나열된 API 키를 사용하십시오. `Include` StartReplicationTaskAssessmentRun

주제

- 테이블이 Innodb 이외의 스토리지 엔진을 사용하는지 검증하십시오.
- 마이그레이션에 사용되는 모든 테이블에서 자동 증분이 활성화되어 있는지 확인합니다.

- 데이터베이스 binlog 형식이 DMS CDC를 ROW 지원하도록 설정되었는지 확인합니다.
- 데이터베이스 binlog 이미지가 DMS CDC를 지원하도록 FULL 설정되었는지 확인합니다.
- 원본 데이터베이스가 MariaDB 읽기 전용 복제본인지 확인
- 테이블에 파티션이 있는지 확인하고 전체 로드 작업 설정을 TRUNCATE_BEFORE_LOAD 권장하거나 DO_NOTHING 권장합니다.
- DMS가 데이터베이스 버전을 지원하는지 확인하십시오.
- 대상 데이터베이스가 1로 local_infile 설정되도록 구성되었는지 확인합니다.
- 대상 데이터베이스에 외래 키가 있는 테이블이 있는지 확인합니다.
- 작업 범위의 원본 테이블에 계단식 제약 조건이 있는지 검증하십시오.
- 작업 범위의 원본 테이블에 생성된 열이 있는지 확인합니다.
- 제한 시간 값이 MariaDB 소스에 적합한지 확인합니다.
- 제한 시간 값이 MariaDB 대상에 적합한지 확인합니다.

테이블이 Innodb 이외의 스토리지 엔진을 사용하는지 검증하십시오.

API 키: `mariadb-check-table-storage-engine`

이 업그레이드 평가는 소스 MariaDB 데이터베이스의 테이블에 사용되는 스토리지 엔진이 Innodb가 아닌 다른 엔진인지 여부를 검증합니다. DMS는 기본적으로 InnoDB 스토리지 엔진으로 대상 테이블을 생성합니다. InnoDB 이외의 스토리지 엔진을 사용해야 하는 경우 대상 데이터베이스에 테이블을 수동으로 생성하고 전체 로드 작업 설정으로 `TRUNCATE_BEFORE_LOAD` 사용하거나 `DO_NOTHING` 사용할 DMS 작업을 구성해야 합니다. 전체 로드 작업 설정에 대한 자세한 내용은 을 참조하십시오. 전체 로드 작업 설정

MariaDB 엔드포인트 제한에 대한 자세한 내용은 을 참조하십시오. AWS DMS에서 MySQL 데이터베이스를 소스로 사용 시 적용되는 제한 사항

마이그레이션에 사용되는 모든 테이블에서 자동 증분이 활성화되어 있는지 확인합니다.

API 키: `mariadb-check-auto-increment`

이 업그레이드 평가는 작업에 사용되는 원본 테이블에 자동 증분이 활성화되어 있는지 여부를 검증합니다. DMS는 열의 AUTO_INCREMENT 속성을 대상 데이터베이스로 마이그레이션하지 않습니다.

MariaDB 엔드포인트 제한에 대한 자세한 내용은 을 참조하십시오. AWS DMS에서 MySQL 데이터베이스를 소스로 사용 시 적용되는 제한 사항 MariaDB에서 ID 열을 처리하는 방법에 대한 자세한 내용은 2부의 ID 열 AWS DMS처리를 참조하십시오.

데이터베이스 binlog 형식이 DMS CDC를 **ROW** 지원하도록 설정되었는지 확인합니다.

API 키: `mariadb-check-binlog-format`

이 업그레이드 평가는 원본 데이터베이스 binlog 형식이 DMS 변경 데이터 캡처 (CDC) 를 ROW 지원하도록 설정되었는지 여부를 검증합니다.

binlog 형식을 로 설정하려면 다음과 같이 하십시오. ROW

- Amazon RDS의 경우 데이터베이스의 파라미터 그룹을 사용합니다. RDS 파라미터 그룹 사용에 대한 자세한 내용은 Amazon RDS 사용 설명서의 MySQL 바이너리 로깅 구성을 참조하십시오.
- 온프레미스 또는 Amazon EC2에 호스팅되는 데이터베이스의 경우 `my.ini` (Microsoft Windows) `my.cnf` 또는 (UNIX) 에서 값을 설정합니다`binlog_format`.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자체 호스팅 MariaDB 서버에 대한 자세한 내용은 을 참조하십시오. AWS DMS용 소스로 자체 관리형 MySQL 호환 데이터베이스 사용

데이터베이스 binlog 이미지가 DMS CDC를 지원하도록 **FULL** 설정되었는지 확인합니다.

API 키: `mariadb-check-binlog-image`

이 업그레이드 평가는 원본 데이터베이스의 binlog 이미지가 로 설정되었는지 여부를 확인합니다. FULL MariaDB에서 변수는 형식을 `binlog_row_image` 사용할 때 바이너리 로그 이벤트가 작성되는 방식을 결정합니다. ROW DMS와의 호환성을 보장하고 CDC를 지원하려면 변수를 로 설정하십시오. `binlog_row_image` FULL 이 설정을 통해 DMS는 마이그레이션 중에 대상 데이터베이스의 전체 데이터 조작 언어 (DML) 를 구성하기에 충분한 정보를 수신할 수 있습니다.

binlog 이미지를 로 FULL 설정하려면 다음과 같이 하십시오.

- Amazon RDS의 경우 FULL 기본적으로 이 값이 사용됩니다.
- 온프레미스 또는 Amazon EC2에 호스팅되는 데이터베이스의 경우 `my.ini` (Microsoft Windows) `my.cnf` 또는 (UNIX) 에서 값을 설정합니다`binlog_row_image`.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

자체 호스팅 MariaDB 서버에 대한 자세한 내용은 을 참조하십시오. AWS DMS용 소스로 자체 관리형 MySQL 호환 데이터베이스 사용

원본 데이터베이스가 MariaDB 읽기 전용 복제본인지 확인

API 키: `mariadb-check-database-role`

이 업그레이드 평가는 원본 데이터베이스가 읽기 전용 복제본인지 여부를 확인합니다. 읽기 전용 복제본에 연결된 상태에서 DMS에 대한 CDC 지원을 활성화하려면 파라미터를 로 설정합니다. `log_slave_updates` `True` 자체 관리형 MySQL 데이터베이스 사용에 대한 자세한 내용은 을 참조하십시오. AWS DMS용 소스로 자체 관리형 MySQL 호환 데이터베이스 사용

`log_slave_updates`값을 로 `True` 설정하려면 다음과 같이 하십시오.

- Amazon RDS의 경우 데이터베이스의 파라미터 그룹을 사용합니다. RDS 데이터베이스 파라미터 그룹 사용에 대한 자세한 내용은 Amazon RDS 사용 설명서의 파라미터 그룹 사용을 참조하십시오.
- 온프레미스 또는 Amazon EC2에 호스팅되는 데이터베이스의 경우 `my.ini` (Microsoft Windows) `my.cnf` 또는 (UNIX) 에서 값을 설정합니다`log_slave_updates`.

이 평가는 전체 로드 마이그레이션 및 CDC 마이그레이션 또는 CDC 전용 마이그레이션에만 유효합니다. 이 평가는 전체 로드 전용 마이그레이션에서는 유효하지 않습니다.

테이블에 파티션이 있는지 확인하고 전체 로드 작업 설정을 **`TRUNCATE_BEFORE_LOAD`** 권장하거나 **`DO_NOTHING`** 권장합니다.

API 키: `mariadb-check-table-partition`

이 업그레이드 평가는 원본 데이터베이스에 파티션이 있는 테이블이 있는지 확인합니다. DMS는 MariaDB 타겟에 파티션이 없는 테이블을 생성합니다. 파티션을 나눈 테이블을 대상의 파티션을 나눈 테이블로 마이그레이션하려면 다음을 수행해야 합니다.

- 대상 MariaDB 데이터베이스에서 파티션을 나눈 테이블을 미리 생성합니다.
- `TRUNCATE_BEFORE_LOAD`사용하거나 `DO_NOTHING` 전체 로드 작업 설정으로 사용하도록 DMS 작업을 구성하십시오.

MariaDB 엔드포인트 제한에 대한 자세한 내용은 을 참조하십시오. AWS DMS에서 MySQL 데이터베이스를 소스로 사용 시 적용되는 제한 사항

DMS가 데이터베이스 버전을 지원하는지 확인하십시오.

API 키: `mariadb-check-supported-version`

이 프리미그레이션 평가는 원본 데이터베이스 버전이 DMS와 호환되는지 여부를 확인합니다. CDC는 Amazon RDS MariaDB 버전 10.4 이하 또는 MySQL 버전 10.11보다 큰 버전에서는 지원되지 않습니다. 지원되는 MariaDB 버전에 대한 자세한 내용은 을 참조하십시오. 데이터 마이그레이션에 사용할 수 있는 소스 엔드포인트

대상 데이터베이스가 1로 **`local_infile`** 설정되도록 구성되었는지 확인합니다.

API 키: `mariadb-check-target-localinfile-set`

이 업그레이드 평가는 대상 데이터베이스의 `local_infile` 파라미터가 1로 설정되었는지 여부를 확인합니다. DMS를 사용하려면 대상 데이터베이스가 완전히 로드되는 동안 'local_infile' 매개변수를 1로 설정해야 합니다. 자세한 정보는 AWS DMS를 사용하여 MySQL에서 MySQL로 마이그레이션.을 참조하세요.

이 평가는 전체 로드 작업에만 유효합니다.

대상 데이터베이스에 외래 키가 있는 테이블이 있는지 확인합니다.

API 키: `mariadb-check-fk-target`

이 업그레이드 평가는 MariaDB 데이터베이스로 마이그레이션하는 전체 로드 또는 전체 및 CDC 작업에 외래 키가 있는 테이블이 있는지 확인합니다. DMS의 기본 설정은 테이블을 알파벳 순으로 로드하는 것입니다. 외래 키와 참조 무결성 제약 조건이 있는 테이블은 상위 테이블과 하위 테이블을 동시에 로드하지 못할 수 있으므로 로드가 실패할 수 있습니다.

DMS의 참조 무결성에 대한 자세한 내용은 해당 항목의 인덱스, 트리거 및 참조 무결성 제약 조건 사용을 참조하십시오. AWS DMS 마이그레이션 성능 개선

작업 범위의 원본 테이블에 계단식 제약 조건이 있는지 검증하십시오.

API 키: `mariadb-check-cascade-constraints`

이 업그레이드 평가는 MariaDB 소스 테이블에 캐스케이드 제약 조건이 있는지 확인합니다. MariaDB는 binlog에 이러한 이벤트의 변경 사항을 기록하지 않기 때문에 계단식 제약 조건은 DMS 작업에 의해 마이그레이션되거나 복제되지 않습니다. 는 이러한 제약 조건을 AWS DMS 지원하지 않지만 관계형 데이터베이스 대상에는 해결 방법을 사용할 수 있습니다.

캐스케이스 제약 조건 및 기타 제약 조건 지원에 대한 자세한 내용은 마이그레이션 작업 문제 해결 항목을 참조하십시오인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음. AWS DMS

작업 범위의 원본 테이블에 생성된 열이 있는지 확인합니다.

API 키: `mariadb-check-generated-columns`

이 업그레이드 평가는 MariaDB 소스 테이블에 생성된 열이 있는지 여부를 확인합니다. DMS 작업은 생성된 열을 마이그레이션하거나 복제하지 않습니다.

생성된 열을 마이그레이션하는 방법에 대한 자세한 내용은 을 참조하십시오. ???

제한 시간 값이 MariaDB 소스에 적합한지 확인합니다.

API 키: `mariadb-check-source-network-parameter`

이 업그레이드 평가는 작업의 MariaDB 소스 엔드포인트에 `net_wait_timeout` 및 `wait_timeout` 설정이 300초 이상으로 `net_read_timeout` 설정되어 있는지 확인합니다. 이는 마이그레이션 중에 연결이 끊기는 것을 방지하는 데 필요합니다.

자세한 정보는 작업 중에 대상 MySQL 인스턴스 연결이 끊김을 참조하세요.

제한 시간 값이 MariaDB 대상에 적합한지 확인합니다.

API 키: `mariadb-check-target-network-parameter`

이 업그레이드 평가는 작업의 MariaDB 대상 엔드포인트에 `net_wait_timeout` 및 `wait_timeout` 설정이 300초 이상으로 `net_read_timeout` 설정되어 있는지 확인합니다. 이는 마이그레이션 중에 연결이 끊기는 것을 방지하는 데 필요합니다.

자세한 정보는 작업 중에 대상 MySQL 인스턴스 연결이 끊김을 참조하세요.

PostgreSQL 평가

이 섹션에서는 PostgreSQL 소스 엔드포인트를 사용하는 마이그레이션 작업에 대한 개별 업그레이드 평가에 대해 설명합니다.

주제

- DMS에서 마이그레이션을 위해 소스 데이터베이스 버전을 지원하는지 확인하십시오.
- 원본 데이터베이스에서 logical_decoding_work_mem 파라미터를 검증합니다.
- 소스 데이터베이스에 장기 실행 트랜잭션이 있는지 확인합니다.

- 원본 데이터베이스 매개 변수의 유효성을 검사합니다. max_slot_wal_keep_size
- 원본 데이터베이스 매개변수가 CDC를 지원하도록 postgres-check-maxwalsenders 설정되어 있는지 확인하십시오.
- 원본 데이터베이스가 다음과 같이 구성되어 있는지 확인하십시오. PGLOGICAL
- 소스 테이블 기본 키가 LOB 데이터 유형인지 확인하십시오.
- 원본 테이블에 기본 키가 있는지 확인합니다.
- 준비된 트랜잭션이 원본 데이터베이스에 있는지 확인합니다.
- DMS CDC를 지원하는 데 필요한 최소 값으로 wal_sender_timeout 설정되어 있는지 확인하십시오.
- 원본 데이터베이스에서 wal_level 논리로 설정되어 있는지 확인합니다.

DMS에서 마이그레이션을 위해 소스 데이터베이스 버전을 지원하는지 확인하십시오.

API 키: `postgres-check-dbversion`

이 업그레이드 평가는 원본 데이터베이스 버전이 호환되는지 여부를 확인합니다. AWS DMS

원본 데이터베이스에서 **`logical_decoding_work_mem`** 파라미터를 검증합니다.

API 키: `postgres-check-for-logical-decoding-work-mem`

이 업그레이드 평가에서는 원본 데이터베이스의 `logical_decoding_work_mem` 파라미터를 조정할 것을 권장합니다. 트랜잭션이 오래 실행되거나 하위 트랜잭션이 많을 수 있는 트랜잭션이 많은 데이터베이스에서는 논리적 디코딩 메모리 사용량이 증가하고 디스크로 넘겨야 할 필요성이 커질 수 있습니다. 이로 인해 복제 중에 DMS 소스 지연 시간이 길어집니다. 이러한 시나리오에서는 `logical_decoding_work_mem` 조정이 필요할 수 있습니다. 이 파라미터는 PostgreSQL 버전 13 이상에서 지원됩니다.

소스 데이터베이스에 장기 실행 트랜잭션이 있는지 확인합니다.

API 키: `postgres-check-longrunningtxn`

이 프리미그레이션 평가는 원본 데이터베이스에 10분 이상 지속된 장기 실행 트랜잭션이 있는지 여부를 확인합니다. 기본적으로 DMS는 작업을 시작하는 동안 열려 있는 트랜잭션을 모두 확인하기 때문에 작업 시작이 실패할 수 있습니다.

원본 데이터베이스 매개 변수의 유효성을 검사합니다. **`max_slot_wal_keep_size`**

API 키: `postgres-check-maxslot-wal-keep-size`

이 업그레이드 평가는 구성된 값을 검증합니다. max_slot_wal_keep_size 를 기본값
이 아닌 값으로 설정하면 필수 WAL 파일이 제거되어 DMS 작업이 실패할 수 있습니다.
max_slot_wal_keep_size

원본 데이터베이스 매개변수가 CDC를 지원하도록 **postgres-check-maxwalsenders** 설정되어 있는지 확인하십시오.

API 키: postgres-check-maxwalsenders

이 업그레이드 평가는 원본 데이터베이스에 구성된 값을 확인합니다max_wal_senders. 변경 데이터 캡처 (CDC) 를 max_wal_senders 지원하려면 DMS를 1보다 크게 설정해야 합니다.

원본 데이터베이스가 다음과 같이 구성되어 있는지 확인하십시오. **PGLOGICAL**

API 키: postgres-check-pglogical

이 프리그레이션 평가를 통해 shared_preload_libraries 값이 CDC pglogical PGLOGICAL 지원으로 설정되었는지 확인합니다. 논리적 복제에 테스트 디코딩을 사용할 계획이라면 이 평가를 무시해도 된다는 점에 유의하세요.

소스 테이블 기본 키가 LOB 데이터 유형인지 확인하십시오.

API 키: postgres-check-pk-lob

이 업그레이드 평가는 테이블의 기본 키가 Large Object (Large Object) 데이터 유형인지 확인합니다. 원본 테이블에 LOB 열이 기본 키로 있는 경우 DMS는 복제를 지원하지 않습니다.

원본 테이블에 기본 키가 있는지 확인합니다.

API 키: postgres-check-pk

이 프리미어그레이션 평가는 작업 범위에서 사용되는 테이블에 기본 키가 존재하는지 확인합니다. 원본 테이블에 복제본 ID가 로 설정되어 있지 않는 한 DMS는 기본 키가 없는 테이블에 대한 복제를 지원하지 않습니다. full

준비된 트랜잭션이 원본 데이터베이스에 있는지 확인합니다.

API 키: postgres-check-preparedtxn

이 프리미어그레이션 평가는 소스 데이터베이스에 준비된 트랜잭션이 있는지 확인합니다. 원본 데이터베이스에 준비된 트랜잭션이 있는 경우 복제 슬롯 생성이 응답하지 않을 수 있습니다.

DMS CDC를 지원하는 데 필요한 최소 값으로 **wal_sender_timeout** 설정되어 있는지 확인하십시오.

API 키: postgres-check-walsenderstimeout

이 프리미어그레이션 평가는 최소 10000밀리초 (10초) 로 wal_sender_timeout 설정되어 있는지 확인합니다. CDC를 사용한 DMS 작업에는 최소 10,000밀리초 (10초) 가 필요하며 값이 10000보다 작으면 실패합니다.

원본 데이터베이스에서 **wal_level** 논리로 설정되어 있는지 확인합니다.

API 키: postgres-check-wallevel

이 업그레이드 평가는 논리적 수준으로 wal_level 설정되어 있는지 확인합니다. DMS CDC가 작동하려면 원본 데이터베이스에서 이 매개변수를 활성화해야 합니다.

데이터 유형 평가 시작 및 보기 (레거시)

> **Note**
>
> 이 섹션에서는 레거시 콘텐츠에 대해 설명합니다. 앞서 설명한 대로 프리미어레이션 평가를 사용하는 것이 좋습니다. 마이그레이션 전 평가 실행 지정, 시작, 보기
> 콘솔에서는 데이터 유형 평가를 사용할 수 없습니다. API 또는 CLI를 사용해서만 데이터 유형 평가를 실행할 수 있으며, 작업의 S3 버킷에서만 데이터 유형 평가 결과를 볼 수 있습니다.

데이터 유형 평가는 대상이 지원하지 않아 제대로 마이그레이션되지 않을 수 있는 원본 데이터베이스의 데이터 유형을 식별합니다. 이 평가 중에 마이그레이션 작업에 사용할 원본 데이터베이스 스키마를 AWS DMS 읽고 열 데이터 유형 목록을 만듭니다. 그런 다음 이 목록을 에서 지원하는 사전 정의된 데이터 유형 목록과 비교합니다. AWS DMS마이그레이션 작업에 지원되지 않는 데이터 유형이 있는 경우 마이그레이션 작업에 지원되지 않는 데이터 유형이 있는지 확인할 수 있는 보고서를 AWS DMS 생성합니다. AWS DMS 마이그레이션 작업에 지원되지 않는 데이터 유형이 없는 경우에는 보고서를 만들지 않습니다.

AWS DMS 다음과 같은 관계형 데이터베이스에 대한 데이터 유형 평가 보고서 생성을 지원합니다.

- Oracle
- SQL Server
- PostgreSQL
- MySQL

- MariaDB
- Amazon Aurora

CLI와 SDK를 사용하여 데이터 유형 평가 보고서를 시작하고 보고 API에 액세스할 수 있습니다. AWS DMS

- CLI는 start-replication-task-assessment 명령을 사용하여 데이터 형식 평가를 시작하고 describe-replication-task-assessment-results 명령을 사용하여 JSON 형식의 최신 데이터 형식 평가 보고서를 확인합니다.
- AWS DMS API는 StartReplicationTaskAssessment작업을 사용하여 데이터 유형 평가를 시작하고 이 DescribeReplicationTaskAssessmentResults작업을 사용하여 JSON 형식의 최신 데이터 유형 평가 보고서를 확인합니다.

데이터 형식 평가 보고서는 지원되지 않는 데이터 형식과 각 데이터 형식에 대한 열 수를 나열한 요약이 포함된 단일 JSON 파일입니다. 여기에는 지원되지 않는 데이터 형식을 가진 스키마, 테이블 및 열을 포함하여 지원되지 않는 각 데이터 형식에 대한 데이터 구조 목록이 포함됩니다. 보고서를 사용하여 소스 데이터 형식을 수정하고 마이그레이션을 개선하여 성공적으로 수행될 수 있도록 합니다.

지원되지 않는 데이터 형식에는 두 가지 수준이 있습니다. 보고서에 지원되지 않는다고 표시되는 데이터 형식은 마이그레이션할 수 없습니다. 보고서에 부분적으로 지원된다고 표시되는 데이터 형식은 다른 데이터 형식으로 변환되어 예상대로 마이그레이션되지 않을 수 있습니다.

다음 예제에서는 볼 수 있는 샘플 데이터 형식 평가 보고서를 보여 줍니다.

```
{
    "summary":{
        "task-name":"test15",
        "not-supported":{
            "data-type": [
                "sql-variant"
            ],
            "column-count":3
        },
        "partially-supported":{
            "data-type":[
                "float8",
                "jsonb"
            ],
            "column-count":2
```

```
        }
    },
    "types":[
        {
            "data-type":"float8",
            "support-level":"partially-supported",
            "schemas":[
                {
                    "schema-name":"schema1",
                    "tables":[
                        {
                            "table-name":"table1",
                            "columns":[
                                "column1",
                                "column2"
                            ]
                        },
                        {
                            "table-name":"table2",
                            "columns":[
                                "column3",
                                "column4"
                            ]
                        }
                    ]
                },
                {
                    "schema-name":"schema2",
                    "tables":[
                        {
                            "table-name":"table3",
                            "columns":[
                                "column5",
                                "column6"
                            ]
                        },
                        {
                            "table-name":"table4",
                            "columns":[
                                "column7",
                                "column8"
                            ]
                        }
                    ]
```

```
                }
            ]
        },
        {
            "datatype":"int8",
            "support-level":"partially-supported",
            "schemas":[
                {
                    "schema-name":"schema1",
                    "tables":[
                        {
                            "table-name":"table1",
                            "columns":[
                                "column9",
                                "column10"
                            ]
                        },
                        {
                            "table-name":"table2",
                            "columns":[
                                "column11",
                                "column12"
                            ]
                        }
                    ]
                }
            ]
        }
    ]
}
```

AWS DMS AWS DMS 계정에서 생성한 Amazon S3 버킷에 최신 및 모든 이전 데이터 유형 평가를 저장합니다. Amazon S3 버킷 이름의 형식은 다음과 같습니다. 여기서 *customerId*는 고객 ID이고 *customerDNS*는 내부 식별자입니다.

```
dms-customerId-customerDNS
```

> **Note**
>
> 기본적으로 AWS 계정 각각에서 최대 100개의 Amazon S3 버킷을 만들 수 있습니다. 계정에 버킷을 AWS DMS 생성하므로 버킷 한도를 초과하지 않도록 해야 합니다. 그렇지 않으면 데이터 형식 평가가 실패합니다.

지정된 마이그레이션 작업에 대한 모든 데이터 형식 평가 보고서는 작업 식별자로 이름이 지정된 버킷 폴더에 저장됩니다. 각 보고서의 파일 이름은 yyyy-mm-dd-hh -mm 형식의 데이터 유형 평가 날짜입니다. Amazon S3 관리 콘솔에서 이전 데이터 형식 평가 보고서를 조회하고 비교할 수 있습니다.

AWS DMS 또한 이러한 보고서용으로 생성된 S3 버킷에 대한 액세스를 허용하는 AWS Identity and Access Management (IAM) 역할을 생성합니다. 역할 이름은 `dms-access-for-tasks`입니다. 이 역할에는 `AmazonDMSRedshiftS3Role` 정책이 사용됩니다. 실행할 `StartReplicationTaskAssessment` 때 ResourceNotFoundFault오류가 발생하는 경우 수동으로 `dms-access-for-tasks` 역할을 생성하는 방법에 대한 자세한 내용은 문제 해결 섹션을 참조하십시오ResourceNotFoundFault.

평가 실행 문제 해결

다음에서 평가 보고서를 실행할 때 발생하는 문제 해결에 대한 항목을 찾을 수 AWS Database Migration Service있습니다. 이러한 항목은 일반적인 문제를 해결하는 데 도움이 될 수 있습니다.

주제

- ResourceNotFoundFault 실행 시 StartReplicationTaskAssessment

ResourceNotFoundFault 실행 시 StartReplicationTaskAssessment

StartReplicationTaskAssessment액션을 실행할 때 다음과 같은 예외가 발생할 수 있습니다.

```
An error occurred (ResourceNotFoundFault) when calling the
 StartReplicationTaskAssessment operation: Task assessment has not been run or dms-
access-for-tasks IAM Role not configured correctly
```

이 예외가 발생하는 경우 다음을 수행하여 dms-access-for-tasks역할을 생성하십시오.

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.

3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 선택 페이지에서 신뢰할 수 있는 엔터티 유형으로 사용자 지정 신뢰 정책을 선택합니다.
5. 기존 텍스트를 대체하여 편집기에 다음 JSON을 붙여넣습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
                "Service": "dms.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

위 정책은 에 `sts:AssumeRole` 권한을 부여합니다. AWS DMSAmazonDMSRedshifts3Role Amazondshifts3role 정책을 추가하면 DMS가 사용자 계정에 S3 버킷을 생성하고 데이터 유형 평가 결과를 이 S3 버킷에 넣을 수 있습니다.

6. 다음을 선택합니다.
7. 권한 추가 페이지에서 Amazondmsredshifts3role 정책을 검색하여 추가합니다. 다음을 선택합니다.
8. 이름, 검토 및 생성 페이지에서 역할 이름을 지정합니다. dms-access-for-tasks 역할 생성을 선택합니다.

작업 설정에 대한 보충 데이터 지정

일부 AWS DMS 엔드포인트에 대한 복제 작업을 만들거나 수정할 때 마이그레이션을 수행하기 위해 작업에 추가 정보가 필요할 수 있습니다. DMS 콘솔의 옵션을 사용하여 이 추가 정보를 지정할 수 있습니다. 또는 DMS API 작업 `CreateReplicationTask` 또는 `ModifyReplicationTask`에 대한 `TaskData` 파라미터를 사용하여 지정할 수 있습니다.

대상 엔드포인트가 Amazon Neptune인 경우, 테이블 매핑을 보완하는 매핑 데이터를 지정해야 합니다. 이 보충 매핑 데이터는 소스 관계형 데이터를 Neptune 데이터베이스가 사용할 수 있는 대상 그래프 데이터로 변환하는 방법을 지정합니다. 이 경우 두 가지 형식 중 하나를 사용할 수 있습니다. 자세한 내

용은 Amazon Neptune에 대한 Gremlin 및 R2RML을 대상으로 사용하여 그래프 매핑 규칙 지정 섹션을 참조하세요.

AWS DMS 태스크 모니터링

모니터링은 AWS DMS와 사용자 AWS 솔루션의 안정성, 가용성 및 성능을 유지하는 중요한 역할을 합니다. 다중 지점 실패가 발생할 경우 보다 쉽게 디버깅할 수 있도록 AWS 솔루션의 모든 부분으로부터 모니터링 데이터를 수집해야 합니다. AWS는 AWS DMS 작업 및 리소스를 모니터링하고 잠재적 인시던트에 대응하기 위한 여러 도구를 제공합니다.

AWS DMS 이벤트 및 알림

AWS DMS는 Amazon Simple Notification Service(Amazon SNS)를 사용하여 AWS DMS 이벤트(예: 복제 인스턴스 생성 또는 삭제)가 발생할 때 알림을 제공합니다. AWS DMS는 구독할 수 있는 범주로 이벤트를 그룹화하므로 사용자는 해당 범주의 이벤트가 발생할 때 알림을 받을 수 있습니다. 예를 들어 임의의 복제 인스턴스에 대한 생성 카테고리를 구독할 경우 생성 관련 이벤트가 발생하여 복제 인스턴스에 영향을 끼칠 때마다 알림 메시지가 수신됩니다. AWS 리전에 따라 Amazon SNS에서 지원하는 형식으로 이메일 메시지, 문자 또는 HTTP 엔드포인트 직접 호출 등의 알림을 사용할 수 있습니다. 자세한 정보는 AWS Database Migration Service에서 Amazon SNS 이벤트 및 알림 사용 섹션을 참조하세요.

작업 상태

작업 상태를 확인하고 작업의 제어 테이블을 확인하여 작업 진행률을 모니터링할 수 있습니다. 작업 상태는 AWS DMS 작업 및 연결된 리소스의 상태를 나타냅니다. 여기에는 작업이 생성, 시작, 실행 또는 중지되는 경우와 같은 표시가 포함됩니다. 또한 테이블 전체 로드가 시작되었거나 진행 중인 경우와 같이 작업이 마이그레이션하는 테이블의 현재 상태와 테이블에서 발생한 삽입, 삭제 및 업데이트 횟수와 같은 세부 정보가 포함됩니다. 작업 및 작업 리소스 상태 모니터링에 대한 자세한 내용은 작업 상태 및 작업 중 테이블 상태 섹션을 참조하세요. 제어 테이블에 대한 자세한 내용은 제어 테이블 작업 설정 섹션을 참조하세요.

Amazon CloudWatch 경보 및 로그

Amazon CloudWatch 경보를 사용하여 지정한 기간 동안 하나 이상의 작업 지표를 감시합니다. 지표가 지정된 임계값을 초과하면 Amazon SNS 주제로 알림이 전송됩니다. CloudWatch 경보는 단순히 특정 상태에 있다고 해서 작업을 호출하지 않습니다. 상태가 변경되어 지정된 기간 동안 유지되어야 합니다. 또한 AWS DMS는 마이그레이션 프로세스 도중 CloudWatch를 사용하여 태스크 정보를 기록합니다. AWS CLI 또는 AWS DMS API를 사용하여 작업 로그에 대한 정보를 볼 수 있습니다. CloudWatch를 AWS DMS와 함께 사용하는 자세한 방법은 Amazon CloudWatch를 사용한 복제 태스크 모니터링 섹션을 참조하세요. AWS DMS 지표 모니터링에 대한 자세한 내용은 AWS Database Migration Service 지표 섹션을 참조하세요. AWS DMS 작업 로그 사용에 대한 자세한 내용은 AWS DMS 태스크 로그 보기 및 관리 섹션을 참조하세요.

Time Travel 로그

AWS DMS Time Travel을 사용하여 복제 태스크를 기록하고 디버깅할 수 있습니다. 이 방식에서는 Amazon S3를 사용하여 로그를 저장하고 암호화 키를 사용하여 로그를 암호화합니다. 날짜-시간 필터를 사용하여 S3 로그를 검색한 다음, 필요에 따라 로그를 보고 다운로드하고 난독화할 수 있습니다. 이렇게 하면 '과거로 이동'하여 데이터베이스 활동을 조사할 수 있습니다.

Time Travel은 DMS가 지원하는 PostgreSQL 소스 엔드포인트와 DMS가 지원하는 PostgreSQL 및 MySQL 대상 엔드포인트와 함께 사용할 수 있습니다. Time Travel은 전체 로드 및 CDC 태스크 및 CDC 전용 태스크에만 사용할 수 있습니다. Time Travel을 활성화하거나 기존의 Time Travel 설정을 수정하려면 태스크를 중지해야 합니다.

Time Travel 로그에 대한 자세한 내용은 Time Travel 작업 설정 섹션을 참조하세요. Time Travel 로그 사용에 대한 모범 사례는 Time Travel을 통한 복제 태스크 문제 해결 섹션을 참조하세요.

AWS CloudTrail 로그

AWS DMS는 AWS DMS에서 사용자, IAM 역할 또는 AWS 서비스가 수행한 작업에 대한 레코드를 제공하는 서비스인 AWS CloudTrail와 통합됩니다. CloudTrail은 AWS DMS 콘솔의 호출, AWS DMS API 작업에 대한 코드 호출을 포함하여 AWS DMS에 대한 모든 API 직접 호출을 이벤트로 캡처합니다. 추적을 생성하면 AWS DMS 이벤트를 포함한 CloudTrail 이벤트를 지속적으로 Amazon S3 버킷에 배포할 수 있습니다. 추적을 구성하지 않은 경우에도 CloudTrail 콘솔의 이벤트 기록에서 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집한 정보를 사용하여 AWS DMS에 수행된 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다. 자세한 내용은 AWS CloudTrail을 사용하여 AWS DMS API 호출 로깅 섹션을 참조하세요.

데이터베이스 로그

AWS 데이터베이스 서비스에 대해 AWS Management Console, AWS CLI 또는 API를 사용하여 작업 엔드포인트의 데이터베이스 로그를 보고, 다운로드하고, 감시할 수 있습니다. 자세한 내용은 AWS 설명서에서 해당 데이터베이스 서비스 설명서를 참조하세요.

자세한 내용은 다음 항목을 참조하세요.

주제

- 작업 상태
- 작업 중 테이블 상태
- Amazon CloudWatch를 사용한 복제 태스크 모니터링
- AWS Database Migration Service 지표

- AWS DMS 태스크 로그 보기 및 관리
- AWS CloudTrail을 사용하여 AWS DMS API 호출 로깅
- AWS DMS 컨텍스트 로깅

작업 상태

태스크 상태는 말 그대로 태스크의 상태를 나타냅니다. 다음 테이블은 작업이 가질 수 있는 가능한 상태를 나타냅니다.

작업 상태	설명
[생성 중]	AWS DMS가 태스크를 생성하는 중입니다.
[실행 중]	작업이 지정된 마이그레이션 업무를 수행하고 있습니다.
[중지됨]	작업이 중지됩니다.
[중지 중]	작업이 중지되고 있습니다. 대개 작업에서 사용자 개입을 나타냅니다.
[삭제 중]	보통 사용자 개입 요청을 통해 작업이 삭제되고 있습니다.
실패함	작업에 실패했습니다. 자세한 내용은 태스크 로그 파일을 참조하세요.
오류	오류로 인해 태스크가 중지되었습니다. 태스크 오류에 대한 간략한 설명은 개요 탭의 마지막 실패 메시지 섹션에 나와 있습니다.
실행 중(오류 포함)	태스크가 오류 상태로 실행 중입니다. 이는 일반적으로 태스크에 있는 하나 이상의 테이블을 마이그레이션할 수 없음을 나타냅니다. 태스크는 선택 규칙에 따라 다른 테이블을 계속 로드합니다.
[시작됨]	작업이 복제 인스턴스와 원본 및 대상 엔드포인트에 연결되고 있습니다. 모든 필터와 변환이 적용되고 있습니다.

작업 상태	설명
[준비됨]	작업을 실행할 준비가 되어 있습니다. 이 상태 다음에는 대개 "생성" 상태가 나옵니다.
수정 중	작업이 수정되고 있습니다. 대개 작업 설정을 수정한 사용자 작업으로 인한 것입니다.
이동 중	태스크가 다른 복제 인스턴스로 이동하는 중입니다. 이동이 완료될 때까지 복제는 이 상태로 유지됩니다. 복제 태스크가 이동 중인 동안에는 태스크 삭제 작업만 허용됩니다.
이동 실패	어떤 이유로든 태스크 이동이 실패했습니다(예: 대상 복제 인스턴스에 스토리지 공간이 부족). 복제 태스크가 이 상태일 때는 태스크를 시작, 수정, 이동 또는 삭제할 수 있습니다.
테스트	이 태스크에 지정된 데이터베이스 마이그레이션이 StartReplicationTaskAssessmentRun 또는 StartReplicationTaskAssessment 작업 실행에 대한 응답으로 테스트되고 있습니다.

작업 상태 표시줄은 작업 진행률의 추정치를 나타냅니다. 이 추정치의 품질은 소스 데이터베이스의 테이블 통계 품질에 따라 달라지며, 테이블 통계 품질이 좋을수록 추정 정확도가 높아집니다. 추정 행 통계 없이 단 하나의 테이블을 사용하는 작업의 경우, 모든 유형의 완료율 추정치를 제공할 수 없습니다. 이 경우에 작업 상태와 로드되는 행 표시를 사용하여 작업이 실제로 실행되고 진행되고 있음을 확인할 수 있습니다.

DMS 콘솔은 '마지막 업데이트' 열에 AWS DMS가 테이블의 테이블 통계 레코드를 마지막으로 업데이트한 시간만 표시합니다. 테이블에 대한 마지막 업데이트 시간을 표시하지 않습니다.

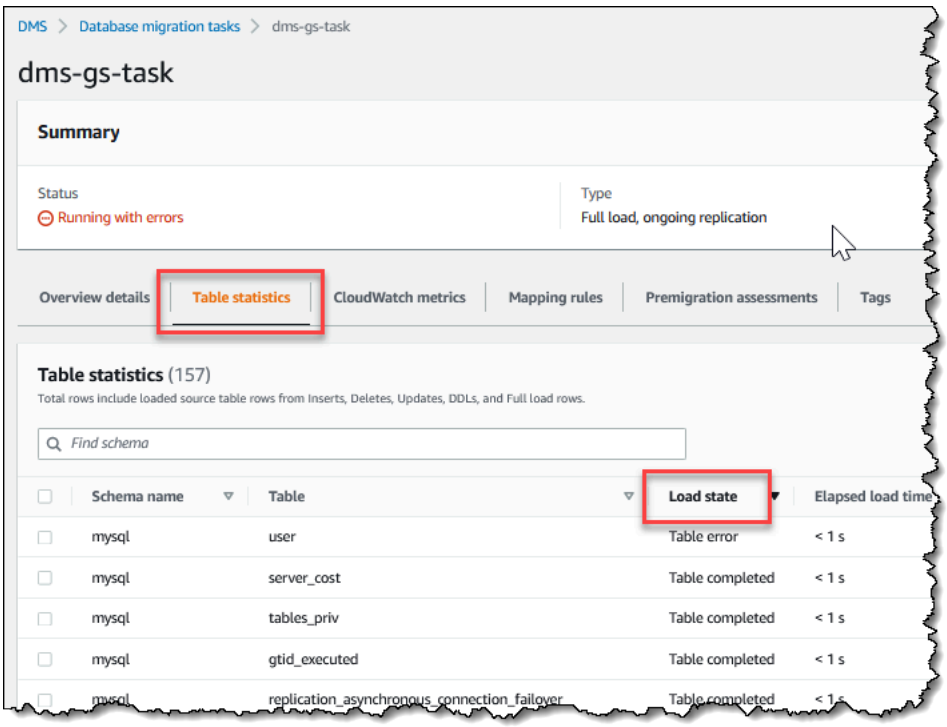
DMS 콘솔을 사용하는 것 외에도 다음 예와 같이 AWS CLI의 `aws dms describe-replication-tasks` 명령을 사용하여 태스크 상태를 비롯한 현재 복제 태스크에 대한 설명을 출력할 수 있습니다.

```
{
    "ReplicationTasks": [
        {
            "ReplicationTaskIdentifier": "moveit2",
```

```
            "SourceEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:6GGI6YPWWGAYUVLKIB732KEVWA",
            "TargetEndpointArn": "arn:aws:dms:us-
east-1:123456789012:endpoint:EOM4SFKCZEYHZBFGAGZT3QEC5U",
            "ReplicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:T3OM7OUB5NM2LCVZF7JPGJRNUE",
            "MigrationType": "full-load",
            "TableMappings": ...output omitted... ,
            "ReplicationTaskSettings": ...output omitted... ,
            "Status": "stopped",
            "StopReason": "Stop Reason FULL_LOAD_ONLY_FINISHED",
            "ReplicationTaskCreationDate": 1590524772.505,
            "ReplicationTaskStartDate": 1590619805.212,
            "ReplicationTaskArn": "arn:aws:dms:us-
east-1:123456789012:task:K55IUCGBASJS5VHZJIINA45FII",
            "ReplicationTaskStats": {
                "FullLoadProgressPercent": 100,
                "ElapsedTimeMillis": 0,
                "TablesLoaded": 0,
                "TablesLoading": 0,
                "TablesQueued": 0,
                "TablesErrored": 0,
                "FreshStartDate": 1590619811.528,
                "StartDate": 1590619811.528,
                "StopDate": 1590619842.068
            }
        }
    ]
}
```

작업 중 테이블 상태

AWS DMS 콘솔은 마이그레이션 중에 테이블 상태에 대한 정보를 업데이트합니다. 다음 표에는 가능한 상태 값이 나와 있습니다.



상태	설명
테이블 없음	AWS DMS가 소스 엔드포인트에서 테이블을 찾을 수 없습니다.
로드 전	전체 로드 프로세스가 활성화되었지만, 아직 시작되지 않았습니다.
전체 로드	전체 로드 프로세스가 진행되고 있습니다.
테이블 완료	전체 로드가 완료되었습니다.
테이블 취소	테이블 로드가 취소되었습니다.
테이블 오류	테이블을 로드하는 동안 오류가 발생했습니다.

Amazon CloudWatch를 사용한 복제 태스크 모니터링

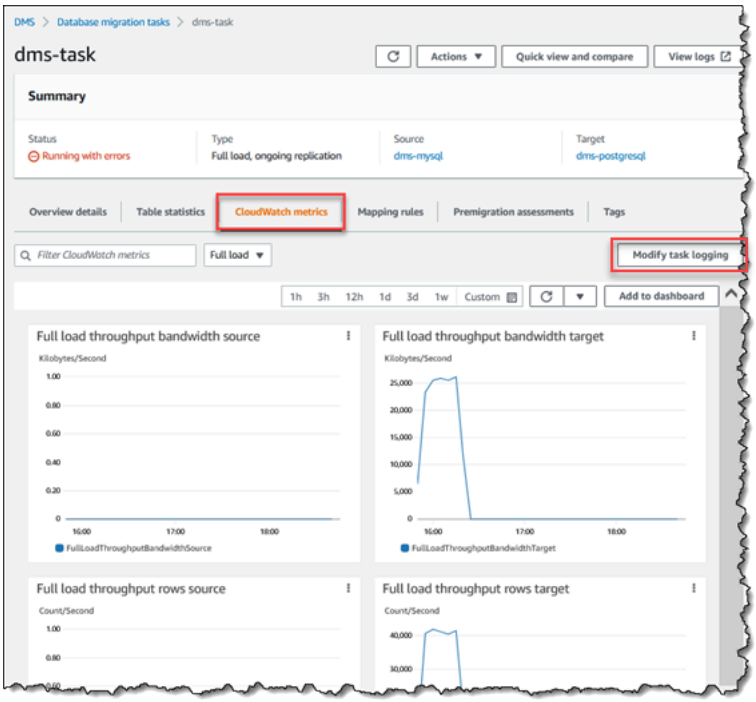
Amazon CloudWatch 경보 또는 이벤트를 사용하여 마이그레이션을 더욱 면밀하게 추적할 수 있습니다. Amazon CloudWatch에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon

CloudWatch, Amazon CloudWatch Events 및 Amazon CloudWatch Logs란 무엇입니까?를 참조하세요. Amazon CloudWatch 사용 시 비용이 청구됩니다.

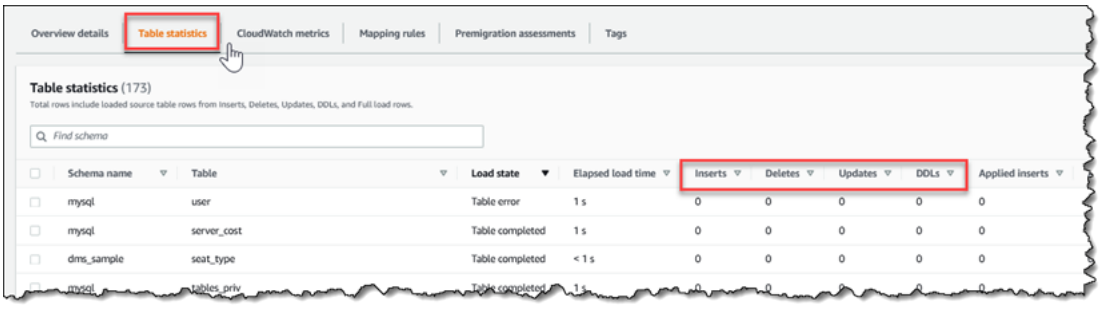
복제 태스크에서 CloudWatch 로그가 생성되지 않는 경우 문제 해결 안내서에서 AWS DMS CloudWatch 로그를 생성하지 않습니다. 섹션을 참조하세요.

AWS DMS 콘솔에는 다음 그림과 같이 작업 상태, 완료율, 경과 시간 및 테이블 통계 등 각 작업별로 기본 CloudWatch 통계가 표시됩니다. 복제 태스크를 선택한 다음 CloudWatch 지표 탭을 선택합니다.

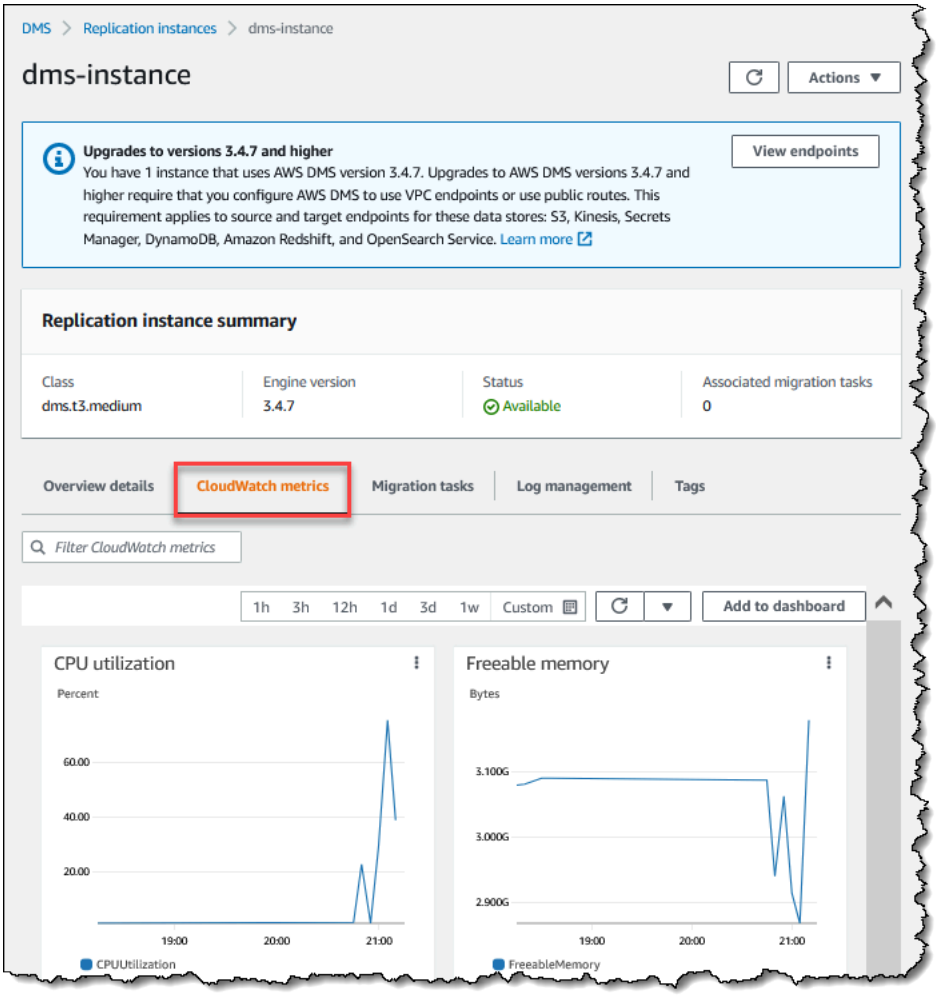
CloudWatch 태스크 로그 설정을 보고 수정하려면 태스크 로깅 수정을 선택합니다. 자세한 내용은 작업 설정 로깅 섹션을 참조하세요.



AWS DMS 콘솔에서 테이블 통계 탭을 선택하면 삽입, 삭제, 업데이트 개수 등 각 테이블별로 성능 통계가 표시됩니다.



또한, 복제 인스턴스 페이지에서 복제 인스턴스를 선택하는 경우 CloudWatch 지표 탭을 선택하여 인스턴스의 성능 지표를 확인할 수 있습니다.



AWS Database Migration Service 지표

AWS DMS는 다음 항목의 통계를 제공합니다.

- 호스트 지표 - Amazon CloudWatch에서 제공하는 복제 호스트에 대한 성능 및 사용률 통계. 사용 가능한 전체 측정치 목록은 복제 인스턴스 지표 섹션을 참조하세요.
- 복제 태스크 지표 - 들어오는 변경 사항과 커밋된 변경 사항, 복제 호스트와 소스 및 대상 데이터베이스 사이의 지연 시간을 포함한 복제 태스크 통계. 사용 가능한 전체 측정치 목록은 복제 작업 지표 섹션을 참조하세요.
- 테이블 지표 - 완료된 삽입, 업데이트, 삭제 및 DDL 문 개수를 포함하여 마이그레이션 중인 테이블에 대한 통계.

작업 지표는 복제 호스트와 원본 엔드포인트 간의 통계, 그리고 복제 호스트와 대상 엔드포인트 간의 통계로 나뉩니다. 관련 통계 2개를 모두 추가하여 작업의 총 통계를 확인할 수 있습니다. 예를 들어,

[CDCLatencySource]와 [CDCLatencyTarget] 값을 결합하여 작업의 총 지연 시간 또는 복제 지연을 확인할 수 있습니다.

작업 지표 값은 원본 데이터베이스에서 현재 작업의 영향을 받을 수 있습니다. 예를 들어, 트랜잭션이 시작되었지만 커밋되지 않은 경우, [CDCLatencySource] 지표는 트랜잭션이 커밋될 때까지 계속 증가합니다.

복제 인스턴스의 경우, [FreeableMemory] 측정치에 대한 명확한 이해가 필요합니다. 여유 메모리는 사용 가능한 실제 여유 메모리를 나타내지 않습니다. 해제하여 다른 용도로 사용할 수 있는 현재 사용 중인 메모리로, 여유 복제 인스턴스에 사용 중인 버퍼 및 캐시의 조합입니다.

[FreeableMemory] 측정치에는 사용 가능한 실제 여유 메모리가 반영되지 않지만, [FreeableMemory] 및 [SwapUsage] 측정치를 보고 복제 인스턴스가 오버로드되었는지 여부를 알 수 있습니다.

다음 조건에 대해 이 두 측정치를 모니터링합니다.

- • FreeableMemory 측정치가 0에 가까움.
- • SwapUsage 측정치가 증가하거나 변동이 있음.

이러한 두 가지 상태 중 하나에 해당하는 경우 이는 대규모 복제 인스턴스로 전환하는 것을 고려해야 함을 나타냅니다. 또한 복제 인스턴스에서 실행 중인 작업 수와 유형을 줄이기는 것도 고려해 보아야 합니다. 전체 로드 작업에는 변경 내용만을 복제하는 작업보다 더 많은 메모리가 요구됩니다.

AWS DMS 마이그레이션 태스크에 필요한 실제 메모리 요구 사항을 대략적으로 추정하려면 다음 파라미터를 사용할 수 있습니다.

LOB 열

마이그레이션 범위 내 각 테이블의 평균 LOB 열 수입니다.

[Maximum number of tables to load in parallel]

AWS DMS가 단일 태스크에서 병렬로 로드되는 최대 테이블 수입니다.

기본값은 8입니다.

LOB 청크 크기

AWS DMS가 데이터를 대상 데이터베이스에 복제하는 데 사용하는 LOB 청크의 크기(KB)입니다.

전체 로드 중 커밋 비율

AWS DMS가 병렬로 전송할 수 있는 최대 레코드 수입니다.

기본값은 10,000입니다.

LOB 크기

개별 LOB의 최대 크기(KB)입니다.

대량 배열 크기

엔드포인트 드라이버에서 가져오거나 처리하는 최대 행 수입니다. 이 값은 드라이버 설정에 따라 달라집니다.

기본값은 1000입니다.

이러한 값을 결정한 후 다음 방법 중 하나를 사용하여 마이그레이션 태스크에 필요한 메모리 양을 추정할 수 있습니다. 이러한 방법은 마이그레이션 태스크의 LOB 열 설정에서 선택한 옵션에 따라 달라집니다.

- 전체 LOB 모드의 경우 다음 공식을 사용합니다.

  ```
  Required memory = (LOB columns) * (Maximum number of tables to load in parallel) * (LOB chunk size) * (Commit rate during full load)
  ```

  소스 테이블에 평균 2개의 LOB 열이 포함되어 있고 LOB 청크 크기가 64KB인 경우를 예로 들어 보겠습니다. `Maximum number of tables to load in parallel` 및 `Commit rate during full load`에 기본값을 사용하는 경우 태스크에 필요한 메모리 양은 다음과 같습니다.

  ```
  Required memory = 2 * 8 * 64 * 10,000 = 10,240,000 KB
  ```

  > **Note**
  >
  > 전체 로드 중 커밋 비율 값을 줄이려면 AWS DMS 콘솔을 열고 데이터베이스 마이그레이션 태스크를 선택한 다음, 태스크를 생성하거나 수정합니다. 고급 설정을 확장하고 전체 로드 중 커밋 비율 값을 입력합니다.

- 제한적 LOB 모드의 경우 다음 공식을 사용합니다.

  ```
  Required memory = (LOB columns) * (Maximum number of tables to load in parallel) * (LOB size) * (Bulk array size)
  ```

  소스 테이블에 평균 2개의 LOB 열이 포함되어 있고 개별 LOB의 최대 크기가 4,096KB인 경우를 예로 들어 보겠습니다. `Maximum number of tables to load in parallel` 및 `Bulk array size`에 기본값을 사용하는 경우 태스크에 필요한 메모리 양은 다음과 같습니다.

```
Required memory = 2 * 8 * 4,096 * 1,000 = 65,536,000 KB
```

AWS DMS가 변환을 최적으로 수행하려면 변환이 발생할 때 CPU를 사용할 수 있어야 합니다. CPU에 과부하가 걸리고 CPU 리소스가 충분하지 않으면 마이그레이션 속도가 느려질 수 있습니다. AWS DMS는 CPU 집약적일 수 있는데, 특히 Oracle에서 PostgreSQL로 마이그레이션하는 것과 같은 이기종 마이그레이션 및 복제를 수행할 때 그렇습니다. C4 복제 인스턴스 클래스 사용은 이러한 상황에서 훌륭한 선택이 될 수 있습니다. 자세한 내용은 마이그레이션에 적합한 AWS DMS 복제 인스턴스 선택 섹션을 참조하세요.

복제 인스턴스 지표

복제 인스턴스 모니터링에는 다음 통계의 Amazon CloudWatch 지표가 포함됩니다.

지표	설명
AvailableMemory	스와핑 없이 새 애플리케이션을 시작하는 데 사용할 수 있는 메모리의 크기. 자세한 내용은 Linux man-pages 페이지의 `/proc/memInfo` 단원에서 `MemAvailable` 값을 참조하세요. 단위: 바이트
CPUAllocated	태스크에 최대로 할당된 CPU의 비율(0은 제한이 없음을 의미함). AWS DMS는 CloudWatch 콘솔의 `ReplicationInstanceIdentifer` 및 `ReplicationTaskIdentifier` 차원을 합친 것과 비교하여 이 지표를 높입니다. `ReplicationInstanceIdentifier, ReplicationTaskIdentifier` 범주를 사용하여 이 지표를 볼 수 있습니다. 단위: 백분율
CPUUtilization	인스턴스에서 현재 사용 중인 할당된 vCPU(가상 CPU)의 비율입니다. 단위: 백분율
DiskQueueDepth	디스크 액세스를 대기 중인 읽기/쓰기 요청(I/O) 수입니다. 단위: 개수

지표	설명
FreeStorageSpace	사용 가능한 스토리지 공간 크기입니다. 단위: 바이트
FreeMemory	애플리케이션, 페이지 캐시, 커널 자체 데이터 구조에서 사용할 수 있는 물리적 메모리의 양입니다. 자세한 내용은 Linux man-pages 페이지의 `/proc/memInfo` 단원에서 `MemFree` 값을 참조하세요. 단위: 바이트
FreeableMemory	사용 가능한 RAM 크기입니다. 단위: 바이트
MemoryAllocated	태스크에 대한 최대 메모리 할당량입니다(0은 제한이 없음을 의미함). AWS DMS는 CloudWatch 콘솔의 `ReplicationInstanceIdentifer` 및 `ReplicationTaskIdentifier` 차원을 합친 것과 비교하여 이 지표를 높입니다. `ReplicationInstanceIdentifier, ReplicationTaskIdentifier` 범주를 사용하여 이 지표를 볼 수 있습니다. 단위: MiB
WriteIOPS	초당 평균 디스크 쓰기 I/O 연산 수 단위: 개수/초
ReadIOPS	초당 평균 디스크 읽기 I/O 연산 수 단위: 개수/초
WriteThroughput	초당 디스크에 쓴 평균 바이트 수. 단위: 바이트/초
ReadThroughput	초당 디스크에서 읽은 평균 바이트 수입니다. 단위: 바이트/초

지표	설명
WriteLatency	디스크 I/O (출력) 연산당 평균 처리 시간. 단위: 밀리초
ReadLatency	디스크 I/O (입력) 연산당 평균 처리 시간. 단위: 밀리초
SwapUsage	복제 인스턴스에서 사용된 스왑 공간 크기. 단위: 바이트
NetworkTransmitThroughput	복제 인스턴스의 송신 네트워크 트래픽입니다(고객 데이터베이스 트래픽과 모니터링과 복제에 사용되는 AWS DMS 트래픽을 모두 포함). 단위: 바이트/초
NetworkReceiveThroughput	복제 인스턴스의 수신 네트워크 트래픽입니다(고객 데이터베이스 트래픽과 모니터링 및 복제에 사용된 AWS DMS 트래픽을 모두 포함). 단위: 바이트/초

복제 작업 지표

복제 태스크 모니터링에는 다음 통계의 지표가 포함됩니다.

지표	설명
FullLoadThroughputBandwidthTarget	대상 전체 로드의 발신 데이터(초당 KB 단위).
FullLoadThroughputRowsTarget	대상 전체 로드의 발신 변경 사항(초당 행 수 단위).
CDCIncomingChanges	대상에 적용되기를 기다리고 있는 특정 시점의 총 변경 이벤트 수입니다. 이 값은 소스 엔드포인트의 트랜잭션 변경 비율 측정치와 같지 않습

지표	설명
	니다. 이 측정치의 값이 크면 일반적으로 AWS DMS가 캡처한 변경 사항을 적시에 적용할 수 없어서 대상 지연 시간이 길어짐을 의미합니다.
CDCChangesMemorySource	메모리에서 누적되면서 원본에서 커밋되기를 대기하는 행 개수. 이 지표를 CDCChangesDiskSource와 함께 확인할 수 있습니다.
CDCChangesMemoryTarget	메모리에서 누적되면서 대상으로 커밋되기를 대기하는 행 개수. 이 지표를 CDCChangesDiskTarget과 함께 확인할 수 있습니다.
CDCChangesDiskSource	디스크에서 누적되면서 원본에서 커밋되기를 대기하는 행 개수. 이 지표를 CDCChangesMemorySource와 함께 확인할 수 있습니다.
CDCChangesDiskTarget	디스크에서 누적되면서 대상으로 커밋되기를 대기하는 행 개수. 이 지표를 CDCChangesMemoryTarget과 함께 확인할 수 있습니다.
CDCThroughputBandwidthTarget	대상의 발신 데이터(초당 KB 단위). CDCThroughputBandwidth는 샘플링 포인트에서 전송된 송신 데이터를 기록합니다. 어떠한 태스크 네트워크 트래픽도 찾을 수 없다면 이 값은 0입니다. CDC는 장시간 트랜잭션을 할 수 없기 때문에 네트워크 트래픽은 기록되지 않을 수 있습니다.
CDCThroughputRowsSource	원본의 수신 작업 수신 변경 사항 개수(초당 행 수 단위).
CDCThroughputRowsTarget	대상의 발신 작업 변경 사항 개수(초당 행 수 단위).

지표	설명
CDCLatencySource	소스 엔드포인트에서 캡처한 마지막 이벤트와 AWS DMS 인스턴스의 현재 시스템 타임스탬프 간의 간격(초)입니다. CDCLatencySource는 소스 인스턴스와 복제 인스턴스 간의 지연 시간을 나타냅니다. CDCLatencySource가 높으면 소스에서 변경 사항을 캡처하는 프로세스가 지연되었음을 나타냅니다. 지속적 복제의 지연 시간을 식별하기 위해 이 지표를 CDCLatencyTarget과 함께 확인할 수 있습니다. CDCLatencySource와 CDCLatencyTarget이 모두 높으면 CDCLatencySource를 먼저 조사하세요. 소스 인스턴스와 복제 인스턴스 간에 복제 지연이 없는 경우 CDCSourceLatency는 0이 될 수 있습니다. 복제 태스크에서 소스의 트랜잭션 로그에서 다음 이벤트를 읽으려고 시도하고 소스에서 마지막으로 읽은 때와 비교하여 새 이벤트가 없는 경우에도 CDCSourceLatency는 0이 될 수 있습니다. 이 경우 태스크는 CDCSourceLatency를 0으로 재설정합니다.
CDCLatencyTarget	대상에 커밋되기를 대기하는 첫 이벤트 타임스탬프와 AWS DMS 인스턴스의 현재 타임스탬프 사이의 간격(초)입니다. 대상 지연 시간은 복제 인스턴스 서버 시간과 대상 구성 요소에 전달된 확인되지 않은 가장 오래된 이벤트 ID 간의 차이입니다. 즉, 대상 지연 시간은 복제 인스턴스와 적용되었지만 TRG 엔드포인트에서 확인되지 않은 가장 오래된 이벤트 간의 타임스탬프 차이입니다(99%). CDCLatencyTarget이 높으면 대상에 변경 이벤트를 적용하는 프로세스가 지연되었음을 나타냅니다. 지속적 복제의 지연 시간을 식별하기 위해 이 지표를 CDCLatencySource와 함께 확인할 수 있습니다. CDCLatencyTarget은 높지만 CDCLatencySource는 높지 않은 경우 다음을 조사하세요. • 대상에 프라이머리 키 또는 인덱스가 없음 • 대상 또는 복제 인스턴스에서 리소스 병목 현상이 발생함 • 복제 인스턴스와 대상 사이에 네트워크 문제가 있음

지표	설명
CPUUtilization	태스크가 여러 코어에 걸쳐 사용하는 CPU의 비율입니다. 태스크 CPUUtilization의 의미는 복제 CPUUtilization과 약간 다릅니다. 하나의 vCPU가 완전히 사용되는 경우 100%를 나타내지만 여러 vCPU가 사용되는 경우에는 값이 100%를 초과할 수 있습니다. 단위: 백분율
SwapUsage	태스크가 사용하는 스왑의 양. 단위: 바이트
MemoryUsage	태스크에서 소비한 제어 그룹(cgroup) memory.usage_in_bytes DMS는 cgroup을 사용하여 메모리 및 CPU와 같은 시스템 리소스의 사용을 제어합니다. 이 지표는 해당 태스크에 할당된 cgroup 내에서 태스크의 메모리 사용량을 메가바이트 단위로 나타냅니다. cgroup 한도는 DMS 복제 인스턴스 클래스에 사용할 수 있는 리소스를 기반으로 합니다. memory.usage_in_bytes는 메모리의 RSS(상주 세트 크기), 캐시 및 스왑 구성 요소로 구성됩니다. 필요한 경우 운영 체제가 캐시 메모리를 회수할 수 있습니다. 복제 인스턴스 지표인 AvailableMemory도 모니터링하는 것이 좋습니다. AWS DMS는 CloudWatch 콘솔의 `ReplicationInstanceIdentifer` 및 `ReplicationTaskIdentifier` 차원을 합친 것과 비교하여 이 지표를 높입니다. `ReplicationInstanceIdentifier, ReplicationTaskIdentifier` 범주를 사용하여 이 지표를 볼 수 있습니다.

AWS DMS 태스크 로그 보기 및 관리

Amazon CloudWatch를 사용하여 AWS DMS 마이그레이션 프로세스 중에 태스크 정보를 로깅할 수 있습니다. 작업 설정을 선택할 때 로깅을 활성화합니다. 자세한 내용은 작업 설정 로깅 섹션을 참조하세요.

실행된 작업의 로그를 보려면 다음 단계를 따르세요.

1. AWS DMS 콘솔을 열고 탐색 창에서 데이터베이스 마이그레이션 태스크를 선택합니다. 데이터베이스 마이그레이션 태스크 대화 상자가 나타납니다.
2. 작업 이름을 선택합니다. 개요 세부 정보 대화 상자가 나타납니다.
3. 마이그레이션 태스크 로그 섹션을 찾아 CloudWatch Logs 보기를 선택합니다.

또한 AWS CLI 또는 AWS DMS API를 사용하여 태스크 로그에 대한 정보를 볼 수 있습니다. 이렇게 하려면 `describe-replication-instance-task-logs` AWS CLI 명령 또는 AWS DMS API 작업 `DescribeReplicationInstanceTaskLogs`를 사용하세요.

예를 들어, 다음 AWS CLI 명령은 JSON 형식의 태스크 로그 메타데이터를 보여줍니다.

```
$ aws dms describe-replication-instance-task-logs \
   --replication-instance-arn arn:aws:dms:us-east-1:237565436:rep:CDSFSFSFFFSSUFCAY
```

명령의 샘플 응답은 다음과 같습니다.

```
{
    "ReplicationInstanceTaskLogs": [
        {
            "ReplicationTaskArn": "arn:aws:dms:us-
east-1:237565436:task:MY34U6Z4MSY52GRTIX3O4AY",
            "ReplicationTaskName": "mysql-to-ddb",
            "ReplicationInstanceTaskLogSize": 3726134
        }
    ],
    "ReplicationInstanceArn": "arn:aws:dms:us-east-1:237565436:rep:CDSFSFSFFFSSUFCAY"
}
```

이 응답에 복제 인스턴스와 관련된 한 가지 작업 로그(`mysql-to-ddb`)가 있습니다. 이 로그의 크기는 3,726,124바이트입니다.

`describe-replication-instance-task-logs`에서 반환한 정보를 사용하여 작업 로그 문제를 진단하고 해결할 수 있습니다. 예를 들어, 태스크의 상세 디버그 로깅을 활성화하면 태스크 로그가 빠르게 증가하게 되어 복제 인스턴스에서 사용 가능한 스토리지를 모두 소비하고 인스턴스 상태가 `storage-full`로 변경될 수 있습니다. 작업 로그를 기술하면 더 이상 필요 없는 항목을 파악해 삭제하여 스토리지 공간을 확보할 수 있습니다.

작업의 작업 로그를 삭제하려면 작업 설정 DeleteTaskLogs를 true로 설정하세요. 예를 들어, AWS CLI modify-replication-task 명령 또는 AWS DMS API ModifyReplicationTask 작업을 사용하여 태스크를 수정할 때 다음 JSOM이 태스크 로그를 삭제합니다.

```
{
   "Logging": {
       "DeleteTaskLogs":true
   }
}
```

AWS CloudTrail을 사용하여 AWS DMS API 호출 로깅

AWS DMS는 AWS DMS에서 사용자, 역할, 또는 AWS 서비스가 수행한 작업에 대한 레코드를 제공하는 서비스인 AWS CloudTrail와 통합됩니다. CloudTrail은 AWS DMS 콘솔의 호출, AWS DMS API 작업에 대한 코드 호출을 포함하여 AWS DMS에 대한 모든 API 직접 호출을 이벤트로 캡처합니다. 추적을 생성하면 AWS DMS 이벤트를 포함한 CloudTrail 이벤트를 지속적으로 Amazon S3 버킷에 배포할 수 있습니다. 추적을 구성하지 않은 경우에도 CloudTrail 콘솔의 이벤트 기록에서 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집한 정보를 사용하여 AWS DMS에 수행된 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

CloudTrail에 대한 자세한 내용은 AWS CloudTrail 사용 설명서를 참조하세요.

CloudTrail의 AWS DMS 정보

CloudTrail은 계정 생성 시 AWS 계정에서 사용되도록 설정됩니다. AWS DMS에서 활동이 발생하면 해당 활동이 [이벤트 기록]의 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. AWS 계정에서 최신 이벤트를 확인, 검색 및 다운로드할 수 있습니다. 자세한 내용은 CloudTrail 이벤트 기록을 사용하여 이벤트 보기를 참조하세요.

AWS DMS에 대한 이벤트를 포함하여 AWS 계정에 이벤트를 지속적으로 기록하려면 추적을 생성합니다. CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 콘솔에서 추적을 생성하면 기본적으로 모든 AWS 리전에 추적이 적용됩니다. 추적은 AWS 파티션에 있는 모든 AWS 리전의 이벤트를 로깅하고 지정한 Amazon S3 버킷으로 로그 파일을 전송합니다. 또는 CloudTrail 로그에서 수집된 이벤트 데이터를 추가 분석 및 처리하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 정보는 다음을 참조하세요.

- 추적 생성 개요

- CloudTrail 지원 서비스 및 통합
- CloudTrail에 대한 Amazon SNS 알림 구성
- 여러 AWS 리전에서 CloudTrail 로그 파일 수신 및 여러 계정에서 CloudTrail 로그 파일 수신

모든 AWS DMS 작업은 CloudTrail에서 로깅되고 AWS Database Migration Service API 참조에 기록됩니다. 예를 들어 `CreateReplicationInstance`, `TestConnection`, `StartReplicationTask` 작업을 호출하면 CloudTrail 로그 파일에 항목이 생성됩니다.

모든 이벤트 및 로그 항목에는 요청을 생성한 사용자에 대한 정보가 들어 있습니다. 자격 증명 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트로 했는지 아니면 IAM 사용자 보안 인증 정보로 했는지 여부.
- 역할 또는 연합된 사용자에 대한 임시 보안 자격 증명을 사용하여 요청이 생성되었는지 여부.
- 다른 AWS 서비스에서 요청했는지 여부.

자세한 내용은 CloudTrail userIdentity 요소를 참조하세요.

AWS DMS 로그 파일 항목 이해

추적이란 지정한 Amazon S3 버킷에 이벤트를 로그 파일로 입력할 수 있게 하는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함될 수 있습니다. 이벤트는 모든 소스의 단일 요청을 나타내며 요청된 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보를 포함하고 있습니다. CloudTrail 로그 파일은 퍼블릭 API 호출에 대한 순서 지정된 스택 추적이 아니기 때문에 특정 순서로 표시되지 않습니다.

다음은 `RebootReplicationInstance` 작업을 보여 주는 CloudTrail 로그 항목이 나타낸 예제입니다.

```
{
"eventVersion": "1.05",
"userIdentity": {
    "type": "AssumedRole",
    "principalId": "AKIAIOSFODNN7EXAMPLE:johndoe",
    "arn": "arn:aws:sts::123456789012:assumed-role/admin/johndoe",
    "accountId": "123456789012",
    "accessKeyId": "ASIAYFI33SINADOJJEZW",
    "sessionContext": {
        "attributes": {
```

```
                "mfaAuthenticated": "false",
                "creationDate": "2018-08-01T16:42:09Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAIOSFODNN7EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/admin",
                "accountId": "123456789012",
                "userName": "admin"
            }
        }
    },
    "eventTime": "2018-08-02T00:11:44Z",
    "eventSource": "dms.amazonaws.com",
    "eventName": "RebootReplicationInstance",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "72.21.198.64",
    "userAgent": "console.amazonaws.com",
    "requestParameters": {
        "forceFailover": false,
        "replicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:EX4MBJ2NMRDL3BMAYJOXUGYPUE"
    },
    "responseElements": {
        "replicationInstance": {
            "replicationInstanceIdentifier": "replication-instance-1",
            "replicationInstanceStatus": "rebooting",
            "allocatedStorage": 50,
            "replicationInstancePrivateIpAddresses": [
                "172.31.20.204"
            ],
            "instanceCreateTime": "Aug 1, 2018 11:56:21 PM",
            "autoMinorVersionUpgrade": true,
            "engineVersion": "2.4.3",
            "publiclyAccessible": true,
            "replicationInstanceClass": "dms.t2.medium",
            "availabilityZone": "us-east-1b",
            "kmsKeyId": "arn:aws:kms:us-east-1:123456789012:key/
f7bc0f8e-1a3a-4ace-9faa-e8494fa3921a",
            "replicationSubnetGroup": {
                "vpcId": "vpc-1f6a9c6a",
                "subnetGroupStatus": "Complete",
                "replicationSubnetGroupArn": "arn:aws:dms:us-
east-1:123456789012:subgrp:EDHRVRBAAAPONQAIYWP4NUW22M",
```

```
                "subnets": [
                    {
                        "subnetIdentifier": "subnet-cbfff283",
                        "subnetAvailabilityZone": {
                            "name": "us-east-1b"
                        },
                        "subnetStatus": "Active"
                    },
                    {
                        "subnetIdentifier": "subnet-d7c825e8",
                        "subnetAvailabilityZone": {
                            "name": "us-east-1e"
                        },
                        "subnetStatus": "Active"
                    },
                    {
                        "subnetIdentifier": "subnet-6746046b",
                        "subnetAvailabilityZone": {
                            "name": "us-east-1f"
                        },
                        "subnetStatus": "Active"
                    },
                    {
                        "subnetIdentifier": "subnet-bac383e0",
                        "subnetAvailabilityZone": {
                            "name": "us-east-1c"
                        },
                        "subnetStatus": "Active"
                    },
                    {
                        "subnetIdentifier": "subnet-42599426",
                        "subnetAvailabilityZone": {
                            "name": "us-east-1d"
                        },
                        "subnetStatus": "Active"
                    },
                    {
                        "subnetIdentifier": "subnet-da327bf6",
                        "subnetAvailabilityZone": {
                            "name": "us-east-1a"
                        },
                        "subnetStatus": "Active"
                    }
                ],
```

```
                "replicationSubnetGroupIdentifier": "default-vpc-1f6a9c6a",
                "replicationSubnetGroupDescription": "default group created by console
 for vpc id vpc-1f6a9c6a"
            },
            "replicationInstanceEniId": "eni-0d6db8c7137cb9844",
            "vpcSecurityGroups": [
                {
                    "vpcSecurityGroupId": "sg-f839b688",
                    "status": "active"
                }
            ],
            "pendingModifiedValues": {},
            "replicationInstancePublicIpAddresses": [
                "18.211.48.119"
            ],
            "replicationInstancePublicIpAddress": "18.211.48.119",
            "preferredMaintenanceWindow": "fri:22:44-fri:23:14",
            "replicationInstanceArn": "arn:aws:dms:us-
east-1:123456789012:rep:EX4MBJ2NMRDL3BMAYJOXUGYPUE",
            "replicationInstanceEniIds": [
                "eni-0d6db8c7137cb9844"
            ],
            "multiAZ": false,
            "replicationInstancePrivateIpAddress": "172.31.20.204",
            "patchingPrecedence": 0
        }
    },
    "requestID": "a3c83c11-95e8-11e8-9d08-4b8f2b45bfd5",
    "eventID": "b3c4adb1-e34b-4744-bdeb-35528062a541",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```


AWS DMS 컨텍스트 로깅

AWS DMS는 컨텍스트 로깅을 사용하여 진행 중인 마이그레이션에 대한 정보를 제공합니다. 컨텍스트 로깅은 다음과 같은 정보를 태스크의 CloudWatch 로그에 기록합니다.

- 태스크의 소스 및 대상 데이터베이스 연결에 대한 정보.
- 복제 태스크 동작. 태스크 로그를 사용하여 복제 문제를 진단할 수 있습니다.

- AWS DMS가 소스 및 대상 데이터베이스에서 실행하는 데이터가 없는 SQL 문 SQL 로그를 사용하여 예상치 못한 마이그레이션 동작을 진단할 수 있습니다.
- 각 CDC 이벤트의 위치 세부 정보를 스트리밍합니다.

컨텍스트 로깅은 AWS DMS 버전 3.5.0 이상에서만 사용할 수 있습니다.

AWS DMS는 컨텍스트 로깅을 기본적으로 활성화합니다. 컨텍스트 로깅을 제어하려면 `EnableLogContext` 태스크 설정을 `true` 또는 `false`로 설정하거나 콘솔에서 태스크를 수정합니다.

AWS DMS는 CloudWatch 로그의 복제 태스크에 컨텍스트 로그 정보를 3분마다 기록합니다. 복제 인스턴스에 애플리케이션 로그를 위한 충분한 공간이 있는지 확인하세요. 태스크 로그 관리에 대한 자세한 내용은 AWS DMS 태스크 로그 보기 및 관리 섹션을 참조하세요.

주제

- 객체 유형
- 로깅 예제
- 제한 사항

객체 유형

AWS DMS는 CloudWatch에서 다음 객체 유형에 대한 컨텍스트 로깅을 생성합니다.

객체 유형	설명
`TABLE_NAME`	이러한 로그 항목에는 현재 태스크 매핑 규칙의 범위 내에 있는 테이블에 대한 정보가 들어 있습니다. 이러한 항목을 사용하여 마이그레이션 중 특정 기간 동안의 테이블 이벤트를 검사할 수 있습니다.
`SCHEMA_NAME`	이러한 로그 항목에는 현재 태스크 매핑 규칙에서 사용하는 스키마에 대한 정보가 들어 있습니다. 이러한 항목을 사용하여 AWS DMS가 마이그레이션 중 특정 기간 동안 어떤 스키마를 사용하고 있는지 확인할 수 있습니다.

객체 유형	설명
TRANSACTION_ID	이러한 항목에는 소스 데이터베이스에서 캡처된 각 DML/DDL 변경 사항에 대한 트랜잭션 ID가 포함됩니다. 이러한 로그 항목을 사용하여 특정 트랜잭션 중에 발생한 변경 사항을 확인할 수 있습니다.
CONNECTION_ID	이러한 항목에는 연결 ID가 포함됩니다. 이러한 로그 항목을 사용하여 AWS DMS가 각 마이그레이션 단계에서 사용하는 연결을 확인할 수 있습니다.
STATEMENT	이러한 항목에는 각 마이그레이션 변경 사항을 가져오고, 처리하고, 적용하는 데 사용되는 SQL 코드가 포함됩니다.
STREAM_POSITION	이러한 항목에는 소스 데이터베이스에 대한 각 마이그레이션 작업의 트랜잭션 로그 파일 내 위치가 포함됩니다. 이러한 항목의 형식은 소스 데이터베이스 엔진 유형에 따라 다릅니다. CDC 전용 복제를 구성할 때 이 정보를 사용하여 복구 체크포인트의 시작 위치를 결정할 수도 있습니다.

로깅 예제

이 섹션에는 복제를 모니터링하고 복제 문제를 진단하는 데 사용할 수 있는 로그 레코드 예제가 포함되어 있습니다.

연결 로그 예제

이 섹션에는 연결 ID가 포함된 로그 샘플이 포함되어 있습니다.

```
2023-02-22T10:09:29 [SOURCE_CAPTURE  ]I:  Capture record 1 to internal
 queue from Source  {operation:START_REGULAR (43), connectionId:27598,
 streamPosition:0000124A/6800A778.NOW}  (streamcomponent.c:2920)
```

```
2023-02-22T10:12:30 [SOURCE_CAPTURE  ]I:  Capture record 0 to internal queue from
 Source  {operation:IDLE (51), connectionId:27598}  (streamcomponent.c:2920)

2023-02-22T11:25:27 [SOURCE_CAPTURE  ]I:  Capture record 0 to internal queue
 from Source  {operation:IDLE (51), columnName:region, connectionId:27598}
 (streamcomponent.c:2920)
```

태스크 동작 로그 예제

이 섹션에는 복제 태스크 로그 동작에 대한 로그 샘플이 포함되어 있습니다. 이 정보를 사용하여 IDLE 상태의 태스크와 같은 복제 문제를 진단할 수 있습니다.

다음 SOURCE_CAPTURE 로그는 소스 데이터베이스 로그 파일에서 읽을 수 있는 이벤트가 없음을 나타내며, 대상 데이터베이스에 적용할 AWS DMS CDC 구성 요소로부터 받은 이벤트가 없음을 나타내는 TARGET_APPLY 레코드를 포함합니다. 이러한 이벤트에는 이전에 적용된 이벤트 관련 컨텍스트 세부 정보도 포함됩니다.

```
2023-02-22T11:23:24 [SOURCE_CAPTURE  ]I:  No Event fetched from wal log
 (postgres_endpoint_wal_engine.c:1369)
2023-02-22T11:24:29 [TARGET_APPLY    ]I:  No records received to load
 or apply on target , waiting for data from upstream. The last context
 is  {operation:INSERT (1), tableName:sales_11, schemaName:public,
 txnId:18662441, connectionId:17855, statement:INSERT INTO
 "public"."sales_11"("sales_no","dept_name","sale_amount","sale_date","region") values
 (?,?,?,?,?),
```

SQL 문 로그 예제

이 섹션에는 소스 및 대상 데이터베이스에서 실행되는 SQL 문에 대한 로그 샘플이 포함되어 있습니다. 로그에 표시되는 SQL 문에는 SQL 문만 표시되고 데이터는 표시되지 않습니다. 다음 TARGET_APPLY 로그는 대상에서 실행된 INSERT 문을 보여줍니다.

```
2023-02-22T11:26:07 [TARGET_APPLY    ]I:  Applied record 2193305 to
 target  {operation:INSERT (1), tableName:sales_111, schemaName:public,
 txnId:18761543, connectionId:17855, statement:INSERT INTO
 "public"."sales_111"("sales_no","dept_name","sale_amount","sale_date","region") values
 (?,?,?,?,?),
```

제한 사항

AWS DMS 컨텍스트 로깅에는 다음과 같은 제한 사항이 적용됩니다.

- AWS DMS는 모든 엔드포인트 유형에 대해 최소 로깅을 생성하지만 광범위한 엔진별 컨텍스트 로깅은 다음 엔드포인트 유형에만 사용할 수 있습니다. 이러한 엔드포인트 유형을 사용할 때는 컨텍스트 로깅을 활성화하는 것이 좋습니다.
  - MySQL
  - PostgreSQL
  - Oracle
  - Microsoft SQL Server
  - MongoDB/Amazon DocumentDB
  - Amazon S3

AWS Database Migration Service에서 Amazon EventBridge 이벤트 및 알림 사용

Amazon EventBridge를 사용하여 AWS DMS 이벤트(복제 인스턴스 생성 또는 삭제 등)가 발생하면 알림을 제공할 수 있습니다. EventBridge는 이벤트를 수신하고 이벤트 규칙에 정의된 대로 이벤트 알림을 라우팅합니다. 특정 AWS 리전의 Amazon EventBridge에서 지원하는 모든 형식으로 이러한 알림을 사용할 수 있습니다. Amazon EventBridge 사용에 대한 자세한 내용은 Amazon EventBridge 사용 설명서에서 Amazon EventBridge란 무엇입니까?를 참조하세요.

> **Note**
>
> Amazon EventBridge 이벤트 사용은 AWS DMS 버전 3.4.5 이상에서 지원됩니다.

EventBridge는 AWS DMS 환경의 변화를 나타내는 지표인 이벤트를 수신하고 규칙을 적용하여 이벤트를 알림 메커니즘으로 라우팅합니다. 규칙은 이벤트의 구조(이벤트 패턴이라고 함)를 기반으로 이벤트를 알림 메커니즘과 일치시킵니다.

AWS DMS는 사용자가 이벤트 규칙에 적용할 수 있는 카테고리로 이벤트를 그룹화합니다. 따라서 사용자는 해당 카테고리의 이벤트가 발생했을 때 이에 대한 알림을 받을 수 있습니다. 예를 들어, 특정 복제 인스턴스의 생성 카테고리에 EventBridge 이벤트 규칙을 적용한다고 가정해 보겠습니다. 그러면 복제 인스턴스에 영향을 주는 생성 관련 이벤트가 발생할 때마다 알림을 받게 됩니다. 복제 인스턴스의 구성 변경 카테고리에 규칙을 적용하면 복제 인스턴스의 구성이 변경될 때마다 메시지가 수신됩니다. AWS DMS에서 제공하는 이벤트 카테고리의 목록은 아래의 AWS DMS 이벤트 카테고리 및 이벤트 메시지를 참조하세요.

> **Note**
>
> events.amazonaws.com로부터 게시를 허용하려면 Amazon SNS 주제의 액세스 정책을 업데이트해야 합니다. 자세한 내용은 Amazon EventBridge 사용 설명서의 Amazon EventBridge에 대한 리소스 기반 정책 사용을 참조하세요.
>
> 이벤트 구독을 Amazon EventBridge로 이전하는 방법에 대한 자세한 내용은 활성 이벤트 구독을 DMS에서 Amazon EventBridge로 마이그레이션을 참조하세요.
>
> Amazon SNS 문자 메시지 사용에 대한 자세한 내용은 Amazon SNS를 사용한 SMS 알림 메시지 전송 및 수신을 참조하세요.

AWS DMS에서 Amazon EventBridge 이벤트 규칙 사용

Amazon EventBridge는 사용자가 EventBridge 이벤트 규칙을 생성할 때 입력한 주소로 이벤트 알림을 보냅니다. 여러 가지 규칙을 생성할 수 있습니다. 예를 들어 모든 이벤트 알림 메시지를 수신하는 하나의 규칙과 프로덕션 DMS 리소스의 중요 이벤트만 포함하는 다른 규칙을 생성할 수 있습니다. EventBridge에서 이벤트 알림을 활성화 또는 비활성화할 수도 있습니다.

AWS DMS 이벤트에 응답하는 Amazon EventBridge 규칙을 생성하려면

- Amazon EventBridge 사용 설명서의 이벤트에 응답하는 Amazon EventBridge 규칙 생성에 설명된 단계를 수행하고 AWS DMS 이벤트에 대한 규칙을 생성합니다.

  a. EventBridge가 규칙의 이벤트 패턴과 일치하는 이벤트를 수신할 때 수행할 작업을 지정합니다. 이벤트가 일치하면 EventBridge는 이벤트를 전송하고 규칙에 정의된 작업을 간접 호출합니다.

  b. 서비스 제공업체에서 AWS를 선택합니다.

  c. 서비스 이름에서 Server Migration Service (SMS)를 선택합니다.

이제부터 이벤트 알림 메시지를 수신할 수 있습니다.

다음 JSON 예제는 AWS DMS 서비스에 대한 EventBridge 이벤트 모델을 보여줍니다.

```
{
   "version":"0",
   "id":"11a11b11-222b-333a-44d4-01234a5b67890",
   "detail-type":"DMS Replication Task State Change",
   "source":"aws.dms",
   "account":"0123456789012",
   "time":"1970-01-01T00:00:00Z",
   "region":"us-east-1",
   "resources":[
      "arn:aws:dms:us-east-1:012345678901:task:AAAABBBB0CCCCDDDDEEEEE1FFFF2GGG3FFFFFF3"
   ],
   "detail":{
      "type":"REPLICATION_TASK",
      "category":"StateChange",
      "eventType":"REPLICATION_TASK_STARTED",
      "eventId":"DMS-EVENT-0069",
      "resourceLink":"https://console.aws.amazon.com/dms/v2/home?region=us-
east-1#taskDetails/taskName",
```

```
      "detailMessage":"Replication task started, with flag = fresh start"
   }
}
```

알림을 받을 수 있는 카테고리와 이벤트 목록은 다음 섹션을 참조하세요.

AWS DMS 이벤트 범주 및 이벤트 메시지

AWS DMS는 사용자가 식별할 수 있는 이벤트 카테고리를 다수 생성합니다. 각 카테고리는 복제 인스턴스 또는 복제 태스크 소스 형식에 적용됩니다.

주제

- 복제 인스턴스 이벤트 메시지
- 복제 태스크 이벤트 메시지
- 복제 이벤트 메시지

복제 인스턴스 이벤트 메시지

다음 표에는 복제 인스턴스 소스 유형에서 가능한 카테고리 및 이벤트가 나와 있습니다.

범주	이벤트 ID	설명
생성	DMS-EVENT-0067	복제 인스턴스를 생성 중입니다.
삭제	DMS-EVENT-0066	복제 인스턴스를 삭제 중입니다.
구성 변경	DMS-EVENT-0012	이 복제 인스턴스의 복제 인스턴스 클래스가 변경되고 있습니다.
구성 변경	DMS-EVENT-0018	복제 인스턴스의 스토리지가 증가하고 있습니다.
구성 변경	DMS-EVENT-0024	복제 인스턴스가 다중 AZ 구성으로 전환 중입니다.

범주	이벤트 ID	설명
구성 변경	DMS-EVENT-0030	복제 인스턴스가 단일 AZ 구성으로 전환 중입니다.
유지 관리	DMS-EVENT-0026	복제 인스턴스의 오프라인 유지 관리가 진행 중입니다. 현재 복제 인스턴스를 사용할 수 없습니다.
생성	DMS-EVENT-0005	복제 인스턴스를 생성 중입니다.
삭제	DMS-EVENT-0003	복제 인스턴스가 삭제되었습니다.
구성 변경	DMS-EVENT-0014	이 복제 인스턴스의 복제 인스턴스 클래스가 변경되었습니다.
구성 변경	DMS-EVENT-0017	복제 인스턴스의 스토리지가 증가되었습니다.
구성 변경	DMS-EVENT-0025	복제 인스턴스가 다중 AZ 구성으로 전환이 완료되었습니다.
구성 변경	DMS-EVENT-0029	복제 인스턴스가 단일 AZ 구성으로 전환이 완료되었습니다.
유지 관리	DMS-EVENT-0047	복제 인스턴스의 관리 소프트웨어가 업데이트되었습니다.
유지 관리	DMS-EVENT-0027	복제 인스턴스의 오프라인 유지 관리가 완료되었습니다. 이제 복제 인스턴스를 사용할 수 있습니다.
유지 관리	DMS-EVENT-0068	복제 인스턴스가 업그레이드할 수 없는 상태입니다.
장애 조치	DMS-EVENT-0034	장애 조치를 너무 자주 요청하면 일반 장애 조치 이벤트 대신 이 이벤트가 발생합니다.

범주	이벤트 ID	설명
결함	DMS-EVENT-0031	복제 인스턴스가 %s 상태로 전화되었습니다.
결함	DMS-EVENT-0036	호환되지 않는 네트워크로 인해 복제 인스턴스에 장애가 발생했습니다.
결함	DMS-EVENT-0037	서비스가 데이터 볼륨을 암호화하는 데 사용된 KMS 키에 액세스할 수 없습니다.
결함		복제 인스턴스에서 호환되지 않는 파라미터를 사용했습니다.
장애 조치		안전한 상태에서 사용자 요청 장애 조치를 시작하기 위한 대기 시간이 초과되었습니다.
장애 조치	DMS-EVENT-0013	다중 AZ 복제 인스턴스에서 장애 조치가 시작되었습니다.
장애 조치	DMS-EVENT-0049	다중 AZ 복제 인스턴스에서 장애 조치가 완료되었습니다.
장애 조치	DMS-EVENT-0050	다중 AZ 활성화가 시작되었습니다.
장애 조치	DMS-EVENT-0051	다중 AZ 활성화가 완료되었습니다.
StateChange		일반 및 느린 쿼리 로그가 자동으로 %s(으)로 전환되었습니다.

범주	이벤트 ID	설명
StateChange		AWS DMS가 애플리케이션 인스턴스 %s의 KMS 암호화 키에 액세스할 수 없습니다. 키가 비활성화되었거나 AWS DMS가 액세스할 수 없기 때문일 수 있습니다. 이 상태가 계속되면 애플리케이션이 액세스할 수 없음 상태가 됩니다. 자세한 내용은 AWS DMS 설명서의 문제 해결 섹션을 참조하세요.
StateChange		AWS DMS가 이제 애플리케이션 인스턴스 %s의 KMS 암호화 키에 액세스할 수 있습니다.
StateChange		Amazon DMS가 애플리케이션 인스턴스 %s의 KMS 암호화 키에 액세스할 수 없습니다. 이 애플리케이션은 액세스할 수 없음 상태가 됩니다. 자세한 내용은 Amazon RDS 설명서의 문제 해결 섹션을 참조하세요.
StateChange		복제 인스턴스 생성의 일환으로 HM에서 앱을 다시 시작
StateChange		복제 인스턴스 삭제의 일환으로 HM에서 앱을 종료
장애 조치	DMS-EVENT-0015	스탠바이로의 다중 AZ 장애 조치.
LowStorage	DMS-EVENT-0007	복제 인스턴스의 여유 스토리지가 부족합니다.
LowStorage		할당된 inode가 모두 소진되었습니다. 해결하려면 스토리지 규모를 조정해야 합니다.

복제 태스크 이벤트 메시지

다음 표에는 복제 태스크 소스 유형에서 가능한 카테고리 및 이벤트가 나와 있습니다.

범주	이벤트 ID	설명
결함	DMS-EVENT-0078	복제 태스크가 실패했습니다.
결함	DMS-EVENT-0082	태스크 데이터 정리 호출이 실패했습니다.
상태 변경	DMS-EVENT-0081	테이블 세부 정보 다시 로드가 요청되었습니다.
상태 변경		복제 태스크가 복사되었습니다.
상태 변경		복제 태스크 복사가 실패했습니다.
상태 변경		복제 태스크가 이동되었습니다.
상태 변경		복제 상태 이동이 실패했습니다.
상태 변경		대상 태스크를 생성하지 못했습니다.
상태 변경		복제 태스크 평가 실행이 시작되었습니다.
상태 변경		복제 태스크 평가 실행이 성공적으로 완료되었습니다.
상태 변경		복제 태스크 평가 실행이 완료되었지만 실패했습니다.
StateChange		복제 태스크 평가 실행이 경고와 함께 완료되었습니다.
StateChange		복제 태스크 평가 실행이 오류와 함께 완료되었습니다.

범주	이벤트 ID	설명
StateChange		복제 태스크 평가 실행 %s이(가) 취소되었습니다.
StateChange		복제 태스크 평가 실행 %s이(가) 삭제되었습니다.
StateChange		복제 태스크 평가 실행이 리소스를 프로비저닝하지 못했습니다.
StateChange		복제 태스크가 실패했습니다.
생성		복제 태스크가 생성되었습니다.
ConfigurationChange		복제 태스크가 수정되었습니다.
결함		복제 태스크가 실패했습니다.
StateChange	DMS-EVENT-0091	읽기가 일시 중지되었습니다. 스왑 파일 한도에 도달했습니다.
StateChange	DMS-EVENT-0092	읽기가 일시 중지되었습니다. 디스크 사용량 한도에 도달했습니다.
StateChange	DMS-EVENT-0093	읽기가 일시 중지되었습니다. 디스크 사용량 한도에 도달했습니다.
StateChange	DMS-EVENT-0093	읽기가 다시 시작되었습니다.
StateChange	DMS-EVENT-0069	복제 태스크가 다음 설정으로 시작되었습니다. taskType: %s, startType: %s
StateChange	DMS-EVENT-0079	복제 태스크가 중지되었습니다.
삭제	DMS-EVENT-0073	복제 태스크가 삭제되었습니다.

복제 이벤트 메시지

다음 표에는 복제 소스 유형에서 가능한 카테고리 및 이벤트가 나와 있습니다.

범주	설명
상태 변경	DMS 복제 스케일 업 이벤트.
상태 변경	DMS 복제 스케일 다운 이벤트.
상태 변경	DMS 복제 규모 조정 이벤트가 완료되었습니다.
상태 변경	DMS 복제가 생성되었습니다.
상태 변경	DMS 복제가 초기화되고 있습니다.
상태 변경	DMS 복제가 메타데이터 수집을 위한 리소스를 준비하고 있습니다.
상태 변경	DMS 복제에 연결된 연결을 테스트 중입니다.
상태 변경	DMS 복제가 메타데이터를 가져오는 중입니다.
상태 변경	DMS 복제가 용량을 계산하는 중입니다.
상태 변경	DMS 복제가 용량을 프로비저닝하는 중입니다.
상태 변경	DMS 복제가 프로비저닝되었습니다.
상태 변경	DMS 복제가 시작되었습니다.
상태 변경	DMS 복제가 실행 중입니다.
상태 변경	DMS 복제가 중지 중입니다.
상태 변경	DMS 복제가 중지되었습니다.
상태 변경	DMS 복제가 수정되고 있습니다.
상태 변경	DMS 복제를 삭제 중입니다.
상태 변경	DMS 복제가 용량을 프로비저닝 해제하는 중입니다.

범주	설명
상태 변경	DMS 복제가 프로비저닝 해제되었습니다.
결함	DMS 복제가 실패했습니다.

AWS Database Migration Service에서 Amazon SNS 이벤트 및 알림 사용

AWS DMS 3.4.5 릴리스 및 이후 버전에서는 AWS DMS 이벤트 발생 시 Amazon EventBridge를 사용하여 알림을 제공하는 것이 좋습니다. AWS DMS에서 EventBridge 이벤트를 사용하는 방법에 대한 자세한 내용은 AWS Database Migration Service에서 Amazon EventBridge 이벤트 및 알림 사용 섹션을 참조하세요.

이벤트 구독을 Amazon EventBridge로 이전

다음 AWS CLI 명령을 사용하여 활성 이벤트 구독을 DMS에서 Amazon EventBridge로 한 번에 최대 10개까지 마이그레이션할 수 있습니다.

```
update-subscriptions-to-event-bridge [--force-move | --no-force-move]
```

기본적으로 AWS DMS는 복제 인스턴스가 AWS DMS 3.4.5 이상의 최신 버전인 경우에만 활성 이벤트 구독을 마이그레이션합니다. 이 기본 동작을 재정의하려면 `--force-move` 옵션을 사용합니다. 하지만 복제 인스턴스가 업그레이드되지 않은 경우 Amazon EventBridge에서 일부 유형의 이벤트를 사용하지 못할 수 있습니다.

`update-subscriptions-to-event-bridge` CLI 명령을 실행하려면 AWS Identity and Access Management(IAM) 사용자는 다음 정책 권한이 있어야 합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
              "SNS:GetTopicAttributes",
              "SNS:SetTopicAttributes",
              "events:PutTargets",
              "events:EnableRule",
              "events:PutRule"
          ],
          "Resource": "*"
      }
  ]
}
```

구독을 EventBridge로 이전하는 방법에 대한 자세한 내용은 AWS Database Migration Service API 참조조의 UpdateSubscriptionsToEventBridge를 참조하세요.

Amazon SNS 이벤트 및 알림 작업

AWS DMS 버전 3.4.5 및 이하에서는 다음과 같은 이벤트 및 알림 작업을 지원합니다.

AWS Database Migration Service(AWS DMS)는 Amazon Simple Notification Service(Amazon SNS)를 사용하여 AWS DMS 이벤트(예: 복제 인스턴스 생성 또는 삭제)가 발생할 때 알림을 제공할 수 있습니다. AWS 리전에 따라 Amazon SNS에서 지원하는 형식으로 이메일 메시지, 문자 또는 HTTP 엔드포인트 직접 호출 등의 알림을 사용할 수 있습니다.

AWS DMS는 고객이 구독할 수 있는 카테고리로 이벤트를 그룹화합니다. 따라서 고객은 해당 카테고리의 이벤트가 발생했을 때 이에 대한 알림을 받을 수 있습니다. 예를 들어 임의의 복제 인스턴스에 대한 생성 카테고리를 구독할 경우 생성 관련 이벤트가 발생하여 복제 인스턴스에 영향을 끼칠 때마다 알림 메시지가 수신됩니다. 복제 인스턴스에서 구성 변경 카테고리를 구독하면 복제 인스턴스의 구성이 변경될 때마다 메시지가 수신됩니다. 또한 이벤트 알림 메시지 구독이 변경되어도 알림 메시지가 수신됩니다. AWS DMS에서 제공하는 이벤트 카테고리 목록은 아래에 나오는 AWS DMS 이벤트 카테고리 및 SNS 알림용 이벤트 메시지를 참조하세요.

AWS DMS는 사용자가 이벤트 구독을 생성할 때 제공한 주소로 이벤트 알림을 전송합니다. 모든 이벤트 알림 메시지를 수신하거나 프로덕션 DMS 리소스의 중요 이벤트만 수신하는 등 다른 구독을 복수로 생성하는 것도 가능합니다. AWS 콘솔에서 활성화됨 옵션을 선택 취소하거나, AWS DMS API를 사용하여 `Enabled` 파라미터를 false로 설정하면 구독을 삭제하지 않고 알림 기능을 쉽게 끌 수 있습니다.

> **Note**
>
> SMS 문자 메시지를 사용한 AWS DMS 이벤트 알림 서비스는 현재 Amazon SNS가 지원되는 모든 AWS 리전의 AWS DMS 리소스로 제공되고 있습니다. Amazon SNS가 SMS 메시징을 지원하는 AWS 리전 및 국가 목록은 지원되는 리전 및 국가를 참조하세요.
> SNS 문자 메시지 사용에 대한 자세한 내용은 Amazon SNS를 사용한 SMS 알림 메시지 전송 및 수신을 참조하세요.
> AWS DMS 이벤트 알림은 CloudWatch 또는 EventBridge의 CloudTrail 이벤트와 다릅니다. CloudTrail 이벤트 알림은 모든 API 간접 호출을 통해 생성할 수 있습니다. DMS는 DMS 이벤트가 발생한 경우에만 알림을 보냅니다.

AWS DMS는 구독 식별자를 사용하여 각 구독을 확인합니다. 동일한 Amazon SNS 주제에 게시된 여러 AWS DMS 이벤트를 구독할 수 있습니다. 이벤트 알림을 사용하면 Amazon SNS 요금이 적용됩니다. Amazon SNS 비용 청구에 대한 자세한 내용은 Amazon SNS 요금을 참조하세요.

Amazon SNS를 사용하여 AWS DMS 이벤트를 구독하려면 다음 프로세스를 사용합니다.

1. Amazon SNS 주제를 생성합니다. 주제에서 수신할 알림 유형과 알림이 도달할 주소 또는 번호를 지정합니다.
2. AWS Management Console, AWS CLI 또는 AWS DMS API를 사용하여 AWS DMS 이벤트 알림 서비스 구독을 생성합니다.
3. AWS DMS가 구독 생성 시 제출한 이메일 주소로 승인 이메일 또는 SMS 메시지를 전송합니다. 구독을 확인하려면, 승인 이메일이나 SMS 메시지의 링크를 클릭합니다.
4. 구독을 확인하면 AWS DMS 콘솔의 이벤트 구독 섹션에 구독 상태가 업데이트됩니다.
5. 이제부터 이벤트 알림 메시지가 수신됩니다.

알림을 받을 수 있는 카테고리와 이벤트 목록은 다음 섹션을 참조하세요. AWS DMS 이벤트 구독 및 관련 작업에 대한 자세한 내용은 SNS를 사용한 AWS DMS 이벤트 알림 구독 섹션을 참조하세요.

AWS DMS 이벤트 카테고리 및 SNS 알림용 이벤트 메시지

> **Important**
>
> AWS DMS 3.4.5 릴리스 및 이후 버전에서는 AWS DMS 이벤트 발생 시 Amazon EventBridge를 사용하여 알림을 제공하는 것이 좋습니다. AWS DMS에서 EventBridge 이벤트를 사용하는 방법에 대한 자세한 내용은 AWS Database Migration Service에서 Amazon EventBridge 이벤트 및 알림 사용 섹션을 참조하세요.

AWS DMS는 AWS DMS 콘솔 또는 AWS DMS API를 사용하여 구독할 수 있는 이벤트 카테고리가 매우 많습니다. 각 카테고리는 소스 형식에 적용됩니다. 즉, 현재 AWS DMS는 복제 인스턴스와 복제 태스크 소스 형식을 지원합니다.

다음 표에는 복제 인스턴스 원본 형식에서 가능한 카테고리와 이벤트가 나와 있습니다.

범주	DMS 이벤트 ID	설명
구성 변경	DMS-EVENT-0012	이 복제 인스턴스의 복제 인스턴스 클래스가 변경되고 있습니다.
구성 변경	DMS-EVENT-0014	이 복제 인스턴스의 복제 인스턴스 클래스가 변경되었습니다.
구성 변경	DMS-EVENT-0018	복제 인스턴스의 스토리지가 증가하고 있습니다.
구성 변경	DMS-EVENT-0017	복제 인스턴스의 스토리지가 증가되었습니다.
구성 변경	DMS-EVENT-0024	복제 인스턴스가 다중 AZ 구성으로 전환 중입니다.
구성 변경	DMS-EVENT-0025	복제 인스턴스가 다중 AZ 구성으로 전환이 완료되었습니다.
구성 변경	DMS-EVENT-0030	복제 인스턴스가 단일 AZ 구성으로 전환 중입니다.
구성 변경	DMS-EVENT-0029	복제 인스턴스가 단일 AZ 구성으로 전환이 완료되었습니다.
생성	DMS-EVENT-0067	복제 인스턴스를 생성 중입니다.
생성	DMS-EVENT-0005	복제 인스턴스가 생성되었습니다.
삭제	DMS-EVENT-0066	복제 인스턴스를 삭제 중입니다.
삭제	DMS-EVENT-0003	복제 인스턴스가 삭제되었습니다.
유지 관리	DMS-EVENT-0047	복제 인스턴스의 관리 소프트웨어가 업데이트되었습니다.
유지 관리	DMS-EVENT-0026	복제 인스턴스의 오프라인 유지 관리가 진행 중입니다. 현재 복제 인스턴스를 사용할 수 없습니다.
유지 관리	DMS-EVENT-0027	복제 인스턴스의 오프라인 유지 관리가 완료되었습니다. 이제 복제 인스턴스를 사용할 수 있습니다.
유지 관리	DMS-EVENT-0068	복제 인스턴스가 업그레이드할 수 없는 상태입니다.

범주	DMS 이벤트 ID	설명
LowStorage	DMS-EVENT-0007	복제 인스턴스가 할당된 스토리지의 90% 이상을 사용하였습니다. 여유 스토리지 공간 지표를 사용하여 복제 인스턴스에 대한 스토리지 공간을 모니터링할 수 있습니다.
장애 조치	DMS-EVENT-0013	다중 AZ 복제 인스턴스에서 장애 조치가 시작되었습니다.
장애 조치	DMS-EVENT-0049	다중 AZ 복제 인스턴스에서 장애 조치가 완료되었습니다.
장애 조치	DMS-EVENT-0015	스탠바이로의 다중 AZ 장애 조치가 완료되었습니다.
장애 조치	DMS-EVENT-0050	다중 AZ 활성화가 시작되었습니다.
장애 조치	DMS-EVENT-0051	다중 AZ 활성화가 완료되었습니다.
장애 조치	DMS-EVENT-0034	장애 조치를 너무 자주 요청하면 일반 장애 조치 이벤트 대신 이 이벤트가 발생합니다.
결함	DMS-EVENT-0031	복제 인스턴스에 스토리지 장애가 발생했습니다.
결함	DMS-EVENT-0036	호환되지 않는 네트워크로 인해 복제 인스턴스에 장애가 발생했습니다.
결함	DMS-EVENT-0037	서비스가 데이터 볼륨을 암호화하는 데 사용된 AWS KMS 키에 액세스할 수 없습니다.

다음 표에는 복제 작업 원본 형식에서 가능한 카테고리와 이벤트가 나와 있습니다.

범주	DMS 이벤트 ID	설명
상태 변경	DMS-EVENT-0069	복제 태스크가 시작되었습니다.
상태 변경	DMS-EVENT-0081	테이블 세부 정보 다시 로드가 요청되었습니다.

범주	DMS 이벤트 ID	설명
상태 변경	DMS-EVENT-0079	복제가 중지되었습니다.
상태 변경	DMS-EVENT-0091	읽기가 일시 중지되었습니다. 스왑 파일 한도에 도달했습니다.
상태 변경	DMS-EVENT-0092	읽기가 일시 중지되었습니다. 디스크 사용량 한도에 도달했습니다.
상태 변경	DMS-EVENT-0093	읽기가 재개되었습니다.
결함	DMS-EVENT-0078	복제 태스크가 실패했습니다.
결함	DMS-EVENT-0082	태스크 삭제 호출이 태스크 데이터를 정리하지 못했습니다.
구성 변경	DMS-EVENT-0080	복제 태스크가 수정되었습니다.
삭제	DMS-EVENT-0073	복제 태스크가 삭제되었습니다.
생성	DMS-EVENT-0074	복제 태스크가 생성되었습니다.

다음 예제는 상태 변경 카테고리의 AWS DMS 이벤트 구독을 보여줍니다.

```
        Resources:
            DMSEvent:
                Type: AWS::DMS::EventSubscription
                Properties:
                    Enabled: true
                    EventCategories: State Change
                    SnsTopicArn: arn:aws:sns:us-east-1:123456789:testSNS
                    SourceIds: []
                    SourceType: replication-task
```

SNS를 사용한 AWS DMS 이벤트 알림 구독

> **Important**
>
> AWS DMS 3.4.5 릴리스 및 이후 버전에서는 AWS DMS 이벤트 발생 시 Amazon EventBridge를 사용하여 알림을 제공하는 것이 좋습니다. AWS DMS에서 EventBridge 이벤트를 사용하는 방법에 대한 자세한 내용은 AWS Database Migration Service에서 Amazon EventBridge 이벤트 및 알림 사용 섹션을 참조하세요.

AWS DMS 이벤트 알림 구독을 생성하여 AWS DMS 이벤트가 발생하면 알림을 받을 수 있습니다. 가장 간단한 구독 생성 방법은 AWS DMS 콘솔을 이용하는 것입니다. 알림 구독에서는 알림을 보내는 방법 및 위치를 선택합니다. 알림 메시지를 받으려는 소스 유형을 지정합니다. 현재 AWS DMS는 복제 인스턴스와 복제 태스크 소스 유형을 지원합니다. 선택한 소스 유형에 따라 이벤트 알림 메시지를 수신하려는 이벤트 카테고리 및 소스를 선택합니다.

AWS Management Console 사용

> **Important**
>
> AWS DMS 3.4.5 릴리스 및 이후 버전에서는 AWS DMS 이벤트 발생 시 Amazon EventBridge를 사용하여 알림을 제공하는 것이 좋습니다. AWS DMS에서 EventBridge 이벤트를 사용하는 방법에 대한 자세한 내용은 AWS Database Migration Service에서 Amazon EventBridge 이벤트 및 알림 사용 섹션을 참조하세요.

콘솔을 사용하여 Amazon SNS를 통한 AWS DMS 이벤트 알림을 구독하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/dms/v2/에서 AWS DMS 콘솔을 엽니다.

   IAM 사용자로 로그인한 경우 AWS DMS에 액세스하기 위한 적절한 권한이 있어야 합니다.
2. 탐색 창에서 [이벤트 구독]을 선택합니다.
3. 이벤트 구독 페이지에서 이벤트 구독 생성을 선택합니다.
4. 이벤트 구독 생성 페이지에서 다음을 수행합니다.

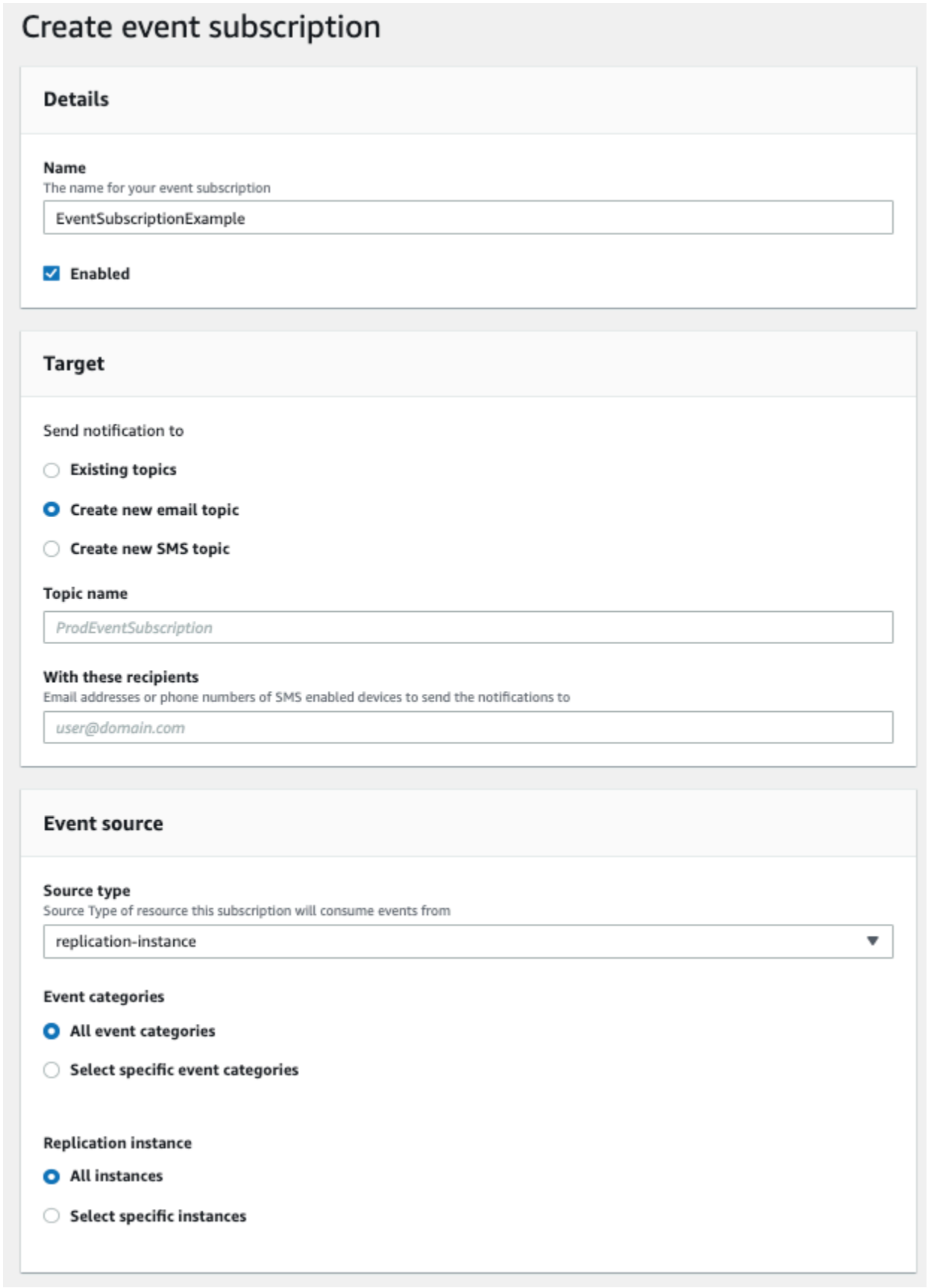
   a. 세부 정보의 이름에서 이벤트 알림 구독의 이름을 입력합니다.

b. 활성화됨을 선택하여 구독을 활성화합니다. 구독만 생성하고 알림 메시지를 아직 전송하지 않으려면 활성화됨을 선택하지 마세요.

c. 대상에서 기존 주제, 새 이메일 주제 생성 또는 새 SMS 주제 생성을 선택하여 알림을 전송합니다. 알림을 전송할 기존 Amazon SNS 주제가 있거나 주제를 생성해야 합니다. 주제를 생성하는 경우 알림이 전송될 이메일 주소를 입력할 수 있습니다.

d. 이벤트 소스의 소스 유형에서 소스 유형을 선택합니다. 옵션은 복제 인스턴스 및 복제 태스크뿐입니다.

e. 선택한 소스 유형에 따라 이벤트 알림 메시지를 수신하고자 하는 이벤트 카테고리와 소스를 선택합니다.


Create event subscription
Details
Name
The name for your event subscription
EventSubscriptionExample
Enabled
Target
Send notification to
Existing topics
Create new email topic
Create new SMS topic
Topic name
ProdEventSubscription
With these recipients
Email addresses or phone numbers of SMS enabled devices to send the notifications to
user@domain.com
Event source
Source type
Source Type of resource this subscription will consume events from
replication-instance
Event categories
All event categories
Select specific event categories
Replication instance
All instances
Select specific instances

f. 이벤트 구독 생성을 선택합니다.

AWS DMS 콘솔에 현재 구독 생성 중으로 나옵니다.

> **Note**
>
> 또한 AWS DMS API 및 CLI를 사용하여 Amazon SNS 이벤트 알림 구독을 생성할 수도 있습니다. 자세한 내용은 AWS DMS API 참조의 CreateEventSubscription 및 AWS DMS CLI 참조의 create-event-subscription을 참조하세요.

SNS 주제의 액세스 정책 검증

SNS 액세스 정책에는 AWS DMS가 SNS 주제에 이벤트를 게시하도록 허용하는 권한이 필요합니다. 다음 절차에 설명된 대로 액세스 정책을 검증하고 업데이트할 수 있습니다.

액세스 정책을 검증하려면

1. Amazon SNS 콘솔을 엽니다.
2. 탐색 패널에서 주제를 선택하고 DMS 알림을 받을 주제를 선택합니다.
3. 액세스 정책 탭을 선택합니다.

SNS 액세스 정책에서 AWS DMS가 SNS 주제에 이벤트를 게시하도록 허용하지 않는 경우 정책을 업데이트할 수 있습니다.

액세스 정책을 업데이트하려면

1. 주제 페이지의 세부 정보 섹션에서 편집을 선택합니다.
2. 액세스 정책 섹션을 확장하고 다음 정책을 JSON 편집기에 연결합니다.

```
{
      "Sid": "dms-allow-publish",
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sns:Publish",
      "Resource": "your-SNS-topic-ARN"
```

```
    }
```

이벤트를 주제에 게시하는 DMS EventSubscription Arn인 `aws:SourceArn` 조건을 지정하여 SNS 주제에 대한 액세스를 추가로 제한하는 것이 좋습니다.

```
...
"Resource": "your-SNS-topic-ARN"
"Condition": {
    "StringEquals": {
       "aws:SourceArn": "arn:partition:dms:your-AWS-region:your-AWS-account-
ID:es:your-dms-es-arn or *"
 }
```

3. 변경 사항 저장을 선택합니다.

AWS DMS 데이터 검증

주제

- 복제 작업 통계
- Amazon CloudWatch를 사용한 복제 태스크 통계
- 작업 중 테이블 다시 검증
- JSON 편집기를 사용하여 검증 규칙 수정
- 검증 전용 태스크
- 문제 해결
- Redshift 검증 성능
- 제한 사항
- Amazon S3 대상 데이터 검증

AWS DMS는 데이터가 원본에서 대상으로 정확히 마이그레이션되었는지 확인하기 위한 데이터 검증을 지원합니다. 활성화된 경우 테이블에 대해 전체 로드가 수행된 직후 검증이 시작됩니다. 검증은 CDC 지원 태스크에 대해 증분 변경 사항이 발생할 경우 이러한 변경 사항을 비교합니다.

데이터를 검증하는 동안 AWS DMS는 소스의 각 행과 대상의 해당 행을 비교한 후 이 행의 데이터가 동일한지 확인하고 불일치 항목이 있을 경우 해당 내용을 보고합니다. 이 작업을 수행하기 위해 AWS DMS는 적절한 쿼리를 실행하여 데이터를 검색합니다. 이러한 쿼리는 소스 및 대상의 추가 리소스와 추가 네트워크 리소스를 사용합니다.

검증이 활성화된 CDC 전용 태스크의 경우, 새 데이터의 검증을 시작하기 전에 테이블에 있는 기존 데이터를 모두 검증합니다.

데이터 검증은 AWS DMS가 소스 엔드포인트로서 다음과 같은 소스 데이터베이스를 지원할 때마다 해당 데이터베이스와 연동합니다.

- Oracle
- PostgreSQL 호환 데이터베이스(PostgreSQL, Aurora PostgreSQL 또는 Aurora Serverless for PostgreSQL)
- MySQL 호환 데이터베이스(MySQL, MariaDB, Aurora MySQL 또는 Aurora Serverless for MySQL)
- Microsoft SQL Server

- IBM Db2 LUW

데이터 검증은 AWS DMS가 대상 엔드포인트로서 다음 데이터베이스를 지원할 때마다 해당 데이터베이스와 연동합니다.

- Oracle
- PostgreSQL 호환 데이터베이스(PostgreSQL, Aurora PostgreSQL 또는 Aurora Serverless for PostgreSQL)
- MySQL 호환 데이터베이스(MySQL, MariaDB, Aurora MySQL 또는 Aurora Serverless for MySQL)
- Microsoft SQL Server
- IBM Db2 LUW
- Amazon Redshift
- Amazon S3. Amazon S3 대상 데이터 유효성 검사에 대한 자세한 내용은 Amazon S3 대상 데이터 검증 단원을 참조하세요.

지원되는 엔드포인트에 대한 자세한 내용은 AWS DMS 엔드포인트 작업 단원을 참조하십시오.

데이터 검증에는 마이그레이션 자체에 소요되는 시간 외에도 추가 시간이 요구됩니다. 필요한 추가 시간은 마이그레이션되는 데이터의 양에 따라 달라집니다.

이러한 설정에 대한 자세한 내용은 데이터 검증 작업 설정 섹션을 참조하세요.

JSON 파일의 `ValidationSettings` 태스크 설정에 대한 예제는 작업 설정 예제 섹션을 참조하세요.

복제 작업 통계

데이터 검증이 활성화되면 AWS DMS는 테이블 수준에서 다음 통계를 제공합니다.

- ValidationState - 테이블의 검증 상태입니다. 이 파라미터의 값은 다음과 같습니다.
  - 활성화되지 않음 - 마이그레이션 작업의 테이블에 대해 검증이 활성화되지 않습니다.
  - 보류 중인 레코드 - 테이블의 일부 레코드가 검증 대기 중입니다.
  - 일치하지 않는 레코드 - 테이블의 일부 레코드가 원본과 대상 간에 일치하지 않습니다. 불일치가 발생하는 이유에는 여러 가지가 있으므로 자세한 내용은 대상 엔드포인트의 `awsdms_control.awsdms_validation_failures_v1` 테이블을 참조하십시오.
  - 일시 중지된 레코드 - 테이블의 일부 레코드를 검증할 수 없습니다.
  - 프라이머리 키 없음 - 프라이머리 키가 없어 테이블을 검증할 수 없습니다.

  - 테이블 오류 - 테이블이 오류 상태이고 일부 데이터가 마이그레이션되지 않아 테이블이 검증되지 않았습니다.
  - 검증됨 - 테이블의 모든 행이 검증되었습니다. 테이블이 업데이트된 경우 상태가 Validated에서 변경할 수 있습니다.
  - 오류 - 예상치 못한 오류로 인해 테이블을 검증할 수 없습니다.
  - 검증 보류 중 - 테이블이 검증 대기 중입니다.
  - 테이블 준비 중 - 마이그레이션 태스크에서 활성화된 테이블을 검증하기 위해 준비 중입니다.
  - 재검증 보류 중 - 테이블이 업데이트된 후 테이블의 모든 행이 검증 보류 중입니다.
- ValidationPending - 대상에 마이그레이션되었지만 아직 검증되지 않은 레코드 수입니다.
- ValidationSuspended - AWS DMS에서 비교할 수 없는 레코드 수입니다. 예를 들어, 원본의 레코드를 지속적으로 업데이트하고 있는 경우 AWS DMS가 원본과 대상을 비교할 수 없습니다.
- ValidationFailed - 데이터 검증 단계를 통과하지 않은 레코드 수입니다.

JSON 파일의 `ValidationSettings` 태스크 설정에 대한 예제는 작업 설정 예제 섹션을 참조하세요.

콘솔, AWS CLI 또는 AWS DMS API를 사용하여 데이터 검증 정보를 볼 수 있습니다.

- 콘솔에서 작업을 생성하거나 수정할 때 작업을 검증하도록 선택할 수 있습니다. 콘솔을 사용하여 데이터 검증 보고서를 보려면 [Tasks] 페이지에서 작업을 선택하고 세부 정보 섹션에서 [Table statistics] 탭을 선택합니다.
- 데이터 검증을 시작하도록 작업을 생성하거나 수정하려는 경우 CLI를 사용하여 `EnableValidation` 파라미터를 `true`로 설정합니다. 다음 예제에서는 작업을 생성하고 데이터 검증을 활성화합니다.

```
create-replication-task
  --replication-task-settings '{"ValidationSettings":{"EnableValidation":true}}'
  --replication-instance-arn arn:aws:dms:us-east-1:5731014:
     rep:36KWVMB7Q
  --source-endpoint-arn arn:aws:dms:us-east-1:5731014:
     endpoint:CSZAEFQURFYMM
  --target-endpoint-arn arn:aws:dms:us-east-1:5731014:
     endpoint:CGPP7MF6WT4JQ
  --migration-type full-load-and-cdc
  --table-mappings '{"rules": [{"rule-type": "selection", "rule-id": "1",
     "rule-name": "1", "object-locator": {"schema-name": "data_types", "table-name":
 "%"},
     "rule-action": "include"}]}'
```

`describe-table-statistics` 명령을 사용하여 JSON 형식의 데이터 검증 보고서를 받습니다. 다음 명령은 데이터 검증 보고서를 표시합니다.

```
aws dms  describe-table-statistics --replication-task-arn arn:aws:dms:us-
east-1:5731014:
rep:36KWVMB7Q
```

이 보고서는 다음과 비슷합니다.

```
{
    "ReplicationTaskArn": "arn:aws:dms:us-west-2:5731014:task:VFPFTYKK2RYSI",
    "TableStatistics": [
        {
            "ValidationPendingRecords": 2,
            "Inserts": 25,
            "ValidationState": "Pending records",
            "ValidationSuspendedRecords": 0,
            "LastUpdateTime": 1510181065.349,
            "FullLoadErrorRows": 0,
            "FullLoadCondtnlChkFailedRows": 0,
            "Ddls": 0,
            "TableName": "t_binary",
            "ValidationFailedRecords": 0,
            "Updates": 0,
            "FullLoadRows": 10,
            "TableState": "Table completed",
            "SchemaName": "d_types_s_sqlserver",
            "Deletes": 0
        }
}
```

- AWS DMS API에서 CreateReplicationTask 작업을 사용하여 태스크 하나를 생성하고, `EnableValidation` 파라미터를 true로 설정하여 태스크를 통해 마이그레이션되는 데이터를 검증합니다. [DescribeTableStatistics] 작업을 사용하여 JSON 형식의 데이터 검증 보고서를 받습니다.

Amazon CloudWatch를 사용한 복제 태스크 통계

Amazon CloudWatch가 활성화되면 AWS DMS에서는 다음과 같은 복제 태스크 통계를 제공합니다.

- ValidationSucceededRecordCount - AWS DMS에서 검증한 분당 행 수입니다.

- ValidationAttemptedRecordCount - 검증을 시도한 분당 행 수입니다.
- ValidationFailedOverallCount - 검증이 실패한 행 수입니다.
- ValidationSuspendedOverallCount - 검증이 일시 중지된 행 수입니다.
- ValidationPendingOverallCount - 검증이 보류 중인 행 수입니다.
- ValidationBulkQuerySourceLatency - AWS DMS는 변경 사항이 많을 경우, 전체 복제 또는 지속적 복제 동안 특정 시나리오를 중심으로 일괄적으로 데이터를 검증할 수 있습니다. 이 지표는 소스 엔드포인트 대량의 데이터 세트를 읽는 데 필요한 지연 시간을 나타냅니다.
- ValidationBulkQueryTargetLatency - AWS DMS는 변경 사항이 많을 경우, 전체 복제 또는 지속적 복제 동안 특정 시나리오를 중심으로 일괄적으로 데이터를 검증할 수 있습니다. 이 지표는 대상 엔드포인트의 대량의 데이터 세트를 읽는 데 필요한 지연 시간을 나타냅니다.
- ValidationItemQuerySourceLatency - 지속적 복제 동안 데이터 검증은 지속적 변경 사항을 식별하고 이러한 변경 사항을 검증합니다. 이 지표는 소스에서 이런 변경을 읽는 데 필요한 지연 시간을 나타냅니다. 검증은 변경의 수를 근거로, 검증 동안 오류가 있을 시 필요한 것보다 더 많은 쿼리를 실행할 수 있습니다.
- ValidationItemQueryTargetLatency - 지속적 복제 동안 데이터 검증은 지속적 변경 사항을 식별하고 행 단위로 변경 사항을 검증할 수 있습니다. 이 지표는 대상에서 이런 변경을 읽는 데 필요한 지연 시간을 제공합니다. 검증은 변경의 수를 근거로, 검증 동안 오류가 있을 시 필요한 것보다 더 많은 쿼리를 실행할 수도 있습니다.

CloudWatch를 사용한 통계에서 데이터 검증 정보를 수집하려면 콘솔을 사용하여 태스크를 생성하거나 수정할 때 CloudWatch 로그 활성화를 선택합니다. 그런 다음, 데이터가 소스에서 대상으로 정확히 마이그레이션되었는지 확인하기 위한 데이터 검증 정보를 확인하려면 다음 작업을 수행합니다.

1. 데이터베이스 마이그레이션 태스크 페이지에서 태스크를 선택합니다.
2. CloudWatch 지표 탭을 선택합니다.
3. 드롭다운 메뉴에서 검증을 선택합니다.

작업 중 테이블 다시 검증

작업 실행 중에 AWS DMS를 요청하여 데이터 검증을 수행할 수 있습니다.

AWS Management Console

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/dms/v2/에서 AWS DMS 콘솔을 엽니다.

AWS Identity and Access Management(IAM) 사용자로 로그인한 경우, AWS DMS에 액세스하기 위한 적절한 권한이 있어야 합니다. 필요한 권한에 대한 내용은 AWS DMS 사용에 필요한 IAM 권한 섹션을 참조하세요.

2. 탐색 창에서 [작업]을 선택합니다.
3. 다시 검증할 테이블이 있는 실행 중인 작업을 선택합니다.
4. 테이블 통계 탭을 선택합니다.
5. 다시 검증할 테이블을 선택합니다(한 번에 최대 10개의 테이블을 선택할 수 있음). 작업이 더 이상 실행되고 있지 않으면 테이블을 다시 검증할 수 없습니다.
6. Revalidate(다시 검증)를 선택합니다.

JSON 편집기를 사용하여 검증 규칙 수정

AWS DMS 콘솔에서 JSON 편집기를 사용하여 태스크에 검증 규칙을 추가하려면 다음 작업을 수행합니다.

1. 데이터베이스 마이그레이션 태스크를 선택합니다.
2. 마이그레이션 태스크 목록에서 태스크를 선택합니다.
3. 태스크가 실행 중인 경우 작업 드롭다운 메뉴에서 중지를 선택합니다.
4. 태스크가 중지된 후 태스크를 수정하려면 작업 드롭다운 메뉴에서 수정을 선택합니다.
5. 테이블 매핑 섹션에서 JSON 편집기를 선택하고 테이블 매핑에 검증 규칙을 추가합니다.

예를 들어, 다음과 같은 검증 규칙을 추가하여 소스에서 replace 함수를 실행할 수 있습니다. 이 경우 검증 규칙에서 null 바이트가 발견되면 해당 바이트는 공백으로 확인됩니다.

```
{
 "rule-type": "validation",
 "rule-id": "1",
 "rule-name": "1",
 "rule-target": "column",
 "object-locator": {
  "schema-name": "Test-Schema",
  "table-name": "Test-Table",
  "column-name": "Test-Column"
 },
 "rule-action": "override-validation-function",
```

```
 "source-function": "REPLACE(${column-name}, chr(0), chr(32))",
 "target-function": "${column-name}"
}
```

검증 전용 태스크

마이그레이션이나 데이터 복제를 수행하지 않고도 데이터를 미리 보고 검증하는 검증 전용 태스크를 생성할 수 있습니다. 검증 전용 태스크를 생성하려면 `EnableValidation` 및 `ValidationOnly` 설정을 `true`로 설정합니다. `ValidationOnly`를 활성화하면 추가 요구 사항이 적용됩니다. 자세한 내용은 데이터 검증 작업 설정 섹션을 참조하세요.

전체 로드 전용 마이그레이션 유형의 경우, 실패가 많이 보고되면 검증 전용 태스크가 CDC 태스크보다 훨씬 더 빨리 완료됩니다. 그러나 소스 또는 대상 엔드포인트에 대한 변경 사항은 전체 로드 모드에서 실패로 보고되며, 이는 단점이 될 수 있습니다.

CDC 검증 전용 태스크는 평균 지연 시간을 기준으로 검증을 지연시키며, 실패를 보고하기 전에 여러 번 재시도합니다. 대부분의 데이터 비교 결과가 실패로 이어질 경우, CDC 모드의 검증 전용 태스크는 속도가 매우 느리며, 이는 단점이 될 수 있습니다.

검증 전용 태스크는 복제 태스크와 같은 방향으로 설정해야 하며, 특히 CDC의 경우에는 더욱 그렇습니다. CDC 검증 전용 태스크는 소스의 변경 로그를 기준으로, 어떤 행이 변경되었고 재검증해야 하는지 감지하기 때문입니다. 대상이 소스로 지정된 경우, DMS에서 대상으로 전송한 변경 사항만 인식하며 복제 오류를 포착하지 못할 수 있습니다.

전체 로드 검증 전용

AWS DMS 3.4.6 이상 버전부터 전체 로드 검증 전용 태스크는 소스 및 대상 테이블의 모든 행을 단일 패스로 빠르게 비교하고, 오류가 발생하면 즉시 보고한 다음 종료합니다. 이 모드에서는 실패로 인해 검증이 일시 중지되지 않으며, 속도에 최적화되어 있습니다. 그러나 소스 또는 대상 엔드포인트에 대한 변경 사항은 실패로 보고됩니다.

> **Note**
>
> AWS DMS 3.4.6 이상 버전부터 이 검증 동작은 검증이 활성화된 전체 로드 마이그레이션 태스크에도 적용됩니다.

CDC 검증 전용

CDC 검증 전용 태스크는 새로 시작할 때 소스 테이블과 대상 테이블 사이에 있는 모든 기존 행을 검증합니다. 또한 CDC 검증 전용 태스크는 지속적으로 실행되며, 지속적 복제의 변경 사항을 재검증하고, 각 패스에서 보고된 실패 횟수를 제한하고, 불일치 행이 실패하기 전에 다시 시도합니다. 이는 거짓 긍정을 방지하도록 최적화되었습니다.

`FailureMaxCount` 또는 `TableFailureMaxCount` 임계값을 위반하면 테이블(또는 전체 태스크)에 대한 검증이 일시 중지됩니다. 이는 검증이 활성화된 CDC 또는 전체 로드+CDC 마이그레이션 태스크에도 적용됩니다. 그리고 검증이 활성화된 CDC 태스크는 평균 소스 및 대상 지연 시간을 기준으로 변경된 각 행에 대한 재검증을 지연시킵니다.

하지만 CDC 검증 전용 태스크는 데이터를 마이그레이션하지 않으며 지연 시간도 발생하지 않습니다. 기본적으로 `ValidationQueryCdcDelaySeconds`는 180으로 설정됩니다. 그리고 지연 시간이 긴 환경을 고려하여 용량을 늘리고 거짓 긍정을 방지할 수 있습니다.

검증 전용 사용 사례

마이그레이션 또는 복제 태스크의 데이터 검증 부분을 별도의 검증 전용 태스크로 분할하는 사용 사례에는 다음 경우가 포함되나, 이에 국한되지 않습니다.

- 검증이 수행되는 시기를 정확하게 제어 - 검증 쿼리는 소스 및 대상 엔드포인트 양쪽 모두에 추가적인 부하를 가중시킵니다. 따라서 먼저 한 태스크에서 데이터를 마이그레이션 또는 복제한 다음, 다른 태스크에서 결과를 검증하는 편이 유리할 수 있습니다.
- 복제 인스턴스의 부하 감소 - 데이터 검증을 분할하여 자체 인스턴스에서 실행하는 편이 유리할 수 있습니다.
- 특정 시점에 일치하지 않는 행의 수를 신속하게 파악 - 예를 들어, 유지 관리 기간 직전에 또는 도중에 프로덕션이 대상 엔드포인트로 전환될 경우 전체 로드 검증 전용 태스크를 생성하여 질문에 대한 답을 얻을 수 있습니다.
- CDC 구성 요소를 사용한 마이그레이션 태스크에서 검증 실패가 예상될 경우 - 예를 들어, Oracle `varchar2`를 PostgreSQL `jsonb`으로 마이그레이션할 경우 CDC 검증은 이러한 실패한 행을 계속 재시도하고 매번 보고되는 실패 횟수를 제한합니다. 하지만 전체 로드 검증 전용 태스크를 생성하면 더 빠른 답을 얻을 수 있습니다.
- 검증 실패 테이블을 읽는 데이터 복구 스크립트/유틸리티를 개발한 경우 - (문제 해결도 참조하세요). 전체 로드 검증 태스크는 데이터 복구 스크립트가 조치를 취해야 할 오류를 신속하게 보고합니다.

JSON 파일의 `ValidationSettings` 태스크 설정에 대한 예제는 작업 설정 예제 섹션을 참조하세요.

문제 해결

검증하는 동안 AWS DMS는 대상 엔드포인트에서 새 테이블인 `awsdms_control.awsdms_validation_failures_v1`을 생성합니다. 레코드 상태가 ValidationSuspended 또는 ValidationFailed가 되면 AWS DMS는 `awsdms_control.awsdms_validation_failures_v1`에 진단 정보를 씁니다. 이 테이블을 쿼리하여 검증 오류 문제를 해결할 수 있습니다.

대상에서 테이블이 생성되는 기본 스키마를 변경하는 방법에 대한 내용은 제어 테이블 태스크 설정 섹션을 참조하세요.

다음은 `awsdms_control.awsdms_validation_failures_v1` 테이블에 대한 설명입니다.

열 명칭	데이터 유형	Description
TASK_NAME	VARCHAR(128) NOT NULL	AWS DMS 작업 식별자
TABLE_OWNER	VARCHAR(128) NOT NULL	테이블의 스키마(소유자).
TABLE_NAME	VARCHAR(128) NOT NULL	테이블 이름.
FAILURE_TIME	DATETIME(3) NOT NULL	실패가 발생한 시간.
KEY_TYPE	VARCHAR(128) NOT NULL	나중에 사용할 수 있도록 예약됩니다(값은 항상 'Row').
KEY	TEXT NOT NULL	이 키는 행 레코드 유형의 기본 키입니다.
FAILURE_TYPE	VARCHAR(128) NOT NULL	검증 오류의 심각도. RECORD_DIFF , MISSING_SOURCE 또는 MISSING_TARGET 입니다.
DETAILS	VARCHAR(8000) NOT NULL	지정된 키와 일치하지 않는 모든 소스/대상 열 값의 JSON 형식 문자열입니다.

다음 쿼리는 awsdms_control.awsdms_validation_failures_v1 테이블을 쿼리하여 작업에 대한 모든 실패를 표시합니다. 작업 이름은 작업의 외부 리소스 ID여야 합니다. 작업의 외부 리소스 ID는 작업 ARN의 마지막 값입니다. 예를 들어, ARN 값이 arn:aws:dms:us-west-2:5599:task:VFPFKH4FJR3FTYKK2RYSI인 작업의 경우, 작업의 외부 리소스 ID는 VFPFKH4FJR3FTYKK2RYSI입니다.

```
select * from awsdms_validation_failures_v1 where TASK_NAME = 'VFPFKH4FJR3FTYKK2RYSI'

TASK_NAME       VFPFKH4FJR3FTYKK2RYSI
TABLE_OWNER     DB2PERF
TABLE_NAME      PERFTEST
FAILURE_TIME    2020-06-11 21:58:44
KEY_TYPE        Row
KEY             {"key":  ["3451491"]}
FAILURE_TYPE    RECORD_DIFF
DETAILS         [[{'MYREAL': '+1.10106036e-01'}, {'MYREAL': '+1.10106044e-01'}],]
```

DETAILS 필드를 보고 일치하지 않는 열을 확인할 수 있습니다. 실패한 레코드의 프라이머리 키가 있으므로, 소스 및 대상 엔드포인트를 쿼리하여 레코드에서 일치하지 않는 부분을 찾아낼 수 있습니다.

Redshift 검증 성능

Amazon Redshift는 열 기반 스토리지, MPP, 데이터 압축 및 기타 요소 등 여러 가지 면에서 관계형 데이터베이스와 다릅니다. 이러한 차이로 인해 Redshift는 관계형 데이터베이스와는 다른 성능 프로파일을 제공합니다.

전체 로드 복제 단계 중 검증에서는 PartitionSize 설정에 따라 데이터 크기가 제어되는 범위 쿼리를 사용합니다. 이러한 범위 기반 쿼리에서는 소스 테이블의 모든 레코드가 선택됩니다.

지속적인 복제의 경우 쿼리는 범위 기반과 개별 레코드 가져오기 간에 전환됩니다. 쿼리 유형은 다음과 같은 여러 요인에 따라 동적으로 결정됩니다.

- 쿼리 볼륨
- 소스 테이블의 DML 쿼리 유형
- 작업 지연 시간
- 총 레코드 수

- PartitionSize 등의 검증 설정

검증 쿼리로 인해 Amazon Redshift 클러스터에 추가 로드가 발생할 수 있습니다. 위의 요인은 사용 사례에 따라 다르므로 검증 쿼리 성능을 검토하고 그에 따라 클러스터와 테이블을 조정해야 합니다. 성능 문제를 완화하기 위한 몇 가지 옵션은 다음과 같습니다.

- PartitionSize 및 ThreadCount 설정을 줄이면 전체 로드 검증 중에 워크로드를 줄이는 데 도움이 됩니다. 이렇게 하면 데이터 검증 속도가 느려진다는 점에 유의하세요.
- Redshift는 프라이머리 키를 적용하지 않지만 AWS DMS는 데이터 검증을 위해 프라이머리 키를 사용하여 대상의 레코드를 고유하게 식별합니다. 전체 로드 검증 쿼리가 더 빠르게 실행되기 위해 가능하면 프라이머리 키가 정렬 키를 미러링하도록 설정합니다.

제한 사항

- 데이터를 검증하려면 테이블에 기본 키나 고유한 인덱스가 있어야 합니다.
  - 기본 키 열은 CLOB, BLOB 또는 BYTE 형식이 아니어야 합니다.
  - VARCHAR 또는 CHAR 형식의 기본 키 열은 길이가 1024보다 작아야 합니다. 데이터 유형에 길이를 지정해야 합니다. 무제한 데이터 유형을 데이터 검증의 프라이머리 키로 사용할 수 없습니다.
  - NOVALIDATE 절을 사용하여 생성된 Oracle 키는 프라이머리 키 또는 고유 인덱스로 간주되지 않습니다.
  - 프라이머리 키가 없고 고유 키만 있는 Oracle 테이블의 경우에는 고유 제약 조건이 있는 열에도 NOT NULL 제약 조건이 있어야 합니다.
- NULL PK/UK 값의 검증은 지원되지 않습니다.
- 대상 PostgreSQL 인스턴스의 기본 키 열의 콜레이션이 "C"로 설정되어 있지 않으면 기본 키의 정렬 순서가 Oracle의 정렬 순서와 달라집니다. PostgreSQL과 Oracle의 정렬 순서가 다르면 데이터 검증을 통해 레코드를 검증할 수 없습니다.
- 데이터 검증을 수행하면 원본 및 대상 데이터베이스에 대해 추가 쿼리가 생성됩니다. 두 데이터베이스에 이 추가 로드를 처리할 수 있을 만큼 충분한 리소스가 확보되었는지 확인해야 합니다. Redshift 대상의 경우 특히 그렇습니다. 자세한 내용은 Redshift 검증 성능 단원을 참조하십시오.
- 여러 데이터베이스를 하나로 통합할 때는 데이터 검증이 지원되지 않습니다.
- 원본 또는 대상 Oracle 엔드포인트의 경우 AWS DMS에서는 DBMS_CRYPTO를 사용하여 LOB를 검증합니다. Oracle 엔드포인트에 LOB가 사용되는 경우 Oracle 엔드포인트에 액세스하는 데 사용되는 사용자 계정에 dbms_crypto에 대한 실행 권한을 부여해야 합니다. 다음 명령문을 실행하여 이 권한을 호출할 수 있습니다.

```
grant execute on sys.dbms_crypto to dms_endpoint_user;
```

- 검증하는 동안 대상 데이터베이스를 AWS DMS 외부에서 수정하는 경우 불일치 사항이 정확히 보고되지 않을 수도 있습니다. AWS DMS가 동일한 테이블을 검증하는 동안 애플리케이션 중 하나가 데이터를 대상 테이블에 쓰는 경우에 이 결과가 발생할 수 있습니다.
- 검증하는 동안 하나 이상의 행이 지속적으로 수정되면 AWS DMS가 이 행을 검증할 수 없습니다.
- AWS DMS에서 10,000개가 넘는 실패 또는 일시 중지된 레코드를 검색하면 검증이 중지됩니다. 계속 진행하기 전에 데이터에 잠재된 문제를 해결하십시오.
- AWS DMS는 보기의 데이터 검증을 지원하지 않습니다.
- AWS DMS는 문자 대체 태스크 설정을 사용할 경우 데이터 검증을 지원하지 않습니다.
- AWS DMS는 Oracle LONG 유형의 검증을 지원하지 않습니다.
- AWS DMS는 이기종 마이그레이션 중에는 Oracle Spatial 유형의 검증을 지원하지 않습니다.

S3 대상 검증 사용에 대한 제한 사항은 S3 대상 검증 사용에 대한 제한 사항 섹션을 참조하세요.

Amazon S3 대상 데이터 검증

AWS DMS는 Amazon S3 대상에서 복제된 데이터의 검증을 지원합니다. AWS DMS는 Amazon S3에 복제된 데이터를 플랫 파일로 저장하므로, Amazon Athena CREATE TABLE AS SELECT(CTAS) 쿼리를 사용하여 데이터를 검증합니다.

Amazon S3에 저장된 데이터에 대한 쿼리는 컴퓨팅 집약적입니다. 따라서 AWS DMS는 변경 데이터 캡처(CDC) 중 하루에 한 번 (00:00) UTC에 Amazon S3 데이터에 대한 검증을 실행합니다. AWS DMS가 매일 실행하는 각각의 검증을 간격 검증이라고 합니다. 간격 검증 과정에서 AWS DMS는 지난 24시간 동안 대상 Amazon S3 버킷으로 마이그레이션된 모든 변경 레코드를 검증합니다. 간격 검증의 제한 사항에 대한 자세한 내용은 S3 대상 검증 사용에 대한 제한 사항 섹션을 참조하세요.

Amazon S3 대상 검증은 Amazon Athena를 사용하므로 추가 비용이 적용됩니다. 자세한 내용은 Amazon Athena 요금을 참조하세요.

> **Note**
>
> S3 대상 검증을 사용하려면 AWS DMS 버전 3.5.0 이상이 필요합니다.

주제

- S3 대상 검증 사전 조건
- S3 대상 검증을 사용하기 위한 권한
- S3 대상 검증 사용에 대한 제한 사항
- S3 대상 검증과 함께 검증 전용 태스크 사용

S3 대상 검증 사전 조건

S3 대상 검증을 사용하기 전에 다음과 같은 설정 및 권한을 확인합니다.

- 엔드포인트의 S3Settings에 대한 `DataFormat` 값을 `parquet`로 설정합니다. 자세한 설명은 S3의 Parquet 설정 섹션을 참조하세요.
- 마이그레이션 태스크를 생성하는 데 사용된 사용자 계정에 할당된 역할은 올바른 권한 집합을 보유하고 있어야 합니다. 아래의 권한 섹션을 참조하세요.

지속적 복제(CDC)를 사용하는 태스크의 경우 다음 설정을 확인하세요.

- CDC 데이터에 전체 레코드를 남기려면 보충 로깅을 활성화합니다. 보충 로깅 활성화에 대한 내용은 이 가이드의 문제 해결 및 진단 지원 섹션에서 보충 로깅을 Oracle 소스 엔드포인트에 자동으로 추가합니다. 부분을 참조하세요.
- 대상 엔드포인트의 `TimestampColumnName` 파라미터를 설정합니다. 타임스탬프 열 이름에는 제한 사항이 없습니다. 자세한 내용은 S3Settings 섹션을 참조하세요.
- 대상의 날짜 기반 폴더 분할을 설정합니다. 자세한 설명은 날짜 기반 폴더 파티셔닝 사용 섹션을 참조하세요.

S3 대상 검증을 사용하기 위한 권한

S3 대상 검증을 위한 액세스 권한을 설정하려면 마이그레이션 작업 생성에 사용된 사용자 계정에 할당된 역할은 다음과 같은 권한 집합을 보유하고 있어야 합니다. 샘플 값을 내가 생각한 값으로 바꿉니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "athena:StartQueryExecution",
```

```
                "athena:GetQueryExecution",
                "athena:CreateWorkGroup"
            ],
            "Resource": "arn:aws:athena:<endpoint_region_code>:<account_id>:workgroup/
dms_validation_workgroup_for_task_*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateDatabase",
                "glue:DeleteDatabase",
                "glue:GetDatabase",
                "glue:GetTables",
                "glue:CreateTable",
                "glue:DeleteTable",
                "glue:GetTable"
            ],
            "Resource": [
                "arn:aws:glue:<endpoint_region_code>:<account_id>:catalog",
                "arn:aws:glue:<endpoint_region_code>:<account_id>:database/
aws_dms_s3_validation_*",
                "arn:aws:glue:<endpoint_region_code>:<account_id>:table/
aws_dms_s3_validation_*/*",
                "arn:aws:glue:<endpoint_region_code>:<account_id>:userDefinedFunction/
aws_dms_s3_validation_*/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucketMultipartUploads",
                "s3:AbortMultipartUpload",
                "s3:ListMultipartUploadParts"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket_name>",
                "arn:aws:s3:::<bucket_name>/*"
            ]
        }
    ]
}
```

S3 대상 검증 사용에 대한 제한 사항

S3 대상 검증을 사용할 때 적용되는 아래의 추가적인 제한 사항을 확인합니다. 모든 검증에 적용되는 제한 사항은 제한 사항 섹션을 참조하세요.

- `DatePartitionSequence` 값에는 '일' 구성 요소가 필요합니다. S3 대상 검증은 `YYYYMM` 형식을 지원하지 않습니다.
- CDC 중에 간격 검증을 실행하면 `awsdms_validation_failures_v1` 표에 거짓 검증 오류가 표시될 수 있습니다. 이러한 오류가 발생하는 이유는 AWS DMS가 간격 검증을 진행하는 중에 도착한 변경 사항을 익일 파티션 폴더로 마이그레이션하기 때문입니다. 일반적으로 이러한 변경 사항은 당일의 파티션 폴더에 작성됩니다. 이러한 거짓 오류는 동적 소스 데이터베이스에서 정적 대상(예: Amazon S3)으로 복제된 항목을 검증하는 과정에서 발생하는 제한 사항입니다. 이러한 거짓 오류를 조사하려면 이런 오류가 일반적으로 나타나는 검증 기간이 종료되는 시점에(00:00 UTC) 레코드를 확인합니다.

  거짓 오류의 수를 최소화하려면 태스크에 대한 `CDCLatencySource`가 낮아야 합니다. 모니터링 지연 시간에 대한 내용은 복제 작업 지표 섹션을 참조하세요.
- `failed` 또는 `stopped` 상태의 태스크는 전날의 변경 사항을 검증하지 않습니다. 예상치 못한 실패로 인한 검증 오류를 최소화하려면 동일한 테이블 매핑, 소스 및 대상 엔드포인트가 포함된 별도의 검증 전용 태스크를 생성합니다. 검증 전용 태스크에 대한 자세한 내용은 S3 대상 검증과 함께 검증 전용 태스크 사용 섹션을 참조하세요.
- 테이블 통계의 검증 상태 열에는 가장 최근의 간격 검증 상태가 반영됩니다. 따라서 불일치 항목이 있는 테이블은 다음 날 간격 검증이 끝난 후에 검증된 것으로 표시될 수 있습니다. 대상 Amazon S3 버킷의 `s3_validation_failures folder`를 검사하여 하루 이상 전에 발생한 불일치 항목이 있는지 확인합니다.
- S3 검증은 Amazon Athena의 버킷 테이블 기능을 사용합니다. 이를 통해 S3 검증에서 대상 테이블 데이터의 버킷 사본을 만들 수 있습니다. 즉, 테이블 데이터의 복사본은 DMS 검증의 내부 파티셔닝과 일치하는 하위 집합으로 분할됩니다. Athena 버킷 테이블의 최대 버킷 수는 100,000개입니다. S3 검증이 시도하는 테이블 중 이 한도를 초과하는 모든 테이블은 검증에 실패합니다. S3 검증이 생성하려고 시도하는 버킷 수는 다음과 같습니다.

```
(#records in the table) / (validation partition size setting)
```

이 제한을 해결하려면 S3 Validation에서 생성되는 버킷 수가 100,000개 미만이 되도록 검증 파티션 크기 설정을 늘리십시오. 버킷팅에 대한 자세한 내용은 Amazon Athena 사용 설명서의 Athena에서의 파티셔닝 및 버켓팅을 참조하십시오.

S3 대상 검증과 함께 검증 전용 태스크 사용

검증 전용 태스크는 마이그레이션을 실행하지 않고 마이그레이션할 데이터에 대한 검증을 실행합니다.

마이그레이션 태스크가 중지되더라도, 검증 전용 태스크는 계속 실행되므로 AWS DMS는 00:00 UTC 간격 검증 기간을 누락하지 않습니다.

Amazon S3 대상 엔드포인트에서 검증 전용 태스크를 사용할 경우 다음과 같은 제한 사항이 적용됩니다.

- 검증 전용 설정이 활성화된 전체 로드 태스크에 대해 Amazon S3 검증이 지원되지만, 다른 엔드포인트의 전체 로드 및 검증 전용 태스크와는 다르게 작동합니다. S3를 대상으로 하는 경우, 이 유형의 태스크는 S3 대상의 전체 로드 데이터에 대해서만 검증을 수행하며 CDC 마이그레이션의 일부로 마이그레이션된 데이터에 대해서는 검증을 수행하지 않습니다. 이 기능은 전체 로드 전용 태스크로 생성된 데이터를 검증할 때만 사용하세요. 이 모드를 사용하여 활성 CDC 태스크가 실행 중인 대상의 데이터를 검증하면 효과적인 검증이 이루어지지 않습니다.
- 검증 전용 태스크는 마지막 간격 검증 기간(00:00 UTC) 이후의 변경 사항만 검증합니다. 검증 전용 태스크는 전날의 전체 로드 데이터 또는 CDC 데이터를 검증하지 않습니다.

AWS Database Migration Service의 리소스에 태그 지정

AWS Database Migration Service(AWS DMS)에서 태그를 사용하여 리소스에 메타데이터를 추가할 수 있습니다. 또한 AWS Identity and Access Management(IAM) 정책과 함께 이러한 태그를 사용하여 AWS DMS 리소스에 대한 액세스를 관리하고 AWS DMS 리소스에 적용 가능한 작업을 제어할 수 있습니다. 마지막으로 비슷하게 태그가 지정된 리소스에 대한 비용을 그룹화하여 이러한 태그로 비용을 추적할 수 있습니다.

모든 AWS DMS 리소스에 태그를 지정할 수 있습니다.

- 인증서
- 데이터 공급자
- 데이터 마이그레이션
- 엔드포인트
- 이벤트 구독
- 인스턴스 프로파일
- 마이그레이션 프로젝트
- 복제 인스턴스
- 복제 서브넷 그룹
- 복제 작업

AWS DMS 태그는 사용자가 정의하고 AWS DMS 리소스와 연결하는 이름-값 페어입니다. 이 이름을 키라고 합니다. 키 값을 제공하는 것은 선택 사항입니다. 태그를 사용하여 AWS DMS 리소스에 임의의 정보를 배정할 수 있습니다. 범주 정의 등에 태그 키를 사용할 수 있으며 태그 값은 해당 범주의 항목일 수 있습니다. 예를 들어, 태그 키를 'project'로 정의하고 태그 값을 'Salix'로 정의하여 AWS DMS 리소스가 Salix 프로젝트에 할당됨을 나타냅니다. environment=test나 environment=production 등의 키를 사용하여 태그로 AWS DMS 리소스를 테스트나 프로덕션에 사용되는 것으로 지정할 수도 있습니다. AWS DMS 리소스와 연결된 메타데이터를 더 쉽게 추적할 수 있게 일관성 있는 태그 키 세트를 사용하는 것이 좋습니다.

태그를 사용하여 비용 구조를 반영하도록 AWS 청구서를 구성합니다. 이렇게 하려면 가입하여 태그 키 값이 포함된 AWS 계정 청구서를 가져옵니다. 그런 다음 같은 태그 키 값을 가진 리소스에 따라 결제 정보를 구성하여 리소스 비용의 합을 볼 수 있습니다. 예를 들어, 특정 애플리케이션 이름으로 여러 리소스에 태그를 지정한 다음 결제 정보를 구성하여 여러 서비스에 걸친 해당 애플리케이션의 총 비용을 볼 수 있습니다. 자세한 내용은 AWS Billing 사용 설명서의 비용 할당 태그 사용을 참조하십시오.

각 AWS DMS 리소스에는 해당 AWS DMS 리소스에 배정되는 모든 태그를 포함하는 태그 세트가 있습니다. 태그 세트는 최대 10개의 태그를 포함하거나 비어 있을 수 있습니다. AWS DMS 리소스의 기존 태그와 동일한 키를 갖는 태그를 리소스에 추가하면 새 값이 이전 값을 덮어씁니다.

AWS는 태그에 의미론적 의미를 적용하지 않으며 태그는 엄격히 문자열로 해석됩니다.AWS DMS는 리소스를 생성할 때 사용하는 설정에 따라, AWS DMS 리소스에 태그를 설정할 수 있습니다.

다음 목록에서는 AWS DMS 태그의 특징을 설명합니다.

- 태그 키는 태그의 필수 이름입니다. 문자열 값은 길이가 1~128자(유니코드 문자)이며 "aws:" 또는 "dms:"로 시작할 수 없습니다. 문자열에는 유니코드 문자, 숫자, 공백, '_', '.', '/', '=', '+', '-'(Java regex: "^([\\p{L}\\p{Z}\\p{N}_.:/=+\\-]*)$")만 포함될 수 있습니다.

- 태그 값은 태그의 선택적 문자열 값입니다. 문자열 값은 길이가 1~256자(유니코드 문자)이며 "aws:" 또는 "dms:"로 시작할 수 없습니다. 문자열에는 유니코드 문자, 숫자, 공백, '_', '.', '/', '=', '+', '-'(Java regex: "^([\\p{L}\\p{Z}\\p{N}_.:/=+\\-]*)$")만 포함될 수 있습니다.

  값은 태그 세트에서 고유할 필요는 없으며 null일 수 있습니다. 예를 들어, project/Trinity 및 cost-center/Trinity의 태그 세트에 키-값 페어가 있을 수 있습니다.

AWS CLI 또는 AWS DMS API를 사용하여 AWS DMS 리소스에서 태그를 추가, 나열, 삭제할 수 있습니다. AWS CLI 또는 AWS DMS API를 사용할 때는 작업하려는 AWS DMS 리소스에 대한 Amazon 리소스 이름(ARN)을 제공해야 합니다. ARN 생성에 대한 자세한 내용은 에 대한 아마존 리소스 이름 (ARN) 생성 AWS DMS 주제단원을 참조하십시오.

권한 부여 목적으로 태그가 캐시됩니다. 이 때문에 AWS DMS 리소스의 태그에 대한 추가나 업데이트가 제공되는 데 몇 분 정도 걸릴 수 있습니다.

API

AWS DMS API를 사용한 AWS DMS 리소스에 대한 태그를 추가, 나열 또는 제거할 수 있습니다.

- AWS DMS 리소스에 태그를 추가하려면 `AddTagsToResource` 작업을 사용합니다.
- AWS DMS 리소스에 할당된 태그를 나열하려면 `ListTagsForResource` 작업을 사용합니다.
- AWS DMS 리소스에서 태그를 제거하려면 `RemoveTagsFromResource` 작업을 사용합니다.

필수 ARN을 생성하는 방법에 대해 자세히 알아보려면 에 대한 아마존 리소스 이름 (ARN) 생성 AWS DMS 단원을 참조하십시오.

AWS DMS API를 사용한 XML 작업 시 다음 스키마를 사용합니다.

```
<Tagging>
  <TagSet>
   <Tag>
    <Key>Project</Key>
    <Value>Trinity</Value>
   </Tag>
   <Tag>
    <Key>User</Key>
    <Value>Jones</Value>
   </Tag>
  </TagSet>
</Tagging>
```

다음 표에는 허용되는 XML 태그와 해당 특성의 목록이 나와 있습니다. Key 및 Value 값은 대/소문자를 구분합니다. 예를 들어, project=Trinity와 PROJECT=Trinity는 서로 다른 두 개의 태그입니다.

태그 지정 요소	설명
TagSet	태그 세트에는 Amazon RDS 리소스에 배정된 모든 태그가 포함됩니다. 리소스당 하나의 태그 세트만 있을 수 있습니다. AWS DMS API를 통해서만 TagSet로 작업합니다.
Tag	태그는 사용자가 정의하는 키-값 페어입니다. 태그 세트에 1~10개의 태그가 있을 수 있습니다.
키	키는 태그의 필수 이름입니다. 문자열 값은 길이가 1~128자(유니코드 문자)이며 "dms:" 또는 "aws:"로 시작할 수 없습니다. 문자열에는 유니코드 문자, 숫자, 공백, '_', '.', '/', '=', '+', '-'(Java regex: "^([\\p{L}\\p{Z}\\p{N}_.:/=+\\-]*)$")만 포함될 수 있습니다. 키는 태그 집합에 대해 고유해야 합니다. 예를 들어, 태그 세트에 project/Trinity와 project/Xanadu처럼 키는 같지만 값은 다른 키-페어가 있을 수 없습니다.
값	값은 태그의 선택적 값입니다. 문자열 값은 길이가 1~256자(유니코드 문자)이며 "dms:" 또는 "aws:"로 시작할 수 없습니다. 문자열에는 유니코드 문자, 숫자, 공백, '_', '.', '/', '=', '+', '-'(Java regex: "^([\\p{L}\\p{Z}\\p{N}_.:/=+\\-]*)$")만 포함될 수 있습니다.

태그 지정 요소	설명
	값은 태그 세트에서 고유할 필요는 없으며 null일 수 있습니다. 예를 들어, project/Trinity 및 cost-center/Trinity의 태그 세트에 키-값 페어가 있을 수 있습니다.

AWS Database Migration Service의 보안

AWS에서는 클라우드 보안을 가장 중요하게 생각합니다. AWS 고객은 보안에 가장 보안에 민감한 조직의 요구 사항에 부합하도록 구축된 데이터 센터 및 네트워크 아키텍처의 혜택을 누릴 수 있습니다.

보안은 AWS과 사용자의 공동 책임입니다. 공동 책임 모델은 이 사항을 클라우드 내 보안 및 클라우드의 보안으로 설명합니다.

- 클라우드의 보안 - AWS는 AWS클라우드에서 AWS서비스를 실행하는 인프라를 보호합니다.AWS는 또한 안전하게 사용할 수 있는 서비스를 제공합니다. 타사 감사원은 정기적으로 AWS 규제 준수 프로그램의 일환으로 보안 효과를 테스트하고 검증합니다. AWS DMS에 적용되는 규정 준수 프로그램에 대해 알아보려면 규정 준수 프로그램 제공 범위 내 AWS서비스를 참조하세요.
- 클라우드 내 보안 – 귀하의 책임은 사용하는 AWS 서비스에 따라 결정됩니다. 또한 데이터의 민감도, 조직의 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는 AWS DMS 사용 시 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 다음 주제에서는 보안 및 규정 준수 목표를 충족하도록 AWS DMS을(를) 구성하는 방법을 보여줍니다. 또한 AWS DMS 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법을 알아봅니다.

AWS DMS 리소스 및 데이터베이스(DB)에 대한 액세스를 관리합니다. 액세스를 관리하는 데 사용하는 방법은 AWS DMS를 사용하여 수행해야 하는 복제 작업에 따라 다릅니다.

- AWS Identity and Access Management(IAM) 정책을 사용하여 AWS DMS 리소스를 관리할 수 있는 사용자를 결정하는 권한을 할당합니다. AWS DMS에서는 IAM 사용자로 로그인하려면 적절한 권한이 있어야 합니다. 예를 들면, IAM을 사용하여 DB 인스턴스 및 클러스터를 생성, 설명, 수정, 삭제하거나 리소스에 태그를 지정하거나 보안 그룹을 수정할 수 있는 사용자를 결정할 수 있습니다. IAM에 대한 정보 및 AWS DMS에서 사용하는 방법에 대한 자세한 내용은 에 대한 ID 및 액세스 관리 AWS Database Migration Service 섹션을 참조하세요.
- AWS DMS는 TLS(전송 계층 보안)와의 엔드포인트 연결에 SSL(보안 소켓 계층)을 사용합니다. AWS DMS에서 SSL/TLS를 사용하는 방법에 대한 자세한 내용은 AWS Database Migration Service에서 SSL 사용 섹션을 참조하세요.
- AWS DMS는 AWS Key Management Service(AWS KMS) 암호화 키를 사용하여 복제 인스턴스와 해당 엔드포인트 연결 정보에 사용되는 스토리지를 암호화합니다. 또한 AWS DMS는 Amazon S3 및 Amazon Redshift 대상 엔드포인트에 대해 AWS KMS 암호화 키를 사용하여 저장 시 대상 데이터를 보호합니다. 자세한 내용은 암호화 키 설정 및 AWS KMS 권한 지정 섹션을 참조하세요.

- AWS DMS는 네트워크 액세스 제어를 최대한 강화할 목적으로 항상 Amazon VPC 서비스를 기반으로 한 Virtual Private Cloud(VPC)에서 복제 인스턴스를 실행합니다. DB 인스턴스 및 인스턴스 클러스터의 경우 복제 인스턴스와 동일한 VPC를 사용하거나 이 액세스 제어 수준에 맞게 추가 VPC를 사용합니다. 사용하는 각 Amazon VPC는 모든 포트의 모든 트래픽이 VPC 외부로 나가도록(송신) 허용하는 규칙이 있는 보안 그룹과 연결되어야 합니다. 해당 엔드포인트에서 올바른 수신이 활성화되어 있는 한 이 접근 방식을 통해 통신을 복제 인스턴스에서 원본과 대상 데이터베이스 엔드포인트로 전달할 수 있습니다.

  AWS DMS에 사용할 수 있는 네트워크 구성에 대한 자세한 내용은 복제 인스턴스용으로 네트워크 설정 섹션을 참조하세요. VPC에서 DB 인스턴스 또는 인스턴스 클러스터를 생성하는 방법에 대한 자세한 내용은 AWS 설명서에서 Amazon 데이터베이스의 보안 및 클러스터 관리 설명서를 참조하세요. AWS DMS를 지원하는 네트워크 구성에 대한 자세한 내용은 복제 인스턴스용으로 네트워크 설정 섹션을 참조하세요.
- 데이터베이스 마이그레이션 로그를 확인하려면, 사용 중인 IAM 역할에 적합한 Amazon CloudWatch Log 권한이 필요합니다. AWS DMS 로깅에 대한 자세한 내용은 Amazon CloudWatch를 사용한 복제 태스크 모니터링 섹션을 참조하세요.

주제

- AWS Database Migration Service 데이터 보호
- 에 대한 ID 및 액세스 관리 AWS Database Migration Service
- AWS Database Migration Service의 규정 준수 확인
- AWS Database Migration Service의 복원성
- AWS Database Migration Service에서 인프라 보안
- 리소스 이름 및 태그를 사용하여 세분화된 액세스 제어
- 암호화 키 설정 및 AWS KMS 권한 지정
- AWS Database Migration Service 네트워크 보안
- AWS Database Migration Service에서 SSL 사용
- 데이터베이스 암호 변경

AWS Database Migration Service 데이터 보호

데이터 암호화

지원되는 AWS DMS 대상 엔드포인트의 데이터 리소스에 대해 암호화를 활성화할 수 있습니다. AWS DMS는 AWS DMS에 대한 연결 및 AWS DMS와 모든 소스 및 대상 엔드포인트 간 연결도 암호화합니다. 또한 AWS DMS 및 지원되는 대상 엔드포인트가 이 암호화를 활성화하기 위해 사용하는 키를 관리할 수 있습니다.

주제

- 저장 시 암호화
- 전송 중 암호화
- 키 관리

저장 시 암호화

AWS DMS는 AWS DMS 대상 엔드포인트에 복사하기 전에 복제된 데이터를 로 푸시하는 데 사용할 서버 측 암호화 모드를 사용자가 지정할 수 있게 허용하여 저장 시 데이터 암호화를 지원합니다. 엔드포인트에 대한 `encryptionMode` 추가 연결 속성을 설정하여 이 암호화 모드를 지정할 수 있습니다. 이 `encryptionMode` 설정이 KMS 키 암호화 모드를 지정하는 경우 다음 AWS DMS 대상 엔드포인트에 대해 특별히 대상 데이터를 암호화하기 위한 사용자 지정 AWS KMS 키를 생성할 수도 있습니다.

- Amazon Redshift - `encryptionMode` 설정에 대한 자세한 내용은 Amazon Redshift를 AWS DMS의 대상으로 사용 시 엔드포인트 설정 섹션을 참조하세요. 사용자 지정 AWS KMS 암호화 키 생성에 대한 자세한 내용은 Amazon Redshift 대상 데이터 암호화를 위한 AWS KMS 키 생성 및 사용 섹션을 참조하세요.
- Amazon S3 - `encryptionMode` 설정에 대한 자세한 내용은 Amazon S3를 AWS DMS의 대상으로 사용 시 엔드포인트 설정 섹션을 참조하세요. 사용자 지정 AWS KMS 암호화 키 생성에 대한 자세한 내용은 AWS KMS 키를 생성하여 Amazon S3 대상 객체 암호화 섹션을 참조하세요.

전송 중 암호화

AWS DMS는 복제하는 데이터가 소스 엔드포인트에서 대상 엔드포인트로 안전하게 이동하도록 전송 중 데이터 암호화를 지원합니다. 여기에는 데이터가 복제 파이프라인을 통해 이동할 때 복제 작업에서 중간 스토리지에 사용하는 복제 인스턴스의 S3 버킷을 암호화하는 작업이 포함됩니다. 소스 및 대상 엔드포인트에 대한 작업 연결을 암호화하기 위해 AWS DMS는 보안 소켓 계층(SSL) 또는 전송 계

층 보안(TLS)을 사용합니다. 두 엔드포인트에 대한 연결을 암호화함으로써 AWS DMS는 데이터가 소스 엔드포인트에서 복제 작업으로 또한 작업에서 대상 엔드포인트로 이동할 때 데이터를 보호합니다. AWS DMS에서 SSL/TLS를 사용하는 방법에 대한 자세한 내용은 AWS Database Migration Service에서 SSL 사용 섹션을 참조하세요.

AWS DMS는 기본 키와 사용자 지정 키를 모두 지원하여 중간 복제 스토리지와 연결 정보를 모두 암호화합니다. AWS KMS를 사용하여 이러한 키를 관리합니다. 자세한 내용은 암호화 키 설정 및 AWS KMS 권한 지정 섹션을 참조하세요.

키 관리

AWS DMS는 복제 스토리지, 연결 정보 및 특정 대상 엔드포인트에 대한 대상 데이터 스토리지를 암호화하는 기본 또는 사용자 지정 키를 지원합니다. AWS KMS를 사용하여 이러한 키를 관리합니다. 자세한 내용은 암호화 키 설정 및 AWS KMS 권한 지정 섹션을 참조하세요.

인터네트워크 트래픽 개인 정보

온프레미스에서 실행하든 클라우드에서 AWS 서비스의 일부로 실행하든 관계 없이 동일한 AWS 리전에서 AWS DMS와 소스 및 대상 엔드포인트 간 연결이 보호됩니다. (하나 이상의 엔드포인트(소스 또는 대상)가 클라우드에서 AWS 서비스의 일부로 실행되어야 합니다.) 이러한 보호는 VPC가 모두 동일한 AWS 리전에 있는 경우 이러한 구성 요소가 동일한 VPC(가상 프라이빗 클라우드)를 공유하는지 또는 별도의 VPC에 존재하는지 관계없이 적용됩니다. AWS DMS에 대해 지원되는 네트워크 구성에 대한 자세한 내용은 복제 인스턴스용으로 네트워크 설정 섹션을 참조하세요. 이러한 네트워크 구성 사용 시 보안 고려 사항에 대한 자세한 내용은 AWS Database Migration Service 네트워크 보안 섹션을 참조하세요.

DMS Fleet Advisor의 데이터 보호

DMS Fleet Advisor는 데이터베이스 메타데이터를 수집 및 분석하여 마이그레이션 대상의 적절한 크기를 결정합니다. DMS Fleet Advisor는 테이블의 데이터에 액세스하지 않으며 데이터를 전송하지도 않습니다. 또한 DMS Fleet Advisor는 데이터베이스 기능 사용을 추적하지 않으며 사용 통계에 액세스하지 않습니다.

DMS Fleet Advisor가 데이터베이스 작업에 사용하는 데이터베이스 사용자를 생성하면 데이터베이스에 대한 액세스를 제어할 수 있습니다. 이러한 사용자에게 필요한 권한을 부여합니다. DMS Fleet Advisor를 사용하려면 데이터베이스 사용자에게 읽기 권한을 부여해야 합니다. DMS Fleet Advisor는 데이터베이스를 수정하지 않으며 쓰기 권한이 필요하지 않습니다. 자세한 내용은 AWS DMS Fleet Advisor용 데이터베이스 사용자 생성 섹션을 참조하세요.

데이터베이스에서 데이터 암호화를 사용할 수 있습니다. 또한 AWS DMS는 DMS Fleet Advisor 및 해당 데이터 수집기 내의 연결을 암호화합니다.

DMS 데이터 수집기는 데이터 보호 애플리케이션 프로그래밍 인터페이스(DPAPI)를 사용하여 고객의 환경 및 데이터베이스 보안 인증 정보에 대한 정보를 암호화, 보호, 저장합니다. DMS Fleet Advisor는 이 암호화된 데이터를 DMS 데이터 수집기가 작동하는 서버의 파일에 저장합니다. DMS Fleet Advisor는 이 데이터를 이 서버 외부로 전송하지 않습니다. DPAPI에 대한 자세한 내용은 방법: 데이터 보호 사용을 참조하세요.

DMS 데이터 수집기를 설치한 후에는 이 애플리케이션이 지표를 수집하기 위해 실행하는 모든 쿼리를 볼 수 있습니다. 오프라인 모드에서 DMS 데이터 수집기를 실행한 다음 서버에서 수집된 데이터를 검토할 수 있습니다. 또한 Amazon S3 버킷에서 이렇게 수집된 데이터를 검토할 수 있습니다. 자세한 내용은 DMS 데이터 수집기는 어떻게 작동합니까? 섹션을 참조하세요.

에 대한 ID 및 액세스 관리 AWS Database Migration Service

AWS Identity and Access Management (IAM) 은 관리자가 AWS 리소스에 대한 액세스를 안전하게 제어할 수 AWS 서비스 있도록 도와줍니다. IAM 관리자는 리소스를 사용할 수 있는 인증 (로그인) 및 권한 부여 (권한 보유) 를 받을 수 있는 사용자를 제어합니다. AWS DMS IAM은 추가 AWS 서비스 비용 없이 사용할 수 있습니다.

주제

- 고객
- 보안 인증을 통한 인증
- 정책을 사용한 액세스 관리
- IAM의 AWS Database Migration Service 작동 방식
- AWS Database Migration Service ID 기반 정책 예제
- 에 대한 리소스 기반 정책 예제 AWS KMS
- 보안 암호를 사용하여 AWS Database Migration Service 엔드포인트에 액세스하기
- AWS DMS에 서비스 연결 역할 사용
- AWS Database Migration Service ID 및 액세스 문제 해결
- AWS DMS 사용에 필요한 IAM 권한
- AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성
- 교차 서비스 혼동된 대리인 방지
- AWS Database Migration Service의 AWS 관리형 정책

고객

사용하는 방식 AWS Identity and Access Management (IAM) 은 수행하는 작업에 따라 다릅니다. AWS DMS

서비스 사용자 - AWS DMS 서비스를 사용하여 작업을 수행하는 경우 관리자가 필요한 자격 증명과 권한을 제공합니다. 더 많은 AWS DMS 기능을 사용하여 작업을 수행함에 따라 추가 권한이 필요할 수 있습니다. 액세스 권한 관리 방식을 이해하면 적절한 권한을 관리자에게 요청할 수 있습니다. AWS DMS의 기능에 액세스할 수 없는 경우 AWS Database Migration Service ID 및 액세스 문제 해결 섹션을 참조하세요.

서비스 관리자 — 회사에서 AWS DMS 리소스를 담당하는 경우 전체 액세스 권한이 있을 수 AWS DMS 있습니다. 서비스 사용자가 액세스해야 하는 AWS DMS 기능과 리소스를 결정하는 것은 여러분

의 몫입니다. 그런 다음, IAM 관리자에게 요청을 제출하여 서비스 사용자의 권한을 변경해야 합니다. 이 페이지의 정보를 검토하여 IAM의 기본 개념을 이해하세요. 회사에서 IAM을 어떻게 사용할 수 있는지 자세히 AWS DMS 알아보려면 을 참조하십시오IAM의 AWS Database Migration Service 작동 방식.

IAM 관리자 - IAM 관리자라면 AWS DMS에 대한 액세스 권한 관리 정책 작성 방법을 자세히 알고 싶을 것입니다. IAM에서 사용할 수 있는 AWS DMS ID 기반 정책의 예를 보려면 을 참조하십시오. AWS Database Migration Service ID 기반 정책 예제

보안 인증을 통한 인증

인증은 자격 증명 자격 증명을 AWS 사용하여 로그인하는 방법입니다. IAM 사용자로 인증 (로그인 AWS) 하거나 IAM 역할을 맡아 인증 (로그인) 해야 합니다. AWS 계정 루트 사용자

ID 소스를 통해 제공된 자격 증명을 사용하여 페더레이션 ID로 로그인할 수 있습니다. AWS AWS IAM Identity Center (IAM ID 센터) 사용자, 회사의 싱글 사인온 인증, Google 또는 Facebook 자격 증명이 페더레이션 ID의 예입니다. 페더레이션형 자격 증명으로 로그인할 때 관리자가 이전에 IAM 역할을 사용하여 자격 증명 페더레이션을 설정했습니다. 페더레이션을 사용하여 액세스하는 경우 AWS 간접적으로 역할을 맡게 됩니다.

사용자 유형에 따라 AWS Management Console 또는 AWS 액세스 포털에 로그인할 수 있습니다. 로그인에 대한 자세한 내용은 AWS 로그인 사용 설명서의 내 로그인 방법을 참조하십시오. AWS AWS 계정

AWS 프로그래밍 방식으로 액세스하는 경우 자격 증명을 사용하여 요청에 암호화 방식으로 서명할 수 있는 소프트웨어 개발 키트 (SDK) 와 명령줄 인터페이스 (CLI) 를 AWS 제공합니다. AWS 도구를 사용하지 않는 경우 요청에 직접 서명해야 합니다. 권장 방법을 사용하여 직접 요청에 서명하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 AWS API 요청 서명을 참조하십시오.

사용하는 인증 방법에 상관없이 추가 보안 정보를 제공해야 할 수도 있습니다. 예를 들어, AWS 계정의 보안을 강화하기 위해 다단계 인증 (MFA) 을 사용할 것을 권장합니다. 자세한 내용은 AWS IAM Identity Center 사용 설명서의 다중 인증 및 IAM 사용 설명서의 AWS에서 다중 인증(MFA) 사용을 참조하세요.

AWS 계정 루트 사용자

계정을 AWS 계정 만들 때는 먼저 계정의 모든 AWS 서비스 리소스에 대한 완전한 액세스 권한을 가진 하나의 로그인 ID로 시작합니다. 이 ID를 AWS 계정 루트 사용자라고 하며, 계정을 만들 때 사용한 이메일 주소와 비밀번호로 로그인하여 액세스할 수 있습니다. 일상적인 작업에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자 자격 증명을 보호하고 루트 사용자만 수행할 수 있는 태스크를 수행하는 데 사용합니다. 루트 사용자로 로그인해야 하는 전체 작업 목록은 IAM 사용 설명서의 Tasks that require root user credentials를 참조하세요.

IAM 사용자 및 그룹

IAM 사용자는 단일 사용자 또는 애플리케이션에 대한 특정 권한을 가진 사용자 내의 자격 증명입니다. AWS 계정 가능하면 암호 및 액세스 키와 같은 장기 자격 증명이 있는 IAM 사용자를 생성하는 대신 임시 자격 증명을 사용하는 것이 좋습니다. 하지만 IAM 사용자의 장기 자격 증명이 필요한 특정 사용 사례가 있는 경우 액세스 키를 교체하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 장기 보안 인증이 필요한 사용 사례의 경우 정기적으로 액세스 키 교체 섹션을 참조하세요.

IAM 그룹은 IAM 사용자 컬렉션을 지정하는 자격 증명입니다. 사용자는 그룹으로 로그인할 수 없습니다. 그룹을 사용하여 여러 사용자의 권한을 한 번에 지정할 수 있습니다. 그룹을 사용하면 대규모 사용자 집합의 권한을 더 쉽게 관리할 수 있습니다. 예를 들어, IAMAdmins라는 그룹이 있고 이 그룹에 IAM 리소스를 관리할 권한을 부여할 수 있습니다.

사용자는 역할과 다릅니다. 사용자는 한 사람 또는 애플리케이션과 고유하게 연결되지만, 역할은 해당 역할이 필요한 사람이라면 누구나 수임할 수 있습니다. 사용자는 영구적인 장기 보안 인증을 가지고 있지만, 역할은 임시 자격 증명만 제공합니다. 자세한 정보는 IAM 사용 설명서의 IAM 사용자를 만들어야 하는 경우(역할이 아님) 섹션을 참조하세요.

IAM 역할

IAM 역할은 특정 권한을 가진 사용자 AWS 계정 내의 자격 증명입니다. IAM 사용자와 유사하지만, 특정 개인과 연결되지 않습니다. 역할을 AWS Management Console 전환하여 에서 일시적으로 IAM 역할을 맡을 수 있습니다. AWS CLI 또는 AWS API 작업을 호출하거나 사용자 지정 URL을 사용하여 역할을 수임할 수 있습니다. 역할 사용 방법에 대한 자세한 정보는 IAM 사용 설명서의 IAM 역할 사용을 참조하세요.

임시 보안 인증이 있는 IAM 역할은 다음과 같은 상황에서 유용합니다.

- 페더레이션 사용자 액세스 - 페더레이션 자격 증명에 권한을 부여하려면 역할을 생성하고 해당 역할의 권한을 정의합니다. 페더레이션 자격 증명이 인증되면 역할이 연결되고 역할에 정의된 권한이 부여됩니다. 페더레이션 역할에 대한 자세한 내용은 IAM 사용 설명서의 타사 자격 증명 공급자의 역할 생성을 참조하세요. IAM 자격 증명 센터를 사용하는 경우 권한 집합을 구성합니다. 인증 후 자격 증명이 액세스할 수 있는 항목을 제어하기 위해 IAM 자격 증명 센터는 권한 세트를 IAM의 역할과 연관짓습니다. 권한 세트에 대한 자세한 내용은 AWS IAM Identity Center 사용 설명서의 권한 세트 섹션을 참조하세요.
- 임시 IAM 사용자 권한 - IAM 사용자 또는 역할은 IAM 역할을 맡아 특정 작업에 대한 다양한 권한을 임시로 받을 수 있습니다.
- 크로스 계정 액세스 - IAM 역할을 사용하여 다른 계정의 사용자(신뢰할 수 있는 보안 주체)가 내 계정의 리소스에 액세스하도록 허용할 수 있습니다. 역할은 계정 간 액세스를 부여하는 기본적인 방법입

니다. 그러나 일부 AWS 서비스 경우에는 역할을 프록시로 사용하는 대신 정책을 리소스에 직접 연결할 수 있습니다. 교차 계정 액세스를 위한 역할과 리소스 기반 정책의 차이점을 알아보려면 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이 섹션을 참조하세요.

- 서비스 간 액세스 — 일부는 다른 AWS 서비스 서비스의 기능을 AWS 서비스 사용합니다. 예를 들어 서비스에서 직접적으로 호출을 수행하면 일반적으로 해당 서비스는 Amazon EC2에서 애플리케이션을 실행하거나 Amazon S3에 객체를 저장합니다. 서비스는 직접적으로 호출하는 보안 주체의 권한을 사용하거나, 서비스 역할을 사용하거나, 또는 서비스 연결 역할을 사용하여 이 태스크를 수행할 수 있습니다.
  - 순방향 액세스 세션 (FAS) — IAM 사용자 또는 역할을 사용하여 작업을 수행하는 경우 보안 AWS 주체로 간주됩니다. 일부 서비스를 사용하는 경우 다른 서비스에서 다른 작업을 시작하는 작업을 수행할 수 있습니다. FAS는 전화를 거는 주체의 권한을 다운스트림 AWS 서비스 서비스에 AWS 서비스 요청하기 위한 요청과 결합하여 사용합니다. FAS 요청은 다른 서비스 AWS 서비스 또는 리소스와의 상호 작용이 필요한 요청을 서비스가 수신한 경우에만 이루어집니다. 이 경우 두 작업을 모두 수행할 수 있는 권한이 있어야 합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.
  - 서비스 역할 - 서비스 역할은 서비스가 사용자를 대신하여 태스크를 수행하기 위해 맡는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 정보는 IAM 사용 설명서의 AWS 서비스에 대한 권한을 위임할 역할 생성을 참조하세요.
  - 서비스 연결 역할 — 서비스 연결 역할은 에 연결된 서비스 역할의 한 유형입니다. AWS 서비스 서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은 사용자에게 AWS 계정 표시되며 해당 서비스가 소유합니다. IAM 관리자는 서비스 링크 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.
- Amazon EC2에서 실행되는 애플리케이션 — IAM 역할을 사용하여 EC2 인스턴스에서 실행되고 API 요청을 AWS CLI 하는 애플리케이션의 임시 자격 증명을 관리할 수 있습니다. AWS 이는 EC2 인스턴스 내에 액세스 키를 저장할 때 권장되는 방법입니다. EC2 인스턴스에 AWS 역할을 할당하고 모든 애플리케이션에서 사용할 수 있게 하려면 인스턴스에 연결된 인스턴스 프로필을 생성합니다. 인스턴스 프로필에는 역할이 포함되어 있으며 EC2 인스턴스에서 실행되는 프로그램이 임시 자격 증명을 얻을 수 있습니다. 자세한 정보는 IAM 사용 설명서의 IAM 역할을 사용하여 Amazon EC2 인스턴스에서 실행되는 애플리케이션에 권한 부여 섹션을 참조하세요.

IAM 역할을 사용할지 또는 IAM 사용자를 사용할지를 알아보려면 IAM 사용 설명서의 IAM 역할(사용자 대신)을 생성하는 경우 섹션을 참조하세요.

정책을 사용한 액세스 관리

정책을 생성하고 이를 AWS ID 또는 리소스에 AWS 연결하여 액세스를 제어할 수 있습니다. 정책은 ID 또는 리소스와 연결될 때 AWS 해당 권한을 정의하는 객체입니다. AWS 주도자 (사용자, 루트 사용자 또는 역할 세션) 가 요청할 때 이러한 정책을 평가합니다. 정책의 권한이 요청 허용 또는 거부 여부를 결정합니다. 대부분의 정책은 JSON 문서로 AWS 저장됩니다. JSON 정책 문서의 구조와 콘텐츠에 대한 자세한 정보는 IAM 사용 설명서의 JSON 정책 개요 섹션을 참조하세요.

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

기본적으로, 사용자와 역할에는 어떠한 권한도 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다. 그런 다음 관리자가 IAM 정책을 역할에 추가하고, 사용자가 역할을 수임할 수 있습니다.

IAM 정책은 작업을 수행하기 위해 사용하는 방법과 상관 없이 작업에 대한 권한을 정의합니다. 예를 들어, `iam:GetRole` 작업을 허용하는 정책이 있다고 가정합니다. 해당 정책을 사용하는 사용자는 AWS Management Console, AWS CLI, 또는 AWS API에서 역할 정보를 가져올 수 있습니다.

보안 인증 기반 정책

보안 인증 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 보안 인증에 연결할 수 있는 JSON 권한 정책 설명서입니다. 이러한 정책은 사용자와 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 제어합니다. ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성을 참조하세요.

보안 인증 기반 정책은 인라인 정책 또는 관리형 정책으로 한층 더 분류할 수 있습니다. 인라인 정책은 단일 사용자, 그룹 또는 역할에 직접 포함됩니다. 관리형 정책은 내 여러 사용자, 그룹 및 역할에 연결할 수 있는 독립형 정책입니다. AWS 계정 관리형 정책에는 AWS 관리형 정책과 고객 관리형 정책이 포함됩니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용 설명서의 관리형 정책과 인라인 정책의 선택을 참조하세요.

리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 연동 사용자 등이 포함될 수 있습니다. AWS 서비스

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. IAM의 AWS 관리형 정책은 리소스 기반 정책에 사용할 수 없습니다.

액세스 제어 목록(ACL)

액세스 제어 목록(ACL)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는지를 제어합니다. ACL은 JSON 정책 설명서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

ACL을 지원하는 서비스의 예로는 아마존 S3와 아마존 VPC가 있습니다. AWS WAF ACL에 대해 자세히 알아보려면 Amazon Simple Storage Service 개발자 안내서의 ACL(액세스 제어 목록) 개요 섹션을 참조하세요.

기타 정책 타입

AWS 일반적이지 않은 추가 정책 유형을 지원합니다. 이러한 정책 타입은 더 일반적인 정책 타입에 따라 사용자에게 부여되는 최대 권한을 설정할 수 있습니다.

- 권한 경계 – 권한 경계는 자격 증명 기반 정책에 따라 IAM 개체(IAM 사용자 또는 역할)에 부여할 수 있는 최대 권한을 설정하는 고급 기능입니다. 개체에 대한 권한 경계를 설정할 수 있습니다. 그 결과로 얻는 권한은 엔터티의 자격 증명 기반 정책과 그 권한 경계의 교집합입니다. `Principal` 필드에서 사용자나 역할을 보안 주체로 지정하는 리소스 기반 정책은 권한 경계를 통해 제한되지 않습니다. 이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 권한 경계에 대한 자세한 정보는 IAM 사용 설명서의 IAM 엔터티에 대한 권한 경계 섹션을 참조하세요.
- 서비스 제어 정책 (SCP) - SCP는 조직 또는 조직 단위 (OU) 에 대한 최대 권한을 지정하는 JSON 정책입니다. AWS Organizations AWS Organizations 사업체가 소유한 여러 AWS 계정 개를 그룹화하고 중앙에서 관리하는 서비스입니다. 조직에서 모든 기능을 활성화할 경우 서비스 통제 정책(SCP)을 임의의 또는 모든 계정에 적용할 수 있습니다. SCP는 구성원 계정의 엔티티 (각 엔티티 포함) 에 대한 권한을 제한합니다. AWS 계정 루트 사용자 조직 및 SCP에 대한 자세한 정보는 AWS Organizations 사용 설명서의 SCP 작동 방식을 참조하세요.
- 세션 정책 – 세션 정책은 역할 또는 페더레이션 사용자에 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 결과적으로 얻는 세션의 권한은 사용자 또는 역할의 보안 인증 기반 정책의 교차와 세션 정책입니다. 또한 권한을 리소스 기반 정책에서 가져올 수도 있습니다. 이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 자세한 정보는 IAM 사용 설명서의 세션 정책을 참조하세요.

여러 정책 타입

여러 정책 타입이 요청에 적용되는 경우 결과 권한은 이해하기가 더 복잡합니다. 여러 정책 유형이 관련되어 있을 때 요청을 허용할지 여부를 AWS 결정하는 방법을 알아보려면 IAM 사용 설명서의 정책 평가 로직을 참조하십시오.

IAM의 AWS Database Migration Service 작동 방식

IAM을 사용하여 액세스를 AWS DMS 관리하기 전에 먼저 사용할 수 있는 IAM 기능이 무엇인지 이해해야 합니다. AWS DMS 기타 AWS 서비스가 AWS DMS IAM과 연동되는 방식을 자세히 알아보려면 IAM 사용 설명서에서 IAM과 연동되는 AWS 서비스를 참조하십시오.

주제

- AWS DMS ID 기반 정책
- AWS DMS 리소스 기반 정책
- AWS DMS 태그 기반 인증
- IAM 역할: AWS DMS
- DMS Fleet Advisor의 ID 및 액세스 관리

AWS DMS ID 기반 정책

IAM 자격 증명 기반 정책을 사용하면 허용되거나 거부되는 작업 및 리소스를 지정할 수 있을 뿐 아니라 작업이 허용되거나 거부되는 조건도 지정할 수 있습니다. AWS DMS 는 특정한 작업, 리소스 및 조건 키를 지원합니다. JSON 정책에서 사용하는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하세요.

작업

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 `Action` 요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 작업을 설명합니다. 정책 작업은 일반적으로 관련 AWS API 작업과 이름이 같습니다. 일치하는 API 작업이 없는 권한 전용 작업 같은 몇 가지 예외도 있습니다. 정책에서 여러 작업이 필요한 몇 가지 작업도 있습니다. 이러한 추가 작업을 종속 작업이라고 합니다.

연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하십시오.

정책 조치는 조치 앞에 다음 접두사를 AWS DMS 사용합니다. dms: 예를 들어 AWS DMS CreateReplicationTask API 작업을 사용하여 복제 작업을 생성할 권한을 누군가에게 부여하려면 해당 작업을 정책에 포함해야 합니다. dms:CreateReplicationTask 정책 설명에는 Action OR NotAction 요소가 포함되어야 합니다. AWS DMS 이 서비스로 수행할 수 있는 작업을 설명하는 고유한 작업 집합을 정의합니다.

명령문 하나에 여러 작업을 지정하려면 다음과 같이 쉼표로 구분합니다.

```
"Action": [
      "dms:action1",
      "dms:action2"
```

와일드카드(*)를 사용하여 여러 작업을 지정할 수 있습니다. 예를 들어, Describe라는 단어로 시작하는 모든 작업을 지정하려면 다음 작업을 포함합니다.

```
"Action": "dms:Describe*"
```

AWS DMS 작업 목록을 보려면 IAM 사용 설명서의 정의된 AWS Database Migration Service 작업을 참조하십시오.

리소스

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

Resource JSON 정책 요소는 작업이 적용되는 하나 이상의 객체를 지정합니다. 보고서에는 Resource 또는 NotResource 요소가 반드시 추가되어야 합니다. 모범 사례에 따라 Amazon 리소스 이름(ARN)을 사용하여 리소스를 지정합니다. 리소스 수준 권한이라고 하는 특정 리소스 유형을 지원하는 작업에 대해 이 작업을 수행할 수 있습니다.

작업 나열과 같이 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

```
"Resource": "*"
```

AWS DMS 다음 리소스와 함께 작동합니다.

- 인증서
- 엔드포인트

- 이벤트 구독
- 복제 인스턴스
- 복제 서브넷(보안) 그룹
- 복제 작업

AWS DMS 필요한 리소스 또는 리소스는 호출하는 작업 또는 작업에 따라 다릅니다. 리소스 ARN에서 지정한 관련 리소스에 대해 이러한 작업을 허용하는 정책이 필요합니다.

예를 들어, AWS DMS 엔드포인트 리소스의 ARN은 다음과 같습니다.

```
arn:${Partition}:dms:${Region}:${Account}:endpoint/${InstanceId}
```

ARN 형식에 대한 자세한 내용은 Amazon 리소스 이름 (ARN) 및 AWS 서비스 네임스페이스를 참조하십시오.

예를 들어 문에서 `use-east-2` 리전의 `1A2B3C4D5E6F7G8H9I0J1K2L3M` 엔드포인트 인스턴스를 지정하려면 다음 ARN을 사용합니다.

```
"Resource": "arn:aws:dms:us-east-2:987654321098:endpoint/1A2B3C4D5E6F7G8H9I0J1K2L3M"
```

특정 계정에 속하는 모든 엔드포인트를 지정하려면 와일드카드(*)를 사용합니다.

```
"Resource": "arn:aws:dms:us-east-2:987654321098:endpoint/*"
```

리소스 생성 작업과 같은 일부 AWS DMS 작업은 특정 리소스에서 수행할 수 없습니다. 이러한 경우, 와일드카드(*)를 사용해야 합니다.

```
"Resource": "*"
```

일부 AWS DMS API 작업에는 여러 리소스가 포함됩니다. 예를 들어 `StartReplicationTask`를 시작하고 복제 작업을 두 개의 데이터베이스 엔드포인트 리소스(소스 및 대상)에 연결하므로 IAM 사용자는 소스 엔드포인트를 읽고 대상 엔드포인트에 쓸 수 있는 권한이 있어야 합니다. 단일 문에서 여러 리소스를 지정하려면 ARN을 쉼표로 구분합니다.

```
"Resource": [
      "resource1",
      "resource2" ]
```

정책을 사용하여 AWS DMS 리소스에 대한 액세스를 제어하는 방법에 대한 자세한 내용은 을 참조하십시오리소스 이름을 사용하여 액세스 제어. AWS DMS 리소스 유형 및 해당 ARN의 목록을 보려면 IAM 사용 설명서의 AWS Database Migration Service에서 정의된 리소스를 참조하십시오. 각 리소스의 ARN을 지정할 수 있는 작업을 알아보려면 AWS Database Migration Service가 정의한 작업을 참조하십시오.

조건 키

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

`Condition` 요소(또는 `Condition` 블록)를 사용하면 정책이 발효되는 조건을 지정할 수 있습니다. `Condition` 요소는 옵션입니다. 같음이나 미만 같은 조건 연산자를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수 있습니다.

한 문에서 여러 `Condition` 요소를 지정하거나 단일 `Condition` 요소에서 여러 키를 지정하는 경우 AWS 은(는) 논리적 `AND` 작업을 사용하여 평가합니다. 단일 조건 키에 여러 값을 지정하는 경우 는 논리적 `OR` 연산을 사용하여 조건을 AWS 평가합니다. 명령문의 권한을 부여하기 전에 모든 조건을 충족해야 합니다.

조건을 지정할 때 자리 표시자 변수를 사용할 수도 있습니다. 예를 들어, IAM 사용자에게 IAM 사용자 이름으로 태그가 지정된 경우에만 리소스에 액세스할 수 있는 권한을 부여할 수 있습니다. 자세한 정보는 IAM 사용 설명서의 IAM 정책 요소: 변수 및 태그 섹션을 참조하세요.

AWS 글로벌 조건 키 및 서비스별 조건 키를 지원합니다. 모든 AWS 글로벌 조건 키를 보려면 IAM 사용 AWS 설명서의 글로벌 조건 컨텍스트 키를 참조하십시오.

AWS DMS 자체 조건 키 세트를 정의하며 일부 글로벌 조건 키 사용도 지원합니다. 모든 AWS 글로벌 조건 키를 보려면 IAM 사용 설명서의 AWS 글로벌 조건 컨텍스트 키를 참조하십시오.

AWS DMS 조건 키에 사용할 수 있는 표준 태그 세트를 정의하고 사용자 지정 태그를 정의할 수도 있습니다. 자세한 설명은 태그를 사용하여 액세스 제어 섹션을 참조하세요.

AWS DMS 조건 키 목록을 보려면 IAM 사용 설명서의 조건 키를 참조하십시오 AWS Database Migration Service. 조건 키를 사용할 수 있는 작업과 리소스에 대해 알아보려면 AWS Database Migration Service에서 정의한 작업 및 AWS Database Migration Service에서 정의한 리소스를 참조하십시오.

예

AWS DMS 자격 증명 기반 정책의 예를 보려면 을 참조하십시오. AWS Database Migration Service ID 기반 정책 예제

AWS DMS 리소스 기반 정책

리소스 기반 정책은 지정된 보안 주체가 특정 리소스에서 어떤 조건에서 수행할 수 있는 작업을 지정하는지를 지정하는 JSON 정책 문서입니다. AWS DMS AWS DMS 지원되는 대상 엔드포인트로 마이그레이션된 데이터를 암호화하기 위해 생성한 AWS KMS 암호화 키에 대한 리소스 기반 권한 정책을 지원합니다. 지원되는 대상 엔드포인트에는 Amazon Redshift 및 Amazon S3 등이 있습니다. 리소스 기반 정책을 사용하여 각 대상 엔드포인트의 다른 계정에 이러한 암호화 키를 사용할 수 있는 권한을 부여할 수 있습니다.

크로스 계정 액세스를 활성화하려는 경우, 전체 계정이나 다른 계정의 IAM 엔터티를 리소스 기반 정책의 보안 주체로 지정할 수 있습니다. 리소스 기반 정책에 크로스 계정 보안 주체를 추가하는 것은 트러스트 관계 설정의 절반밖에 되지 않는다는 것을 유념하세요. 보안 주체와 리소스가 서로 다른 AWS 계정에 있는 경우 보안 주체 엔티티에 리소스에 액세스할 수 있는 권한도 부여해야 합니다. 엔터티에 보안 인증 기반 정책을 연결하여 권한을 부여합니다. 하지만 리소스 기반 정책이 동일 계정의 보안 주체에 액세스를 부여하는 경우, 추가 자격 증명 기반 정책이 필요하지 않습니다. 자세한 정보는 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이 섹션을 참조하세요.

이 AWS DMS 서비스는 AWS KMS 암호화 키에 연결된 키 정책이라는 한 가지 유형의 리소스 기반 정책만 지원합니다. 이 정책은 지원되는 대상 엔드포인트에서 마이그레이션된 데이터를 암호화할 수 있는 보안 주체 엔터티(계정, 사용자, 역할 및 페더레이션 사용자)를 정의합니다.

지원되는 대상 엔드포인트에 대해 생성한 암호화 키에 리소스 기반 정책을 연결하는 방법을 알아보려면 Amazon Redshift 대상 데이터 암호화를 위한 AWS KMS 키 생성 및 사용 및 AWS KMS 키를 생성하여 Amazon S3 대상 객체 암호화 단원을 참조하십시오.

예

AWS DMS 리소스 기반 정책의 예는 을 참조하십시오. 에 대한 리소스 기반 정책 예제 AWS KMS

AWS DMS 태그 기반 인증

AWS DMS 리소스에 태그를 첨부하거나 요청에 태그를 전달할 수 있습니다. AWS DMS 태그를 기반으로 액세스를 제어하려면 `dms:ResourceTag/`*`key-name`*`aws:RequestTag/`*`key-name`*, 또는 `aws:TagKeys`조건 키를 사용하여 정책의 조건 요소에 태그 정보를 제공합니다. AWS DMS 조건 키에 사용할 수 있는 표준 태그 세트를 정의하며 사용자 지정 태그를 정의할 수도 있습니다. 자세한 설명은 태그를 사용하여 액세스 제어 섹션을 참조하세요.

태그를 기반으로 리소스에 대한 액세스를 제한하는 자격 증명 기반 정책의 예는 태그를 기반으로 AWS DMS 리소스에 액세스 단원을 참조하십시오.

IAM 역할: AWS DMS

IAM 역할은 AWS 계정 내에서 특정 권한을 가진 엔티티입니다.

임시 자격 증명 사용: AWS DMS

임시 보안 인증을 사용하여 페더레이션으로 로그인하거나 IAM 역할을 수임하거나 교차 계정 역할을 수임할 수 있습니다. AssumeRole또는 와 같은 AWS STS API 작업을 호출하여 임시 보안 자격 증명을 얻습니다 GetFederationToken.

AWS DMS 임시 자격 증명 사용을 지원합니다.

서비스 링크 역할

서비스 연결 역할을 사용하면 AWS 서비스가 다른 서비스의 리소스에 액세스하여 사용자를 대신하여 작업을 완료할 수 있습니다. 서비스 연결 역할은 IAM 계정에 나타나며 서비스가 소유합니다. IAM 관리자는 서비스 연결 역할의 권한을 볼 수 있지만 편집할 수 없습니다.

AWS DMS 서비스 연결 역할을 만들거나 관리하는 방법에 대한 자세한 내용은 을 참조하십시오. 서비스 연결 역할 사용

Service roles(서비스 역할)

이 기능을 사용하면 서비스가 사용자를 대신하여 서비스 역할을 수임할 수 있습니다. 이 역할을 사용하면 서비스가 다른 서비스의 리소스에 액세스해 사용자를 대신해 작업을 완료할 수 있습니다. 서비스 역할은 IAM 계정에 나타나고, 해당 계정이 소유합니다. 즉, IAM 관리자가 이 역할에 대한 권한을 변경할 수 있습니다. 그러나 권한을 변경하면 서비스의 기능이 손상될 수 있습니다.

AWS DMS 특정 소스 또는 대상 엔드포인트를 사용하기 위해 생성해야 하는 두 가지 유형의 서비스 역할을 지원합니다.

- 다음 소스 및 대상 엔드포인트 (또는 해당 리소스) 에 대한 AWS DMS 액세스를 허용할 권한이 있는 역할:
  - 대상으로서 Amazon DynamoDB – 자세한 내용은 DynamoDB를 AWS Database Migration Service의 대상으로 사용하기 위한 사전 조건를 참조하세요.
  - OpenSearch 대상으로 — 자세한 내용은 을 참조하십시오. Amazon OpenSearch Service 서비스를 AWS Database Migration Service의 대상으로 사용하기 위한 사전 요구 사항

- 대상으로서 Amazon Kinesis – 자세한 내용은 Kinesis 데이터 스트림을 대상으로 사용하기 위한 사전 요구 사항 AWS Database Migration Service를 참조하세요.
- 대상으로서 Amazon Redshift – 대상 데이터를 암호화하기 위한 사용자 지정 KMS 암호화 키를 생성하거나 중간 작업 스토리지를 보관할 사용자 지정 S3 버킷을 지정하는 경우에만 지정된 역할을 생성해야 합니다. 자세한 내용은 Amazon Redshift 대상 데이터 암호화를 위한 AWS KMS 키 생성 및 사용 또는 Amazon S3 버킷 설정를 참조하세요.
- 소스 또는 대상으로서 Amazon S3 – 자세한 내용은 Amazon S3를 AWS DMS 소스로 사용 시 사전 요구 사항 또는 Amazon S3를 대상으로 사용하기 위한 사전 조건 단원을 참조하십시오.

  예를 들어, S3 소스 엔드포인트에서 데이터를 읽거나 S3 대상 엔드포인트로 데이터를 푸시하려면 이러한 엔드포인트 작업 각각에서 S3에 액세스하기 위한 전제 조건으로 서비스 역할을 생성해야 합니다.

- AWS CLI 및 AWS DMS API를 사용하는 데 필요한 권한이 있는 역할 — 생성해야 하는 두 개의 IAM 역할은 및 입니다`dms-vpc-role`. `dms-cloudwatch-logs-role` Amazon Redshift를 대상 데이터베이스로 사용하는 경우 IAM `dms-access-for-endpoint` 역할도 생성하여 계정에 추가해야 합니다. AWS 자세한 설명은 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성 섹션을 참조하세요.

에서 IAM 역할 선택 AWS DMS

데이터베이스 마이그레이션에 AWS CLI 또는 AWS DMS API를 사용하는 경우 DMS 기능을 사용하려면 먼저 AWS 계정에 특정 IAM 역할을 추가해야 합니다. AWS 이 중 두 가지는 `dms-vpc-role`과 `dms-cloudwatch-logs-role`입니다. Amazon Redshift를 대상 데이터베이스로 사용하는 경우 계정에 IAM `dms-access-for-endpoint` 역할도 추가해야 합니다. AWS 자세한 설명은 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성 섹션을 참조하세요.

DMS Fleet Advisor의 ID 및 액세스 관리

IAM 자격 증명 기반 정책을 사용하면 허용되거나 거부되는 작업과 리소스를 지정할 수 있으며 작업이 허용되거나 거부되는 조건도 지정할 수 있습니다. DMS Fleet Advisor는 특정 작업, 리소스 및 조건 키를 지원합니다. JSON 정책에서 사용하는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하십시오.

DMS Fleet Advisor는 IAM 역할을 사용하여 Amazon Simple Storage Service에 액세스합니다. IAM 역할은 AWS 계정 내에서 특정 권한을 가진 엔티티입니다. 자세한 설명은 IAM 리소스 생성 섹션을 참조하세요.

AWS Database Migration Service ID 기반 정책 예제

기본적으로 IAM 사용자 및 역할은 AWS DMS 리소스를 생성하거나 수정할 수 있는 권한이 없습니다. 또한 AWS Management Console AWS CLI, 또는 AWS API를 사용하여 작업을 수행할 수 없습니다. IAM 관리자는 지정된 리소스에서 특정 API 작업을 수행할 수 있는 권한을 사용자와 역할에게 부여하는 IAM 정책을 생성해야 합니다. 그런 다음 관리자는 해당 권한이 필요한 IAM 사용자 또는 그룹에 이러한 정책을 연결해야 합니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM 보안 인증 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 JSON 탭에서 정책 생성을 참조하십시오.

주제

- 정책 모범 사례
- AWS DMS 콘솔 사용
- 사용자가 자신의 고유한 권한을 볼 수 있도록 허용
- 하나의 Amazon S3 버킷에 액세스
- 태그를 기반으로 AWS DMS 리소스에 액세스

정책 모범 사례

ID 기반 정책은 누군가가 계정에서 AWS DMS 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부를 결정합니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. 자격 증명 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르세요.

- AWS 관리형 정책으로 시작하여 최소 권한 권한으로 이동 — 사용자와 워크로드에 권한을 부여하려면 여러 일반적인 사용 사례에 권한을 부여하는 AWS 관리형 정책을 사용하세요. 해당 내용은 에서 사용할 수 있습니다. AWS 계정 사용 사례에 맞는 AWS 고객 관리형 정책을 정의하여 권한을 더 줄이는 것이 좋습니다. 자세한 정보는 IAM 사용 설명서의 AWS 관리형 정책 또는 직무에 대한 AWS 관리형 정책 섹션을 참조하세요.
- 최소 권한 적용 – IAM 정책을 사용하여 태스크를 설정하는 경우 태스크를 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 least-privilege permissions(최소 권한)으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용 설명서의 IAM의 정책 및 권한을 참조하세요.
- Use conditions in IAM policies to further restrict access(IAM 정책의 조건을 사용하여 액세스 추가 제한) – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어 SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 생성할 수 있습니다. 예를 AWS 서

비스 들어 특정 작업을 통해 서비스 작업을 사용하는 경우 조건을 사용하여 서비스 작업에 대한 액세스 권한을 부여할 수도 AWS CloudFormation 있습니다. 자세한 정보는 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.

- IAM Access Analyzer를 통해 IAM 정책을 검증하여 안전하고 기능적인 권한 보장 – IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 신규 및 기존 정책을 검증합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 권장 사항을 제공하여 안전하고 기능적인 정책을 생성하도록 돕습니다. 자세한 정보는 IAM 사용 설명서의 IAM Access Analyzer 정책 검증을 참조하세요.
- 멀티 팩터 인증 (MFA) 필요 - IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 AWS 계정 MFA를 활성화하십시오. API 작업을 직접적으로 호출할 때 MFA가 필요하면 정책에 MFA 조건을 추가합니다. 자세한 정보는 IAM 사용 설명서의 MFA 보호 API 액세스 구성 섹션을 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 IAM의 보안 모범 사례를 참조하세요.

AWS DMS 콘솔 사용

다음 정책은 AWS AWS DMS 콘솔을 포함한 DMS에 대한 액세스를 제공하고 Amazon EC2와 같은 다른 Amazon 서비스에 필요한 특정 작업에 대한 권한도 지정합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "dms:*",
            "Resource": "arn:aws:dms:region:account:resourcetype/id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:ListAliases",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "iam:GetRole",
                "iam:PassRole",
                "iam:CreateRole",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVpcs",
                "ec2:DescribeInternetGateways",
                "ec2:DescribeAvailabilityZones",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:Get*",
                "cloudwatch:List*"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:DescribeLogStreams",
                "logs:FilterLogEvents",
                "logs:GetLogEvents"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
    ]
}
```

이러한 권한을 분석하면 콘솔을 사용하는 데 각 권한이 필요한 이유를 더 잘 이해할 수 있습니다.

다음 단원은 사용자가 콘솔에 표시할 수 있는 AWS KMS 키와 별칭을 나열할 수 있도록 허용하는 데 필요합니다. KMS 키에 대한 Amazon 리소스 이름(ARN)을 알고 있고 AWS Command Line Interface (AWS CLI)만을 사용할 때는 이 항목이 필요하지 않습니다.

```
{
            "Effect": "Allow",
            "Action": [
                "kms:ListAliases",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
```

엔드포인트를 사용하여 역할 ARN을 전달해야 하는 특정 엔드포인트 유형에는 다음 단원이 필요합니다. 또한 필요한 AWS DMS 역할을 미리 생성하지 않은 경우 AWS DMS 콘솔에서 역할을 생성할 수 있습니다. 모든 역할이 사전에 구성된 경우, `iam:GetRole` 및 `iam:PassRole`만 있으면 됩니다. 역할에 관한 자세한 내용은 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성 단원을 참조하세요.

```
{
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:PassRole",
                "iam:CreateRole",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
```

Amazon EC2 인스턴스를 생성하고 생성된 복제 인스턴스에 맞게 네트워크를 AWS DMS 구성해야 하므로 다음 섹션이 필요합니다. 이러한 리소스는 고객의 계정에 있으므로 고객 대신에 이러한 작업을 수행하는 기능이 필요합니다.

```
{
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVpcs",
                "ec2:DescribeInternetGateways",
                "ec2:DescribeAvailabilityZones",
```

```
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
```

다음 단원에서는 사용자가 복제 인스턴스 지표를 볼 수 있도록 허용해야 합니다.

```
{
            "Effect": "Allow",
            "Action": [
                "cloudwatch:Get*",
                "cloudwatch:List*"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
```

이 단원에서는 사용자가 복제 로그를 볼 수 있도록 허용해야 합니다.

```
{
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:DescribeLogStreams",
                "logs:FilterLogEvents",
                "logs:GetLogEvents"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
```

AWS DMS 콘솔은 AWS DMS 콘솔을 사용할 때 AWS 계정에 자동으로 연결되는 여러 역할을 생성합니다. 마이그레이션에 AWS Command Line Interface (AWS CLI) 또는 AWS DMS API를 사용하는 경우 계정에 이러한 역할을 추가해야 합니다. 이러한 역할을 추가하는 방법에 관한 자세한 내용은 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성 단원을 참조하십시오.

이 정책을 사용하여 AWS DMS에 액세스하기 위한 요구 사항에 대한 자세한 내용은 을 참조하십시오. AWS DMS 사용에 필요한 IAM 권한

사용자가 자신의 고유한 권한을 볼 수 있도록 허용

이 예제는 IAM 사용자가 자신의 사용자 자격 증명에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 또는 API를 사용하여 프로그래밍 방식으로 이 작업을 완료할 수 있는 AWS CLI 권한이 포함됩니다. AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

하나의 Amazon S3 버킷에 액세스

AWS DMS는 Amazon S3 버킷을 데이터베이스 마이그레이션을 위한 중간 스토리지로 사용합니다. 일반적으로 AWS DMS는 이러한 목적으로 기본 S3 버킷을 관리합니다. 그러나 특정 경우, 특히 AWS CLI

또는 AWS DMS API를 사용하는 경우 DMS를 사용하여 자체 AWS S3 버킷을 대신 지정할 수 있습니다. 예를 들어, 데이터를 Amazon Redshift 대상 엔드포인트로 마이그레이션하기 위한 자체 S3 버킷을 지정할 수 있습니다. 이 경우 AWS `AmazonDMSRedshiftS3Role` -managed 정책을 기반으로 권한이 있는 역할을 생성해야 합니다.

다음 예제에서는 `AmazonDMSRedshiftS3Role` 정책의 한 버전을 보여 줍니다. 이를 통해 AWS DMS는 AWS 계정의 IAM 사용자에게 Amazon S3 버킷 중 하나에 대한 액세스 권한을 부여할 수 있습니다. 또한 사용자가 객체를 추가, 업데이트 및 삭제할 수 있게 허용합니다.

이 정책에서는 `s3:PutObject`, `s3:GetObject` 및 `s3:DeleteObject` 권한을 사용자에게 부여할 뿐만 아니라 `s3:ListAllMyBuckets`, `s3:GetBucketLocation` 및 `s3:ListBucket` 권한도 부여합니다. 이러한 권한은 콘솔에 필요한 추가 권한입니다. 다른 권한을 통해 AWS DMS는 버킷 수명 주기를 관리할 수 있습니다. 또한 객체를 복사할 수 있도록 하려면 `s3:GetObjectAcl` 작업이 필요합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
                "s3:ListBucket",
                "s3:DeleteBucket",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:PutObject",
                "s3:DeleteObject",
                "s3:GetObjectVersion",
                "s3:GetBucketPolicy",
                "s3:PutBucketPolicy",
                "s3:GetBucketAcl",
                "s3:PutBucketVersioning",
                "s3:GetBucketVersioning",
                "s3:PutLifecycleConfiguration",
                "s3:GetLifecycleConfiguration",
                "s3:DeleteBucketPolicy"
            ],
            "Resource": "arn:aws:s3:::dms-*"
        }
    ]
}
```

이 정책을 기반으로 역할을 생성하는 방법에 관한 자세한 내용은 Amazon S3 버킷 설정 단원을 참조하십시오.

태그를 기반으로 AWS DMS 리소스에 액세스

자격 증명 기반 정책의 조건을 사용하여 태그를 기반으로 AWS DMS 리소스에 대한 액세스를 제어할 수 있습니다. 이 예제는 모든 AWS DMS 엔드포인트에 대한 액세스를 허용하는 정책을 만드는 방법을 보여줍니다. 하지만 권한은 엔드포인트 데이터베이스 태그 `Owner`가 해당 사용자의 사용자 이름을 값으로 갖는 경우에만 부여됩니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "dms:*",
            "Resource": "arn:aws:dms:*:*:endpoint/*",
            "Condition": {
                "StringEquals": {"dms:endpoint-tag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

이 정책을 계정의 IAM 사용자에게 연결할 수 있습니다. 라는 사용자가 AWS DMS 엔드포인트에 `richard-roe` 액세스하려고 시도하는 경우 엔드포인트 데이터베이스에 또는 태그를 `Owner=richard-roe` 지정해야 합니다. `owner=richard-roe` 그렇지 않으면 이 사용자는 액세스가 거부됩니다. 조건 키 이름은 대소문자를 구분하지 않기 때문에 조건 태그 키 `Owner`는 `Owner` 및 `owner`와 모두 일치합니다. 자세한 정보는 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하십시오.

에 대한 리소스 기반 정책 예제 AWS KMS

AWS DMS를 사용하면 사용자 지정 AWS KMS 암호화 키를 생성하여 지원되는 대상 엔드포인트 데이터를 암호화할 수 있습니다. 지원되는 대상 데이터 암호화를 위해 생성하는 암호화 키에 대한 키 정책을 생성하고 연결하는 방법을 알아보려면 Amazon Redshift 대상 데이터 암호화를 위한 AWS KMS 키 생성 및 사용 및 AWS KMS 키를 생성하여 Amazon S3 대상 객체 암호화 단원을 참조하십시오.

주제

- Amazon Redshift 대상 데이터를 암호화하기 위한 사용자 지정 AWS KMS 암호화 키에 대한 정책

- Amazon S3 대상 데이터를 암호화하기 위한 사용자 지정 AWS KMS 암호화 키에 대한 정책

Amazon Redshift 대상 데이터를 암호화하기 위한 사용자 지정 AWS KMS 암호화 키에 대한 정책

다음 예제에서는 Amazon Redshift 대상 데이터를 암호화하기 위해 생성하는 AWS KMS 암호화 키에 대해 생성된 키 정책의 JSON을 보여줍니다.

```
{
  "Id": "key-consolepolicy-3",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:root"
        ]
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow access for Key Administrators",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:role/Admin"
        ]
      },
      "Action": [
        "kms:Create*",
        "kms:Describe*",
        "kms:Enable*",
        "kms:List*",
        "kms:Put*",
        "kms:Update*",
        "kms:Revoke*",
        "kms:Disable*",
        "kms:Get*",
        "kms:Delete*",
```

```
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ScheduleKeyDeletion",
        "kms:CancelKeyDeletion"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow use of the key",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:role/DMS-Redshift-endpoint-access-role"
        ]
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow attachment of persistent resources",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:role/DMS-Redshift-endpoint-access-role"
        ]
      },
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": "*",
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": true
        }
      }
    }
```

```
  ]
}
```

여기서 키 정책이 키를 생성하기 전에 Amazon Redshift 대상 엔드포인트 데이터에 액세스하기 위해 생성한 역할을 참조하는 위치를 확인할 수 있습니다. 이 예제에서는 `DMS-Redshift-endpoint-access-role`입니다. 또한 다른 보안 주체(사용자 및 역할)에게 허용된 다양한 키 작업도 볼 수 있습니다. 예를 들어 `DMS-Redshift-endpoint-access-role`이 부여된 모든 사용자는 대상 데이터를 암호화, 해독 및 다시 암호화할 수 있습니다. 또한 이러한 사용자는 내보내기용 데이터 키를 생성하여 외부 데이터를 암호화할 수 있습니다. AWS KMS 또한 방금 생성한 AWS KMS 키와 같은 키에 대한 세부 정보를 반환할 수도 있습니다. 이 밖에도 이러한 사용자는 대상 엔드포인트와 같은 AWS 리소스에 대한 연결을 관리할 수 있습니다.

Amazon S3 대상 데이터를 암호화하기 위한 사용자 지정 AWS KMS 암호화 키에 대한 정책

다음 예제에서는 Amazon S3 대상 데이터를 암호화하기 위해 생성하는 AWS KMS 암호화 키에 대해 생성된 키 정책의 JSON을 보여줍니다.

```
{
  "Id": "key-consolepolicy-3",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:root"
        ]
      },
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow access for Key Administrators",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:role/Admin"
        ]
      },
```

```
      "Action": [
        "kms:Create*",
        "kms:Describe*",
        "kms:Enable*",
        "kms:List*",
        "kms:Put*",
        "kms:Update*",
        "kms:Revoke*",
        "kms:Disable*",
        "kms:Get*",
        "kms:Delete*",
        "kms:TagResource",
        "kms:UntagResource",
        "kms:ScheduleKeyDeletion",
        "kms:CancelKeyDeletion"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow use of the key",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:role/DMS-S3-endpoint-access-role"
        ]
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Allow attachment of persistent resources",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::987654321098:role/DMS-S3-endpoint-access-role"
        ]
      },
      "Action": [
```

```
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": "*",
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": true
        }
      }
    }
  ]
```

여기서 키 정책이 키를 생성하기 전에 Amazon S3 대상 엔드포인트 데이터에 액세스하기 위해 생성한 역할을 참조하는 위치를 확인할 수 있습니다. 이 예제에서는 `DMS-S3-endpoint-access-role`입니다. 또한 다른 보안 주체(사용자 및 역할)에게 허용된 다양한 키 작업도 볼 수 있습니다. 예를 들어 `DMS-S3-endpoint-access-role`이 부여된 모든 사용자는 대상 데이터를 암호화, 해독 및 다시 암호화할 수 있습니다. 또한 이러한 사용자는 내보내기용 데이터 키를 생성하여 외부 데이터를 암호화할 수 있습니다. AWS KMS 또한 방금 생성한 AWS KMS 키와 같은 키에 대한 세부 정보를 반환할 수도 있습니다. 이 밖에도 이러한 사용자는 대상 엔드포인트와 같은 AWS 리소스에 대한 연결을 관리할 수 있습니다.

보안 암호를 사용하여 AWS Database Migration Service 엔드포인트에 액세스하기

예를 들어 AWS DMS, 암호는 암호 인증을 통해 지원되는 AWS DMS 원본 또는 대상 엔드포인트에 대한 데이터베이스 연결을 인증하기 위해 일련의 사용자 자격 증명을 나타내는 데 사용할 수 있는 암호화된 키입니다. Oracle ASM (자동 저장소 관리) 도 사용하는 Oracle 엔드포인트의 경우 Oracle ASM에 액세스하기 위한 사용자 자격 증명을 나타내는 추가 암호가 AWS DMS 필요합니다.

클라우드와 온프레미스에서 애플리케이션 AWS Secrets Manager, 서비스 및 IT 리소스에 액세스하기 위한 자격 증명을 안전하게 생성, 저장 및 검색하는 서비스를 사용하여 비밀 인증에 AWS DMS 필요한 암호 또는 암호를 생성할 수 있습니다. 여기에는 사용자 개입 없이 암호화된 보안 암호 값을 자동으로 주기적으로 교체할 수 있는 지원이 포함되므로 보안 인증에 대한 보안 수준이 더 향상됩니다. AWS Secrets Manager 또한 암호 값 순환을 활성화하면 암호에 의존하는 데이터베이스 마이그레이션에 영향을 주지 않고 이러한 비밀 값 순환이 이루어집니다. 엔드포인트 데이터베이스 연결을 비밀리에 인증하려면 `SecretsManagerSecretId`에 할당하는 ID 또는 ARN에 대한 보안 암호를 생성하십시오. 이 암호는 엔드포인트 설정에 포함해야 합니다. Oracle 엔드포인트의 일부로서 Oracle ASM을 비밀리에

인증하려면 SecretsManagerOracleAsmSecretId에 할당하는 ID 또는 ARN에 대한 보안 암호를 생성하십시오. 이 암호는 엔드포인트 설정에도 포함해야 합니다.

> **Note**
>
> Amazon RDS Aurora에서 관리하는 마스터 보안 인증은 사용할 수 없습니다. 이러한 자격 증명에는 연결을 설정하는 데 AWS DMS 필요한 호스트 또는 포트 정보가 포함되지 않습니다. 그 대신에 새 사용자와 보안 암호를 생성합니다. 사용자 및 보안 암호 생성에 관한 자세한 내용은 다음 를 AWS Management Console 사용하여 비밀 및 비밀 액세스 역할 생성을 참조하세요.

에 대한 자세한 내용은 AWS Secrets Manager AWS Secrets Manager란? 을 참조하십시오. AWS Secrets Manager 사용 설명서에서.

AWS DMS 지원되는 소스 및 대상 엔드포인트에서 다음과 같은 온프레미스 또는 AWS 관리형 데이터베이스에 대한 비밀 인증을 지원합니다.

- Amazon DocumentDB
- IBM Db2 LUW
- Microsoft SQL Server
- MongoDB
- MySQL
- Oracle
- PostgreSQL
- Amazon Redshift
- SAP ASE

이러한 데이터베이스에 연결하려면 엔드포인트 설정의 일부로 다음 값 집합 중 하나를 입력할 수 있지만 둘 다 입력할 수는 없습니다.

- UserName, Password, ServerName 및 Port 설정을 사용하여 데이터베이스 연결을 인증하기 위한 일반 텍스트 값. Oracle ASM을 함께 사용하는 Oracle 엔드포인트의 경우, AsmUserName, AsmPassword 및 AsmServerName 설정을 사용하여 ASM을 인증하기 위한 추가 일반 텍스트 값을 포함하십시오.
- SecretsManagerSecretId 및 SecretsManagerAccessRoleArn 설정 값을 사용하는 보안 암호 인증. Oracle ASM을 사용하는 Oracle 엔드포인트의 경우,

SecretsManagerOracleAsmSecretId 및 SecretsManagerOracleAsmAccessRoleArn 설정에 대한 추가 값을 포함하십시오. 이러한 설정의 보안 암호 값에는 다음과 같은 항목이 포함될 수 있습니다.

- SecretsManagerSecretId – AWS Secrets Manager에서 엔드포인트 데이터베이스 액세스를 위해 생성한 보안 암호의 전체 Amazon 리소스 이름(ARN), 부분 ARN 또는 기억하기 쉬운 이름.
- SecretsManagerAccessRoleArn— 사용자를 대신하여 이 SecretsManagerSecretId 비밀에 대한 액세스를 제공하기 위해 IAM에서 생성한 보안 AWS DMS 액세스 역할의 ARN.
- SecretsManagerOracleAsmSecretId – AWS Secrets Manager에서 Oracle ASM에 액세스하기 위해 생성한 보안 암호의 전체 Amazon 리소스 이름(ARN), 부분 ARN 또는 기억하기 쉬운 이름.
- SecretsManagerOracleAsmAccessRoleArn – 사용자를 대신하여 이 SecretsManagerOracleAsmSecretId 보안 암호에 대한 AWS DMS 액세스를 제공하기 위해 IAM에서 생성한 보안 암호 액세스 역할의 ARN.

> **Note**
>
> 또한 단일 암호 액세스 역할을 사용하여 암호와 SecretsManagerSecretId 암호 모두에 AWS DMS 대한 액세스를 제공할 수 있습니다. SecretsManagerOracleAsmSecretId 두 보안 암호에 대해 이 단일 보안 암호 액세스 역할을 생성하는 경우, 이 액세스 역할에 대한 동일한 ARN을 SecretsManagerAccessRoleArn 및 SecretsManagerOracleAsmAccessRoleArn에 모두 할당해야 합니다. 예를 들어, 두 보안 암호에 대한 보안 액세스 역할에서 해당 ARN ARN2xsecrets이 변수에 할당된 경우, 다음과 같이 ARN 설정을 지정할 수 있습니다.
>
> ```
> SecretsManagerAccessRoleArn = ARN2xsecrets;
> SecretsManagerOracleAsmAccessRoleArn = ARN2xsecrets;
> ```

이러한 값을 생성하는 방법에 관한 자세한 내용은 를 AWS Management Console 사용하여 비밀 및 비밀 액세스 역할 생성 단원을 참조하십시오.

엔드포인트에 필요한 보안 암호 및 보안 암호 액세스 역할 엔드포인트 설정을 생성하고 지정한 후에는 이러한 보안 암호 세부 정보로 CreateEndpoint 또는 ModifyEndpoint API 요청을 실행할 사용자 계정의 권한을 업데이트하십시오. 이러한 계정 권한에 보안 액세스 역할에 대한 권한과 비밀에 대한 SecretsManager:DescribeSecret 권한이 포함되는지 확인하십시오. IAM:GetRole AWS DMS 액세스 역할과 암호를 모두 검증하려면 이러한 권한이 필요합니다.

필수 사용자 권한을 제공하고 확인하려면

1. 에 AWS Management Console 로그인하고 에서 AWS Identity and Access Management 콘솔을 엽니다https://console.aws.amazon.com/iam/.
2. 사용자를 선택한 다음, `CreateEndpoint` 및 `ModifyEndpoint` API 직접 호출에 사용되는 사용자 ID를 선택합니다.
3. 권한 탭에서 {} JSON을 선택합니다.
4. 사용자에게 다음과 같은 권한이 있는지 확인하십시오.

```
{
 "Statement": [{
   "Effect": "Allow",
   "Action": [
    "iam:GetRole",
    "iam:PassRole"
   ],
   "Resource": "SECRET_ACCESS_ROLE_ARN"
  },
  {
   "Effect": "Allow",
   "Action": "secretsmanager:DescribeSecret",
   "Resource": "SECRET_ARN"
  }
 ]
}
```

5. 사용자에게 해당 권한이 없는 경우 권한을 추가하세요.
6. IAM 역할을 사용하여 DMS API 직접 호출을 하는 경우, 각 역할에 대해 위 단계를 반복하세요.
7. 터미널을 열고 AWS CLI 를 사용하여 위에서 사용한 역할 또는 사용자를 가정하여 권한이 올바르게 부여되었는지 확인합니다.

   a. IAM `get-role` 명령어 SecretAccessRole 사용에 대한 사용자의 권한을 확인합니다.

   ```
   aws iam get-role --role-name ROLE_NAME
   ```

   *ROLE_NAME*을 `SecretsManagerAccessRole`의 이름으로 바꿉니다.

   명령에서 오류 메시지가 반환되는 경우, 권한이 올바르게 제공되었는지 확인하십시오.

b. Secrets Manager `describe-secret` 명령을 사용하여 보안 암호에 대한 사용자 권한을 확인합니다.

```
aws secretsmanager describe-secret --secret-id SECRET_NAME OR SECRET_ARN --
region=REGION_NAME
```

사용자는 기억하기 쉬운 이름, 부분 ARN 또는 전체 ARN일 수 있습니다. 자세한 내용은 describe-secret을 참조하십시오.

명령에서 오류 메시지가 반환되는 경우, 권한이 올바르게 제공되었는지 확인하십시오.

를 AWS Management Console 사용하여 비밀 및 비밀 액세스 역할 생성

를 사용하여 엔드포인트 인증을 위한 비밀번호를 생성하고 사용자 대신 비밀에 액세스할 수 AWS DMS 있도록 허용하는 정책 및 역할을 생성할 수 있습니다. AWS Management Console

원본 및 대상 엔드포인트 연결에 AWS Management Console 대한 데이터베이스를 인증하는 데 사용할 AWS DMS 수 있는 암호를 사용하여 암호를 만들려면

1. 에 AWS Management Console 로그인하고 에서 AWS Secrets Manager https://console.aws.amazon.com/secretsmanager/ 콘솔을 엽니다.
2. 새 보안 암호 저장을 선택합니다.
3. 새 보안 암호 저장 페이지의 보안 암호 유형 선택에서 다른 유형의 보안 암호를 선택한 다음, 일반 텍스트를 선택합니다.

> **Note**
>
> 이 시점부터 엔드포인트 데이터베이스에 연결하기 위해 일반 텍스트 보안 인증 정보를 입력해야 하는 곳은 여기뿐입니다.

4. 일반 텍스트 필드에서:

   - `SecretsManagerSecretId`에 할당할 ID에 대한 보안 암호의 경우, 다음 JSON 구조를 입력합니다.

```
{
  "username": db_username,
```

```
  "password": db_user_password,
  "port": db_port_number,
  "host": db_server_name
}
```

> **Note**
>
> 이것은 엔드포인트 데이터베이스를 인증하는 데 필요한 JSON 멤버의 최소 목록입니다. 원하는 모든 추가 JSON 엔드포인트 설정을 모두 소문자로 JSON 멤버로 추가할 수 있습니다. 하지만 AWS DMS 는 엔드포인트 인증을 위한 추가 JSON 멤버를 무시합니다.

여기서 *db_username*은 다음 예제와 같이 데이터베이스에 액세스하는 사용자의 이름이고, *db_user_password*는 데이터베이스 사용자의 비밀번호이며, *db_port_number*는 데이터베이스에 액세스하기 위한 포트 번호이고, *db_server_name*은 웹상의 데이터베이스 서버 이름(주소)입니다.

```
{
  "username": "admin",
  "password": "some_password",
  "port": "8190",
  "host": "oracle101.abcdefghij.us-east-1.rds.amazonaws.com"
}
```

- `SecretsManagerOracleAsmSecretId`에 할당할 ID에 대한 보안 암호의 경우, 다음 JSON 구조를 입력합니다.

```
{
  "asm_user": asm_username,
  "asm_password": asm_user_password,
  "asm_server": asm_server_name
}
```

> **Note**
>
> 이것은 Oracle 엔드포인트에 대한 Oracle ASM을 인증하는 데 필요한 JSON 멤버의 최소 목록입니다. 또한 사용 가능한 Oracle ASM 엔드포인트 설정을 기반으로 지정할 수 있는 전체 목록이기도 합니다.

여기서 *asm_username*은 다음 예제와 같이 Oracle ASM에 액세스하는 사용자의 이름이고, *asm_user_password*는 Oracle ASM 사용자의 비밀번호이며, *asm_server_name*은 포트를 포함한 웹상의 Oracle ASM 서버 이름(주소)입니다.

```
{
  "asm_user": "oracle_asm_user",
  "asm_password": "oracle_asm_password",
  "asm_server": "oracle101.abcdefghij.us-east-1.rds.amazonaws.com:8190/+ASM"
}
```

5. 암호를 암호화할 AWS KMS 암호화 키를 선택합니다. 를 사용하여 서비스용으로 생성한 기본 암호화 키를 그대로 AWS Secrets Manager 사용하거나 직접 생성한 AWS KMS 키를 선택할 수 있습니다.
6. 이 보안 암호를 참조할 이름과 설명(선택 사항)을 지정합니다. 이 이름은 `SecretsManagerSecretId` 또는 `SecretsManagerOracleAsmSecretId` 값으로 사용할 수 있는 기억하기 쉬운 이름입니다.
7. 암호에서 자동 교체를 활성화하려면 설명된 대로 암호의 자격 증명을 교체할 권한이 있는 AWS Lambda 함수를 선택하거나 생성해야 합니다. 그러나 Lambda 함수를 사용하도록 자동 교체를 설정하기 전에 함수에 대한 구성 설정이 `EXCLUDE_CHARACTERS` 환경 변수 값에 다음 4자를 추가하는지 확인하십시오.

   ```
   ;.:+{}
   ```

   AWS DMS 엔드포인트 자격 증명에 사용되는 암호에는 이러한 문자를 사용할 수 없습니다. 이러한 문자를 제외하도록 Lambda 함수를 구성하면 AWS Secrets Manager 은 이러한 문자를 교체된 비밀번호 값의 일부로 생성할 수 없습니다. Lambda 함수를 사용하도록 자동 교체를 설정한 후 AWS Secrets Manager , 즉시 시크릿을 교체하여 시크릿 구성을 검증합니다.

   > **Note**
   >
   > 데이터베이스 엔진 구성에 따라 데이터베이스는 교체된 보안 인증을 가져오지 못할 수도 있습니다. 이 경우, 보안 인증을 새로 고치려면 작업을 수동으로 다시 시작해야 합니다.

8. 시크릿을 검토하고 저장하십시오. AWS Secrets Manager 그런 다음 에서 AWS Secrets Manager 알기 쉬운 이름으로 각 암호를 검색한 다음 엔드포인트 데이터베이스 연결 및 Oracle ASM

(사용하는 경우) 에 대한 액세스를 인증하는 데 적합한 SecretsManagerSecretId 값이나 SecretsManagerOracleAsmSecretId 암호 ARN을 검색할 수 있습니다.

보안 액세스 정책 및 역할을 생성하여 적절한 비밀번호에 대한 액세스를 허용하는 **SecretsManagerAccessRoleArnSecretsManagerOracleAsmAccessRoleArn**AWS DMS OR 을 AWS Secrets Manager 설정하려면

1. AWS Management Console 로그인하고 https://console.aws.amazon.com/iam/ 에서 AWS Identity and Access Management (IAM) 콘솔을 엽니다.
2. 정책을 선택한 후 정책 생성을 선택합니다.
3. JSON을 선택하고 다음 정책을 입력하여 보안 암호에 대한 액세스 및 암호 해독을 활성화합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": secret_arn,
        },
        {
             "Effect": "Allow",
             "Action": [
                        "kms:Decrypt",
                        "kms:DescribeKey"
                      ],
             "Resource": kms_key_arn,
        }
     ]
}
```

여기서 *secret_arn*은 보안 암호의 ARN(적절하다면 SecretsManagerSecretId 또는 SecretsManagerOracleAsmSecretId에서 가져올 수 있음)이고, *kms_key_arn*은 다음 예제와 같이 보안 암호를 암호화하는 데 사용하는 AWS KMS 키의 ARN입니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:us-
east-2:123456789012:secret:MySQLTestSecret-qeHamH"
        },
        {
             "Effect": "Allow",
             "Action": [
                       "kms:Decrypt",
                       "kms:DescribeKey"
                     ],
             "Resource": "arn:aws:kms:us-
east-2:123456789012:key/761138dc-0542-4e58-947f-4a3a8458d0fd"
        }
     ]
}
```

> **Note**
>
> 에서 생성한 AWS Secrets Manager 기본 암호화 키를 사용하는 경우 AWS KMS *kms_key_arn* 권한을 지정하지 않아도 됩니다.
> 정책에서 두 보안 암호에 대한 액세스를 제공하도록 하려면 다른 *secret_arn*에 대해 추가 JSON 리소스 객체를 지정하면 됩니다.
> 보안 암호가 다른 계정에 있는 경우, SecretsManagerAccessRoleArn 역할에는 계정 간 보안 암호를 확인하기 위한 추가 정책이 필요합니다. 이러한 사용 사례의 경우, 정책에 작업 secretsmanager:DescribeSecret을 추가하세요. 교차 계정 암호 설정에 대한 자세한 내용은 다른 계정의 사용자에 대한 AWS Secrets Manager 암호에 대한 권한을 참조하십시오.

4. 기억하기 쉬운 이름과 설명(선택 사항)이 있는 정책을 검토하고 생성합니다.
5. 역할을 선택한 다음, 역할 생성을 선택합니다.
6. 신뢰할 수 있는 엔터티 유형으로서 AWS 서비스를 선택합니다.
7. 서비스 목록에서 DMS를 신뢰할 수 있는 서비스로 선택하고 다음: 권한을 선택합니다.
8. 4단계에서 생성한 정책을 찾아 연결한 다음, 계속 진행하여 태그를 추가하고 역할을 검토합니다. 이제 AWS DMS 지역 서비스 보안 주체를 신뢰할 수 있는 개체로 사용하도록 역할의 신뢰 관계를 편집하십시오. 이 보안 주체의 형식은 다음과 같습니다.

```
dms.region-name.amazonaws.com
```

여기서 *region-name*은 리전의 이름입니다(예: us-east-1). 따라서 이 지역의 AWS DMS 지역 서비스 주체는 다음과 같습니다.

```
dms.us-east-1.amazonaws.com
```

9. 역할의 신뢰할 수 있는 엔터티를 편집한 후 기억하기 쉬운 이름과 선택적 설명을 사용하여 역할을 생성합니다. 이제 IAM에서 기억하기 쉬운 이름으로 새 역할을 검색한 다음, 역할 ARN을 SecretsManagerAccessRoleArn 또는 SecretsManagerOracleAsmAccessRoleArn 값으로 검색하여 엔드포인트 데이터베이스 연결을 인증할 수 있습니다.

프라이빗 서브넷의 복제 인스턴스와 함께 Secrets Manager를 사용하려면

1. Secrets Manager VPC 엔드포인트를 생성하고 이 엔드포인트의 DNS를 기록해 둡니다. Secrets Manager VPC 엔드포인트를 생성하는 방법에 관한 자세한 내용은 AWS Secrets Manager 사용 설명서의 VPC 엔드포인트를 통해 Secrets Manager에 연결하기를 참조하십시오.
2. 복제 인스턴스 보안 그룹을 Secret Manager VPC 엔드포인트에 연결합니다.
3. 복제 인스턴스 보안 그룹 송신 규칙의 경우, 대상 0.0.0.0/0으로 전송되는 모든 트래픽을 허용하십시오.
4. 다음 예제와 같이 Secret Manager VPC 엔드포인트 DNS를 제공하려면 엔드포인트 추가 연결 속성 secretsManagerEndpointOverride=*secretsManager endpoint DNS*를 설정합니다.

```
secretsManagerEndpointOverride=vpce-1234a5678b9012c-12345678.secretsmanager.eu-
west-1.vpce.amazonaws.com
```

AWS DMS에 서비스 연결 역할 사용

AWS Database Migration Service은 AWS Identity and Access Management(IAM) 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 AWS DMS에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 AWS DMS에서 사전 정의하며 서비스에서 다른 AWS 서비스를 자동으로 호출하기 위해 필요한 모든 권한을 포함합니다.

서비스 연결 역할을 통해 AWS DMS 설정이 쉬워지는데 필요한 권한을 수동으로 추가할 필요가 없기 때문입니다. AWS DMS에서 서비스 연결 역할 권한을 정의하므로, 달리 정의되지 않은 한 AWS DMS에서만 해당 역할을 맡을 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며, 이 권한 정책은 다른 IAM 엔터티에 연결할 수 없습니다.

먼저 관련 리소스를 삭제한 후에만 서비스 연결 역할을 삭제할 수 있습니다. 이렇게 하면 리소스에 대한 액세스 권한을 부주의로 삭제할 수 없기 때문에 AWS DMS 리소스가 보호됩니다.

서비스 연결 역할을 지원하는 기타 서비스에 관한 자세한 내용을 알아보려면 IAM으로 작업하는 AWS 서비스를 참조하고 서비스 연결 역할 열에 예가 표시된 서비스를 찾으세요. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예 링크를 선택합니다.

AWS DMS 기능에 대한 서비스 연결 역할

주제

- AWS DMS Fleet Advisor의 서비스 연결 역할
- AWS DMS Serverless의 서비스 연결 역할

AWS DMS Fleet Advisor의 서비스 연결 역할

AWS DMS Fleet Advisor는 AWSServiceRoleForDMSFleetAdvisor라는 서비스 연결 역할을 사용 – DMS FleetAdvisor는 서비스 연결 역할을 사용하여 Amazon CloudWatch 지표를 관리합니다. 이 서비스 연결 역할은 관리형 정책 `AWSDMSFleetAdvisorServiceRolePolicy`에 연결됩니다. 이 정책에 대한 업데이트는 AWS Database Migration Service의 AWS 관리형 정책 단원을 참조하세요.

AWSServiceRoleForDMSFleetAdvisor 서비스 연결 역할은 역할을 수임하기 위해 다음 서비스를 신뢰합니다.

- `dms-fleet-advisor.amazonaws.com`

이름이 AWSDMSFleetAdvisorServiceRolePolicy인 역할 권한 정책은 지정된 리소스에 대해 AWS DMS Fleet Advisor가 다음 작업을 완료하도록 허용합니다.

- 작업: `all AWS resources`에 대한 `cloudwatch:PutMetricData`

  이 권한은 보안 주체가 Amazon CloudWatch에 지표 데이터 포인트를 게시할 수 있도록 허용합니다. AWS DMS Fleet Advisor가 CloudWatch에서 데이터베이스 지표와 함께 차트를 표시하려면 이 권한이 필요합니다.

다음 코드 예제는 AWSDMSFleetAdvisorServiceRolePolicy 역할을 생성하는 데 사용하는 AWSDMSFleetAdvisorServiceRolePolicy 정책을 보여줍니다.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Resource": "*",
        "Action": "cloudwatch:PutMetricData",
        "Condition": {
            "StringEquals": {
                "cloudwatch:namespace": "AWS/DMS/FleetAdvisor"
            }
        }
    }
}
```

IAM 엔터티(사용자, 그룹, 역할 등)가 서비스 연결 역할을 생성하고 편집하거나 삭제할 수 있도록 허용하는 권한을 구성해야 합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하세요.

AWS DMS Fleet Advisor에 대한 서비스 연결 역할 생성

IAM 콘솔을 사용하여 DMS – Fleet Advisor 사용 사례에서 서비스 연결 역할을 생성할 수 있습니다. AWS CLI 또는 AWS API에서 `dms-fleet-advisor.amazonaws.com` 서비스 이름의 서비스 연결 역할을 생성합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 생성을 참조하세요. 이 서비스 연결 역할을 삭제하면 동일한 프로세스를 사용하여 역할을 다시 생성할 수 있습니다.

데이터 수집기를 생성하기 전에 이 역할을 생성해야 합니다. DMS Fleet Advisor는 이 역할을 사용하여 AWS Management Console에서 데이터베이스 지표가 포함된 차트를 표시합니다. 자세한 내용은 데이터 수집기 생성 섹션을 참조하세요.

AWS DMS Fleet Advisor에 대한 서비스 연결 역할 편집

AWS DMS는 AWSServiceRoleForDMSFleetAdvisor 서비스 연결 역할을 편집하도록 허용하지 않습니다. 서비스 연결 역할을 생성한 후에는 다양한 엔터티가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

AWS DMS Fleet Advisor에 대한 서비스 연결 역할 삭제

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 결국 적극적으로 모니터링하거나 유지 관리하지 않는 미사용 개체가 없게 됩니다. 단, 서비스 연결 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

> **Note**
>
> 리소스를 삭제하려 할 때 AWS DMS 서비스가 역할을 사용 중이면 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

AWSServiceRoleForDMSFleetAdvisor에서 사용하는 AWS DMS 리소스를 삭제하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/dms/v2/에서 AWS DMS 콘솔을 엽니다.
2. 탐색 창에서 검색 아래에 있는 데이터 수집기를 선택합니다. 데이터 수집기 페이지가 열립니다.
3. 데이터 수집기를 선택하고 삭제를 선택합니다.
4. 그런 다음, 텍스트 입력 필드에 데이터 수집기 이름을 입력합니다. 다음으로 삭제를 선택합니다.

> **Important**
>
> DMS 데이터 수집기를 삭제하면 DMS Fleet Advisor는 이 수집기를 사용하여 검색한 모든 데이터베이스를 인벤토리에서 삭제합니다.

데이터 수집기를 모두 삭제한 후 서비스 연결 역할을 삭제할 수 있습니다.

IAM을 사용하여 수동으로 서비스 연결 역할을 삭제하려면

IAM 콘솔, AWS CLI 또는 AWS API를 사용하여 AWSServiceRoleForDMSFleetAdvisor 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서에서 서비스 연결 역할 삭제를 참조하세요.

AWS DMS Fleet Advisor 서비스 연결 역할에 대해 지원되는 리전

AWS DMS Fleet Advisor는 서비스가 제공되는 모든 리전에서 서비스 연결 역할을 사용하도록 지원합니다. 자세한 내용은 지원되는 AWS 리전 섹션을 참조하세요.

AWS DMS Serverless의 서비스 연결 역할

AWS DMS 서버리스는 이름이 지정된 서비스 연결 역할을 사용합니다. AWSServiceRoleForDMSServerless AWS DMS 는 이 서비스 연결 역할을 사용하여 Amazon CloudWatch Metrics와 같은 AWS DMS 리소스를 사용자 대신 생성하고 관리합니다. AWS DMS 이 역할을 사용하므로 복제에만 신경 쓰면 됩니다. 이 서비스 연결 역할은 관리형 정책

AWSDMSServerlessServiceRolePolicy에 연결됩니다. 이 정책에 대한 업데이트는 AWS Database Migration Service의 AWS 관리형 정책 단원을 참조하세요.

AWSServiceRoleForDMSServerless 서비스 연결 역할은 다음 서비스가 역할을 맡을 것으로 신뢰합니다.

- dms.amazonaws.com

다음 코드 예제는 역할을 생성하는 데 사용하는 AWSDMSServerlessServiceRolePolicy 정책을 보여줍니다. AWSServiceRoleForDMSServerless

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "id0",
            "Effect": "Allow",
            "Action": [
                "dms:CreateReplicationInstance",
                "dms:CreateReplicationTask"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "dms:req-tag/ResourceCreatedBy": "DMSServerless"
                }
            }
        },
        {
            "Sid": "id1",
            "Effect": "Allow",
            "Action": [
                "dms:DescribeReplicationInstances",
                "dms:DescribeReplicationTasks"
            ],
            "Resource": "*"
        },
        {
            "Sid": "id2",
            "Effect": "Allow",
            "Action": [
                "dms:StartReplicationTask",
```

```
                "dms:StopReplicationTask",
                "dms:DeleteReplicationTask",
                "dms:DeleteReplicationInstance"
            ],
            "Resource": [
                "arn:aws:dms:*:*:rep:*",
                "arn:aws:dms:*:*:task:*"
            ],
            "Condition": {
                "StringEqualsIgnoreCase": {
                    "aws:ResourceTag/ResourceCreatedBy": "DMSServerless"
                }
            }
        },
        {
            "Sid": "id3",
            "Effect": "Allow",
            "Action": [
                "dms:TestConnection",
                "dms:DeleteConnection"
            ],
            "Resource": [
                "arn:aws:dms:*:*:rep:*",
                "arn:aws:dms:*:*:endpoint:*"
            ]
        }
    ]
}
```

IAM 엔터티(사용자, 그룹, 역할 등)가 서비스 연결 역할을 생성하고 편집하거나 삭제할 수 있도록 허용하는 권한을 구성해야 합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하십시오.

AWS DMS Serverless에 대한 서비스 연결 역할 생성

복제를 만들면 AWS DMS 서버리스는 프로그래밍 방식으로 AWS DMS 서버리스 서비스 연결 역할을 생성합니다. 이 역할은 IAM 콘솔에서 볼 수 있습니다. 이 역할을 수동으로 생성하도록 선택할 수도 있습니다. 역할을 수동으로 생성하려면 IAM 콘솔을 사용하여 DMS 사용 사례와 함께 서비스 연결 역할을 생성하십시오. AWS CLI 또는 AWS API에서 서비스 이름을 사용하여 서비스 연결 역할을 생성합니다. `dms.amazonaws.com` 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 생성 섹션을 참조하세요. 이 서비스 연결 역할을 삭제하면 동일한 프로세스를 사용하여 역할을 다시 생성할 수 있습니다.

> **Note**
>
> 계정에 복제가 있는 상태에서 역할을 삭제하면 복제가 실패합니다.

AWS DMS Serverless에 대한 서비스 연결 역할 편집

AWS DMS AWSServiceRoleForDMSServerless 서비스 연결 역할을 편집할 수 없습니다. 서비스 연결 역할을 생성한 후에는 다양한 엔터티가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하십시오.

AWS DMS Serverless에 대한 서비스 연결 역할 삭제

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 결국 적극적으로 모니터링하거나 유지 관리하지 않는 미사용 개체가 없게 됩니다. 단, 서비스 링크 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

> **Note**
>
> AWS DMS 서비스가 역할을 사용하고 있을 때 리소스를 삭제하려고 하면 삭제가 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

에서 사용하는 AWS DMS 리소스를 삭제하려면 AWSServiceRoleForDMSServerless

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 탐색 창의 검색에서 Serverless를 선택합니다. Serverless 페이지가 열립니다.
3. 서버리스 복제를 선택하고 삭제를 선택합니다.
4. 삭제를 확인하려면 텍스트 입력 필드에 서버리스 복제 이름을 입력합니다. 다음으로 삭제를 선택합니다.

서버리스 복제를 모두 삭제한 후 서비스 연결 역할을 삭제할 수 있습니다.

IAM을 사용하여 수동으로 서비스 링크 역할을 삭제하려면

IAM 콘솔 AWS CLI, 또는 AWS API를 사용하여 AWSServiceRoleForDMSServerless 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서에서 서비스 연결 역할 삭제를 참조하세요.

AWS DMS Serverless 서비스 연결 역할에 대해 지원되는 리전

AWS DMS 서버리스는 서비스를 사용할 수 있는 모든 지역에서 서비스 연결 역할을 사용할 수 있도록 지원합니다.

AWS Database Migration Service ID 및 액세스 문제 해결

다음 정보를 사용하면 및 IAM을 사용할 때 발생할 수 있는 일반적인 문제를 AWS DMS 진단하고 해결하는 데 도움이 됩니다.

주제

- 저는 다음과 같은 작업을 수행할 권한이 없습니다. AWS DMS
- 저는 IAM을 수행할 권한이 없습니다. PassRole
- 저는 관리자이며 다른 사람들이 액세스할 수 있도록 허용하고 싶습니다. AWS DMS
- 내 AWS 계정 외부의 사용자가 내 리소스에 액세스할 수 있도록 허용하고 싶습니다. AWS DMS

저는 다음과 같은 작업을 수행할 권한이 없습니다. AWS DMS

작업을 수행할 권한이 없다는 AWS Management Console 메시지가 표시되면 관리자에게 도움을 요청해야 합니다. 관리자는 사용자 이름과 비밀번호를 제공한 사람입니다.

다음 예제 오류는 `mateojackson` IAM 사용자가 콘솔을 사용하여 AWS DMS 엔드포인트에 대한 세부 정보를 보려고 하지만 권한이 없는 경우 `dms: DescribeEndpoint` 발생합니다.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 dms:DescribeEndpoint on resource: my-postgresql-target
```

이 경우 Mateo는 `dms:DescribeEndpoint` 작업을 사용하여 `my-postgresql-target` 엔드포인트 리소스에 액세스하도록 허용하는 정책을 업데이트할 것을 관리자에게 요청합니다.

저는 IAM을 수행할 권한이 없습니다. PassRole

`iam:PassRole` 작업을 수행할 수 있는 권한이 없다는 오류가 수신되면 AWS DMS에 역할을 전달할 수 있도록 정책을 업데이트해야 합니다.

새 서비스 역할 또는 서비스 연결 역할을 만드는 대신 기존 역할을 해당 서비스에 전달할 AWS 서비스 수 있는 기능도 있습니다. 이렇게 하려면 사용자가 서비스에 역할을 전달할 수 있는 권한을 가지고 있어야 합니다.

다음 예 오류는 marymajor라는 IAM 사용자가 콘솔을 사용하여 AWS DMS에서 작업을 수행하려고 하는 경우에 발생합니다. 하지만 작업을 수행하려면 서비스 역할이 부여한 권한이 서비스에 있어야 합니다. Mary는 서비스에 역할을 전달할 수 있는 권한을 가지고 있지 않습니다.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

이 경우 Mary가 iam:PassRole 작업을 수행할 수 있도록 Mary의 정책을 업데이트해야 합니다.

도움이 필요하면 관리자에게 문의하세요. AWS 관리자는 로그인 자격 증명을 제공한 사람입니다.

저는 관리자이며 다른 사람들이 액세스할 수 있도록 허용하고 싶습니다. AWS DMS

다른 사람이 액세스할 수 있도록 하려면 액세스가 AWS DMS 필요한 개인 또는 애플리케이션을 위한 IAM 엔티티 (사용자 또는 역할) 를 생성해야 합니다. 다른 사용자들은 해당 엔터티에 대한 보안 인증을 사용해 AWS에 액세스합니다. 그런 다음 AWS DMS에 대한 올바른 권한을 부여하는 정책을 엔터티에 연결해야 합니다.

바로 시작하려면 IAM 사용 설명서의 첫 번째 IAM 위임 사용자 및 그룹 생성을 참조하십시오.

내 AWS 계정 외부의 사용자가 내 리소스에 액세스할 수 있도록 허용하고 싶습니다. AWS DMS

다른 계정의 사용자 또는 조직 외부의 사람이 리소스에 액세스할 때 사용할 수 있는 역할을 생성할 수 있습니다. 역할을 수임할 신뢰할 수 있는 사람을 지정할 수 있습니다. 리소스 기반 정책 또는 액세스 제어 목록(ACL)을 지원하는 서비스의 경우 이러한 정책을 사용하여 다른 사람에게 리소스에 대한 액세스 권한을 부여할 수 있습니다.

자세히 알아보려면 다음을 참조하세요.

- 이러한 기능의 AWS DMS 지원 여부를 알아보려면 을 참조하십시오IAM의 AWS Database Migration Service 작동 방식.
- 소유한 리소스에 대한 액세스 권한을 AWS 계정 부여하는 방법을 알아보려면 IAM 사용 설명서에서 자신이 소유한 다른 AWS 계정 IAM 사용자에게 액세스 권한 제공을 참조하십시오.
- 제3자에게 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 타사 AWS 계정AWS 계정 소유에 대한 액세스 제공을 참조하십시오.
- ID 페더레이션을 통해 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 외부에서 인증된 사용자에게 액세스 권한 제공(ID 페더레이션)을 참조하세요.

- 크로스 계정 액세스를 위한 역할과 리소스 기반 정책 사용의 차이점을 알아보려면 IAM 사용 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하세요.

AWS DMS 사용에 필요한 IAM 권한

AWS DMS를 사용하려면 특정 IAM 권한 및 IAM 역할을 사용합니다. IAM 사용자로 로그인한 상태에서 사용하려는 경우 계정 관리자가 이 섹션에서 설명하는 정책을 실행에 사용하는 AWS DMS IAM 사용자, 그룹 또는 역할에 연결해야 합니다. AWS DMS IAM 권한에 관한 자세한 내용은 IAM 사용 설명서를 참조하십시오.

다음 정책은 IAM AWS DMS, Amazon EC2 및 Amazon과 같은 AWS KMS 다른 Amazon 서비스에서 필요한 특정 작업에 대한 액세스 및 권한을 제공합니다. CloudWatch CloudWatch AWS DMS 마이그레이션을 실시간으로 모니터링하고 마이그레이션 진행 상황을 나타내는 지표를 수집 및 추적합니다. CloudWatch 로그를 사용하여 작업 문제를 디버깅할 수 있습니다.

> **Note**
>
> 태그 지정을 사용하여 AWS DMS 리소스에 대한 액세스를 추가로 제한할 수 있습니다. 태깅을 사용하여 AWS DMS 리소스에 대한 액세스를 제한하는 방법에 대한 자세한 내용은 을 참조하십시오. 리소스 이름 및 태그를 사용하여 세분화된 액세스 제어

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "dms:*",
            "Resource": "arn:aws:dms:region:account:resourcetype/id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:ListAliases",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:PassRole",
```

```
                "iam:CreateRole",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVpcs",
                "ec2:DescribeInternetGateways",
                "ec2:DescribeAvailabilityZones",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:Get*",
                "cloudwatch:List*"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:DescribeLogStreams",
                "logs:FilterLogEvents",
                "logs:GetLogEvents"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
    ]
}
```

이러한 권한을 분석하면 각 권한이 필요한 이유를 더 잘 이해할 수 있습니다.

사용자가 AWS DMS API 작업을 호출할 수 있도록 하려면 다음 섹션이 필요합니다.

```
{
            "Effect": "Allow",
            "Action": "dms:*",
            "Resource": "arn:aws:dms:region:account:resourcetype/id"
}
```

사용자가 사용 가능한 AWS KMS 키와 별칭을 나열하여 콘솔에 표시할 수 있도록 하려면 다음 섹션이 필요합니다. KMS 키의 Amazon 리소스 이름 (ARN) 을 알고 있고 () 만 사용하는 경우에는 이 항목이 필요하지 않습니다. AWS Command Line Interface AWS CLI

```
{
            "Effect": "Allow",
            "Action": [
                "kms:ListAliases",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
```

엔드포인트를 사용하여 IAM 역할 ARN을 전달해야 하는 특정 엔드포인트 유형에는 다음 섹션이 필요합니다. 또한 필요한 AWS DMS 역할을 미리 생성하지 않은 경우 AWS DMS 콘솔에서 역할을 생성할 수 있습니다. 모든 역할이 사전에 구성된 경우, `iam:GetRole` 및 `iam:PassRole`만 있으면 됩니다. 역할에 관한 자세한 내용은 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성 단원을 참조하세요.

```
{
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:PassRole",
                "iam:CreateRole",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
```

Amazon EC2 인스턴스를 생성하고 생성된 복제 인스턴스에 맞게 네트워크를 AWS DMS 구성해야 하므로 다음 섹션이 필요합니다. 이러한 리소스는 고객의 계정에 있으므로 고객 대신에 이러한 작업을 수행하는 기능이 필요합니다.

```
{
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVpcs",
                "ec2:DescribeInternetGateways",
                "ec2:DescribeAvailabilityZones",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
```

다음 단원에서는 사용자가 복제 인스턴스 지표를 볼 수 있도록 허용해야 합니다.

```
{
            "Effect": "Allow",
            "Action": [
                "cloudwatch:Get*",
                "cloudwatch:List*"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
```

이 단원에서는 사용자가 복제 로그를 볼 수 있도록 허용해야 합니다.

```
{
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:DescribeLogStreams",
                "logs:FilterLogEvents",
                "logs:GetLogEvents"
            ],
            "Resource": "arn:aws:service:region:account:resourcetype/id"
        }
```

AWS DMS 콘솔은 여러 역할을 생성합니다. 이 역할은 AWS DMS 콘솔을 사용할 때 AWS 계정에 자동으로 연결됩니다. 마이그레이션에 AWS Command Line Interface (AWS CLI) 또는 AWS DMS API를

사용하는 경우 계정에 이러한 역할을 추가해야 합니다. 이러한 역할을 추가하는 방법에 관한 자세한 내용은 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성 단원을 참조하십시오.

AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성

데이터베이스 마이그레이션에 AWS CLI 또는 AWS DMS API를 사용하는 경우, AWS 계정에 IAM 역할 3개를 추가해야 의 기능을 사용할 수 있습니다. AWS DMS 이 중 두 가지는 dms-vpc-role과 dms-cloudwatch-logs-role입니다. Amazon Redshift를 대상 데이터베이스로 사용하는 경우 계정에 IAM dms-access-for-endpoint 역할도 추가해야 합니다. AWS

관리형 정책은 자동으로 업데이트됩니다. IAM 역할에서 사용자 지정 정책을 사용하는 경우, 이 설명서의 관리형 정책 업데이트를 정기적으로 확인해야 합니다. 관리형 정책의 세부 정보를 보려면 get-policy와 get-policy-version 명령을 조합하여 사용하면 됩니다.

예를 들어 다음 get-policy 명령은 지정된 IAM 역할에 대한 정보를 검색합니다.

```
aws iam get-policy --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonDMSVPCManagementRole
```

이 명령에서 반환되는 정보는 다음과 같습니다.

```
{
    "Policy": {
        "PolicyName": "AmazonDMSVPCManagementRole",
        "Description": "Provides access to manage VPC settings for AWS managed customer
 configurations",
        "CreateDate": "2015-11-18T16:33:19Z",
        "AttachmentCount": 1,
        "IsAttachable": true,
        "PolicyId": "ANPAJHKIGMBQI4AEFFSYO",
        "DefaultVersionId": "v3",
        "Path": "/service-role/",
        "Arn": "arn:aws:iam::aws:policy/service-role/AmazonDMSVPCManagementRole",
        "UpdateDate": "2016-05-23T16:29:57Z"
    }
}
```

다음 get-policy-version 명령은 IAM 정책 정보를 검색합니다.

```
aws iam get-policy-version --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonDMSVPCManagementRole --version-id v3
```

이 명령에서 반환되는 정보는 다음과 같습니다.

```
{
    "PolicyVersion": {
        "CreateDate": "2016-05-23T16:29:57Z",
        "VersionId": "v3",
        "Document": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": [
                    "ec2:CreateNetworkInterface",
                    "ec2:DescribeAvailabilityZones",
                    "ec2:DescribeInternetGateways",
                    "ec2:DescribeSecurityGroups",
                    "ec2:DescribeSubnets",
                    "ec2:DescribeVpcs",
                    "ec2:DeleteNetworkInterface",
                    "ec2:ModifyNetworkInterfaceAttribute"
                ],
                "Resource": "arn:aws:service:region:account:resourcetype/id",
                "Effect": "Allow"
            }
        ]
    },
    "IsDefaultVersion": true
    }
}
```

동일한 명령을 사용하여 AmazonDMSCloudWatchLogsRole 및 AmazonDMSRedshiftS3Role 관리형 정책에 관한 정보를 가져올 수 있습니다.

> **Note**
>
> AWS DMS 콘솔을 사용하여 데이터베이스 마이그레이션을 수행하는 경우 이러한 역할이 AWS 계정에 자동으로 추가됩니다.

다음 절차에 따라 dms-vpc-role, dms-cloudwatch-logs-role, dms-access-for-endpoint IAM 역할을 생성합니다.

AWS CLI 또는 AWS DMS API와 함께 사용할 dms-vpc-role IAM 역할을 만들려면

1. 다음과 같은 IAM 정책이 있는 JSON 파일을 생성합니다. JSON 파일의 이름을 dmsAssumeRolePolicyDocument.json으로 지정합니다.

```
{
   "Version": "2012-10-17",
   "Statement": [
   {
     "Effect": "Allow",
     "Principal": {
        "Service": "dms.amazonaws.com"
     },
   "Action": "sts:AssumeRole"
   }
 ]
}
```

   다음 명령을 사용하여 역할을 생성합니다. AWS CLI

```
aws iam create-role --role-name dms-vpc-role --assume-role-policy-document file://
dmsAssumeRolePolicyDocument.json
```

2. 다음 명령을 사용하여 AmazonDMSVPCManagementRole 정책을 dms-vpc-role에 연결합니다.

```
aws iam attach-role-policy --role-name dms-vpc-role --policy-arn
 arn:aws:iam::aws:policy/service-role/AmazonDMSVPCManagementRole
```

AWS CLI 또는 AWS DMS API와 함께 사용할 dms-cloudwatch-logs-role IAM 역할을 만들려면

1. 다음과 같은 IAM 정책이 있는 JSON 파일을 생성합니다. JSON 파일의 이름을 dmsAssumeRolePolicyDocument2.json으로 지정합니다.

```
{
   "Version": "2012-10-17",
   "Statement": [
   {
     "Effect": "Allow",
     "Principal": {
        "Service": "dms.amazonaws.com"
     },
   "Action": "sts:AssumeRole"
   }
 ]
}
```

   다음 명령을 사용하여 역할을 생성합니다. AWS CLI

```
aws iam create-role --role-name dms-cloudwatch-logs-role --assume-role-policy-
document file://dmsAssumeRolePolicyDocument2.json
```

2. 다음 명령을 사용하여 AmazonDMSCloudWatchLogsRole 정책을 dms-cloudwatch-logs-role에 연결합니다.

```
aws iam attach-role-policy --role-name dms-cloudwatch-logs-role --policy-arn
 arn:aws:iam::aws:policy/service-role/AmazonDMSCloudWatchLogsRole
```

대상 데이터베이스로서 Amazon Redshift를 사용하는 경우, IAM 역할 dms-access-for-endpoint를 생성하여 Amazon S3에 대한 액세스 권한을 부여해야 합니다.

Amazon Redshift와 함께 대상 데이터베이스로 사용할 dms-access-for-endpoint IAM 역할을 만들려면

1. 다음과 같은 IAM 정책이 있는 JSON 파일을 생성합니다. JSON 파일의 이름을 dmsAssumeRolePolicyDocument3.json으로 지정합니다.

```
 {
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
        "Service": "dms.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    },
    {
      "Sid": "2",
      "Effect": "Allow",
      "Principal": {
        "Service": "redshift.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. 다음 명령을 사용하여 역할을 생성합니다. AWS CLI

```
  aws iam create-role --role-name dms-access-for-endpoint --assume-role-policy-
document file://dmsAssumeRolePolicyDocument3.json
```

3. 다음 명령을 사용하여 AmazonDMSRedshiftS3Role 정책을 dms-access-for-endpoint 역할에 연결합니다.

```
aws iam attach-role-policy --role-name dms-access-for-endpoint \
```

```
    --policy-arn arn:aws:iam::aws:policy/service-role/AmazonDMSRedshiftS3Role
```

이제 AWS CLI 또는 AWS DMS API를 사용할 수 있는 IAM 정책을 마련해야 합니다.

교차 서비스 혼동된 대리인 방지

혼동된 대리인 문제는 작업을 수행할 권한이 없는 개체가 권한이 더 많은 개체에게 작업을 수행하도록 강요할 수 있는 보안 문제입니다. 에서 AWS크로스 서비스 사칭으로 인해 대리인 문제가 발생할 수 있습니다. 교차 서비스 가장은 한 서비스(호출하는 서비스)가 다른 서비스(호출되는 서비스)를 호출할 때 발생할 수 있습니다. 직접적으로 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 AWS 에서는 계정의 리소스에 대한 액세스 권한이 부여된 서비스 보안 주체를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 제공합니다.

리소스 정책에 `aws:SourceArn`및 `aws:SourceAccount`글로벌 조건 컨텍스트 키를 사용하여 리소스에 다른 서비스에 AWS Database Migration Service 부여하는 권한을 제한하는 것이 좋습니다. 만약 `aws:SourceArn` 값에 AWS DMS 복제 인스턴스 이름(ARN)과 같은 계정 ID가 포함되어 있지 않은 경우 권한을 제한하려면 두 전역 조건 컨텍스트 키를 모두 사용해야 합니다. 두 전역 조건 컨텍스트 키와 계정을 포함한 `aws:SourceArn` 값을 모두 사용하는 경우, `aws:SourceAccount` 값 및 `aws:SourceArn` 값의 계정은 동일한 정책 명령문에서 사용할 경우 반드시 동일한 계정 ID를 사용해야 합니다. 하나의 리소스만 교차 서비스 액세스와 연결되도록 허용하려는 경우 `aws:SourceArn`을(를) 사용하세요. 해당 계정의 모든 리소스가 교차 서비스 사용과 연결되도록 허용하려는 경우 `aws:SourceAccount`을(를) 사용하십시오.

AWS DMS 3.4.7 버전 이상부터는 혼동된 보조 옵션을 지원합니다. 자세한 정보는 AWS Database Migration Service 3.4.7 릴리스 노트을 참조하세요. 복제 인스턴스가 AWS DMS 버전 3.4.6 이하를 사용하는 경우 혼동된 대리자 옵션을 설정하기 전에 최신 버전으로 업그레이드해야 합니다.

혼동된 대리인 문제로부터 보호하는 가장 효과적인 방법은 리소스의 전체 ARN이 포함된 `aws:SourceArn` 글로벌 조건 컨텍스트 키를 사용하는 것입니다. 리소스의 전체 ARN을 모르거나 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드 문자(*)를 포함한 `aws:SourceArn` 글로벌 조건 컨텍스트 키를 사용합니다. 예를 들어 `arn:aws:dms:*:123456789012:rep:*`입니다.

주제

- 서비스 간 혼란을 방지하기 위해 AWS DMS API와 함께 사용할 IAM 역할
- 교차 서비스 혼동된 대리자 방지를 위해 Amazon S3에 사전 평가를 저장하기 위한 IAM 정책
- Amazon DynamoDB를 대상 AWS DMS 엔드포인트로 사용하여 서비스 간 혼란을 방지하는 데 사용할 수 있습니다.

서비스 간 혼란을 방지하기 위해 AWS DMS API와 함께 사용할 IAM 역할

데이터베이스 마이그레이션에 AWS CLI 또는 AWS DMS API를 사용하려면 먼저 `dms-vpc-role` 및 `dms-cloudwatch-logs-role` IAM 역할을 AWS 계정에 추가해야 의 기능을 사용할 수 있습니다. AWS DMS자세한 정보는 AWS CLI 및 AWS DMS API와 함께 사용할 IAM 역할 생성을 참조하세요.

다음 예는 `my-replication-instance` 복제 인스턴스에서 `dms-vpc-role` 역할을 사용하기 위한 정책을 보여줍니다. 혼동된 대리자 문제를 방지하려면 이들 정책을 사용하세요.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "dms.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "your_account_id"
                },
                "ArnEqual": {
                    "AWS:SourceArn": "arn:aws:dms:your_region:your_account_id:rep:my-
replication-instance"
                }
            }
        }
    ]
}
```

교차 서비스 혼동된 대리자 방지를 위해 Amazon S3에 사전 평가를 저장하기 위한 IAM 정책

사전 평가 결과를 S3 버킷에 저장하려면 AWS DMS 가 Amazon S3의 객체를 관리하도록 허용하는 IAM 정책을 생성합니다. 자세한 정보는 IAM 리소스 생성 을 참조하세요.

다음 예는 지정된 사용자 계정으로 모든 작업 및 평가 실행에 액세스할 수 AWS DMS 있도록 허용하는 IAM 역할에 설정된 혼동된 보조 조건이 포함된 신뢰 정책을 보여줍니다.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "dms.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "your_account_id"
                },
                "ArnLike": {
                    "AWS:SourceArn": [
                        "arn:aws:dms:your_region:your_account_id:assessment-run:*",
                        "arn:aws:dms:region:your_account_id:task:*"
                    ]
                }
            }
        }
    ]
}
```

Amazon DynamoDB를 대상 AWS DMS 엔드포인트로 사용하여 서비스 간 혼란을 방지하는 데 사용할 수 있습니다.

Amazon DynamoDB를 데이터베이스 마이그레이션의 대상 엔드포인트로 사용하려면 DynamoDB 테이블에 대한 액세스 권한을 위임하고 AWS DMS 부여할 수 있는 IAM 역할을 생성해야 합니다. 그런 다음 이 역할을 사용하여 AWS DMS에서 대상 DynamoDB 엔드포인트를 생성합니다. 자세한 정보는 대상으로 Amazon DynamoDB 사용을 참조하세요.

다음 예는 모든 AWS DMS 엔드포인트가 DynamoDB 테이블에 액세스할 수 있도록 허용하는 IAM 역할에 설정된 혼동된 보조 조건이 포함된 신뢰 정책을 보여줍니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "dms.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "your_account_id"
                },
                "ArnLike": {
                    "AWS:SourceArn":
 "arn:aws:dms:your_region:your_account_id:endpoint:*"
                }
            }
        }
    ]
}
```

AWS Database Migration Service의 AWS 관리형 정책

주제

- AWS 관리형 정책: AWSDMSServerlessServiceRolePolicy
- AWS 관리형 정책: AmazonDMSCloudWatchLogsRole
- AWS 관리형 정책: AWSDMSFleetAdvisorServiceRolePolicy
- AWS DMS가 AWS 관리형 정책으로 업데이트

AWS 관리형 정책: AWSDMSServerlessServiceRolePolicy

이 정책은 AWS DMS에서 사용자를 대신하여 작업을 수행할 수 있도록 `AWSServiceRoleForDMSServerless` 역할에 연결됩니다. 자세한 내용은 AWS DMS Serverless의 서비스 연결 역할 섹션을 참조하세요.

이 정책은 AWS DMS가 복제 리소스를 관리할 수 있도록 허용하는 권한을 기여자에게 부여합니다.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- `dms` – 보안 주체가 AWS DMS 리소스와 상호 작용할 수 있도록 허용합니다.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "id0",
            "Effect": "Allow",
            "Action": [
                "dms:CreateReplicationInstance",
                "dms:CreateReplicationTask"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "dms:req-tag/ResourceCreatedBy": "DMSServerless"
                }
            }
        },
        {
            "Sid": "id1",
            "Effect": "Allow",
            "Action": [
                "dms:DescribeReplicationInstances",
                "dms:DescribeReplicationTasks"
            ],
            "Resource": "*"
        },
        {
            "Sid": "id2",
            "Effect": "Allow",
            "Action": [
                "dms:StartReplicationTask",
                "dms:StopReplicationTask",
                "dms:DeleteReplicationTask",
                "dms:DeleteReplicationInstance"
            ],
            "Resource": [
                "arn:aws:dms:*:*:rep:*",
                "arn:aws:dms:*:*:task:*"
            ],
            "Condition": {
                "StringEqualsIgnoreCase": {
                    "aws:ResourceTag/ResourceCreatedBy": "DMSServerless"
                }
            }
        },
```

```
        {
            "Sid": "id3",
            "Effect": "Allow",
            "Action": [
                "dms:TestConnection",
                "dms:DeleteConnection"
            ],
            "Resource": [
                "arn:aws:dms:*:*:rep:*",
                "arn:aws:dms:*:*:endpoint:*"
            ]
        }
    ]
}
```

AWS 관리형 정책: AmazonDMSCloudWatchLogsRole

이 정책은 AWS DMS에서 사용자를 대신하여 작업을 수행할 수 있도록 `dms-cloudwatch-logs-role` 역할에 연결됩니다. 자세한 내용은 AWS DMS에 서비스 연결 역할 사용 섹션을 참조하세요.

이 정책은 AWS DMS가 복제 로그를 CloudWatch 로그에 게시할 수 있도록 허용하는 권한을 기여자에게 부여합니다.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- `logs` – 보안 주체가 로그를 CloudWatch Logs에 게시하도록 허용합니다. AWS DMS가 CloudWatch를 사용하여 복제 로그를 표시하려면 이 권한이 필요합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowDescribeOnAllLogGroups",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups"
            ],
            "Resource": [
```

```
                "*"
            ]
        },
        {
            "Sid": "AllowDescribeOfAllLogStreamsOnDmsTasksLogGroup",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogStreams"
            ],
            "Resource": [
                "arn:aws:logs:*:*:log-group:dms-tasks-*",
                "arn:aws:logs:*:*:log-group:dms-serverless-replication-*"
            ]
        },
        {
            "Sid": "AllowCreationOfDmsLogGroups",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup"
            ],
            "Resource": [
                "arn:aws:logs:*:*:log-group:dms-tasks-*",
                "arn:aws:logs:*:*:log-group:dms-serverless-replication-*:log-stream:"
            ]
        },
        {
            "Sid": "AllowCreationOfDmsLogStream",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream"
            ],
            "Resource": [
                "arn:aws:logs:*:*:log-group:dms-tasks-*:log-stream:dms-task-*",
                "arn:aws:logs:*:*:log-group:dms-serverless-replication-*:log-
stream:dms-serverless-*"
            ]
        },
        {
            "Sid": "AllowUploadOfLogEventsToDmsLogStream",
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
```

```
                "arn:aws:logs:*:*:log-group:dms-tasks-*:log-stream:dms-task-*",
                "arn:aws:logs:*:*:log-group:dms-serverless-replication-*:log-
stream:dms-serverless-*"
            ]
        }
    ]
}
```

AWS 관리형 정책: AWSDMSFleetAdvisorServiceRolePolicy

AWSDMSFleetAdvisorServiceRolePolicy를 IAM 엔터티에 연결할 수 없습니다. 이 정책은 AWS DMS Fleet Advisor가 사용자를 대신하여 작업을 수행할 수 있도록 서비스 연결 역할에 연결됩니다. 자세한 내용은 AWS DMS에 서비스 연결 역할 사용 섹션을 참조하세요.

이 정책은 AWS DMS Fleet Advisor가 Amazon CloudWatch 지표를 게시할 수 있도록 허용하는 권한을 기여자에게 부여합니다.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- `cloudwatch` – 보안 주체가 Amazon CloudWatch에 지표 데이터 포인트를 게시하도록 허용합니다. AWS DMS Fleet Advisor가 CloudWatch를 사용하여 데이터베이스 지표와 함께 차트를 표시하려면 이 권한이 필요합니다.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Resource": "*",
        "Action": "cloudwatch:PutMetricData",
        "Condition": {
            "StringEquals": {
                "cloudwatch:namespace": "AWS/DMS/FleetAdvisor"
            }
        }
    }
```

```
}
```

AWS DMS가 AWS 관리형 정책으로 업데이트

이 서비스가 이러한 변경 내용을 추적하기 시작한 이후부터 AWS DMS의 AWS 관리형 정책 업데이트에 관한 세부 정보를 봅니다. 이 페이지의 변경 사항에 관한 자동 알림을 받으려면 AWS DMS 문서 기록 페이지에서 RSS 피드를 구독하세요.

변경 사항	설명	날짜
AWSDMSServerlessServiceRolePolicy – 새 정책	AWS DMS는 Amazon CloudWatch 지표 게시와 같은 서비스를 AWS DMS가 사용자 대신 생성하고 관리할 수 있도록 허용하는 `AWSDMSServerlessServiceRolePolicy` 역할을 추가했습니다.	2023년 5월 22일
AmazonDMSCloudWatchLogsRole – 변경 사항	AWS DMS는 서버리스 복제 구성에서 CloudWatch Logs로 AWS DMS 복제 로그를 업로드할 수 있도록 부여된 각 권한에 서버리스 리소스용 ARN을 추가했습니다.	2023년 5월 22일
AWSDMSFleetAdvisorServiceRolePolicy – 새 정책	AWS DMS Fleet Advisor는 Amazon CloudWatch에 지표 데이터 포인트를 게시하도록 허용하는 새 정책을 추가했습니다.	2023년 3월 6일
AWS DMS에서 변경 사항 추적 시작	AWS DMS가 AWS 관리형 정책에 대한 변경 사항 추적을 시작했습니다.	2023년 3월 6일

AWS Database Migration Service의 규정 준수 확인

타사 감사자는 여러 AWS Database Migration Service 규정 준수 프로그램의 일환으로 AWS의 보안 및 규정 준수를 평가합니다. 여기에는 다음 프로그램이 포함됩니다.

- SOC
- PCI
- ISO
- FedRAMP
- DoD CC SRG
- HIPAA BAA
- MTCS
- CS
- K-ISMS
- ENS High
- OSPAR
- HITRUST CSF

특정 규정 준수 프로그램의 범위 내에 있는 AWS 서비스 목록은 규정 준수 프로그램 제공 범위 내 AWS 서비스를 참조하세요. 일반적인 내용은 AWS 규정 준수 프로그램을 참조하세요.

AWS Artifact를 사용하여 타사 감사 보고서를 다운로드할 수 있습니다. 자세한 내용은 AWS Artifact에서 보고서 다운로드 를 참조하세요.

AWS DMS 사용 시 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표 및 관련 법률과 규정에 따라 결정됩니다. AWS에서는 규정 준수를 지원할 다음과 같은 리소스를 제공합니다.

- 보안 및 규정 준수 빠른 시작 안내서 – 이 배포 안내서에서는 아키텍처 고려 사항에 관해 설명하고 AWS에서 보안 및 규정 준수에 중점을 둔 기본 환경을 배포하기 위한 단계를 제공합니다.
- Architecting for HIPAA security and compliance on Amazon Web Services 백서 - 이 백서는 기업에서 AWS를 사용하여 HIPAA를 준수하는 애플리케이션을 생성하는 방법을 설명합니다.
- AWS 규정 준수 리소스 - 고객 조직이 속한 산업 및 위치에 적용될 수 있는 워크북 및 가이드 모음입니다.
- AWS Config – 이 AWS 서비스로 리소스 구성이 내부 관행, 업계 지침 및 규정을 준수하는 정도를 평가할 수 있습니다.

- AWS Security Hub - 이 AWS 서비스는 보안 산업 표준 및 모범 사례 규정 준수 여부를 확인하는 데 도움이 되도록 AWS 내 보안 상태를 종합적으로 보여줍니다.

AWS Database Migration Service의 복원성

AWS 글로벌 인프라는 AWS 리전 및 가용 영역을 중심으로 구축됩니다. AWS 리전은 물리적으로 분리되고 격리된 다수의 가용 영역을 제공하며 이러한 가용 영역은 짧은 지연 시간, 높은 처리량 및 높은 중복성을 갖춘 네트워크에 연결되어 있습니다. 가용 영역을 사용하면 중단 없이 가용 영역 간에 자동으로 장애 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 관한 자세한 내용은 AWS 글로벌 인프라를 참조하세요.

AWS 글로벌 인프라 외에도 AWS DMS는 Multi-AZ 옵션을 선택할 때 다중 AZ 배포를 사용해 복제 인스턴스에 대한 고가용성 및 장애 조치를 지원합니다.

다중 AZ 배포에서 AWS DMS는 서로 다른 가용 영역에 복제 인스턴스의 예비 복제본을 자동으로 프로비저닝하고 유지합니다. 기본 복제 인스턴스는 대기 복제본에 동기식으로 복제됩니다. 기본 복제 인스턴스에 장애가 발생하거나 무응답 상태가 되면 대기 복제본은 중단을 최소화하면서 실행 중이던 작업을 다시 시작합니다. 기본 복제본은 자신의 상태를 끊임없이 대기 복제본으로 복제하기 때문에 다중 AZ 배포는 약간의 성능 오버헤드를 발생시킵니다.

다중 AZ 배포 사용에 관한 자세한 내용은 AWS DMS 복제 인스턴스 사용 단원을 참조하십시오.

AWS Database Migration Service에서 인프라 보안

관리형 서비스인 AWS Database Migration Service은 AWS 글로벌 네트워크 보안으로 보호됩니다. AWS 보안 서비스와 AWS의 인프라 보호 방법에 관한 자세한 내용은 AWS 클라우드 보안을 참조하세요. 인프라 보안에 대한 모범 사례를 사용하여 AWS 환경을 설계하려면 보안 원칙 AWS Well-Architected Framework의 인프라 보호를 참조하세요.

AWS에서 게시한 API 직접 호출을 사용하여 네트워크를 통해 AWS DMS에 액세스합니다. 고객은 다음을 지원해야 합니다.

- 전송 계층 보안(TLS). TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군. Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

또한 요청은 액세스 키 ID 및 IAM 주체와 관련된 비밀 액세스 키를 사용하여 서명해야 합니다. 또는 AWS Security Token Service(AWS STS)를 사용하여 임시 보안 인증을 생성하여 요청에 서명할 수 있습니다.

이러한 API 작업은 모든 네트워크 위치에서 호출할 수 있습니다. 또한 AWS DMS는 리소스 기반 액세스 정책도 지원하는데 이 정책은 이를테면 소스 IP 주소를 기반으로 작업 및 리소스에 대한 제한 사항을 지정할 수 있습니다. 또한 AWS DMS 정책을 사용하여 특정 Amazon VPC 엔드포인트 또는 특정 VPC(Virtual Private Cloud)에서 액세스를 제어할 수도 있습니다. 그러면 AWS 네트워크 내에 있는 특정 VPC에서만 특정 AWS DMS 리소스에 대한 네트워크 액세스가 효과적으로 격리됩니다. 예제를 포함하여 AWS DMS에서 리소스 기반 액세스 정책을 사용하는 방법에 관한 자세한 내용은 리소스 이름 및 태그를 사용하여 세분화된 액세스 제어 단원을 참조하십시오.

AWS DMS와의 통신을 단일 VPC 내에서만 실행하려는 경우, AWS PrivateLink를 통해 AWS DMS에 연결할 수 있도록 VPC 인터페이스 엔드포인트를 생성하면 됩니다. AWS PrivateLink는 AWS DMS에 대한 모든 호출 및 그와 관련된 결과가 인터페이스 엔드포인트가 생성된 특정 VPC에만 계속 나타나도록 지원합니다. 그런 다음, AWS CLI 또는 SDK를 사용하여 실행하는 모든 AWS DMS 명령에 이 인터페이스 엔드포인트의 URL을 옵션으로 지정하면 됩니다. 이렇게 하면 AWS DMS와의 전체 통신이 VPC에만 나타나고 퍼블릭 인터넷에서는 보이지 않게 할 수 있습니다.

단일 VPC에서 DMS에 액세스할 수 있도록 인터페이스 엔드포인트를 생성하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/vpc/에서 Amazon VPC 콘솔을 엽니다.

2. 탐색 창에서 엔드포인트를 선택합니다. 그러면 엔드포인트 생성 페이지가 열리는데 이 페이지에서는 VPC에서 AWS DMS까지 인터페이스 엔드포인트를 생성할 수 있습니다.
3. AWS 서비스를 선택한 다음, 다음과 같은 형식의 서비스 이름 값(이 경우에는 AWS DMS)을 검색하여 선택합니다.

   ```
   com.amazonaws.region.dms
   ```

   여기서는 AWS DMS가 실행되는 AWS 리전을 *region*이 지정합니다(예: com.amazonaws.us-west-2.dms).
4. VPC의 경우, 인터페이스 엔드포인트를 생성할 VPC를 선택합니다(예를 들면 vpc-12abcd34).
5. 가용 영역 및 서브넷 ID에서 값을 하나씩 선택합니다. 이 값은 선택한 AWS DMS 엔드포인트가 실행될 수 있는 위치를 나타내야 합니다(예: us-west-2a (usw2-az1) 및 subnet-ab123cd4).
6. DNS 이름을 사용하여 엔드포인트를 생성하려면 DNS 이름 활성화를 선택합니다.
   이 DNS 이름은 임의의 문자열(ab12dc34)과 함께 하이픈으로 구분된 엔드포인트 ID(vpce-12abcd34efg567hij)로 구성됩니다. 이는 점을 통해 역순으로 서비스 이름과 구분되며 vpce(dms.us-west-2.vpce.amazonaws.com)가 추가된 것이 특징입니다.

   예를 들면, vpce-12abcd34efg567hij-ab12dc34.dms.us-west-2.vpce.amazonaws.com입니다.
7. 보안 그룹의 경우, 엔드포인트에 사용할 그룹을 선택합니다.

   보안 그룹을 설정할 때는 해당 그룹 내에서 아웃바운드 HTTPS 호출을 허용해야 합니다. 자세한 내용을 알아보려면 Amazon VPC 사용 설명서의 보안 그룹 생성을 참조하세요.
8. 정책에서 전체 액세스 또는 사용자 지정 값을 선택합니다. 예를 들어, 다음과 비슷한 사용자 지정 정책을 선택하여 특정 작업 및 리소스에 대한 엔드포인트의 액세스를 제한할 수 있습니다.

   ```
   {
     "Statement": [
       {
         "Action": "dms:*",
         "Effect": "Allow",
         "Resource": "*",
         "Principal": "*"
       },
       {
         "Action": [
           "dms:ModifyReplicationInstance",
           "dms:DeleteReplicationInstance"
   ```

```
      ],
      "Effect": "Deny",
      "Resource": "arn:aws:dms:us-west-2:<account-id>:rep:<replication-instance-
id>",
      "Principal": "*"
    }
  ]
}
```

여기서 샘플 정책은 특정 복제 인스턴스의 삭제 또는 수정을 제외한 모든 AWS DMS API 직접 호출을 허용합니다.

이제 6단계에서 생성된 DNS 이름을 사용하여 구성된 URL을 옵션으로 지정할 수 있습니다. 생성된 인터페이스 엔드포인트를 사용하여 서비스 인스턴스에 액세스하기 위해 모든 AWS DMS CLI 명령 또는 API 작업에 대해 이를 지정합니다. 예를 들어, 이 VPC에서 다음과 같이 DMS CLI 명령 `DescribeEndpoints`를 실행할 수 있습니다.

```
$ aws dms describe-endpoints --endpoint-url https://vpce-12abcd34efg567hij-
ab12dc34.dms.us-west-2.vpce.amazonaws.com
```

프라이빗 DNS 옵션을 활성화한 경우, 요청에 엔드포인트 URL을 지정할 필요는 없습니다.

VPC 인터페이스 엔드포인트 생성 및 사용(프라이빗 DNS 옵션 활성화 포함)에 관한 자세한 내용은 Amazon VPC 사용 설명서의 인터페이스 VPC 엔드포인트(AWS PrivateLink)를 참조하십시오.

리소스 이름 및 태그를 사용하여 세분화된 액세스 제어

Amazon 리소스 이름(ARN)을 기반으로 리소스 이름과 리소스 태그를 사용하여 AWS DMS 리소스에 대한 액세스를 관리할 수 있습니다. IAM 정책에 조건문을 포함하거나 허용 작업을 정의하여 이 작업을 수행할 수 있습니다.

리소스 이름을 사용하여 액세스 제어

IAM 사용자 계정을 생성하고 AWS DMS 리소스의 ARN에 따라 정책을 할당할 수 있습니다.

다음 정책은 ARN이 arn:aws:dms:us-east-1:152683116:rep:DOH67ZTOXGLIXMIHKITV인 AWS DMS 복제 인스턴스에 대한 액세스를 거부합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "dms:*"
            ],
            "Effect": "Deny",
            "Resource": "arn:aws:dms:us-east-1:152683116:rep:DOH67ZTOXGLIXMIHKITV"
        }
    ]
}
```

예를 들면, 정책이 적용 중일 때는 다음 명령이 실패합니다.

```
$ aws dms delete-replication-instance
   --replication-instance-arn "arn:aws:dms:us-
east-1:152683116:rep:DOH67ZTOXGLIXMIHKITV"

A client error (AccessDeniedException) occurred when calling the
 DeleteReplicationInstance
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:DeleteReplicationInstance on resource: arn:aws:dms:us-
east-1:152683116:rep:DOH67ZTOXGLIXMIHKITV
```

```
$ aws dms modify-replication-instance
   --replication-instance-arn "arn:aws:dms:us-
east-1:152683116:rep:DOH67ZTOXGLIXMIHKITV"

A client error (AccessDeniedException) occurred when calling the
 ModifyReplicationInstance
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:ModifyReplicationInstance on resource: arn:aws:dms:us-
east-1:152683116:rep:DOH67ZTOXGLIXMIHKITV
```

AWS DMS 엔드포인트와 복제 작업에 대한 액세스를 제한하는 IAM 정책을 지정할 수도 있습니다.

다음 정책은 엔드포인트의 ARN을 사용하여 AWS DMS 엔드포인트에 대한 액세스를 제한합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "dms:*"
            ],
            "Effect": "Deny",
            "Resource": "arn:aws:dms:us-
east-1:152683116:endpoint:D6E37YBXTNHOA6XRQSZCUGX"
        }
    ]
}
```

예를 들면, 엔드포인트의 ARN을 사용하는 정책이 적용 중일 때는 다음 명령이 실패합니다.

```
$ aws dms delete-endpoint
   --endpoint-arn "arn:aws:dms:us-east-1:152683116:endpoint:D6E37YBXTNHOA6XRQSZCUGX"

A client error (AccessDeniedException) occurred when calling the DeleteEndpoint
 operation:
User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
 dms:DeleteEndpoint
on resource: arn:aws:dms:us-east-1:152683116:endpoint:D6E37YBXTNHOA6XRQSZCUGX
```

```
$ aws dms modify-endpoint
   --endpoint-arn "arn:aws:dms:us-east-1:152683116:endpoint:D6E37YBXTNHOA6XRQSZCUGX"


A client error (AccessDeniedException) occurred when calling the ModifyEndpoint
 operation:
User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
 dms:ModifyEndpoint
on resource: arn:aws:dms:us-east-1:152683116:endpoint:D6E37YBXTNHOA6XRQSZCUGX
```

다음 정책은 작업의 ARN을 사용하여 AWS DMS 작업에 대한 액세스를 제한합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "dms:*"
            ],
            "Effect": "Deny",
            "Resource": "arn:aws:dms:us-east-1:152683116:task:UO3YR4N47DXH3ATT4YMWOIT"
        }
    ]
}
```

예를 들면, 작업의 ARN을 사용하는 정책이 적용 중일 때는 다음 명령이 실패합니다.

```
$ aws dms delete-replication-task
   --replication-task-arn "arn:aws:dms:us-
east-1:152683116:task:UO3YR4N47DXH3ATT4YMWOIT"

A client error (AccessDeniedException) occurred when calling the DeleteReplicationTask
 operation:
User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
 dms:DeleteReplicationTask
on resource: arn:aws:dms:us-east-1:152683116:task:UO3YR4N47DXH3ATT4YMWOIT
```

태그를 사용하여 액세스 제어

AWS DMS는 추가 태그 지정 요구 사항 없이 고객 정의 정책에서 사용할 수 있는 공통 키-값 페어 집합을 정의합니다. AWS DMS 리소스 태깅에 대한 자세한 내용은 AWS Database Migration Service의 리소스에 태그 지정 섹션을 참조하세요.

다음은 AWS DMS와 사용할 수 있는 표준 태그 목록입니다.

- aws:CurrentTime – 임시 기준에 따라 액세스 제한을 허용하는 요청 날짜 및 시간을 나타냅니다.
- aws:EpochTime – 이 태그는 위의 aws:CurrentTime 태그와 비슷하지만, 현재 시간이 Unix epoch 이후에 경과된 초 시간으로 표시됩니다.
- aws:MultiFactorAuthPresent – 요청이 멀티 팩터 인증을 통해 서명되었는지 여부를 나타내는 부울 태그입니다.
- aws:MultiFactorAuthAge – 멀티 팩터 인증 토큰의 수명(초)에 액세스할 수 있습니다.
- aws:principaltype - 현재 요청에 대한 보안 주체 유형(사용자, 계정, 연합된 사용자 등)에 액세스할 수 있습니다.
- aws:SourceIp - 요청을 실행하는 사용자의 원본 IP 주소를 나타냅니다.
- aws:UserAgent – 리소스를 요청하는 클라이언트 애플리케이션에 대한 정보를 제공합니다.
- aws:userid – 요청을 실행하는 사용자의 ID에 액세스할 수 있습니다.
- aws:username – 요청을 실행하는 사용자의 이름에 액세스할 수 있습니다.
- dms:InstanceClass – 복제 인스턴스 호스트의 컴퓨팅 크기에 액세스할 수 있습니다.
- dms:StorageSize - 스토리지 볼륨 크기(GB)에 액세스할 수 있습니다.

고유한 태그를 정의할 수도 있습니다. 고객이 정의한 태그는 AWS 태깅 서비스에서 지속되는 단순한 키-값 페어입니다. 이러한 키-값 페어를 복제 인스턴스, 엔드포인트 및 작업을 비롯한 AWS DMS 리소스에 추가할 수 있습니다. 이러한 태그는 정책의 IAM "조건"문을 사용하여 일치되며 특정 조건 태그를 사용하여 참조됩니다. 태그 키에는 "dms" 접두사, 리소스 유형 및 "tag" 접두사가 붙습니다. 태그 형식은 다음과 같습니다.

```
dms:{resource type}-tag/{tag key}={tag value}
```

예를 들어 "stage=production" 태그를 포함하는 복제 인스턴스에 대해 API 호출 성공만 허용하는 정책을 정의하려 한다고 가정해 보겠습니다. 다음 조건문은 리소스와 지정된 태그를 일치시킵니다.

```
"Condition":
```

```
{
    "streq":
        {
            "dms:rep-tag/stage":"production"
        }
}
```

이 정책 조건과 일치하는 복제 인스턴스에 다음 태그를 추가합니다.

```
stage production
```

AWS DMS 리소스에 이미 할당된 태그 외에, 지정된 리소스에 적용할 수 있는 태그 키와 값을 제한할 정책을 작성할 수도 있습니다. 이 경우 태그 접두사는 "req"입니다.

예를 들면, 다음 정책문은 사용자가 지정된 리소스에 할당할 수 있는 태그를 특정 허용 값 목록으로 제한합니다.

```
 "Condition":
{
    "streq":
        {
            "dms:rep-tag/stage": [ "production", "development", "testing" ]
        }
}
```

다음 정책 예제는 리소스 태그에 따라 AWS DMS 리소스에 대한 액세스를 제한합니다.

다음 정책은 태그 값이 "Desktop"이고 태그 키가 "Env"인 복제 인스턴스에 대한 액세스를 제한합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "dms:*"
            ],
            "Effect": "Deny",
            "Resource": "*",
            "Condition": {
```

```
                "StringEquals": {
                    "dms:rep-tag/Env": [
                        "Desktop"
                    ]
                }
            }
        }
    ]
}
```

다음 명령은 태그 값이 "Desktop"이고 태그 키가 "Env"일 때 액세스를 제한하는 IAM 정책에 따라 성공하거나 실패합니다.

```
$ aws dms list-tags-for-resource
   --resource-name arn:aws:dms:us-east-1:152683116:rep:46DHOU7JOJYOJXWDOZNFEN
   --endpoint-url http://localhost:8000
{
    "TagList": [
        {
            "Value": "Desktop",
            "Key": "Env"
        }
    ]
}

$ aws dms delete-replication-instance
   --replication-instance-arn "arn:aws:dms:us-
east-1:152683116:rep:46DHOU7JOJYOJXWDOZNFEN"
A client error (AccessDeniedException) occurred when calling the
 DeleteReplicationInstance
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:DeleteReplicationInstance on resource: arn:aws:dms:us-
east-1:152683116:rep:46DHOU7JOJYOJXWDOZNFEN

$ aws dms modify-replication-instance
   --replication-instance-arn "arn:aws:dms:us-
east-1:152683116:rep:46DHOU7JOJYOJXWDOZNFEN"

A client error (AccessDeniedException) occurred when calling the
 ModifyReplicationInstance
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
```

```
dms:ModifyReplicationInstance on resource: arn:aws:dms:us-
east-1:152683116:rep:46DHOU7JOJYOJXWDOZNFEN

$ aws dms add-tags-to-resource
   --resource-name arn:aws:dms:us-east-1:152683116:rep:46DHOU7JOJYOJXWDOZNFEN
   --tags Key=CostCenter,Value=1234

A client error (AccessDeniedException) occurred when calling the AddTagsToResource
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:AddTagsToResource on resource: arn:aws:dms:us-
east-1:152683116:rep:46DHOU7JOJYOJXWDOZNFEN

$ aws dms remove-tags-from-resource
   --resource-name arn:aws:dms:us-east-1:152683116:rep:46DHOU7JOJYOJXWDOZNFEN
   --tag-keys Env

A client error (AccessDeniedException) occurred when calling the
 RemoveTagsFromResource
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:RemoveTagsFromResource on resource: arn:aws:dms:us-
east-1:152683116:rep:46DHOU7JOJYOJXWDOZNFEN
```

다음 정책은 태그 값이 "Desktop"이고 태그 키가 "Env"인 AWS DMS 엔드포인트에 대한 액세스를 제한합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "dms:*"
            ],
            "Effect": "Deny",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "dms:endpoint-tag/Env": [
                        "Desktop"
                    ]
                }
            }
```

```
        }
    ]
}
```

다음 명령은 태그 값이 "Desktop"이고 태그 키가 "Env"일 때 액세스를 제한하는 IAM 정책에 따라 성공하거나 실패합니다.

```
$ aws dms list-tags-for-resource
   --resource-name arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I
{
    "TagList": [
        {
            "Value": "Desktop",
            "Key": "Env"
        }
    ]
}

$ aws dms delete-endpoint
   --endpoint-arn "arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I"

A client error (AccessDeniedException) occurred when calling the DeleteEndpoint
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:DeleteEndpoint on resource: arn:aws:dms:us-
east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I

$ aws dms modify-endpoint
   --endpoint-arn "arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I"


A client error (AccessDeniedException) occurred when calling the ModifyEndpoint
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:ModifyEndpoint on resource: arn:aws:dms:us-
east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I

$ aws dms add-tags-to-resource
   --resource-name arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I
   --tags Key=CostCenter,Value=1234

A client error (AccessDeniedException) occurred when calling the AddTagsToResource
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
```

```
dms:AddTagsToResource on resource: arn:aws:dms:us-
east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I

$ aws dms remove-tags-from-resource
   --resource-name arn:aws:dms:us-east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I
   --tag-keys Env

A client error (AccessDeniedException) occurred when calling the
 RemoveTagsFromResource
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:RemoveTagsFromResource on resource: arn:aws:dms:us-
east-1:152683116:endpoint:J2YCZPNGOLFY52344IZWA6I
```

다음 정책은 태그 값이 "Desktop"이고 태그 키가 "Env"인 복제 작업에 대한 액세스를 제한합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "dms:*"
            ],
            "Effect": "Deny",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "dms:task-tag/Env": [
                        "Desktop"
                    ]
                }
            }
        }
    ]
}
```

다음 명령은 태그 값이 "Desktop"이고 태그 키가 "Env"일 때 액세스를 제한하는 IAM 정책에 따라 성공하거나 실패합니다.

```
$ aws dms list-tags-for-resource
```

```
   --resource-name arn:aws:dms:us-east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3
{
    "TagList": [
        {
            "Value": "Desktop",
            "Key": "Env"
        }
    ]
}

$ aws dms delete-replication-task
   --replication-task-arn "arn:aws:dms:us-east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3"

A client error (AccessDeniedException) occurred when calling the DeleteReplicationTask
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:DeleteReplicationTask on resource: arn:aws:dms:us-
east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3

$ aws dms add-tags-to-resource
   --resource-name arn:aws:dms:us-east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3
   --tags Key=CostCenter,Value=1234

A client error (AccessDeniedException) occurred when calling the AddTagsToResource
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:AddTagsToResource on resource: arn:aws:dms:us-
east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3

$ aws dms remove-tags-from-resource
   --resource-name arn:aws:dms:us-east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3
   --tag-keys Env

A client error (AccessDeniedException) occurred when calling the
 RemoveTagsFromResource
operation: User: arn:aws:iam::152683116:user/dmstestusr is not authorized to perform:
dms:RemoveTagsFromResource on resource: arn:aws:dms:us-
east-1:152683116:task:RB7N24J2XBUPS3RFABZTG3
```

암호화 키 설정 및 AWS KMS 권한 지정

AWS DMS는 복제 인스턴스와 엔드포인트 연결 정보에서 사용하는 스토리지를 암호화합니다. 복제 인스턴스에서 사용하는 스토리지를 암호화하기 위해 AWS DMS는 AWS 계정에 고유한 AWS Key Management Service(AWS KMS) 키를 사용합니다. AWS KMS에서 이 키를 보고 관리할 수 있습니다. 계정의 기본 KMS 키(`aws/dms`)를 사용하거나 사용자 지정 KMS 키를 생성할 수 있습니다. 기존 KMS 키가 있는 경우, 암호화에 이 키를 사용할 수도 있습니다.

> **Note**
>
> 암호화 키로 사용하는 사용자 지정 또는 기존 AWS KMS 키는 모두 대칭 키여야 합니다. AWS DMS는 비대칭 암호화 키 사용을 지원하지 않습니다. 대칭 및 비대칭 암호화 KMS 키에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서의 https://docs.aws.amazon.com/kms/latest/developerguide/symmetric-asymmetric.html 섹션을 참조하세요.

복제 인스턴스 생성 페이지의 고급 섹션에서 사용자 지정 KMS 키를 선택하지 않은 경우, 복제 인스턴스를 처음 시작할 때 기본 KMS 키(`aws/dms`)가 생성됩니다. 기본 KMS 키를 사용할 경우, 마이그레이션을 위해 사용 중인 IAM 사용자 계정에 부여해야 하는 유일한 권한은 `kms:ListAliases`와 `kms:DescribeKey`입니다. 기본 KMS 키 사용에 대한 자세한 정보는 AWS DMS 사용에 필요한 IAM 권한 섹션을 참조하세요.

사용자 지정 KMS 키를 사용하려면 다음 옵션 중 하나를 사용하여 사용자 지정 KMS 키에 대한 권한을 할당해야 합니다.

- AWS KMS 사용자 지정 키에서 키 관리자 또는 키 사용자로서 마이그레이션에 사용되는 IAM 사용자 계정을 추가합니다. 이렇게 하면 필요한 AWS KMS 권한이 IAM 사용자 계정에 부여됩니다. 이 작업은 AWS DMS를 사용하기 위해 사용자가 IAM 사용자 계정에 부여하는 IAM 권한 외의 것입니다. 키 사용자에게 권한을 부여하는 방법에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서의 키 사용자가 KMS 키를 사용하도록 허용을 참조하세요.
- 사용자 지정 KMS 키에서 IAM 사용자 계정을 키 관리자/키 사용자로서 추가하지 않을 경우, AWS DMS를 사용할 수 있도록 IAM 사용자 계정에 부여해야 하는 IAM 권한에 다음 추가 권한을 부여해야 합니다.

```
{
            "Effect": "Allow",
```

```
            "Action": [
                "kms:ListAliases",
                "kms:DescribeKey",
                "kms:CreateGrant",
                "kms:Encrypt",
                "kms:ReEncrypt*"
            ],
            "Resource": "*"
        },
```

AWS DMS는 KMS 키 별칭에서도 작동합니다. 자체 AWS KMS 키를 생성하고 사용자에게 KMS 키에 대한 액세스 권한을 부여하는 방법에 대한 자세한 내용은 AWS KMS 개발자 안내서를 참조하세요.

KMS 키 식별자를 지정하지 않으면 AWS DMS는 기본 암호화 키를 사용합니다. AWS KMS는 AWS 계정의 AWS DMS에 대한 기본 암호화 키를 생성합니다. AWS 계정에는 AWS 리전마다 다른 기본 암호화 키가 있습니다.

AWS DMS 리소스 암호화에 사용되는 AWS KMS 키를 관리하려면 AWS Key Management Service를 사용합니다. AWS KMS는 클라우드에 맞게 확장된 키 관리 시스템을 제공하기 위해 안전하고 가용성이 높은 하드웨어 및 소프트웨어를 결합합니다. AWS KMS를 사용하면 암호화 키를 생성하고 이 키를 사용할 수 있는 방법을 제어하는 정책을 정의할 수 있습니다.

AWS KMS는 AWS Management Console에서 찾을 수 있습니다.

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/kms에서 AWS Key Management Service(AWS KMS) 콘솔을 엽니다.
2. AWS 리전을 변경하려면 페이지의 오른쪽 상단 모서리에 있는 리전 선택기를 사용합니다.
3. 다음 옵션 중 하나를 선택하여 AWS KMS 키를 사용합니다.
   - AWS에서 계정을 위해 직접 생성하고 관리하는 키를 보려면 탐색 창에서 AWS 고객 관리형 키를 선택합니다.
   - 해당 계정에서 직접 생성하고 관리하는 키를 보려면 탐색 창에서 고객 관리형 키를 선택합니다.

AWS KMS는 AWS CloudTrail를 지원하므로 키가 적절하게 사용되고 있는지 확인하기 위해 키 사용을 감사할 수 있습니다. AWS KMS 키는 Amazon RDS, Amazon S3, Amazon Redshift, Amazon EBS 등 지원되는 AWS 서비스에서 AWS DMS와 함께 사용할 수 있습니다.

또한 특히 다음 AWS DMS 엔드포인트에 대한 대상 데이터를 암호화하기 위해 사용자 지정 AWS KMS 키를 생성할 수 있습니다.

- Amazon Redshift – 자세한 내용은 Amazon Redshift 대상 데이터 암호화를 위한 AWS KMS 키 생성 및 사용 섹션을 참조하세요.
- Amazon S3 – 자세한 내용은 AWS KMS 키를 생성하여 Amazon S3 대상 객체 암호화 섹션을 참조하세요.

KMS 키를 사용하여 AWS DMS 리소스를 생성한 후에는 이 리소스에서 암호화 키를 변경할 수 없습니다. AWS DMS 리소스를 생성하기 전에 암호화 키 요구 사항을 결정해야 합니다.

AWS Database Migration Service 네트워크 보안

AWS Database Migration Service를 사용할 때 생성하는 네트워크의 보안 요구 사항은 네트워크 구성 방법에 따라 달라집니다. AWS DMS의 네트워크 보안에 적용되는 일반 규칙은 다음과 같습니다.

- 복제 인스턴스에는 리소스 및 대상 엔드포인트에 대한 액세스 권한이 있어야 합니다. 복제 인스턴스용 보안 그룹에는 데이터베이스 포트에서 밖으로 데이터베이스 엔드포인트까지 인스턴스를 내보내도록 허용하는 네트워크 ACL 또는 규칙이 있어야 합니다.
- 데이터베이스 엔드포인트에는 복제 인스턴스에서 수신되는 액세스를 허용하는 네트워크 ACL과 보안 그룹 규칙이 포함되어야 합니다. 이 작업을 완료하기 위해 사용자의 구성에 따라 복제 인스턴스의 보안 그룹, 프라이빗 IP 주소, 퍼블릭 IP 주소 또는 NAT 게이트웨이의 퍼블릭 주소를 사용합니다.
- 네트워크에서 VPN 터널을 사용하는 경우, NAT 게이트웨이 역할을 하는 Amazon EC2 인스턴스는 복제 인스턴스가 트래픽을 전송하도록 허용하는 규칙을 가진 보안 그룹을 사용해야 합니다.

기본적으로 AWS DMS 복제 인스턴스에서 사용하는 VPC 보안 그룹에는 모든 포트에서 0.0.0.0/0으로의 송신을 허용하는 규칙이 있습니다. 이 보안 그룹을 수정하거나 자체 보안 그룹을 사용하는 경우, 각 데이터베이스 포트에서 최소한 발신이 원본과 대상 엔드포인트에 허용되어야 합니다.

각 데이터베이스 마이그레이션에서 사용할 수 있는 네트워크를 구성할 때에는 다음 구체적인 보안 사항을 고려해야 합니다.

- 하나의 VPC에 모든 데이터베이스 마이그레이션 구성 요소가 있는 구성 - 엔드포인트에서 사용되는 보안 그룹은 복제 인스턴스의 데이터베이스 포트에서 수신을 허용해야 합니다. 복제 인스턴스에서 사용하는 보안 그룹이 엔드포인트에 대한 수신이 있는지 확인하고, 그렇지 않으면 복제 인스턴스 액세스의 프라이빗 IP 주소를 허용하고 엔드포인트가 사용하는 보안 그룹에 규칙을 생성할 수 있습니다.
- 여러 VPC를 사용한 구성 - 복제 인스턴스에서 사용하는 보안 그룹에는 데이터베이스의 VPC 범위와 DB 포트에 대한 규칙이 있어야 합니다.
- AWS Direct Connect 또는 VPN을 사용하여 VPC에 대한 네트워크 구성 - VPC에서 온프레미스 VPN으로 트래픽 전송을 허용하는 VPN 터널입니다. 이 구성에서 VPC에는 특정 IP 주소나 범위로 지정된 트래픽을 호스트로 전송하는 라우팅 규칙이 포함되어 있어 트래픽을 VPC에서 온프레미스 VPN으로 연결할 수 있습니다. 이 경우에 트래픽을 복제 인스턴스의 프라이빗 IP 주소나 보안 그룹에서 NAT 인스턴스로 전송하도록 허용해야 하는 자체 보안 그룹 설정이 NAT 호스트에 적용되어 있습니다.
- 인터넷을 사용한 네트워크와 VPC 연결 구성 - VPC 보안 그룹에는 트래픽을 VPC로 전달하지 않고 인터넷 게이트웨이로 전송하는 라우팅 규칙이 적용되어 있어야 합니다. 이 구성에서 엔드포인트와의 연결이 복제 인스턴스의 퍼블릭 IP 주소에서 오는 것으로 나타납니다.

- 를 사용하여 VPC에 있지 않은 RDS DB 인스턴스를 VPC의 DB 인스턴스로 구성 ClassicLink - 원본 또는 대상 Amazon RDS DB 인스턴스가 VPC에 없고 보안 그룹을 VPC(복제 인스턴스가 있음)와 공유하지 않을 경우, 프록시 서버를 설정하고 ClassicLink를 사용하여 원본과 대상 데이터베이스를 연결할 수 있습니다.
- 소스 엔드포인트가 복제 인스턴스에서 사용하는 VPC 외부에 있고 NAT 게이트웨이를 사용 - 단일 탄력적 네트워크 인터페이스에 연결된 단일 탄력적 IP 주소를 사용하여 Network Address Translation(NAT) 게이트웨이를 구성할 수 있습니다. 그런 다음 이 탄력적 네트워크 인터페이스는 NAT 식별자(nat-#####)를 수신합니다. VPC에 인터넷 게이트웨이가 아닌 해당 NAT 게이트웨이로의 기본 경로가 포함된 경우, 대신 복제 인스턴스가 인터넷 게이트웨이의 퍼블릭 IP 주소를 사용하여 데이터베이스 엔드포인트에 접촉하는 것으로 나타납니다. 이 경우에 VPC 외부에 있는 데이터베이스 엔드포인트로의 수신은 복제 인스턴스의 퍼블릭 IP 주소가 아닌 NAT 주소의 수신을 허용해야 합니다.
- 비 RDBMS 엔진용 VPC 엔드포인트 - AWS DMS는 비 RDBMS 엔진용 VPC 엔드포인트를 지원하지 않습니다.

AWS Database Migration Service에서 SSL 사용

Secure Sockets Layer(SSL)을 사용하여 소스와 대상 엔드포인트 연결을 암호화할 수 있습니다. 이 작업을 위해 AWS DMS 관리 콘솔 또는 AWS DMS API를 사용하여 엔드포인트에 인증서를 할당할 수 있습니다. 또한 AWS DMS 콘솔을 사용하여 인증서를 관리할 수도 있습니다.

모든 데이터베이스가 동일한 방식으로 SSL을 사용하는 것은 아닙니다. Amazon Aurora MySQL 호환 에디션은 클러스터에 있는 기본 인스턴스의 엔드포인트인 서버 이름을 SSL용 엔드포인트로 사용합니다. Amazon Redshift 엔드포인트는 이미 SSL 연결을 사용하므로 AWS DMS에서 설정한 SSL 연결이 필요하지 않습니다. Oracle 엔드포인트에서는 추가 단계가 필요합니다. 자세한 내용은 Oracle 엔드포인트용 SSL 지원 섹션을 참조하세요.

주제

- AWS DMS에서 SSL을 사용할 때 적용되는 제한 사항
- 인증서 관리
- MySQL 호환, PostgreSQL 또는 SQL Server 엔드포인트에서 SSL 활성화

인증서를 엔드포인트에 할당하기 위해 루트 인증서나 중간 CA 인증서 체인을 제공하여 루트(인증서 번들로서)로 이어집니다. 즉, 이것은 엔드포인트에서 배포되는 서버 SSL 인증서에 서명하는 데 사용됩니다. 인증서는 PEM 형식 X509 파일로만 허용됩니다. 인증서를 가져올 때에는 엔드포인트에서 인증서를 지정하는 데 사용할 수 있는 Amazon 리소스 이름(ARN)을 수신합니다. Amazon RDS를 사용하는 경우 Amazon RDS에서 호스팅하는 `rds-combined-ca-bundle.pem` 파일에 제공된 루트 CA 및 인증서 번들을 다운로드할 수 있습니다. 이 파일을 다운로드하는 방법에 자세한 내용은 Amazon RDS 사용 설명서에서 SSL을 사용하여 DB 인스턴스에 대한 연결 암호화를 참조하세요.

여러 SSL 모드에서 선택하여 SSL 인증서 확인에 사용할 수 있습니다.

- none - 연결이 암호화되지 않습니다. 이 옵션은 보안이 적용되지 않지만, 오버헤드를 줄여야 합니다.
- require - CA 검증 없이 SSL(TLS)를 사용하여 연결이 암호화됩니다. 이 옵션은 보안이 강화되어 오버헤드가 증가합니다.
- verify-ca - 연결이 암호화됩니다. 이 옵션은 보안이 강화되어 오버헤드가 증가합니다. 이 옵션은 서버 인증서를 확인합니다.
- verify-full - 연결이 암호화됩니다. 이 옵션은 보안이 강화되어 오버헤드가 증가합니다. 이 옵션은 서버 인증서를 확인하고 서버 호스트 이름이 인증서의 호스트 이름 속성과 일치하는지 확인합니다.

데이터베이스 엔드포인트에서 작동하지 않는 SSL 모드도 있습니다. 다음 표에는 각 데이터베이스 엔진별로 지원되는 SSL 모드가 나와 있습니다.

DB 엔진	none	[require]	[verify-ca]	[verify-full]
MySQL/MariaDB/ Amazon Aurora MySQL	기본값	지원되지 않음	지원	지원
Microsoft SQL Server	기본값	지원	지원되지 않음	지원
PostgreSQL	기본값	지원	지원	지원
Amazon Redshift	기본값	SSL이 활성화되지 않음	SSL이 활성화되지 않음	SSL이 활성화되지 않음
Oracle	기본값	지원되지 않음	지원	지원되지 않음
SAP ASE	기본값	SSL이 활성화되지 않음	SSL이 활성화되지 않음	지원
MongoDB	기본값	지원	지원되지 않음	지원
Db2 LUW	기본값	지원되지 않음	지원	지원되지 않음
Db2 for z/OS	기본값	지원되지 않음	지원	지원되지 않음

Note

DMS 콘솔 또는 API의 SSL 모드 옵션은 Kinesis, DynamoDB와 같은 일부 데이터 스트리밍 및 NoSQL 서비스에는 적용되지 않습니다. 이러한 스트리밍 및 서비스는 기본적으로 안전하므로 DMS에서는 SSL 모드 설정이 없음으로 표시됩니다(SSL 모드=None). 엔드포인트에서 SSL을 사용하기 위한 추가 구성을 제공할 필요는 없습니다. 예를 들어 Kinesis를 대상 엔드포인트로 사용하는 경우, 기본적으로 안전합니다. Kinesis에 대한 모든 API 직접 호출은 SSL을 사용하므로 DMS 엔드포인트에서는 추가 SSL 옵션이 필요하지 않습니다. Kinesis 데이터 스트림에 연결할 때 기본적으로 DMS가 사용하는 HTTPS 프로토콜을 사용하여 SSL 엔드포인트를 통해 안전하게 데이터를 넣고 검색할 수 있습니다.

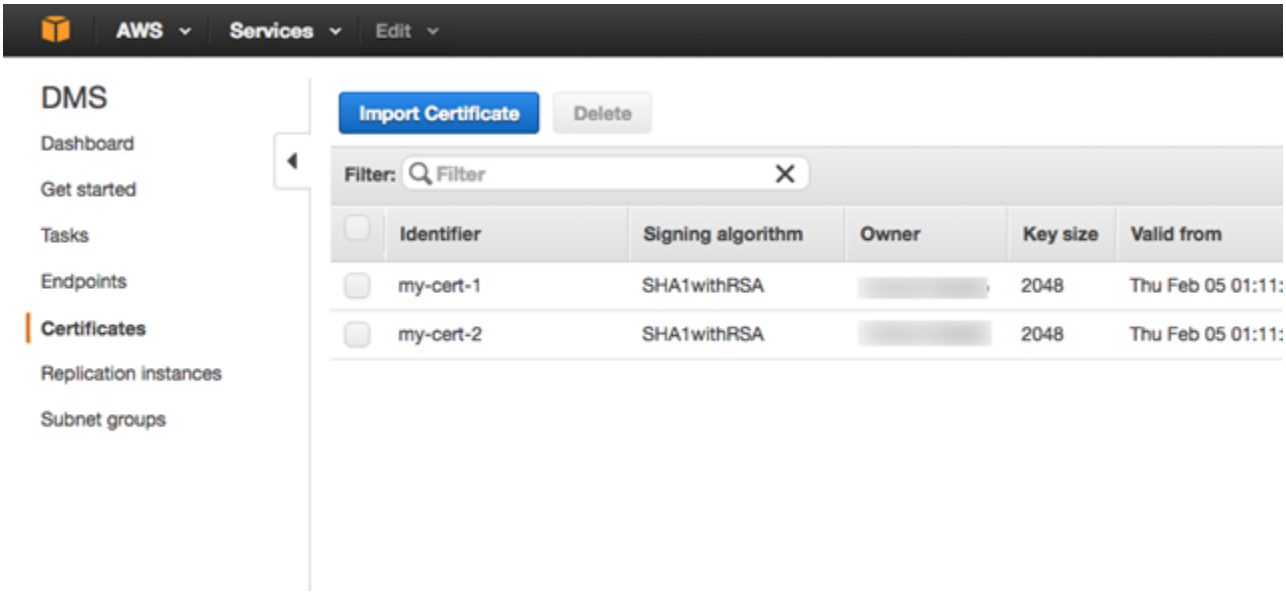
AWS DMS에서 SSL을 사용할 때 적용되는 제한 사항

AWS DMS에서 SSL을 사용할 때 다음과 같은 제한 사항이 있습니다.

- SSL과 Amazon Redshift 대상 엔드포인트 간 연결은 지원되지 않습니다. AWS DMS는 Amazon S3 버킷을 사용하여 데이터를 Amazon Redshift 데이터베이스로 전송합니다. 이 전송은 기본적으로 Amazon Redshift로 암호화됩니다.
- SSL이 활성화된 Oracle 엔드포인트를 통해 CDC(변경 데이터 캡처) 작업을 수행할 때 SQL 시간 초과가 발생할 수 있습니다. CDC 카운터가 예상 수치를 반영하지 않는 문제가 발생할 경우, 작업 설정의 `ChangeProcessingTuning` 섹션에서 `MinimumTransactionSize` 파라미터를 더 낮은 값으로 설정합니다. 최저 100부터 시작할 수 있습니다. `MinimumTransactionSize` 파라미터에 대한 자세한 내용은 변경 처리 튜닝 설정을 참조하세요.
- 인증서는 .pem 및 .sso(Oracle wallet) 형식으로만 가져올 수 있습니다.
- 경우에 따라 중간 CA(인증 기관)에서 서버 SSL 인증서를 서명할 수 있습니다. 이 경우 중간 CA에서 루트 CA까지 전체 인증서 체인을 하나의 .pem 파일로 가져와야 합니다.
- 서버에서 자체 서명 인증서를 사용하는 경우, SSL 모드로서 [require]를 선택합니다. require SSL 모드는 서버의 SSL 인증서를 명시적으로 신뢰하고 이 인증서가 CA에서 서명되었는지 여부를 확인하려고 하지 않습니다.

인증서 관리

DMS 콘솔을 사용하여 SSL 인증서를 보고 관리할 수 있습니다. 또한 DMS 콘솔을 사용하여 인증서를 가져올 수도 있습니다.



MySQL 호환, PostgreSQL 또는 SQL Server 엔드포인트에서 SSL 활성화

새로 생성한 엔드포인트 또는 기존 엔드포인트에 SSL 연결을 추가할 수 있습니다.

SSL을 통해 AWS DMS 엔드포인트를 생성하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/dms/v2/에서 AWS DMS 콘솔을 엽니다.

   AWS Identity and Access Management(IAM) 사용자로 로그인한 경우 AWS DMS에 액세스하기 위한 적절한 권한이 있어야 합니다. 데이터베이스 마이그레이션에 필요한 권한에 대한 자세한 내용은 AWS DMS 사용에 필요한 IAM 권한 섹션을 참조하세요.
2. 탐색 창에서 [Certificates]를 선택합니다.
3. [Import Certificate]를 선택합니다.
4. 엔드포인트와의 연결을 암호화하는 데 사용할 인증서를 업로드합니다.

   > **Note**
   >
   > 데이터베이스 엔드포인트 생성 페이지에서 새 CA 인증서 추가를 선택하여 엔드포인트를 생성하거나 수정할 때 AWS DMS 콘솔을 사용하여 인증서를 업로드할 수도 있습니다. Aurora Serverless를 대상으로 사용하려면 Aurora Serverless에서 TLS/SSL 사용에 설명된 인증서를 받으세요.
5. 2단계: 소스 및 대상 엔드포인트 지정 섹션의 설명과 같이 엔드포인트를 생성합니다.

기존 AWS DMS 엔드포인트를 수정하여 SSL을 사용하려면

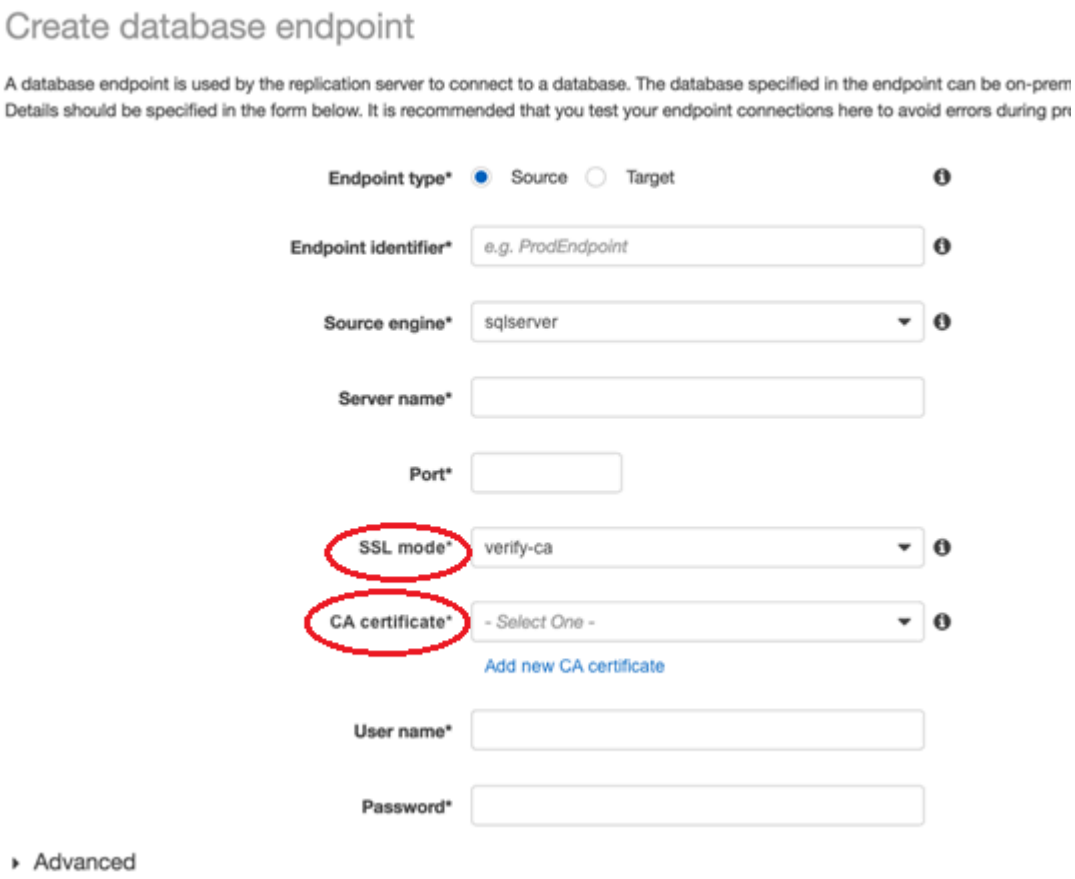
1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/dms/v2/에서 AWS DMS 콘솔을 엽니다.

   IAM 사용자로 로그인한 경우 AWS DMS에 액세스하기 위한 적절한 권한이 있어야 합니다. 데이터베이스 마이그레이션에 필요한 권한에 대한 자세한 내용은 AWS DMS 사용에 필요한 IAM 권한 섹션을 참조하세요.
2. 탐색 창에서 [Certificates]를 선택합니다.
3. [Import Certificate]를 선택합니다.
4. 엔드포인트와의 연결을 암호화하는 데 사용할 인증서를 업로드합니다.

> **Note**
>
> 데이터베이스 엔드포인트 생성 페이지에서 새 CA 인증서 추가를 선택하여 엔드포인트를 생성하거나 수정할 때 AWS DMS 콘솔을 사용하여 인증서를 업로드할 수도 있습니다.

5. 탐색 창에서 [Endpoints]를 선택하고, 수정할 엔드포인트를 선택하고, [Modify]를 선택합니다.
6. SSL 모드의 값을 선택합니다.

   verify-ca 또는 verify-full 모드를 선택할 경우 다음과 같이 CA 인증서에 사용할 인증서를 지정합니다.



7. 수정을 선택합니다.
8. 엔드포인트를 수정할 때 엔드포인트를 선택한 뒤 연결 테스트를 선택하여 SSL 연결이 잘 작동하는지 여부를 확인합니다.

소스와 대상 엔드포인트를 생성한 후 이 엔드포인트를 사용하는 작업을 생성합니다. 작업 생성에 대한 자세한 내용은 3단계: 태스크 생성 및 데이터 마이그레이션 섹션을 참조하세요.

데이터베이스 암호 변경

대다수의 경우에 소스 또는 대상 엔드포인트에서 데이터베이스 암호를 쉽게 변경할 수 있습니다. 마이그레이션이나 복제 작업에 현재 사용하고 있는 엔드포인트의 데이터베이스 암호를 변경해야 하는 경우, 이 과정은 약간 더 복잡합니다. 다음 절차에서는 이 작업을 수행하는 방법을 보여 줍니다.

마이그레이션이나 복제 작업에 현재 사용하고 있는 엔드포인트의 데이터베이스 암호를 변경하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/dms/v2/에서 AWS DMS 콘솔을 엽니다.

   IAM 사용자로 로그인한 경우 AWS DMS에 액세스하기 위한 적절한 권한이 있어야 합니다. 필요한 권한에 대한 자세한 내용은 AWS DMS 사용에 필요한 IAM 권한 섹션을 참조하세요.
2. 탐색 창에서 데이터베이스 마이그레이션 작업을 선택합니다.
3. 변경하려는 엔드포인트를 사용하는 작업을 선택하여 데이터베이스 암호를 변경한 후 [중지]를 선택합니다.
4. 작업을 중지한 동안 데이터베이스에서 사용하는 기본 도구를 사용하여 엔드포인트의 데이터베이스 암호를 변경할 수 있습니다.
5. DMS 관리 콘솔로 돌아가 탐색 창에서 [엔드포인트]를 선택합니다.
6. 암호를 변경할 데이터베이스의 엔드포인트를 선택하고 나서 [수정]을 선택합니다.
7. 암호 상자에 새 암호를 입력하고 저장을 선택합니다.
8. 탐색 창에서 데이터베이스 마이그레이션 작업을 선택합니다.
9. 이전에 중지한 작업을 선택하고 시작/재개를 선택합니다.
10. 작업을 계속할 방법에 따라 다시 시작 또는 재개를 선택하고 작업 시작을 선택합니다.

AWS Database Migration Service에 대한 할당량

다음에서 AWS Database Migration Service(AWS DMS)의 리소스 할당량 및 명명 제약 조건을 확인할 수 있습니다.

AWS DMS가 마이그레이션할 수 있는 데이터베이스의 최대 크기는 여러 요인에 따라 달라집니다. 예를 들어 소스 환경, 소스 데이터베이스에서 데이터 배포, 소스 시스템의 사용 빈도에 따라 다를 수 있습니다.

특정 시스템이 AWS DMS에 알맞은지 여부를 판단하는 최선의 방법은 테스트해 보는 것입니다. 구성이 작동하도록 천천히 시작한 후 복합 객체를 추가합니다. 마지막으로 전체 로드를 테스트로 시도해 봅니다.

AWS Database Migration Service에 대한 리소스 할당량

각 AWS 계정에는 AWS 리전마다 생성할 수 있는 AWS DMS 리소스 수에 할당량이 있습니다. 리소스 할당량에 도달하면 해당 리소스 생성을 위한 추가 호출이 예외와 함께 실패합니다.

다음 표에는 AWS 리전별 AWS DMS 리소스 및 그 할당량이 나열되어 있습니다.

Resource	기본 할당량
API 요청 제한	초당 최대 요청 200개
API 요청 새로 고침 빈도	초당 요청 8개
사용자 계정당 복제 인스턴스 수	60
복제 인스턴스에 대한 총 스토리지 양	30,000GB
사용자 계정당 이벤트 구독 수	60
사용자 계정당 복제 서브넷 그룹 수	60
복제 서브넷 그룹당 서브넷	60
사용자 계정당 엔드포인트 수	1,000
복제 인스턴스당 엔드포인트 수	100

Resource	기본 할당량
사용자 계정당 태스크 수	600
복제 인스턴스당 태스크 수	200
사용자 계정당 인증서 수	100
사용자 계정당 데이터 공급자 수	1,000
사용자 계정당 인스턴스 프로파일 수	60
사용자 계정당 마이그레이션 프로젝트 수	10
사용자 계정당 DMS 데이터 수집기 수	10
한 번에 생성되는 대상 권장 사항 수	100
DMS 데이터 수집기가 시간당 업로드할 수 있는 파일 수	500
사용자 계정당 동종 데이터 마이그레이션 수	600
한 번에 실행되는 동종 데이터 마이그레이션 수	100
마이그레이션 프로젝트당 동종 데이터 마이그레이션 수	10
서버리스 복제	100

API 요청 제한 할당량 및 새로 고침 빈도에 대한 자세한 내용은 API 요청 제한에 대한 이해 섹션을 참조하세요.

30,000GB의 스토리지 할당량은 해당 AWS 리전의 모든 AWS DMS 복제 인스턴스에 적용됩니다. 이 스토리지는 대상이 소스를 유지하고 로그 정보를 저장할 수 없는 경우에 변경 사항을 캐시하는 데 사용됩니다.

API 요청 제한에 대한 이해

AWS DMS는 초당 200개의 API 직접 호출이라는 최대 API 요청 할당량(리전에 따라 다를 수 있음)을 지원합니다. 즉, API 요청이 이 속도를 초과하면 병목 현상이 발생합니다. 또한 다른 API 요청을 하기 전에 AWS DMS가 할당량을 새로 고치는 데 걸리는 시간에 따라 초당 API 직접 호출 수를 줄이도록 제

한할 수도 있습니다. 이 할당량은 API를 직접 호출하는 경우와 AWS DMS 관리 콘솔을 사용하여 사용자 대신 호출하는 경우 모두에 적용됩니다.

API 요청 제한이 작동하는 방식을 이해하려면 AWS DMS가 API 요청을 추적하는 토큰 버킷을 유지 관리한다고 상상해 보는 것이 좋습니다. 이 시나리오에서는 버킷의 각 토큰을 사용하여 단일 API 직접 호출을 실행할 수 있습니다. 버킷에서 한 번에 200개 이상의 토큰을 보유할 수 없습니다. API 직접 호출을 실행할 때마다 AWS DMS가 버킷에서 토큰을 하나 제거합니다. 1초 이내에 200개의 API 직접 호출을 실행하면 버킷이 비어 있고 다른 API 직접 호출을 시도해도 실패합니다. API 직접 호출을 실행하지 않는 매초마다 AWS DMS는 버킷에 8개의 토큰을 최대 200개가 될 때까지 추가합니다. 이것이 AWS DMS API 요청 새로 고침 빈도입니다. 제한 후 언제든지 버킷에 토큰을 추가하면 호출이 다시 제한될 때까지 사용 가능한 토큰만큼 추가 API 직접 호출을 실행할 수 있습니다.

AWS CLI를 사용하여 제한이 발생하는 API 직접 호출을 실행하는 경우 AWS DMS가 다음과 같은 오류를 반환합니다.

```
An error occurred (ThrottlingException) when calling the AwsDmsApiCall operation
 (reached max retries: 2): Rate exceeded
```

여기에서 *AwsDmsApiCall*은 제한이 발생한 AWS DMS API 작업의 이름입니다(예: `DescribeTableStatistics`). 그러면 제한을 방지하도록 충분한 시간이 경과한 후 다시 시도하거나 다른 호출을 시도할 수 있습니다.

> **Note**
>
> Amazon EC2와 같은 몇몇 다른 서비스에서 관리하는 API 요청 제한과 달리 AWS DMS에서 관리하는 API 요청 제한 할당량은 증가를 요청할 수 없습니다.

AWS Database Migration Service에서 마이그레이션 작업 문제 해결

아래에는 AWS Database Migration Service(AWS DMS)와 관련된 문제 해결에 대한 주제가 나와 있습니다. 이러한 주제는 AWS DMS 및 선택한 엔드포인트 데이터베이스를 모두 사용하여 일반적인 문제를 해결하는 데 도움이 될 수 있습니다.

AWS Support 사례를 열면 지원 엔지니어가 엔드포인트 데이터베이스 구성 중 하나와 관련된 잠재적인 문제를 파악할 수 있습니다. 엔지니어가 데이터베이스에 대한 진단 정보를 반환하는 지원 스크립트를 실행할 것을 요청할 수도 있습니다. 이러한 유형의 지원 스크립트에서 진단 정보를 다운로드, 실행, 업로드하는 방법에 대한 자세한 내용은 AWS DMS에서 진단 지원 스크립트 작업 섹션을 참조하세요.

문제 해결을 위해 복제 인스턴스에서 추적 및 덤프 파일을 AWS DMS 수집합니다. 문제 해결이 필요한 문제가 발생하는 경우 AWS Support에 이러한 파일을 제공할 수 있습니다. 기본적으로 DMS는 30일이 지난 추적 및 덤프 파일을 제거합니다. 추적 및 덤프 파일 수집을 거부하려면 AWS Support에 케이스를 여십시오.

주제

- 마이그레이션 태스크가 느리게 실행됨
- 태스크 상태 표시줄이 움직이지 않음
- 태스크가 완료되었지만 아무것도 마이그레이션되지 않음
- 외부 키와 보조 인덱스 누락
- AWS DMS CloudWatch 로그를 생성하지 않습니다.
- Amazon RDS에 연결할 때 문제가 발생함
- 네트워킹 문제 발생
- 전체 로드 후 CDC 중단
- 태스크 재시작 시 프라이머리 키 위반 오류 발생
- 스키마 초기 로드 실패
- 알 수 없는 오류로 인해 태스크 실패
- 시작부터 작업 재시작 시 테이블 로드
- 태스크당 테이블 수로 인해 문제 발생
- LOB 열에 프라이머리 키가 생성되면 태스크가 실패함

- 프라이머리 키가 없는 대상 테이블에서 중복 레코드 발생
- 소스 엔드포인트가 예약된 IP 범위에 속함
- Amazon Athena 쿼리에서 타임스탬프가 깨져 보임
- Oracle 관련 문제 해결
- MySQL 관련 문제 해결
- PostgreSQL 관련 문제 해결
- Microsoft SQL Server 관련 문제 해결
- Amazon Redshift 관련 문제 해결
- Amazon Aurora MySQL 관련 문제 해결
- SAP ASE 관련 문제 해결
- IBM Db2 관련 문제 해결
- AWS Database Migration Service의 지연 시간 문제 해결
- AWS DMS에서 진단 지원 스크립트 작업
- AWS DMS 진단 지원 AMI 작업

마이그레이션 태스크가 느리게 실행됨

마이그레이션 작업이 느리게 실행되거나 연속 작업이 초기 작업보다 느리게 실행되도록 만들 수 있는 여러 문제가 있습니다.

마이그레이션 태스크가 느리게 실행되는 가장 일반적인 이유는 AWS DMS 복제 인스턴스에 부적절한 리소스가 할당되었기 때문입니다. 복제 인스턴스의 CPU, 메모리, 스왑 파일, IOPS 사용량을 확인하여 실행 중인 태스크를 지원하는 리소스가 인스턴스에 충분히 있는지 확인합니다. 예를 들어, 엔드포인트로서 Amazon Redshift에서 여러 태스크를 실행하면 I/O가 집중적으로 사용됩니다. 복제 인스턴스용 IOPS를 늘리거나 여러 복제 인스턴스에서 작업을 분할하면 보다 효율적인 마이그레이션이 가능합니다.

복제 인스턴스 크기 결정에 대한 자세한 내용은 복제 인스턴스에 가장 적합한 크기 선택 섹션을 참조하세요.

다음 작업을 수행하여 초기 마이그레이션 로드 속도를 촉진할 수 있습니다.

- 대상이 Amazon RDS DB 인스턴스인 경우, 다중 AZ가 대상 DB 인스턴스에서 활성화되지 않도록 해야 합니다.

- 로드 중에 대상 데이터베이스에서 자동 백업 또는 로깅 기능을 끄고, 마이그레이션이 완료된 후 이러한 기능을 다시 켭니다.
- 이 기능을 대상에서 사용할 수 있는 경우, 프로비저닝된 IOPS를 사용합니다.
- 마이그레이션 데이터에 LOB가 포함된 경우, 작업이 LOB 마이그레이션에 최적화되어 있는지 확인합니다. LOB를 위한 최적화에 대한 자세한 내용은 대상 메타데이터 작업 설정 섹션을 참조하세요.

태스크 상태 표시줄이 움직이지 않음

작업 상태 표시줄은 작업 진행률의 추정치를 나타냅니다. 이 추정치의 품질은 소스 데이터베이스의 테이블 통계 품질에 따라 달라지며, 테이블 통계 품질이 좋을수록 추정 정확도가 높아집니다.

추정 행 통계 없이 단 하나의 테이블을 사용하는 태스크의 경우, AWS DMS는 모든 유형의 완료율 추정치를 제공할 수 없습니다. 이 경우 태스크 상태와 로드되는 행 표시를 사용하여 태스크가 실제로 실행 및 진행되고 있다는 것을 확인할 수 있습니다.

태스크가 완료되었지만 아무것도 마이그레이션되지 않음

태스크가 완료된 후 아무것도 마이그레이션되지 않은 경우 다음을 수행합니다.

- 엔드포인트를 생성한 사용자에게 마이그레이션하려는 테이블에 대한 읽기 권한이 있는지 확인합니다.
- 마이그레이션하려는 객체가 테이블인지 확인합니다. 보기인 경우, 테이블 매핑을 업데이트하고 객체 로케이터를 'view' 또는 'all'로 지정합니다. 자세한 설명은 콘솔에서 테이블 선택 및 변환 규칙 지정 섹션을 참조하세요.

외부 키와 보조 인덱스 누락

AWS DMS는 테이블과 프라이머리 키를 생성하고 경우에 따라 고유 인덱스도 생성하지만, 소스의 데이터를 효율적으로 마이그레이션하는 데 필요하지 않은 다른 객체는 생성하지 않습니다. 예를 들어 보조 인덱스, 기본이 아닌 키 제약 조건 또는 데이터 기본값을 생성하지 않습니다.

데이터베이스에서 보조 객체를 마이그레이션하려면, 원본 데이터베이스와 동일한 데이터베이스 엔진으로 이동할 경우 데이터베이스의 기본 도구를 사용합니다. 보조 객체를 마이그레이션하기 위해 소스 데이터베이스가 사용하는 것과 다른 데이터베이스 엔진으로 마이그레이션할 경우 AWS Schema Conversion Tool(AWS SCT)을 사용합니다.

AWS DMS CloudWatch 로그를 생성하지 않습니다.

복제 작업에서 CloudWatch 로그가 생성되지 않는 경우 계정에 `dms-cloudwatch-logs-role` 역할이 있는지 확인하세요. 이 역할이 없는 경우 다음을 수행하여 생성합니다.

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 역할 탭을 선택합니다. Create role(역할 생성)을 선택합니다.
3. 신뢰할 수 있는 유형의 엔터티 선택 섹션에서 AWS 서비스를 선택합니다.
4. 사용 사례 선택 섹션에서 DMS를 선택합니다.
5. 다음: 권한을 선택합니다.
6. 검색 **`AmazonDMSCloudWatchLogsRole`** 필드에 을 입력하고 CloudWatchLogsRoleAmazonDMS 옆의 체크박스를 선택합니다. 이렇게 하면 AWS DMS 액세스 권한이 부여됩니다. CloudWatch
7. 다음: 태그를 선택합니다.
8. 다음: 검토를 선택합니다.
9. 역할 이름에 **`dms-cloudwatch-logs-role`**을 입력합니다. 이 이름은 대/소문자를 구분합니다.
10. Create role(역할 생성)을 선택합니다.

Amazon RDS에 연결할 때 문제가 발생함

소스 또는 대상으로 설정한 Amazon RDS DB 인스턴트에 연결할 수 없는 이유에는 여러 가지가 있습니다. 확인해야 할 몇 가지 항목은 다음과 같습니다.

- 사용자 이름과 암호 조합이 올바른지 확인합니다.
- 인스턴스의 Amazon RDS 콘솔에 표시된 엔드포인트 값이 AWS DMS 엔드포인트를 생성하는 데 사용된 엔드포인트 식별자와 동일한지 확인합니다.
- 인스턴스의 Amazon RDS 콘솔에 표시된 포트 값이 AWS DMS 엔드포인트에 할당된 포트와 동일한지 확인합니다.
- Amazon RDS DB 인스턴스에 할당된 보안 그룹이 AWS DMS 복제 인스턴스의 연결을 허용하는지 확인합니다.
- AWS DMS 복제 인스턴스와 Amazon RDS DB 인스턴스가 동일한 Virtual Private Cloud(VPC)에 없는 경우, DB 인스턴스에 공개적으로 액세스할 수 있는지 확인합니다.

오류 메시지: Incorrect thread connection string: incorrect thread value 0

이 오류는 흔히 엔드포인트와의 연결을 테스트할 때 발생할 수 있습니다. 이 오류는 연결 문자열에 오류가 있음을 나타냅니다. 호스트 IP 주소 뒤의 공백을 예로 들 수 있습니다. 또 다른 예로, 연결 문자열에 잘못된 문자를 복사하는 경우도 있습니다.

네트워킹 문제 발생

가장 일반적인 네트워킹 문제에는 AWS DMS 복제 인스턴스에서 사용하는 VPC 보안 그룹이 포함됩니다. 기본적으로 이 보안 그룹에는 모든 포트에서 0.0.0.0/0로의 발신을 허용하는 규칙이 있습니다. 대부분의 경우 이 보안 그룹을 수정하거나 자체 보안 그룹을 사용합니다. 이렇게 할 경우, 최소한 각 데이터베이스 포트의 소스 및 대상 엔드포인트에 송신을 제공해야 합니다.

그 밖의 구성 관련 문제에는 다음이 포함될 수 있습니다.

- 동일한 VPC의 복제 인스턴스와 소스 및 대상 엔드포인트 – 엔드포인트에서 사용하는 보안 그룹은 복제 인스턴스의 데이터베이스 포트에서 수신을 허용해야 합니다. 복제 인스턴스에서 사용하는 보안 그룹에 엔드포인트로의 수신이 있는지 확인합니다. 그렇지 않을 경우, 복제 인스턴스 액세스의 프라이빗 IP 주소를 허용하고 엔드포인트가 사용하는 보안 그룹에 규칙을 생성할 수 있습니다.
- 소스 엔드포인트가 복제 인스턴스에서 사용하는 VPC 외부에 있음(인터넷 게이트웨이 사용) – VPC 보안 그룹에는 VPC로 전달되지 않고 인터넷 게이트웨이로 트래픽을 전송하는 라우팅 규칙이 포함되어야 합니다. 이 구성에서 엔드포인트와의 연결이 복제 인스턴스의 퍼블릭 IP 주소에서 오는 것으로 나타납니다.
- 소스 엔드포인트가 복제 인스턴스에서 사용하는 VPC 외부에 있음(NAT 게이트웨이 사용) - 단일 탄력적 네트워크 인터페이스에 연결된 단일 탄력적 IP 주소를 사용하여 Network Address Translation(NAT) 게이트웨이를 구성할 수 있습니다. 이 NAT 게이트웨이는 NAT 식별자(nat-#####)를 수신합니다.

  경우에 따라 VPC에는 인터넷 게이트웨이 대신, 이러한 NAT 게이트웨이에 대한 기본 경로가 포함됩니다. 이러한 경우 복제 인스턴스는 NAT 게이트웨이의 퍼블릭 IP 주소를 사용하여 데이터베이스 엔드포인트에 접속하는 것으로 보입니다. 이런 상황에서 VPC 외부에 있는 데이터베이스 엔드포인트로의 수신은 복제 인스턴스의 퍼블릭 IP 주소가 아닌 NAT 주소의 수신을 허용해야 합니다.

자체 온프레미스 네임 서버 사용에 대한 자세한 내용은 자체 온프레미스 이름 서버 사용 섹션을 참조하세요.

전체 로드 후 CDC 중단

느리거나 중단된 복제 변경은 여러 AWS DMS 설정이 서로 충돌할 때 전체 로드 마이그레이션 이후에 나타날 수 있습니다.

예를 들어 대상 테이블 준비 모드 파라미터가 아무 작업 안 함 또는 자르기로 설정되어 있다고 가정해 보겠습니다. 이 경우 기본 및 고유 인덱스 생성을 포함하여, 대상 테이블에서 설정을 수행하지 않도록 AWS DMS에 지시했습니다. 대상 테이블에서 기본 또는 고유 키를 생성하지 않은 경우, AWS DMS는 업데이트를 할 때마다 전체 테이블 스캔을 수행합니다. 이런 접근 방식은 성능에 상당한 영향을 미칠 수 있습니다.

태스크 재시작 시 프라이머리 키 위반 오류 발생

이 오류는 데이터가 이전 마이그레이션 작업의 대상 데이터베이스에 있을 때 발생할 수 있습니다. 대상 테이블 준비 모드 옵션을 아무 작업 안 함으로 설정하면, AWS DMS에서는 이전 태스크에서 삽입된 데이터 정리 등을 포함하여, 대상 테이블에서 어떠한 준비도 수행하지 않습니다.

작업을 다시 시작하고 이러한 오류를 방지하려면, 이전 태스크 실행에서 대상 테이블에 삽입된 행을 제거합니다.

스키마 초기 로드 실패

경우에 따라 `Operation:getSchemaListDetails:errType=, status=0, errMessage=, errDetails=` 오류가 발생하여 스키마의 초기 로드가 실패할 수 있습니다.

이러한 경우 AWS DMS가 소스 엔드포인트에 연결하는 데 사용하는 사용자 계정에 필요한 권한이 없는 상태입니다.

알 수 없는 오류로 인해 태스크 실패

알 수 없는 유형의 오류가 발생하는 원인은 다양할 수 있습니다. 하지만 AWS DMS 복제 인스턴스에 할당된 리소스가 부족하기 때문에 문제가 발생하는 경우가 종종 있습니다.

복제 인스턴스의 CPU, 메모리, 스왑 파일, IOPS 사용량을 확인하여 마이그레이션을 수행하는 리소스가 인스턴스에 충분히 있는지 확인합니다. 모니터링에 대한 자세한 내용은 AWS Database Migration Service 지표 섹션을 참조하십시오.

시작부터 작업 재시작 시 테이블 로드

테이블의 초기 로드가 끝나지 않으면 AWS DMS는 테이블 로드를 처음부터 다시 시작합니다. 태스크가 다시 시작되면 AWS DMS는 초기 로드가 완료되지 않은 상태에서 테이블을 처음부터 다시 로드합니다.

태스크당 테이블 수로 인해 문제 발생

복제 태스크당 테이블 수에 대한 제한은 설정되어 있지 않습니다. 그러나 한 태스크의 테이블 수는 60,000개 미만으로 제한하는 것이 좋습니다. 단일 작업에서 60,000개 이상의 테이블을 사용할 때 리소스에 흔히 병목 현상이 발생할 수 있습니다.

LOB 열에 프라이머리 키가 생성되면 태스크가 실패함

FULL LOB 또는 LIMITED LOB 모드에서 AWS DMS는 LOB 데이터 형식인 프라이머리 키의 복제를 지원하지 않습니다.

DMS는 처음에 LOB 열이 null인 행을 마이그레이션한 다음 나중에 LOB 열을 업데이트합니다. 따라서 LOB 열에 기본 키가 생성되면 기본 키가 null이 될 수 없으므로 초기 삽입이 실패합니다. 차선책으로서 다른 열을 프라이머리 키로 추가하고 LOB 열에서 프라이머리 키를 제거합니다.

프라이머리 키가 없는 대상 테이블에서 중복 레코드 발생

전체 로드 및 CDC 태스크를 실행하면 프라이머리 키 또는 고유 인덱스가 없는 대상 테이블에 중복 레코드를 생성할 수 있습니다. 전체 로드 및 CDC 태스크 중에 대상 테이블에 레코드가 중복되지 않도록 하려면 대상 테이블에 프라이머리 키 또는 고유 인덱스가 있어야 합니다.

소스 엔드포인트가 예약된 IP 범위에 속함

AWS DMS 소스 데이터베이스가 예약된 IP 범위 내의 IP 주소 192.168.0.0/24를 사용하는 경우 소스 엔드포인트 연결 테스트가 실패합니다. 다음 단계에서는 가능한 해결 방법을 제공합니다.

1. 192.168.0.0/24에서 소스 데이터베이스와 통신할 수 있는 예약된 범위에 없는 Amazon EC2 인스턴스를 찾습니다.
2. socat 프록시를 설치하고 실행합니다. 다음은 그 한 예입니다.

```
yum install socat

socat -d -d -lmlocal2 tcp4-listen:database port,bind=0.0.0.0,reuseaddr,fork
 tcp4:source_database_ip_address:database_port
&
```

AWS DMS 엔드포인트에 대해 위에서 지정한 Amazon EC2 인스턴스 IP 주소와 데이터베이스 포트를 사용합니다. 엔드포인트에 AWS DMS가 데이터베이스 포트에 액세스할 수 있도록 허용하는 보안 그룹이 있어야 합니다. 단, 프록시는 DMS 태스크 실행 기간에 실행해야 합니다. 사용 사례에 따라, 프록시 설정을 자동화해야 할 수도 있습니다.

Amazon Athena 쿼리에서 타임스탬프가 깨져 보임

Athena 쿼리에서 타임스탬프가 제대로 표시되지 않는 경우 AWS Management Console 또는 ModifyEndpoint작업을 사용하여 Amazon S3 엔드포인트의 `parquetTimestampInMillisecond` 값을 로 설정하십시오. `true` 자세한 내용은 S3Settings 섹션을 참조하세요.

Oracle 관련 문제 해결

아래에서 AWS DMS를 Oracle 데이터베이스와 함께 사용했을 때 발생하는 문제에 대한 해결 방법을 배울 수 있습니다.

주제

- 보기에서 데이터 가져오기
- 12c에서 LOB 마이그레이션
- LogMiner오라클과 바이너리 리더 간 전환
- 오류: Oracle CDC stopped 122301 Oracle CDC maximum retry counter exceeded.
- 보충 로깅을 Oracle 소스 엔드포인트에 자동으로 추가합니다.
- LOB 변경 사항이 캡처되지 않음
- 오류: ORA-12899: value too large for column column-name
- NUMBER 데이터 형식 오해
- 전체 로드 중 레코드 누락
- 테이블 오류

- 오류: Cannot retrieve Oracle archived Redo log destination ids
- Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능 평가

보기에서 데이터 가져오기

보기에서 데이터를 끌어와서 지속적인 복제에 사용할 수 없습니다. 보기에서 데이터를 추출하려면 Oracle 소스 엔드포인트 페이지의 엔드포인트 설정 섹션에 아래와 같은 코드를 추가해야 합니다. 보기에서 데이터를 추출하면 보기가 대상 스키마에서 테이블로 표시됩니다.

```
"ExposeViews": true
```

12c에서 LOB 마이그레이션

AWS DMS두 가지 방법, 즉 바이너리 리더와 Oracle을 사용하여 Oracle 데이터베이스의 변경 내용을 캡처할 수 있습니다. LogMiner 기본적으로 LogMiner Oracle을 AWS DMS 사용하여 변경 사항을 캡처합니다. 하지만 Oracle 12c에서는 오라클이 LOB 열을 LogMiner 지원하지 않습니다. Oracle 12c에서 LOB 열의 변경 사항을 캡처하려면, Binary Reader를 사용합니다.

LogMiner오라클과 바이너리 리더 간 전환

AWS DMS원본 Oracle 데이터베이스의 변경 내용을 캡처하는 데 바이너리 리더와 Oracle이라는 두 가지 방법을 사용할 수 LogMiner 있습니다. LogMiner Oracle이 기본값입니다. 변경 캡처를 위해 Binary Reader 사용으로 전환하려면, 다음 작업을 수행합니다.

변경 캡처를 위해 Binary Reader를 사용하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 엔드포인트를 선택합니다.
3. Binary Reader를 사용할 Oracle 소스 엔드포인트를 선택합니다.
4. 수정을 선택합니다.
5. 고급을 선택한 다음, 추가 연결 속성에 대해 아래의 코드를 추가합니다.

```
useLogminerReader=N
```

6. SQL-Plus와 같은 Oracle 개발자 도구를 사용하여 Oracle 엔드포인트에 연결하는 데 사용되는 AWS DMS 사용자 계정에 추가 권한을 부여합니다.

```
SELECT ON V_$TRANSPORTABLE_PLATFORM
```

오류: Oracle CDC stopped 122301 Oracle CDC maximum retry counter exceeded.

AWS DMS에서 Oracle 보관 로그를 사용하여 변경 사항을 캡처하기 전에 서버에서 필요한 Oracle 보관 로그를 제거할 때 이 오류가 발생합니다. 데이터 서버에서 로그 보존 기간 정책을 증가시킵니다. Amazon RDS 데이터베이스에서는 다음 절차를 실행하여 로그 보존 기간을 증가시킵니다. 예를 들어, 다음 코드는 Amazon RDS DB 인스턴스에서 로그 보존 기간을 24시간으로 증가시킵니다.

```
exec rdsadmin.rdsadmin_util.set_configuration('archivelog retention hours',24);
```

보충 로깅을 Oracle 소스 엔드포인트에 자동으로 추가합니다.

기본적으로 AWS DMS에서는 보충 로깅을 끕니다. 원본 Oracle 엔드포인트에서 보충 로깅을 자동으로 켜려면, 다음을 수행합니다.

보충 로깅을 소스 Oracle 엔드포인트에 추가하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 엔드포인트를 선택합니다.
3. 보충 로깅을 추가할 Oracle 원본 엔드포인트를 선택합니다.
4. 수정을 선택합니다.
5. 고급을 선택하고 나서 추가 연결 속성 텍스트 상자에 다음 코드를 추가합니다.

```
addSupplementalLogging=Y
```

6. 수정을 선택합니다.

LOB 변경 사항이 캡처되지 않음

현재 테이블에는 LOB 변경 사항을 캡처하기 위한 AWS DMS의 기본 키가 있어야 합니다. LOB를 포함하는 테이블에 기본 키가 없는 경우, LOB 변경 사항을 캡처하기 위해 취할 수 있는 몇 가지 조치가 있습니다.

- 테이블에 기본 키를 추가합니다. 이 작업은 트리거를 사용하여 ID 열을 추가하고 이 열을 시퀀스로 채우는 것만큼이나 간편합니다.
- 프라이머리 키로서 시스템에서 생성한 ID를 포함하는 테이블의 구체화된 뷰를 생성하고 테이블 이외에 구체화된 뷰를 마이그레이션합니다.
- 논리 대기를 생성하고, 테이블에 기본 키를 추가하고, 논리 대기에서 마이그레이션합니다.

오류: ORA-12899: value too large for column *column-name*

"ORA-12899: value too large for column *column-name*" 오류는 주로 몇 가지 문제로 인해 발생합니다.

이러한 문제 중 하나는 소스 데이터베이스와 대상 데이터베이스에서 사용하는 문자 집합이 일치하지 않는 것입니다.

또 다른 문제는 두 데이터베이스의 National Language Support(NLS) 설정이 다르다는 점입니다. 이 원인의 일반적인 원인은 원본 데이터베이스 NLS_LENGTH_SEMANTICS 파라미터가 CHAR로 설정되고 대상 데이터베이스 NLS_LENGTH_SEMANTICS 파라미터가 BYTE로 설정되어 있기 때문입니다.

NUMBER 데이터 형식 오해

Oracle NUMBER 데이터 형식은 정밀도와 NUMBER 크기에 따라 여러 AWS DMS 데이터 형식으로 변환됩니다. 이 변환은 Oracle용 소스 데이터 형식에 소개되어 있습니다. 소스 Oracle 엔드포인트에 엔드포인트 설정을 사용할 경우 NUMBER 형식의 변환 방식에 영향을 미칠 수도 있습니다. 이러한 엔드포인트 설정은 Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS에 설명되어 있습니다.

전체 로드 중 레코드 누락

전체 로드를 수행할 경우, AWS DMS는 데이터베이스 수준에서 열린 트랜잭션을 찾은 후 트랜잭션이 커밋될 때까지 대기합니다. 예를 들어, 태스크 설정 `TransactionConsistencyTimeout=600`이 기준인 경우 AWS DMS는 테이블 매핑에 포함되지 않은 테이블에 열린 트랜잭션이 있더라도 10분간 대

기합니다. 그러나 열린 트랜잭션이 테이블 매핑에 포함된 테이블에 있고, 트랜잭션이 제시간에 커밋되지 않으면 대상 테이블 결과에서 레코드가 누락됩니다.

열린 트랜잭션을 커밋하는 데 시간이 오래 걸린다는 점을 알게 된 경우, `TransactionConsistencyTimeout` 태스크 설정을 수정하고 대기 시간을 늘릴 수 있습니다.

또한 `FailOnTransactionConsistencyBreached` 태스크 설정의 기본값은 `false`입니다. 즉, AWS DMS가 다른 트랜잭션은 계속 적용하지만 열린 트랜잭션은 누락됩니다. 열린 트랜잭션이 제 시간에 종료되지 않은 경우 태스크가 실패하도록 설정하려면 `FailOnTransactionConsistencyBreached`를 `true`로 설정하면 됩니다.

테이블 오류

`Table Error` 절이 프라이머리 키 열을 참조하지 않고, 모든 열에 보충 로깅이 사용되지 않을 경우 복제 중에 테이블 통계에 `WHERE`가 표시됩니다.

이 문제를 해결하려면 참조된 테이블의 모든 열에 대해 보충 로깅을 설정합니다. 자세한 설명은 보충 로깅 설정 섹션을 참조하세요.

오류: Cannot retrieve Oracle archived Redo log destination ids

이 오류는 Oracle 소스에 생성된 아카이브 로그가 없거나, V$ARCHIVED_LOG가 비어 있는 경우 발생합니다. 로그를 수동으로 전환하여 오류를 해결할 수 있습니다.

Amazon RDS 데이터베이스에서는 다음 절차를 실행하여 로그 파일을 전환합니다. `switch_logfile` 절차에는 파라미터가 없습니다.

```
exec rdsadmin.rdsadmin_util.switch_logfile;
```

자체 관리형 Oracle 소스 데이터베이스에서는 아래의 명령을 사용하여 로그 전환을 적용합니다.

```
ALTER SYSTEM SWITCH LOGFILE ;
```

Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능 평가

Oracle 소스에서 성능 문제가 발생할 경우, Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능을 평가하여 성능을 개선할 방법을 찾을 수 있습니다. 재실행 또는 아카이브 로그의 읽기 성능을 테스트하려면 AWS DMS 진단 Amazon Machine Image(AMI)를 사용합니다.

AWS DMS 진단 AMI를 사용하면 다음 작업을 수행할 수 있습니다.

- bFile 메서드를 사용하여 재실행 로그 파일의 성능을 평가합니다.
- LogMiner 메서드를 사용하여 리두 로그 파일 성능을 평가하십시오.
- PL/SQL(dbms_lob.read) 메서드를 사용하여 재실행 로그 파일의 성능을 평가합니다.
- 단일 스레드를 사용하여 ASMFile의 읽기 성능을 평가합니다.
- 다중 스레드를 사용하여 ASMFile의 읽기 성능을 평가합니다.
- Direct OS Readfile() Windows 또는 Pread64 Linux 함수를 사용하여 재실행 로그 파일을 평가합니다.

그런 다음, 결과에 따라 수정 단계를 진행할 수 있습니다.

Oracle 재실행 또는 아카이브 로그 파일의 읽기 성능을 테스트하려면

1. AWS DMS 진단 AMI Amazon EC2 인스턴스를 생성한 후 이 인스턴스에 연결합니다.

   자세한 내용은 AWS DMS 진단 AMI 작업 섹션을 참조하세요.
2. awsreplperf 명령을 실행합니다.

   ```
   $ awsreplperf
   ```

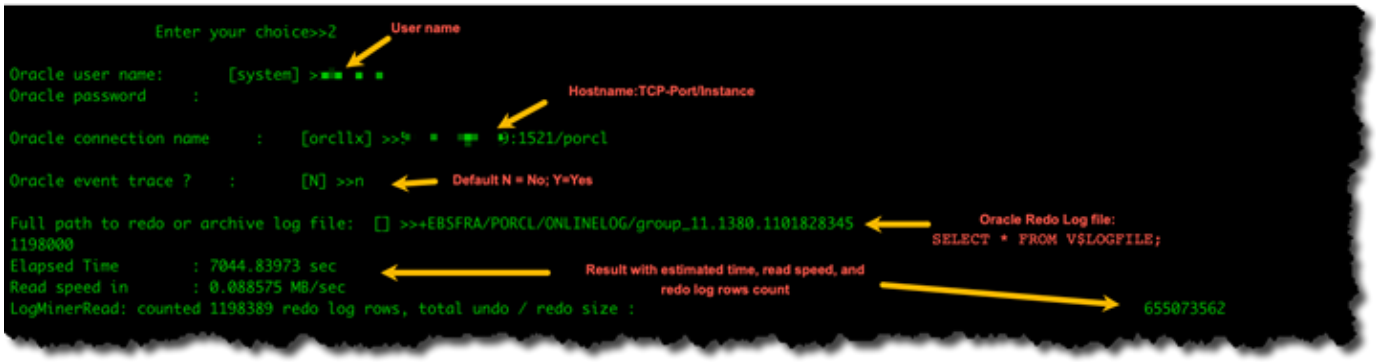
   이 명령은 AWS DMS Oracle 읽기 성능 유틸리티 옵션을 표시합니다.

   ```
   0. Quit
   1. Read using Bfile
   2. Read using LogMiner
   3. Read file PL/SQL (dms_lob.read)
   4. Read ASMFile Single Thread
   5. Read ASMFile Multi Thread
   6. Readfile() function
   ```
3. 목록에서 옵션을 선택합니다.
4. 아래의 데이터베이스 연결 및 아카이브 로그 정보를 입력합니다.

   ```
   Oracle user name [system]:
   Oracle password:

   Oracle connection name [orcllx]:
   Connection format hostname:port/instance
   ```

```
Oracle event trace? [N]:
Default N = No or Y = Yes

Path to redo or archive log file []:
```

5. 표시된 출력에서 관련 읽기 성능 정보를 확인합니다. 예를 들어, 다음은 옵션 번호 2, 읽기 사용을 LogMiner 선택한 결과 발생할 수 있는 출력을 보여줍니다.



6. 유틸리티를 종료하려면 0을 입력합니다.

다음 단계

- 결과에 읽기 속도가 허용 가능한 임계값 미만인 것으로 표시될 경우, 엔드포인트에서 Oracle 진단 지원 스크립트를 실행하고 대기 시간, 로드 프로필, IO 프로필 섹션을 검토합니다. 그런 다음, 읽기 성능을 개선할 수 있는 비정상적인 구성을 조정합니다. 예를 들어, 재실행 로그 파일이 최대 2GB인 경우 LOG_BUFFER를 200MB로 늘리면 성능을 개선하는 데 도움이 될 수 있습니다.
- AWS DMS 모범 사례를 검토하여 DMS 복제 인스턴스, 태스크, 엔드포인트가 최적으로 구성되어 있는지 확인합니다.

MySQL 관련 문제 해결

아래에서 AWS DMS를 MySQL 데이터베이스와 함께 사용했을 때 발생하는 문제에 대한 해결 방법을 배울 수 있습니다.

주제

- 이진 로깅 비활성화로 인해 Amazon RDS DB 인스턴스 엔드포인트에서 CDC 작업 실패
- 작업 중에 대상 MySQL 인스턴스 연결이 끊김
- MySQL 호환 엔드포인트에 자동 커밋 추가
- 대상 MySQL 호환 엔드포인트에서 외래 키 비활성화

- 물음표로 대체된 문자
- 'Bad event' 로그 항목
- MySQL 5.5로 변경 데이터 캡처
- Amazon RDS DB 인스턴스를 위한 이진 로그 보존 기간 증가
- 로그 메시지: 소스 데이터베이스의 일부 변경 사항은 대상 데이터베이스에 적용될 때 영향이 전혀 없었습니다.
- 오류: Identifier too long
- 오류: 지원되지 않는 문자 집합으로 인해 필드 데이터 변환 실패
- 오류: Codepage 1252 to UTF8 [120112] A field data conversion failed
- 인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음

이진 로깅 비활성화로 인해 Amazon RDS DB 인스턴스 엔드포인트에서 CDC 작업 실패

이 문제는 자동 백업이 비활성화되어 있기 때문에 Amazon RDS DB 인스턴스에서 발생합니다. 백업 보존 기간을 0이 아닌 값으로 설정하여 자동 백업을 활성화합니다.

작업 중에 대상 MySQL 인스턴스 연결이 끊김

MySQL 대상과 연결이 끊긴 LOB가 포함된 태스크가 있을 경우, 태스크 로그에 다음과 같은 유형의 오류가 표시될 수 있습니다.

```
[TARGET_LOAD ]E: RetCode: SQL_ERROR SqlState: 08S01 NativeError:
2013 Message: [MySQL][ODBC 5.3(w) Driver][mysqld-5.7.16-log]Lost connection
to MySQL server during query [122502] ODBC general error.
```

```
 [TARGET_LOAD ]E: RetCode: SQL_ERROR SqlState: HY000 NativeError:
2006 Message: [MySQL][ODBC 5.3(w) Driver]MySQL server has gone away
[122502] ODBC general error.
```

이 경우 일부 태스크 설정을 조정해야 할 수도 있습니다.

작업이 MySQL 대상과 연결이 끊기는 문제를 해결하려면, 다음 작업을 수행합니다.

- 데이터베이스 변수 `max_allowed_packet`이 가장 큰 LOB를 보유할 수 있을 정도로 충분히 크게 설정되어 있는지 확인합니다.

- 다음 변수가 큰 시간 초과 값을 갖도록 설정했는지 확인합니다. 이 변수 각각에서 최소 5분 이상의 값을 사용할 것을 제안합니다.
  - `net_read_timeout`
  - `net_write_timeout`
  - `wait_timeout`

MySQL 시스템 변수 설정에 대한 자세한 내용은 MySQL 설명서의 Server System Variables를 참조하세요.

MySQL 호환 엔드포인트에 자동 커밋 추가

대상 MySQL 호환 엔드포인트에 자동 커밋을 추가하려면

1. 에 AWS Management Console 로그인하고 https://console.aws.amazon.com/dms/v2/ 에서 AWS DMS 콘솔을 엽니다.
2. 엔드포인트를 선택합니다.
3. 자동 커밋을 추가할 MySQL과 호환되는 대상 엔드포인트를 선택합니다.
4. 수정을 선택합니다.
5. 고급을 선택하고 나서 추가 연결 속성 텍스트 상자에 다음 코드를 추가합니다.

```
Initstmt= SET AUTOCOMMIT=1
```

6. 수정을 선택합니다.

대상 MySQL 호환 엔드포인트에서 외래 키 비활성화

대상 MySQL, Amazon Aurora MySQL 호환 에디션 또는 MariaDB 엔드포인트의 고급 섹션에 있는 추가 연결 속성에 아래의 코드를 추가하여 MySQL에서 외부 키 검사를 비활성화할 수 있습니다.

대상 MySQL 호환 엔드포인트에서 외래 키를 비활성화하려면

1. https://console.aws.amazon.com/dms/v2/ 에서 AWS Management Console 로그인하고 AWS DMS 콘솔을 엽니다.
2. 엔드포인트를 선택합니다.

3. 외부 키를 비활성화할 MySQL, Aurora MySQL 또는 MariaDB 대상 엔드포인트를 선택합니다.
4. 수정을 선택합니다.
5. 고급을 선택하고 나서 추가 연결 속성 텍스트 상자에 다음 코드를 추가합니다.

```
Initstmt=SET FOREIGN_KEY_CHECKS=0
```

6. 수정을 선택합니다.

물음표로 대체된 문자

이 문제를 유발하는 가장 일반적인 상황은 원본 엔드포인트 문자가 AWS DMS에서 지원되지 않는 문자 집합으로 인코딩되는 경우입니다.

'Bad event' 로그 항목

마이그레이션 로그에서 'Bad event' 항목은 대개 소스 데이터베이스 엔드포인트에서 지원되지 않는 출력 데이터 정의 언어(DDL) 작업이 시도되었음을 나타냅니다. 지원되지 않는 DDL 작업은 복제 인스턴스가 건너뛸 수 없는 이벤트를 유발하므로 잘못된 이벤트가 로깅됩니다.

이 문제를 해결하려면 태스크를 처음부터 다시 시작합니다. 이렇게 하면 지원되지 않는 DDL 작업이 실행된 후 테이블이 다시 로드되고 변경 사항 캡처가 시작됩니다.

MySQL 5.5로 변경 데이터 캡처

Amazon RDS MySQL 호환 데이터베이스를 위한 AWS DMS 변경 데이터 캡처(CDC)에는 전체 이미지 행 기반 바이너리 로깅이 필요하며, 이는 MySQL 버전 5.5 이하에서 지원되지 않습니다. AWS DMS CDC를 사용하려면 Amazon RDS DB 인스턴스를 MySQL 버전 5.6으로 업그레이드해야 합니다.

Amazon RDS DB 인스턴스를 위한 이진 로그 보존 기간 증가

AWS DMS에서는 변경 데이터 캡처를 위한 바이너리 로그 파일 보존이 필요합니다. Amazon RDS 인스턴스에서 로그 보존 기간을 연장하려면, 다음 절차를 실행합니다. 다음은 이진 로그 보존 기간을 24시간으로 연장하는 예제입니다.

```
call mysql.rds_set_configuration('binlog retention hours', 24);
```

로그 메시지: 소스 데이터베이스의 일부 변경 사항은 대상 데이터베이스에 적용될 때 영향이 전혀 없었습니다.

AWS DMS가 MySQL 데이터베이스 열의 값을 기존 값으로 업데이트하면, `zero rows affected`의 메시지가 MySQL에서 반환됩니다. 이러한 동작은 Oracle 및 SQL Server와 같은 다른 데이터베이스 엔진과 다릅니다. 이러한 엔진은 대체 값이 현재 값과 동일한 경우에도 하나의 행을 업데이트합니다.

오류: Identifier too long

다음 오류는 식별자가 너무 길 때 발생합니다.

```
TARGET_LOAD E: RetCode: SQL_ERROR SqlState: HY000 NativeError:
1059 Message: MySQLhttp://ODBC 5.3(w) Driverhttp://mysqld-5.6.10Identifier
name 'name' is too long 122502 ODBC general error. (ar_odbc_stmt.c:4054)
```

대상 데이터베이스에 테이블과 프라이머리 키를 생성하도록 AWS DMS를 설정하는 경우도 있습니다. 이러한 경우 DMS는 현재 소스 데이터베이스에서 사용된 프라이머리 키에 동일한 이름을 사용하지 않습니다. 대신, DMS는 테이블 이름을 기준으로 프라이머리 키 이름을 생성합니다. 테이블 이름이 길면, 자동 생성된 식별자는 MySQL에 허용된 한도보다 길 수 있습니다.

이 문제를 해결하기 위한 현재의 접근 방식은 먼저 대상 데이터베이스에 테이블과 프라이머리 키를 미리 생성하는 것입니다. 그런 다음, 태스크 설정 대상 테이블 준비 모드가 아무 작업 안 함 또는 자르기로 설정된 태스크를 사용하여 대상 테이블을 채웁니다.

오류: 지원되지 않는 문자 집합으로 인해 필드 데이터 변환 실패

다음 오류는 지원되지 않는 문자 집합으로 인해 필드 데이터 변환이 실패할 때 발생합니다.

```
"[SOURCE_CAPTURE ]E: Column 'column-name' uses an unsupported character set [120112]
A field data conversion failed. (mysql_endpoint_capture.c:2154)
```

연결과 관련된 데이터베이스의 파라미터를 확인합니다. 다음 명령은 이 파라미터를 설정하는 데 사용할 수 있습니다.

```
SHOW VARIABLES LIKE '%char%';
```

오류: Codepage 1252 to UTF8 [120112] A field data conversion failed

원본 MySQL 데이터베이스에 codepage-1252 문자가 없을 경우 마이그레이션 중에 다음과 같은 오류가 발생할 수 있습니다.

```
[SOURCE_CAPTURE ]E: Error converting column 'column_xyz' in table
'table_xyz with codepage 1252 to UTF8 [120112] A field data conversion failed.
(mysql_endpoint_capture.c:2248)
```

차선책으로 `CharsetMapping` 추가 연결 속성을 원본 MySQL 엔드포인트와 함께 사용하여 문자 세트 매핑을 지정할 수 있습니다. 이 엔드포인트 설정을 추가하면 AWS DMS 마이그레이션 태스크를 처음부터 다시 시작해야 할 수 있습니다.

예를 들어, 소스 문자 집합이 `Utf8` 또는 `latin1`인 MySQL 소스 엔드포인트에 아래와 같은 엔드포인트 설정을 사용할 수 있습니다. 65001은 UTF8 코드 페이지 식별자입니다.

```
CharsetMapping=utf8,65001
CharsetMapping=latin1,65001
```

인덱스, 외부 키 또는 하위 항목 업데이트 또는 삭제가 마이그레이션되지 않음

AWS DMS는 인덱스 및 외부 키와 같은 보조 객체의 마이그레이션을 지원하지 않습니다. 하위 항목 업데이트 또는 삭제 작업에서 하위 테이블의 변경 사항을 복제하려면 트리거하는 외부 키 제약 조건이 대상 테이블에서 활성화된 상태여야 합니다. 이러한 제한을 해결하려면 대상 테이블에서 외부 키를 수동으로 생성합니다. 그런 다음, 아래에 설명된 대로 전체 로드 및 CDC를 위한 단일 태스크를 생성하거나, 전체 로드 및 CDC를 위한 별도의 태스크 두 개를 생성합니다.

전체 로드 및 CDC를 지원하는 단일 태스크 생성

이 절차에서는 전체 로드 및 CDC를 위한 단일 태스크를 사용하여 외부 키와 인덱스를 마이그레이션하는 방법을 설명합니다.

전체 로드 및 CDC 태스크 생성

1. 대상에서 외부 키와 인덱스가 있는 테이블을 수동으로 생성하여 소스 테이블과 일치하도록 합니다.
2. 대상 AWS DMS 엔드포인트에 아래의 ECA를 추가합니다.

```
Initstmt=SET FOREIGN_KEY_CHECKS=0;
```

3. TargetTablePrepMode를 DO_NOTHING으로 설정하여 AWS DMS 태스크를 생성합니다.
4. Stop task after full load completes 설정값을 StopTaskCachedChangesApplied로 설정합니다.
5. 태스크를 시작합니다. 전체 로드가 완료되면 AWS DMS가 자동으로 태스크를 중지하고 캐시된 변경 사항을 적용합니다.
6. 이전에 추가한 SET FOREIGN_KEY_CHECKS ECA를 제거합니다.
7. 태스크를 재개합니다. 태스크는 CDC 단계에 진입하고, 소스 데이터베이스의 지속적인 변경 사항을 대상에 적용합니다.

전체 로드 태스크 및 CDC 태스크를 별도로 생성

이 절차에서는 전체 로드 및 CDC를 위한 별도의 태스크를 사용하여 외부 키와 인덱스를 마이그레이션하는 방법을 설명합니다.

전체 로드 태스크 생성

1. 대상에서 외부 키와 인덱스가 있는 테이블을 수동으로 생성하여 소스 테이블과 일치하도록 합니다.
2. 대상 AWS DMS 엔드포인트에 아래의 ECA를 추가합니다.

```
Initstmt=SET FOREIGN_KEY_CHECKS=0;
```

3. TargetTablePrepMode 파라미터를 DO_NOTHING으로 설정하고 EnableValidation을 FALSE로 설정하여 AWS DMS 태스크를 생성합니다.
4. 태스크를 시작합니다. 전체 로드가 완료되면 AWS DMS가 자동으로 태스크를 중지하고 캐시된 변경 사항을 적용합니다.
5. 태스크가 완료되면 전체 로드 태스크 시작 시간(UTC) 또는 바이너리 로그 파일의 이름과 위치를 기록하여 CDC 전용 작업을 시작합니다. 로그를 참조하여 초기 전체 로드 시작 시간에서 타임스탬프(UTC)를 가져옵니다.

CDC 전용 태스크 생성

1. 이전에 설정한 `SET FOREIGN_KEY_CHECKS` ECA를 제거합니다.
2. 시작 위치를 이전 단계에서 기록한 전체 로드 시작 시간으로 설정하여 CDC 전용 태스크를 생성합니다. 또는 이전 단계에서 기록된 바이너리 로그 위치를 사용할 수도 있습니다. `TargetTablePrepMode` 설정값을 `DO_NOTHING`로 설정합니다. 필요한 경우 `EnableValidation` 설정을 `TRUE`로 설정하여 데이터 검증을 활성화합니다.
3. CDC 전용 태스크를 시작하고, 로그에 오류가 있는지 모니터링합니다.

> **Note**
>
> 이 해결 방법은 MySQL에서 MySQL로 마이그레이션하는 경우에만 적용됩니다. 일괄 적용 기능을 사용하려면 대상 테이블에 활성화된 외부 키가 없어야 하므로, 이 방법은 일괄 적용 기능과 함께 사용할 수 없습니다.

PostgreSQL 관련 문제 해결

아래에서 AWS DMS를 PostgreSQL 데이터베이스와 함께 사용했을 때 발생하는 문제에 대한 해결 방법을 배울 수 있습니다.

주제

- 잘리는 JSON 데이터 형식
- 사용자 정의 데이터 형식 열이 올바르게 마이그레이션되지 않음
- 오류: 생성하도록 선택된 스키마가 없음
- 테이블 삭제와 업데이트가 CDC를 사용하여 복제되지 않음
- 자르기 문이 전파되지 않고 있음
- PostgreSQL에서 DDL이 캡처되지 않음
- DDL 캡처를 위해 데이터베이스 객체가 생성되는 스키마 선택
- PostgreSQL로 마이그레이션한 후 Oracle 테이블 누락
- ReplicationSlotDiskUsage ETL 워크로드와 같은 긴 트랜잭션 중에는 restart_lsn이 증가하고 restart_lsn이 앞으로 이동하지 않습니다.
- 소스로 보기를 사용한 작업에서 복사된 행이 없음

잘리는 JSON 데이터 형식

AWS DMS에서는 PostgreSQL의 JSON 데이터 형식을 LOB 데이터 형식 열로 간주합니다. 다시 말해서 제한된 LOB 모드를 사용할 때 LOB 크기 제한이 JSON 데이터에 적용됨을 의미합니다.

예를 들어, 제한된 LOB 모드가 4,096KB로 설정되어 있다고 가정해 보겠습니다. 이 경우 4,096KB를 초과하는 모든 JSON 데이터는 4,096KB 한도에서 잘리며 PostgreSQL에서 검증 테스트에 실패합니다.

다음 로그 정보에는 제한된 LOB 모드 설정 및 실패한 검증으로 인해 잘린 JSON 데이터가 나와 있습니다.

```
03:00:49
2017-09-19T03:00:49 [TARGET_APPLY ]E: Failed to execute statement:
  'UPDATE "public"."delivery_options_quotes" SET "id"=? , "enabled"=? ,
  "new_cart_id"=? , "order_id"=? , "user_id"=? , "zone_id"=? , "quotes"=? ,
  "start_at"=? , "end_at"=? , "last_quoted_at"=? , "created_at"=? ,
  "updated_at"=? WHERE "id"=? ' [1022502] (ar_odbc_stmt
2017-09-19T03:00:49 [TARGET_APPLY ]E: Failed to execute statement:
  'UPDATE "public"."delivery_options_quotes" SET "id"=? , "enabled"=? ,
  "new_cart_id"=? , "order_id"=? , "user_id"=? , "zone_id"=? , "quotes"=? ,
  "start_at"=? , "end_at"=? , "last_quoted_at"=? , "created_at"=? ,
  "updated_at"=? WHERE "id"=? ' [1022502] (ar_odbc_stmt.c:2415)
#
03:00:49
2017-09-19T03:00:49 [TARGET_APPLY ]E: RetCode: SQL_ERROR SqlState:
  22P02 NativeError: 1 Message: ERROR: invalid input syntax for type json;,
  Error while executing the query [1022502] (ar_odbc_stmt.c:2421)
2017-09-19T03:00:49 [TARGET_APPLY ]E: RetCode: SQL_ERROR SqlState:
  22P02 NativeError: 1 Message: ERROR: invalid input syntax for type json;,
  Error while executing the query [1022502] (ar_odbc_stmt.c:2421)
```

사용자 정의 데이터 형식 열이 올바르게 마이그레이션되지 않음

PostgreSQL 소스에서 복제할 때 AWS DMS는 모든 열에서 동일한 데이터 형식을 사용하여 대상 테이블을 생성하고, 이때 사용자 정의 데이터 형식을 사용한 열은 제외됩니다. 그러한 경우에 데이터 형식은 대상에서 ‘character varying’으로 생성됩니다.

오류: 생성하도록 선택된 스키마가 없음

경우에 따라 “SQL_ERROR SqlState: 3F000 NativeError: 7 메시지: 오류: 생성할 스키마를 선택하지 않았습니다.” 라는 오류가 표시될 수 있습니다.

JSON 테이블 매핑에 스키마에 대한 와일드카드 값이 포함되어 있지만 소스 데이터베이스가 해당 값을 지원하지 않는 경우, 이 오류가 발생할 수 있습니다.

테이블 삭제와 업데이트가 CDC를 사용하여 복제되지 않음

소스 테이블에 프라이머리 키가 없는 경우, 변경 데이터 캡처(CDC) 중에 삭제 및 업데이트 작업이 무시됩니다. AWS DMS는 프라이머리 키가 있는 PostgreSQL 테이블의 변경 데이터 캡처(CDC)를 지원합니다.

테이블에 프라이머리 키가 없는 경우, Write Ahead Log(WAL)에 데이터베이스 행의 이전 이미지가 포함되지 않습니다. 이 경우 AWS DMS는 테이블을 업데이트할 수 없습니다. 복제할 작업을 삭제하려면 소스 테이블에서 프라이머리 키를 생성합니다.

자르기 문이 전파되지 않고 있음

변경 데이터 캡처(CDC)를 사용할 때 TRUNCATE 작업이 AWS DMS에서 지원되지 않습니다.

PostgreSQL에서 DDL이 캡처되지 않음

아래의 엔드포인트 설정 문을 추가하여 PostgreSQL 대상 엔드포인트가 DDL 문을 캡처하지 않도록 할 수 있습니다.

```
"CaptureDDLs": "N"
```

DDL 캡처를 위해 데이터베이스 객체가 생성되는 스키마 선택

DDL 캡처와 관련된 데이터베이스 객체가 생성되는 스키마를 제어할 수 있습니다. 아래의 엔드포인트 설정 문을 추가합니다. 엔드포인트 설정 파라미터는 소스 엔드포인트의 탭에서 제공됩니다.

```
"DdlArtifactsSchema: "xyzddlschema"
```

PostgreSQL로 마이그레이션한 후 Oracle 테이블 누락

이 경우 일반적으로 테이블과 데이터에는 계속 액세스할 수 있습니다.

Oracle 기본값을 대문자 테이블 이름으로 설정하고, PostgreSQL 기본값을 소문자 테이블 이름으로 설정합니다. Oracle에서 PostgreSQL로 마이그레이션할 경우, 태스크의 테이블 매핑 섹션 아래에 특정 변환 규칙을 제공하는 것이 좋습니다. 이는 테이블 이름의 대소문자를 변환하는 변환 규칙입니다.

변환 규칙을 사용하지 않고 테이블을 마이그레이션하여 테이블 이름의 대소문자를 변환할 경우, 테이블 이름을 참조할 때 테이블 이름을 따옴표로 묶어야 합니다.

ReplicationSlotDiskUsage ETL 워크로드와 같은 긴 트랜잭션 중에는 restart_lsn이 증가하고 restart_lsn이 앞으로 이동하지 않습니다.

논리적 복제가 활성화된 경우 트랜잭션당 메모리에 보관되는 최대 변경 사항 수는 4MB입니다. 그 후에는 변경 사항이 디스크로 유출됩니다. 따라서 트랜잭션이 완료/중지되고 롤백이 완료될 때까지 `ReplicationSlotDiskUsage`가 증가하며 `restart_lsn`이 진행되지 않습니다. 트랜잭션이 길기 때문에 롤백하는 데 시간이 오래 걸릴 수 있습니다.

따라서 논리적 복제가 활성화된 경우, 장기 실행 트랜잭션은 피하세요. 대신 트랜잭션을 여러 개의 작은 트랜잭션으로 나눕니다.

소스로 보기를 사용한 작업에서 복사된 행이 없음

보기를 마이그레이션하려면 `table-type`을 `all` 또는 `view`로 설정합니다. 자세한 설명은 콘솔에서 테이블 선택 및 변환 규칙 지정 섹션을 참조하세요.

보기를 지원하는 소스는 다음과 같습니다.

- Oracle
- Microsoft SQL Server
- MySQL
- PostgreSQL
- IBM Db2 LUW
- SAP Adaptive Server Enterprise(ASE)

Microsoft SQL Server 관련 문제 해결

아래에서 AWS DMS를 Microsoft SQL Server 데이터베이스와 함께 사용했을 때 발생하는 문제에 대한 해결 방법을 배울 수 있습니다.

주제

- SQL Server 데이터베이스에서 변경 사항을 캡처하는 동안 발생하는 오류
- 자격 증명 열 누락
- 오류: SQL Server doesn't support publications
- 대상에 변경 사항이 표시되지 않음
- 파티션 간에 매핑된 비균일 테이블

SQL Server 데이터베이스에서 변경 사항을 캡처하는 동안 발생하는 오류

변경 데이터 캡처(CDC) 중에 발생하는 오류는 흔히 사전 요구 사항 중 하나에 부합하지 않음을 나타낼 수 있습니다. 예를 들어, 자주 간과되는 사전 요구 사항은 전체 데이터베이스 백업입니다. 이 작업 로그에는 다음 오류와 함께 이러한 누락이 나타납니다.

```
SOURCE_CAPTURE E: No FULL database backup found (under the 'FULL' recovery model).
To enable all changes to be captured, you must perform a full database backup.
120438 Changes may be missed. (sqlserver_log_queries.c:2623)
```

Microsoft SQL Server 데이터베이스를 원본으로 사용 AWS DMS에서 원본으로서 SQL Server를 위해 나열된 사전 요구 사항을 검토합니다.

자격 증명 열 누락

AWS DMS는 대상 스키마를 생성할 때 자격 증명 열을 지원하지 않습니다. 초기 로드가 완료된 후 자격 증명 열을 추가해야 합니다.

오류: SQL Server doesn't support publications

SQL Server Express를 원본 엔드포인트로 사용할 때 다음 오류가 생성됩니다.

```
RetCode: SQL_ERROR SqlState: HY000 NativeError: 21106
Message: This edition of SQL Server does not support publications.
```

AWS DMS는 현재 SQL Server Express를 원본 또는 대상으로 지원하지 않습니다.

대상에 변경 사항이 표시되지 않음

AWS DMS에서 변경 사항을 일관성 있게 캡처하려면 원본 SQL Server 데이터베이스가 'FULL' 또는 'BULK LOGGED' 데이터 복구 모델에 있어야 합니다. 'SIMPLE' 모델은 지원되지 않습니다.

SIMPLE 복구 모델은 사용자가 데이터베이스를 복구하도록 허용하는 데 필요한 최소한의 정보를 로깅합니다. 체크포인트가 발생하면 모든 비활성 로그 항목은 자동으로 잘립니다.

모든 작업은 여전히 기록됩니다. 하지만 체크포인트가 발생하는 즉시 로그는 자동으로 잘립니다. 이렇게 잘리면 로그를 재사용할 수 있게 되고, 이전 로그 항목을 덮어쓸 수 있습니다. 로그 항목을 덮어쓰면 변경 사항을 캡처할 수 없습니다. 이 문제는 AWS DMS가 SIMPLE 데이터 복구 모델을 지원하지 않는 이유입니다. SQL Server를 소스로 사용할 때 다른 필요한 사전 요구 사항에 대한 정보는 Microsoft SQL Server 데이터베이스를 원본으로 사용 AWS DMS 섹션을 참조하세요.

파티션 간에 매핑된 비균일 테이블

변경 데이터 캡처(CDC) 중 AWS DMS가 테이블에서 CDC를 제대로 수행할 수 없는 경우 특수한 구조의 테이블 마이그레이션이 일시 중단됩니다. 다음과 같은 메시지가 표시됩니다.

```
[SOURCE_CAPTURE ]W: Table is not uniformly mapped across partitions. Therefore - it is
 excluded from CDC (sqlserver_log_metadata.c:1415)
[SOURCE_CAPTURE ]I: Table has been mapped and registered for CDC.
 (sqlserver_log_metadata.c:835)
```

SQL Server 테이블에서 CDC를 실행할 때 AWS DMS에서 SQL Server tlog를 구문 분석합니다. 각 tlog 레코드에서 AWS DMS는 변경 중에 삽입, 업데이트 또는 삭제된 열에 대한 데이터가 포함된 16진수 값을 구문 분석합니다.

16진수 레코드를 구문 분석하기 위해 AWS DMS는 SQL Server 시스템 테이블에서 테이블 메타데이터를 읽습니다. 이러한 시스템 테이블은 특수 구조화된 테이블 열이 무엇인지 식별하고 “xoffset”및 “null bit position”과 같은 내부 속성 중 일부를 나타냅니다.

AWS DMS는 해당 메타데이터가 테이블의 모든 원시 파티션에 대해 동일할 것으로 예상합니다. 하지만 특수하게 구성된 테이블의 모든 파티션의 경우에는 메타데이터가 동일하지 않을 때도 있습니다. 이러한 경우 AWS DMS는 변경 사항의 구문을 잘못 분석하여 대상에 잘못된 데이터를 제공하는 것을 방지하기 위해 해당 테이블에서 CDC를 일시 중단할 수 있습니다. 해결 방법은 다음과 같습니다.

- 테이블에 클러스터링된 인덱스가 있는 경우 인덱스를 재구축합니다.

- 테이블에 클러스터링된 인덱스가 없으면 테이블에 클러스터링된 인덱스를 추가합니다(원하는 경우 나중에 삭제할 수 있음).

Amazon Redshift 관련 문제 해결

아래에서 AWS DMS를 Amazon Redshift 데이터베이스와 함께 사용했을 때 발생하는 문제에 대한 해결 방법을 배울 수 있습니다.

주제

- 다른 AWS 리전에서 Amazon Redshift 클러스터에 로드
- 오류: Relation "awsdms_apply_exceptions" already exists
- awsdms_changes"로 시작하는 테이블 이름이 있는 오류
- dms.awsdms_changes000000000XXXX와 같은 이름과 함께 클러스터에 테이블 표시
- Amazon Redshift 사용에 필요한 권한

다른 AWS 리전에서 Amazon Redshift 클러스터에 로드

AWS DMS 복제 인스턴스가 아닌 다른 AWS 리전에서 Amazon Redshift 클러스터에 로드할 수 없습니다. DMS를 사용하려면 복제 인스턴스와 Amazon Redshift 클러스터가 동일한 리전에 있어야 합니다.

오류: Relation "awsdms_apply_exceptions" already exists

오류 "Relation ‘awsdms_apply_exceptions’ already exists"는 흔히 Redshift 엔드포인트가 PostgreSQL 엔드포인트로써 지정될 때 발생합니다. 이 문제를 해결하려면, 엔드포인트를 수정하고 [Target engine]을 "redshift"로 변경합니다.

awsdms_changes"로 시작하는 테이블 이름이 있는 오류

'awsdms_changes'로 시작하는 이름과 관련된 테이블 오류 메시지는 데이터를 동일한 Amazon Redshift 클러스터로 로드하려고 하는 태스크 2개를 동시에 실행할 때 발생합니다. 임시 테이블 이름을 지정하는 방식으로 인해 동일한 테이블을 업데이트할 때 동시 작업이 충돌할 수 있습니다.

dms.awsdms_changes000000000XXXX와 같은 이름과 함께 클러스터에 테이블 표시

Amazon S3에 저장된 파일에서 데이터가 로드되면 AWS DMS가 임시 테이블을 생성합니다. 이 임시 테이블의 이름에는 접두사 `dms.awsdms_changes`가 있습니다. 처음 로드될 때와 최종 대상 테이블에 배치되기 전에 AWS DMS가 데이터를 복원할 수 있도록 이 테이블이 필요합니다.

Amazon Redshift 사용에 필요한 권한

Amazon Redshift와 함께 AWS DMS를 사용하려면 Amazon Redshift에 액세스하기 위해 사용하는 사용자 계정에 다음 권한이 있어야 합니다.

- CRUD(선택, 삽입, 업데이트, 삭제)
- 대량 로드
- 생성, 변경, 삭제(필요한 경우 태스크의 정의에 따라)

대상으로서 Amazon Redshift를 사용할 때 필요한 모든 사전 요구 사항을 보려면, AWS Database Migration Service의 대상으로 Amazon Redshift 데이터베이스 사용 섹션을 참조하세요.

Amazon Aurora MySQL 관련 문제 해결

아래에서 AWS DMS를 Amazon Aurora MySQL 데이터베이스와 함께 사용했을 때 발생하는 문제에 대한 해결 방법을 배울 수 있습니다.

주제

- 오류: CHARACTER SET UTF8 fields terminated by ',' enclosed by "" lines terminated by '\n'

오류: CHARACTER SET UTF8 fields terminated by ',' enclosed by "" lines terminated by '\n'

Amazon Aurora MySQL을 대상으로 사용 중인 경우, 로그에 다음과 같은 오류가 표시될 수 있습니다. 이러한 유형의 오류는 일반적으로 ANSI_QUOTES가 SQL_MODE 파라미터의 일부로 포함되어 있음을 나타냅니다. SQL_MODE 파라미터의 일부로서 ANSI_QUOTES가 포함되어 있으면 큰따옴표가 일반 따옴표로 처리되어 태스크를 실행할 때 문제가 발생할 수 있습니다.

이 오류를 수정하려면, SQL_MODE 파라미터에서 ANSI_QUOTES를 제거합니다.

```
2016-11-02T14:23:48 [TARGET_LOAD ]E: Load data sql statement. load data local infile
"/rdsdbdata/data/tasks/7XO4FJHCVON7TYTLQ6RX3CQHDU/data_files/4/LOAD000001DF.csv" into
 table
`VOSPUSER`.`SANDBOX_SRC_FILE` CHARACTER SET UTF8 fields terminated by ','
enclosed by '"' lines terminated by '\n'( `SANDBOX_SRC_FILE_ID`,`SANDBOX_ID`,
`FILENAME`,`LOCAL_PATH`,`LINES_OF_CODE`,`INSERT_TS`,`MODIFIED_TS`,`MODIFIED_BY`,
`RECORD_VER`,`REF_GUID`,`PLATFORM_GENERATED`,`ANALYSIS_TYPE`,`SANITIZED`,`DYN_TYPE`,
`CRAWL_STATUS`,`ORIG_EXEC_UNIT_VER_ID` ) ; (provider_syntax_manager.c:2561)
```

SAP ASE 관련 문제 해결

아래에서 AWS DMS를 SAP ASE 데이터베이스와 함께 사용했을 때 발생하는 문제에 대한 해결 방법을 배울 수 있습니다.

오류: LOB columns have NULL values when source has a composite unique index with NULL values

NULL 값을 허용하는 복합 고유 인덱스로 구성된 테이블이 있는 소스로서 SAP ASE를 사용할 경우, 복제가 진행되는 동안 LOB 값이 마이그레이션되지 않을 수 있습니다. 일반적으로 이러한 동작은 DMS 복제 인스턴스 클라이언트에서 ANSI_NULL의 기본값이 1로 설정되었을 때 발생합니다.

LOB 필드를 올바르게 마이그레이션하려면 엔드포인트 설정 `'AnsiNull=0'`을 AWS DMS 소스 엔드포인트에 포함합니다.

IBM Db2 관련 문제 해결

아래에서 AWS DMS를 IBM Db2 데이터베이스와 함께 사용했을 때 발생하는 문제에 대한 해결 방법을 배울 수 있습니다.

오류: Resume from timestamp is not supported Task

지속적 복제(CDC)의 경우, 특정 타임스탬프에서 복제를 시작하려면 `StartFromContext` 연결 속성을 필수 타임스탬프로 설정해야 합니다. 자세한 내용은 Db2 LUW 사용 시 엔드포인트 설정을 참조하세요. `StartFromContext`를 필수 타임스탬프로 설정하면 아래와 같은 문제를 방지할 수 있습니다.

```
Last Error Resume from timestamp is not supported Task error notification received
 from
```

```
subtask 0, thread 0 [reptask/replicationtask.c:2822] [1020455] 'Start from timestamp'
 was blocked to prevent Replicate from
scanning the log (to find the timestamp). When using IBM DB2 for LUW, 'Start from
 timestamp' is only supported if an actual
change was captured by this Replicate task earlier to the specified timestamp.
```

AWS Database Migration Service의 지연 시간 문제 해결

이 섹션에서는 지속적 복제 단계(CDC) 동안 AWS DMS 태스크 지연 시간이 발생하는 일반적인 원인에 대한 개요를 제공합니다. AWS DMS는 데이터를 비동기식으로 복제합니다. 지연 시간은 변경 사항이 소스에서 커밋된 시간과 변경 사항이 대상에 복제된 시간 사이에 지연이 발생하는 것입니다. 다음과 같은 복제 구성 요소의 잘못된 구성으로 인해 지연 시간이 발생할 수 있습니다.

- 소스 엔드포인트 또는 데이터 소스
- 대상 엔드포인트 또는 데이터 소스
- 복제 인스턴스
- 이러한 구성 요소 간의 네트워크

테스트 마이그레이션을 개념 증명으로 사용하여 복제에 대한 정보를 수집하는 것이 좋습니다. 그런 다음, 이 정보를 사용해 복제 구성을 튜닝하여 지연 시간을 최소화할 수 있습니다. 개념 증명 마이그레이션 실행에 대한 자세한 내용은 개념 증명 실행 섹션을 참조하세요.

주제

- CDC 지연 시간 유형
- CDC 지연 시간이 발생하는 일반적인 원인
- 지연 시간 문제 해결

CDC 지연 시간 유형

이 섹션에는 CDC 중에 발생할 수 있는 복제 지연 시간 유형이 나와 있습니다.

소스 지연 시간

소스 엔드포인트에서 캡처한 마지막 이벤트의 커밋 시간과 복제 인스턴스의 현재 시스템 타임스탬프 간의 지연 시간(초)입니다. `CDCLatencySource` CloudWatch 지표를 사용하여 데이터 소스와 복제 인스턴스 간의 지연 시간을 모니터링할 수 있습니다. `CDCLatencySource` 지표가 높으면 소스에서 변경 사항을 캡처하는 프로세스가 지연되었음을 나타냅니다. 예를 들어, 애플리케이션이 10:00에 소스에 삽

입을 커밋하고, AWS DMS가 10:02에 변경 사항을 사용할 경우 CDCLatencySource 지표는 120초입니다.

의 CloudWatch 측정치에 대한 자세한 내용은 AWS DMS 을 참조하십시오복제 작업 지표.

대상 지연 시간

대상에 커밋되기를 대기하는 첫 이벤트의 소스에 대한 커밋 시간과 DMS 복제 인스턴스의 현재 타임스탬프 사이의 지연 시간(초)입니다. CDCLatencyTarget CloudWatch 지표를 사용하여 데이터 소스와 데이터 대상의 커밋 간 지연 시간을 모니터링할 수 있습니다. 즉, 소스에서 데이터를 읽는 데 걸리는 모든 지연도 CDCLatencyTarget에 포함됩니다. 따라서 CDCLatencyTarget은 항상 CDCLatencySource보다 크거나 같습니다.

예를 들어, 애플리케이션이 10:00에 소스에 삽입을 커밋하고, AWS DMS가 10:02에 변경 사항을 사용한 후 10:05에 이러한 변경 사항을 대상에 쓸 경우 CDCLatencyTarget 지표는 300초입니다.

CDC 지연 시간이 발생하는 일반적인 원인

이 섹션에는 CDC 중에 복제 작업 시 발생할 수 있는 지연 시간 유형이 나와 있습니다.

주제

- 엔드포인트 리소스
- 복제 인스턴스 리소스
- 네트워크 속도 및 대역폭
- DMS 구성
- 복제 시나리오

엔드포인트 리소스

복제 성능 및 지연 시간에 큰 영향을 미치는 요인은 다음과 같습니다.

- 소스 및 대상 데이터베이스 구성
- 인스턴스 크기
- 과소 프로비저닝되었거나 잘못 구성된 소스 또는 대상 데이터 스토어

AWS호스팅된 소스 및 대상의 엔드포인트 문제로 인한 지연 원인을 식별하려면 다음 CloudWatch 지표를 모니터링하십시오.

- `FreeMemory`
- `CPUUtilization`
- 처리량 및 I/O 지표(예: `WriteIOPS`, `WriteThroughput` 또는 `ReadLatency`)
- 트랜잭션 볼륨 지표(예`CDCIncomingChanges`:)

모니터링 CloudWatch 지표에 대한 자세한 내용은 을 참조하십시오AWS Database Migration Service 지표.

복제 인스턴스 리소스

복제 인스턴스 리소스는 복제에 매우 중요합니다. 이러한 리소스로 인해 소스 및 대상 지연 시간이 모두 발생할 수 있으므로, 리소스 병목 현상이 없는지 확인해야 합니다.

복제 인스턴스에 리소스 병목 현상이 있는지 알아보려면 다음 사항을 확인합니다.

- CPU, 메모리, 초당 I/O 및 스토리지와 같은 중요한 CloudWatch 지표에는 급증하거나 지속적으로 높은 값이 발생하지 않습니다.
- 복제 인스턴스의 크기가 워크로드에 알맞게 적정합니다. 복제 인스턴스의 알맞은 크기 결정에 대한 자세한 내용은 복제 인스턴스에 가장 적합한 크기 선택 섹션을 참조하세요.

네트워크 속도 및 대역폭

네트워크 대역폭은 데이터 전송에 영향을 미치는 요인입니다. 복제의 네트워크 성능을 분석하려면 다음 작업 중 하나를 수행하세요.

- 인스턴스 수준에서 `ReadThroughput` 및 `WriteThroughput` 지표를 확인합니다. 모니터링 CloudWatch 지표에 대한 자세한 내용은 을 참조하십시오. AWS Database Migration Service 지표
- AWS DMS 진단 지원 AMI를 사용합니다. 현재 리전에서 진단 지원 AMI를 사용할 수 없는 경우, 지원되는 임의의 지역에서 이를 다운로드한 후 해당 리전에 복사하여 네트워크 분석을 수행할 수 있습니다. 진단 지원 AMI에 대한 자세한 내용은 AWS DMS 진단 지원 AMI 작업 섹션을 참조하세요.

AWS DMS의 CDC는 단일 스레드 방식이므로 데이터 일관성을 보장합니다. 따라서 단일 스레드 데이터 전송 속도를 계산하여 네트워크에서 지원 가능한 데이터 볼륨을 결정할 수 있습니다. 예를 들어, 100Mbps(초당 메가비트) 네트워크를 사용하여 태스크를 소스에 연결할 경우 복제에 할당되는 이론상의 최대 대역폭은 12.5Mbps(초당 메가바이트)입니다. 이는 시간당 45기가비트입니다. 소스의 트랜잭션 로그 생성 속도가 시간당 45기가비트보다 크면 태스크에 CDC 지연 시간이 발생했다는 것을 의미합

니다. 100Mbps 네트워크의 경우 이러한 속도는 이론상 최대값입니다. 소스 및 대상의 네트워크 트래픽과 리소스 오버헤드 같은 다른 요인으로 인해 실제 사용 가능한 대역폭이 감소합니다.

DMS 구성

이 섹션에는 지연 시간을 줄이는 데 도움이 되는 권장 복제 구성이 나와 있습니다.

- 엔드포인트 설정: 소스 및 대상 엔드포인트 설정으로 인해 복제 인스턴스의 성능이 저하될 수 있습니다. 리소스를 많이 사용하는 기능을 활성화하는 엔드포인트 설정은 성능에 영향을 미칩니다. 예를 들어 Oracle 엔드포인트의 경우 Binary Reader를 사용하지 않도록 설정하고 LogMiner 사용하면 LogMiner 리소스가 많이 사용되므로 성능이 향상됩니다. 다음과 같은 엔드포인트 설정은 Oracle 엔드포인트의 성능을 향상합니다.

  ```
  useLogminerReader=N;useBfile=Y
  ```

  엔드포인트 설정에 대한 자세한 내용은 AWS DMS 엔드포인트 작업 주제의 소스 및 대상 엔드포인트 엔진에 대한 설명서를 참조하세요.
- 태스크 설정: 특정 복제 시나리오의 일부 태스크 설정으로 인해 복제 인스턴스의 성능이 저하될 수 있습니다. 예를 들어, AWS DMS는 Amazon Redshift를 제외한 모든 엔드포인트의 경우 CDC에 기본적으로 트랜잭션 적용 모드(`BatchApplyEnabled=false`)를 사용합니다. 하지만 변경 횟수가 많은 소스의 경우 `BatchApplyEnabled`를 `true`로 설정하면 성능이 향상될 수 있습니다.

  작업 설정에 관한 자세한 내용은 AWS Database Migration Service 작업에 대한 작업 설정 지정 섹션을 참조하십시오.
- CDC 전용 태스크의 시작 위치: 과거의 위치 또는 타임스탬프에서 CDC 전용 태스크를 시작하면 CDC 소스 지연 시간이 증가한 상태에서 작업이 시작됩니다. 소스의 변경 사항 볼륨에 따라 태스크 지연 시간이 감소하는 데 시간이 걸릴 수 있습니다.
- LOB 설정: 대용량 객체 데이터 유형은 AWS DMS가 대용량 바이너리 데이터를 복제하는 방식으로 인해 복제 성능에 지장을 줄 수 있습니다. 자세한 정보는 다음 주제를 참조하십시오.
  - 태스크의 소스 데이터베이스에 대한 LOB 지원 설정 AWS DMS
  - 대용량 이진 객체(LOB) 마이그레이션.

복제 시나리오

이 섹션에서는 특정 복제 시나리오를 살펴보고 이러한 시나리오가 지연 시간에 어떤 영향을 미칠 수 있는지 설명합니다.

주제

- 장기간 태스크 중지
- 캐시된 변경 사항
- 교차 리전 복제

장기간 태스크 중지

태스크를 중지하면 AWS DMS는 소스에서 읽은 마지막 트랜잭션 로그의 위치를 저장합니다. 작업을 재개하면 DMS는 동일한 트랜잭션 로그 위치에서 읽기 작업을 계속 수행하려고 합니다. 몇 시간 또는 며칠 후에 작업을 재개하면 DMS가 트랜잭션 백로그 사용을 마칠 때까지 CDC 소스 지연 시간이 증가합니다.

캐시된 변경 사항

캐시된 변경 사항은 AWS DMS가 전체 로드 복제 단계를 실행하는 동안 애플리케이션이 데이터 소스에 작성하는 변경 사항입니다. DMS는 전체 로드 단계가 완료되고 CDC 단계가 시작될 때까지 이러한 변경 사항을 적용하지 않습니다. 트랜잭션 수가 많은 소스의 경우, 캐시된 변경 사항을 적용하는 데 시간이 오래 걸리므로 CDC 단계가 시작되면 소스 지연 시간이 증가합니다. 캐시된 변경 사항 수를 최소화하려면 트랜잭션 볼륨이 적을 때 전체 로드 단계를 실행하는 것이 좋습니다.

교차 리전 복제

DMS 엔드포인트 또는 복제 인스턴스를 다른 AWS 리전에 배치하면 네트워크 지연 시간이 증가합니다. 이 경우 복제 지연 시간도 증가합니다. 최상의 성능을 얻으려면 소스 엔드포인트, 대상 엔드포인트, 복제 인스턴스를 동일한 AWS 리전에 배치하세요.

지연 시간 문제 해결

이 섹션에는 복제 지연 시간에 대한 문제 해결 단계가 나와 있습니다.

지연 시간 문제를 해결하려면 다음을 수행합니다.

- 먼저, 태스크의 지연 시간 유형과 양을 확인합니다. DMS 콘솔 또는 CLI에서 태스크의 테이블 통계 섹션을 확인합니다. 카운터가 변경되면 데이터 전송이 진행 중인 것입니다. `CDCLatencySource` 및 `CDCLatencyTarget` 지표를 함께 확인하여 CDC 중에 병목 현상이 발생했는지 확인합니다.
- 높은 `CDCLatencySource` 또는 `CDCLatencyTarget` 지표가 복제에서 병목 현상이 발생했음을 나타내는 경우 다음 사항을 확인하세요.

- CDCLatencySource가 높고 CDCLatencyTarget이 CDCLatencySource와 같은 경우, 소스 엔드포인트에 병목 현상이 발생했으며 AWS DMS가 대상에 데이터를 원활하게 쓰고 있음을 나타냅니다. 아래의 소스 지연 시간 문제 해결 섹션을 참조하세요.
- CDCLatencySource가 낮고 CDCLatencyTarget이 높은 경우, 대상 엔드포인트에 병목 현상이 발생했으며 AWS DMS가 소스에서 데이터를 원활하게 읽고 있음을 나타냅니다. 아래의 대상 지연 시간 문제 해결 섹션을 참조하세요.
- CDCLatencySource가 높고 CDCLatencyTarget가 CDCLatencySource보다 훨씬 더 높은 경우, 소스 읽기와 대상 쓰기 양쪽 모두에 병목 현상이 발생했음을 나타냅니다. 먼저 소스 지연 시간을 검사한 다음, 대상 지연 시간을 검사합니다.

DMS 태스크 지표의 모니터링에 대한 자세한 내용은 AWS DMS 태스크 모니터링 섹션을 참조하세요.

소스 지연 시간 문제 해결

아래의 주제에서는 소스 엔드포인트 유형별 복제 시나리오를 설명합니다.

주제

- Oracle 엔드포인트 문제 해결
- MySQL 엔드포인트 문제 해결
- PostgreSQL 엔드포인트 문제 해결
- SQL Server 엔드포인트 문제 해결

Oracle 엔드포인트 문제 해결

이 섹션에는 Oracle과 관련된 복제 시나리오가 나와 있습니다.

소스 읽기 일시 중지

다음과 같은 시나리오에서 AWS DMS는 Oracle 소스에서 읽기를 일시 중지합니다. 이 동작은 설계에 따른 것입니다. 태스크 로그를 사용하여 이 문제의 원인을 조사할 수 있습니다. 태스크 로그에서 다음과 비슷한 메시지를 찾아보세요. 태스크 로그에 대한 자세한 내용은 AWS DMS 태스크 로그 보기 및 관리 섹션을 참조하세요.

- SORTER 메시지: DMS가 복제 인스턴스에서 트랜잭션을 캐싱하고 있음을 나타냅니다. 자세한 내용은 태스크 로그의 SORTER 메시지 단원을 참조하십시오.
- 디버그 태스크 로그: DMS가 읽기 프로세스를 중단할 경우, 태스크는 컨텍스트 필드 또는 타임스탬프를 변경하지 않고 다음과 같은 메시지를 디버그 태스크 로그에 반복적으로 기록합니다.

- Binary reader:

```
[SOURCE_CAPTURE  ]T:  Produce CTI event:
context '00000020.f23ec6e5.00000002.000a.00.0000:190805.3477731.16'
xid [00000000001e0018] timestamp '2021-07-19 06:57:55'
thread 2  (oradcdc_oralog.c:817)
```

- Logminer:

```
[SOURCE_CAPTURE  ]T:  Produce INSERT event:
object id 1309826 context
 '000000000F2CECAA010000010005A8F500000275016C0000000000000F2CEC58'
xid [000014e06411d996] timestamp '2021-08-12 09:20:32' thread 1
 (oracdc_reader.c:2269)
```

- AWS DMS는 모든 새로운 재실행 로그 작업 또는 아카이브된 로그 작업에 대해 다음과 같은 메시지를 기록합니다.

```
00007298: 2021-08-13T22:00:34 [SOURCE_CAPTURE ]I: Start processing archived
 Redo log sequence 14850 thread 2 name XXXXX/XXXXX/ARCHIVELOG/2021_08_14/
thread_2_seq_14850.22977.1080547209 (oradcdc_redo.c:754)
```

  소스에 새로운 재실행 로그 작업 또는 아카이브된 로그 작업이 있고 AWS DMS가 이러한 메시지를 로그에 쓰지 않을 경우, 이는 태스크가 이벤트를 처리하고 있지 않음을 의미합니다.

높은 재실행 생성률

태스크가 재실행 로그 또는 아카이브된 로그를 처리하는 중인데도 소스 지연 시간이 여전히 높은 경우, 재실행 로그 생성률 및 생성 패턴을 파악해 보세요. 재실행 로그 생성 수준이 높을 경우, 복제된 테이블과 관련된 변경 사항을 가져오기 위해 태스크가 모든 재실행 로그 및 아카이브 로그를 읽으므로 소스 지연 시간이 증가합니다.

재실행 생성률을 확인하려면 아래의 쿼리를 사용합니다.

- 일별 재실행 생성률:

```
select trunc(COMPLETION_TIME,'DD') Day, thread#,
round(sum(BLOCKS*BLOCK_SIZE)/1024/1024/1024) GB,
count(*) Archives_Generated from v$archived_log
where completion_time > sysdate- 1
```

```
group by trunc(COMPLETION_TIME,'DD'),thread# order by 1;
```

- 시간당 재실행 생성률:

```
Alter session set nls_date_format = 'DD-MON-YYYY HH24:MI:SS';
select trunc(COMPLETION_TIME,'HH') Hour,thread# ,
round(sum(BLOCKS*BLOCK_SIZE)/1024/1024) "REDO PER HOUR (MB)",
count(*) Archives from v$archived_log
where completion_time > sysdate- 1
group by trunc(COMPLETION_TIME,'HH'),thread#  order by 1 ;
```

이 시나리오의 지연 시간 문제를 해결하려면 다음 사항을 확인하세요.

- 복제의 네트워크 대역폭과 단일 스레드 성능을 확인하여 기본 네트워크가 소스 재실행 생성률을 지원할 수 있는지 확인합니다. 네트워크 대역폭이 복제 성능에 어떤 영향을 미치는지에 대한 자세한 내용은 이전의 네트워크 속도 및 대역폭 섹션을 참조하세요.
- 보충 로깅을 올바르게 설정했는지 확인합니다. 소스에 대한 추가 로깅(예: 테이블의 모든 열에 대한 로깅 활성화)을 사용하지 않아야 합니다. 보충 로깅 설정에 대한 자세한 내용은 보충 로깅 설정 섹션을 참조하세요.
- 올바른 API를 사용하여 재실행 로그 또는 아카이빙된 로그를 읽고 있는지 확인합니다. 오라클 LogMiner 또는 AWS DMS 바이너리 리더를 사용할 수 있습니다. 온라인 리두 로그와 아카이브된 리두 로그 파일을 LogMiner 읽는 동안 Binary Reader는 원시 리두 로그 파일을 직접 읽고 분석합니다. 따라서 Binary Reader의 성능이 더 높습니다. 재실행 로그 생성률이 시간당 10GB를 초과할 경우 Binary Reader를 사용하는 것이 좋습니다. 자세한 설명은 CDC용 오라클 LogMiner 또는 AWS DMS 바이너리 리더 사용 섹션을 참조하세요.
- `ArchivedLogsOnly`를 Y로 설정했는지 확인합니다. 이 엔드포인트 설정이 설정되어 있으면 AWS DMS는 아카이브된 재실행 로그에서 읽기를 수행합니다. 이렇게 하면 읽기를 시작하기 전에 AWS DMS는 온라인 재실행 로그가 아카이브될 때까지 대기하므로, 소스 지연 시간이 증가합니다. 자세한 내용은 을 참조하십시오. ArchivedLogsOnly
- Oracle 소스가 Automatic Storage Management(ASM)를 사용하는 경우, 데이터 스토어를 올바르게 구성하는 방법에 대한 자세한 내용은 Oracle을 소스로 사용하는 경우 Oracle ASM에 REDO를 저장합니다. AWS DMS 섹션을 참조하세요. 추가 연결 속성(ECA) `asmUsePLSQLArray`를 사용하여 읽기 성능을 더욱 최적화할 수도 있습니다. `asmUsePLSQLArray` 사용에 대한 자세한 내용은 Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS을 참조하십시오.

MySQL 엔드포인트 문제 해결

이 섹션에는 MySQL과 관련된 복제 시나리오가 나와 있습니다. AWS DMS는 MySQL 바이너리 로그를 주기적으로 스캔하여 변경 사항을 복제합니다. 다음과 같은 시나리오에서는 이러한 프로세스 때문에 지연 시간이 증가할 수 있습니다.

주제

- 소스에서 장시간 실행 중인 트랜잭션
- 소스의 높은 워크로드
- 바이너리 로그 경합

소스에서 장시간 실행 중인 트랜잭션

MySQL은 커밋된 트랜잭션만 바이너리 로그에 작성하므로, 트랜잭션이 장시간 실행되면 쿼리 실행 시간에 비례하여 지연 시간이 급증합니다.

장시간 실행 중인 트랜잭션을 파악하려면 아래의 쿼리를 사용하거나 느린 쿼리 로그를 사용합니다.

```
SHOW FULL PROCESSLIST;
```

느린 쿼리 로그 사용에 대한 자세한 내용은 MySQL 설명서의 The Slow Query Log를 참조하세요.

장시간 실행 중인 트랜잭션으로 인한 지연 시간 급증을 방지하려면 소스 트랜잭션을 재구성하여 쿼리 실행 시간을 줄이거나 커밋 빈도를 늘립니다.

소스의 높은 워크로드

DMS CDC는 단일 스레드 방식이므로, 트랜잭션 수가 많으면 소스 지연 시간이 증가할 수 있습니다. 소스 지연 시간이 과중한 워크로드로 인한 것인지 확인하려면, 지연 시간 동안 생성된 바이너리 로그의 수와 크기를 지연 시간 이전에 생성된 로그와 비교합니다. 바이너리 로그와 DMS CDC 스레드 상태를 확인하려면 아래의 쿼리를 사용합니다.

```
SHOW BINARY LOGS;
SHOW PROCESSLIST;
```

CDC 바이너리 로그 덤프 스레드 상태에 대한 자세한 내용은 Replication Source Thread States를 참조하세요.

소스에서 생성된 최신 바이너리 로그 위치를 DMS가 현재 처리 중인 이벤트와 비교하여 지연 시간을 확인할 수 있습니다. 소스 최신 바이너리 로그를 확인하려면 다음을 수행합니다.

- SOURCE_CAPTURE 구성 요소에 대한 디버그 로그를 활성화합니다.
- 태스크 디버그 로그에서 DMS 처리 바이너리 로그와 위치 세부 정보를 검색합니다.
- 소스에 대한 최신 바이너리 로그를 확인하려면 아래의 쿼리를 사용합니다.

```
SHOW MASTER STATUS;
```

성능을 더 최적화하려면 `EventsPollInterval`을 튜닝합니다. 기본적으로 DMS는 5초마다 바이너리 로그를 폴링하지만, 이 값을 줄여 성능을 개선할 수 있습니다. `EventsPollInterval` 설정에 관한 자세한 내용은 MySQL을 AWS DMS 소스로 사용할 경우 엔드포인트 설정를 참조하십시오.

바이너리 로그 경합

데이터의 양이 많은 여러 테이블을 마이그레이션할 경우, MySQL 5.7.2 이상에서는 테이블을 별도의 태스크로 분할하는 것이 좋습니다. MySQL 버전 5.7.2 이상에서는 마스터 덤프 스레드가 잠금 경합을 줄이고 처리량을 개선합니다. 따라서 덤프 스레드는 이벤트를 읽을 때마다 더 이상 바이너리 로그를 잠그지 않습니다. 이렇게 되면 여러 덤프 스레드가 바이너리 로그 파일을 동시에 읽을 수 있습니다. 이는 클라이언트가 바이너리 로그에 쓰는 동안 덤프 스레드가 바이너리 로그를 읽을 수 있다는 의미이기도 합니다. 덤프 스레드에 대한 자세한 내용은 Replication Threads 및 MySQL 5.7.2 릴리스 정보를 참조하세요.

5.7.2 이전 버전의 MySQL 소스의 복제 성능을 개선하려면 CDC 구성 요소로 작업을 통합해 보시기 바랍니다.

PostgreSQL 엔드포인트 문제 해결

이 섹션에는 PostgreSQL과 관련된 복제 시나리오가 나와 있습니다.

주제

- 소스에서 장시간 실행 중인 트랜잭션
- 소스의 높은 워크로드
- 높은 네트워크 처리량
- Aurora PostgreSQL에서 파일을 유출하세요

소스에서 장시간 실행 중인 트랜잭션

소스 데이터베이스에 장시간 실행 중인 트랜잭션이 있는 경우(예: 단일 트랜잭션에 수천 개의 삽입이 있는 경우), 트랜잭션이 완료될 때까지 DMS CDC 이벤트 및 트랜잭션 카운터가 증가하지 않습니다. 이러한 지연으로 인해 `CDCLatencyTarget` 지표로 측정 가능한 지연 시간 문제가 발생할 수 있습니다.

장시간 실행 중인 트랜잭션을 검토하려면 다음 중 하나를 수행합니다.

- `pg_replication_slots` 보기를 사용합니다. `restart_lsn` 값이 업데이트되지 않을 경우, 장시간 실행 중인 활성 트랜잭션으로 인해 PostgreSQL이 Write Ahead Log(WAL)를 릴리스하지 못할 수 있습니다. `pg_replication_slots` 보기에 대한 자세한 내용은 PostgreSQL 15.4 설명서의 pg_replication_slots 섹션을 참조하세요.
- 아래의 쿼리를 사용하여 데이터베이스의 모든 활성 쿼리 목록을 관련 정보와 함께 반환합니다.

  ```
  SELECT pid, age(clock_timestamp(), query_start), usename, query
  FROM pg_stat_activity WHERE query != '<IDLE>'
  AND query NOT ILIKE '%pg_stat_activity%'
  ORDER BY query_start desc;
  ```

  쿼리 결과의 `age` 필드에는 각 쿼리의 활성 기간이 표시되므로, 장시간 실행 중인 쿼리를 확인하는 데 사용할 수 있습니다.

소스의 높은 워크로드

소스에 PostgreSQL의 워크로드가 높은 경우, 다음 사항을 확인하여 지연 시간을 줄입니다.

- 초당 트랜잭션(TPS) 값이 높은 소스 데이터베이스에서 테이블의 하위 집합을 마이그레이션하는 동안 `test_decoding` 플러그인을 사용하면 지연 시간이 길어질 수 있습니다. 그 이유는 `test_decoding` 플러그인이 모든 데이터베이스 변경 사항을 복제 인스턴스로 보낸 다음, DMS가 태스크의 테이블 매핑을 기반으로 필터링하기 때문입니다. 태스크의 테이블 매핑에 포함되지 않은 테이블의 이벤트는 소스 지연 시간을 증가시킬 수 있습니다.
- 다음 방법 중 하나를 사용하여 TPS 처리량을 확인합니다.
  - Aurora PostgreSQL 소스의 경우 메트릭을 사용하십시오. `CommitThroughput` CloudWatch
  - Amazon RDS 또는 온프레미스에서 실행되는 PostgreSQL의 경우, PSQL 클라이언트 버전 11 이상을 활용하여 아래의 쿼리를 사용합니다(결과를 진행하려면 쿼리 중에 **`enter`**를 누릅니다).

    ```
    SELECT SUM(xact_commit)::numeric as temp_num_tx_ini FROM pg_stat_database; \gset
    select pg_sleep(60);
    ```

```
SELECT SUM(xact_commit)::numeric as temp_num_tx_final FROM pg_stat_database; \gset
select (:temp_num_tx_final - :temp_num_tx_ini)/ 60.0 as "Transactions Per Second";
```

- test_decoding 플러그인 사용 시 지연 시간을 줄이려면 pglogical 플러그인을 대신 사용하는 것이 좋습니다. test_decoding 플러그인과 달리, pglogical 플러그인은 원본에서 Write Ahead Log(WAL) 변경 사항을 필터링하고 관련 변경 사항만 복제 인스턴스로 전송합니다. pglogical 플러그인을 AWS DMS와 함께 사용하는 방법에 대한 내용은 pglogical 플러그인 구성 섹션을 참조하세요.

높은 네트워크 처리량

test_decoding 플러그인을 사용할 경우, 특히 대용량 트랜잭션 시 복제를 수행할 때 네트워크 대역폭 사용량이 높을 수 있습니다. 그 이유는 test_decoding 플러그인은 변경 사항을 처리한 후 이를 사람이 읽을 수 있는 형식으로 변환하는데, 원래 바이너리 형식보다 용량이 크기 때문입니다.

성능을 개선하려면 바이너리 플러그인인 pglogical 플러그인을 대신 사용하는 것이 좋습니다. test_decoding 플러그인과 달리 pglogical 플러그인은 바이너리 형식의 출력을 생성하므로, 압축된 Write Ahead Log(WAL) 스트림 변경 사항이 발생합니다.

Aurora PostgreSQL에서 파일을 유출하세요

PostgreSQL 버전 13 이상에서는 매개 변수가 디코딩 및 logical_decoding_work_mem 스트리밍을 위한 메모리 할당을 결정합니다. logical_decoding_work_mem파라미터에 대한 자세한 내용은 PostgreSQL 13.13 설명서의 PostgreSQL에서의 리소스 사용량을 참조하십시오.

논리적 복제는 해당 트랜잭션이 커밋될 때까지 모든 트랜잭션의 변경 내용을 메모리에 누적합니다. 모든 트랜잭션에 저장된 데이터 양이 데이터베이스 매개 변수에 logical_decoding_work_mem 지정된 양을 초과하면 DMS는 트랜잭션 데이터를 디스크로 넘겨 새 디코딩 데이터에 사용할 메모리를 확보합니다.

트랜잭션이 오래 실행되거나 하위 트랜잭션이 많으면 DMS에서 논리적 디코딩 메모리를 많이 소모할 수 있습니다. 이렇게 메모리 사용량이 증가하면 DMS가 디스크에 유출 파일을 생성하여 복제 중에 소스 지연 시간이 길어집니다.

소스 워크로드 증가로 인한 영향을 줄이려면 다음과 같이 하십시오.

- 장기 실행 트랜잭션을 줄이십시오.
- 하위 트랜잭션 수를 줄이세요.
- 단일 트랜잭션에서 전체 테이블을 삭제하거나 업데이트하는 등 대량의 로그 레코드를 생성하는 작업은 수행하지 마십시오. 대신 작은 일괄 처리로 작업을 수행하세요.

다음 CloudWatch 지표를 사용하여 소스의 워크로드를 모니터링할 수 있습니다.

- `TransactionLogsDiskUsage`: 논리적 WAL이 현재 점유하고 있는 바이트 수입니다. 논리적 복제 슬롯이 새 쓰기 속도를 따라갈 수 없거나 장기간 실행되는 트랜잭션으로 인해 오래된 파일이 불필요한 수집이 불가능한 경우 이 값은 일정하게 증가합니다.
- `ReplicationSlotDiskUsage`: 논리적 복제 슬롯이 현재 사용하는 디스크 공간의 양입니다.

`logical_decoding_work_mem`파라미터를 조정하여 소스 지연 시간을 줄일 수 있습니다. 이 매개변수의 기본값은 64MB입니다. 이 매개 변수는 각 논리적 스트리밍 복제 연결에서 사용되는 메모리 양을 제한합니다. DMS가 `logical_decoding_work_mem` 디스크에 기록하는 디코딩된 변경 사항의 양을 줄이려면 `work_mem` 값을 이 값보다 훨씬 높게 설정하는 것이 좋습니다.

특히 마이그레이션 작업이나 지연 시간이 많은 기간에는 유출 파일이 있는지 정기적으로 확인하는 것이 좋습니다. DMS에서 생성되는 유출 파일의 수가 많으면 논리적 디코딩이 효율적으로 작동하지 않아 지연 시간이 늘어날 수 있다는 뜻입니다. 이 문제를 해결하려면 파라미터 값을 높이십시오. `logical_decoding_work_mem`

함수를 사용하여 현재 트랜잭션 오버플로를 `aurora_stat_file` 확인할 수 있습니다. 자세한 내용은 Amazon 관계형 데이터베이스 서비스 개발자 안내서의 논리적 디코딩을 위한 작업 메모리 조정을 참조하십시오.

SQL Server 엔드포인트 문제 해결

이 섹션에는 SQL Server와 관련된 복제 시나리오가 나와 있습니다. SQL Server에서 복제할 변경 사항을 확인하기 위해, AWS DMS는 트랜잭션 로그를 읽고 소스 데이터베이스에서 주기적인 스캔을 실행합니다. 일반적으로 복제 지연 시간이 발생하는 이유는 리소스 제약으로 인해 SQL Server에서 이러한 스캔을 제한해서 그렇습니다. 짧은 시간 내에 트랜잭션 로그에 작성되는 이벤트 수가 크게 증가하기 때문일 수도 있습니다.

주제

- 인덱스 재구축
- 큰 트랜잭션
- Amazon RDS SQL 서버의 MS-CDC 폴링 간격이 잘못 구성된 경우
- 동일한 소스 데이터베이스에서 복제되는 여러 CDC 태스크

인덱스 재구축

SQL Server는 큰 인덱스를 재구축할 때 단일 트랜잭션을 사용합니다. 이로 인해 많은 이벤트가 생성되며, SQL Server가 한 번에 여러 인덱스를 재구축할 경우 많은 양의 로그 공간을 모두 사용하게 될 수 있습니다. 이렇게 되면 일시적인 복제 급증이 발생할 수 있습니다. SQL Server 소스에서 지속적으로 로그 급증이 발생할 경우 다음 사항을 확인하세요.

- 먼저, `CDCLatencySource` 및 `CDCLatencySource` CloudWatch 측정치를 사용하거나 작업 로그의 처리량 모니터링 메시지를 확인하여 지연 시간이 급증하는 기간을 확인하십시오. 의 CloudWatch 지표에 대한 자세한 내용은 AWS DMS 을 참조하십시오복제 작업 지표.
- 지연 시간이 급증하는 동안 활성 트랜잭션 로그 또는 로그 백업의 크기가 증가했는지 확인합니다. 해당 기간에 유지 관리 작업이나 재구축이 실행되었는지도 확인합니다. 트랜잭션 로그 크기 확인에 대한 자세한 내용은 SQL Server 기술 설명서의 Monitor log space use 섹션을 참조하세요.
- 유지 관리 계획이 SQL Server 모범 사례를 준수하는지 확인합니다. SQL Server 유지 관리 모범 사례에 대한 자세한 내용은 SQL Server 기술 설명서의 Index maintenance strategy 섹션을 참조하세요.

인덱스 재구축 중 지연 시간 문제를 해결하려면 다음을 수행합니다.

- 오프라인 재구축에 `BULK_LOGGED` 복구 모델을 사용하면 태스크가 처리해야 하는 이벤트를 줄일 수 있습니다.
- 가능한 경우, 인덱스 재구축 중에는 태스크를 중지합니다. 또는 사용량이 많지 않은 시간에 인덱스 재구축을 예약하여 지연 시간 급증으로 인한 영향을 완화하시기 바랍니다.
- 디스크 지연 시간 또는 I/O 처리량처럼 DMS 읽기 속도를 저하시키는 리소스 병목 현상을 확인하고 이를 해결합니다.

큰 트랜잭션

이벤트가 많은 트랜잭션 또는 장시간 실행 중인 트랜잭션으로 인해 트랜잭션 로그가 커집니다. 이 경우 DMS 읽기 시간이 길어져 지연 시간이 발생합니다. 이는 인덱스 재구축이 복제 성능에 미치는 영향과 유사합니다.

소스 데이터베이스의 일반적인 워크로드에 익숙하지 않다면 이 문제를 확인하기 어려울 수 있습니다. 이 문제를 해결하려면 다음을 수행합니다.

- 먼저, `ReadThroughput` 및 `WriteThroughput` CloudWatch 지표를 사용하거나 작업 로그의 처리량 모니터링 메시지를 확인하여 지연 시간이 급증한 시간을 식별하십시오.

- 지연 시간이 급증했을 때 소스 데이터베이스에 장시간 실행 중인 쿼리가 있는지 확인합니다. 장시간 실행 중인 쿼리에 대한 자세한 내용은 SQL Server 기술 설명서의 Troubleshoot slow-running queries in SQL Server 섹션을 참조하세요.
- 활성 트랜잭션 로그 또는 로그 백업의 크기가 증가했는지 확인합니다. 자세한 내용은 SQL Server 기술 설명서의 Monitor log space use 섹션을 참조하세요.

이 문제를 해결하려면 다음 중 하나를 수행합니다.

- 가장 좋은 해결 방법은 애플리케이션 측에서 트랜잭션을 재구성하여 트랜잭션이 빠르게 완료되도록 하는 것입니다.
- 트랜잭션을 재구성할 수 없는 경우, 단기적인 해결 방법은 디스크 대기 또는 CPU 경합과 같은 리소스 병목 현상이 있는지 확인하는 것입니다. 소스 데이터베이스에서 병목 현상이 발견되면 소스 데이터베이스의 디스크, CPU, 메모리 리소스를 늘려 지연 시간을 줄일 수 있습니다. 이렇게 하면 시스템 리소스 경합이 감소하여 DMS 쿼리를 더 빨리 완료할 수 있습니다.

Amazon RDS SQL 서버의 MS-CDC 폴링 간격이 잘못 구성된 경우

Amazon RDS 인스턴스에서 폴링 간격 설정이 잘못 구성되면 트랜잭션 로그가 증가할 수 있습니다. 복제로 인해 로그 잘림이 방지되기 때문입니다. 실행 중인 태스크를 지연 시간을 최소화하여 계속 복제할 수 있긴 하지만, 태스크를 중지 및 재개하거나 CDC 전용 작업을 시작할 경우 태스크가 실패할 수 있습니다. 대용량 트랜잭션 로그를 스캔하는 동안 시간 초과가 발생하기 때문입니다.

잘못 구성된 폴링 간격 문제를 해결하려면 다음을 수행합니다.

- 활성 트랜잭션 로그 크기가 증가하고 있는지, 그리고 로그 사용량이 100%에 가까운지 확인합니다. 자세한 내용은 SQL Server 기술 설명서의 Monitor log space use 섹션을 참조하세요.
- `log_reuse_wait_desc value`의 원인이 `REPLICATION`이기 때문에 로그 잘림이 지연되고 있는 것인지 확인합니다. 자세한 내용은 SQL Server 기술 설명서의 The Transaction Log (SQL Server) 섹션을 참조하세요.

이전 목록의 주제에서 문제가 발견될 경우, MS-CDC 폴링 간격을 튜닝합니다. 폴링 간격 튜닝에 대한 자세한 내용은 SQL Server용 Amazon RDS를 원본으로 사용할 때의 권장 설정 AWS DMS 섹션을 참조하세요.

동일한 소스 데이터베이스에서 복제되는 여러 CDC 태스크

전체 로드 단계에서는 태스크 전체에서 테이블을 분할하여 성능을 개선하고, 종속 테이블을 논리적으로 분리하고, 태스크 실패로 인한 영향을 완화하는 것이 좋습니다. 하지만 CDC 단계에서는 태스크를 통합하여 DMS 스캔을 최소화하는 것이 바람직합니다. CDC 단계에서 각 DMS 태스크는 1분에 여러 번 트랜잭션 로그를 스캔하여 새 이벤트가 있는지 확인합니다. 각 태스크는 독립적으로 실행되므로 모든 태스크는 각각의 트랜잭션 로그를 개별적으로 스캔합니다. 이 경우 소스 SQL Server 데이터베이스의 디스크 및 CPU 사용량이 증가합니다. 따라서 많은 태스크를 동시에 실행하면 SQL Server가 DMS 읽기 속도를 제한할 수 있으므로, 지연 시간이 증가하게 됩니다.

여러 태스크가 점진적으로 시작될 경우, 이 문제를 확인하기 어려울 수 있습니다. 이 문제의 가장 일반적인 증상은 대부분의 태스크 스캔에 소요되는 시간이 더 길어지기 시작한다는 것입니다. 이 경우 이러한 스캔의 지연 시간이 길어집니다. SQL Server는 몇 가지 태스크 스캔의 우선 순위를 지정하므로, 일부 태스크에서는 정상적인 지연 시간이 표시됩니다. 이 문제를 해결하려면 모든 태스크에 대한 `CDCLatencySource` 지표를 확인합니다. 일부 태스크는 `CDCLatencySource`가 증가하는 반면, 일부 태스크는 `CDCLatencySource`가 낮을 경우 SQL Server가 일부 태스크의 DMS 읽기 속도를 제한하고 있을 가능성이 높습니다.

CDC 중에 SQL Server가 태스크 읽기 속도를 제한할 경우, 태스크를 통합하여 DMS 스캔 수를 최소화합니다. 경합을 생성하지 않고 소스 데이터베이스에 연결할 수 있는 최대 태스크 수는 소스 데이터베이스 용량, 트랜잭션 로그 증가율, 테이블 수 등과 같은 여러 요인에 따라 달라집니다. 복제 시나리오에 적합한 태스크 수를 확인하려면 프로덕션 환경과 비슷한 테스트 환경에서 복제를 테스트하세요.

대상 지연 시간 문제 해결

이 섹션에는 대상 지연 시간에 영향을 미칠 수 있는 시나리오가 나와 있습니다.

주제

- 인덱싱 문제
- 태스크 로그의 SORTER 메시지
- 데이터베이스 잠금
- 느린 LOB 조회
- 다중 AZ, 감사 로깅, 백업

인덱싱 문제

CDC 단계에서 타겟에서 AWS DMS는 DML 문(삽입, 업데이트, 삭제)을 실행하여 소스의 변경 사항을 복제합니다. DMS를 사용하는 이기종 마이그레이션의 경우, 소스와 대상의 인덱스 최적화 차이로 인해

대상에 쓰기 작업을 수행하는 데 더 오랜 시간이 걸릴 수 있습니다. 이 경우 대상 지연 시간 및 성능 문제가 발생합니다.

인덱싱 문제를 해결하려면 다음을 수행합니다. 이러한 단계의 절차는 데이터베이스 엔진마다 다릅니다.

- 대상 데이터베이스의 쿼리 시간을 모니터링합니다. 대상과 소스의 쿼리 실행 시간을 비교하면 최적화가 필요한 인덱스를 나타낼 수 있습니다.
- 느리게 실행 중인 쿼리에 대한 로깅을 활성화합니다.

장시간 실행 중인 복제의 인덱싱 문제를 해결하려면 다음을 수행합니다.

- 소스 및 대상 데이터베이스의 인덱스를 튜닝하여 소스와 대상의 쿼리 실행 시간이 비슷해지도록 조정합니다.
- 소스와 대상의 DML 쿼리에 사용된 보조 인덱스를 비교합니다. 대상의 DML 성능이 소스 DML 성능과 비슷하거나 더 나은지 확인합니다.

참고로, 인덱스를 최적화하는 절차는 데이터베이스 엔진에 따라 다릅니다. 소스 및 대상 인덱스를 튜닝할 수 있는 DMS 기능은 없습니다.

태스크 로그의 SORTER 메시지

대상 엔드포인트가 AWS DMS가 대상 엔드포인트에 쓰는 변경 사항의 볼륨을 따라가지 못할 경우, 태스크는 복제 인스턴스의 변경 사항을 캐시합니다. 캐시가 내부 임계값보다 커지면 태스크는 소스에서 추가 변경 사항을 읽는 것을 중단합니다. DMS는 복제 인스턴스의 스토리지가 부족해지거나, 대기 중인 대량의 이벤트를 읽는 동안 태스크가 중단되는 것을 방지하기 위해 이러한 조치를 수행합니다.

이 문제를 해결하려면 CloudWatch 로그에서 다음 중 하나와 유사한 메시지가 있는지 확인하십시오.

```
[SORTER ]I: Reading from source is paused. Total disk usage exceeded the limit 90%
 (sorter_transaction.c:110)
[SORTER ]I: Reading from source is paused. Total storage used by swap files exceeded
 the limit 1048576000 bytes  (sorter_transaction.c:110)
```

로그에 첫 번째 메시지와 유사한 메시지가 있는 경우, 태스크에 대한 모든 추적 로깅을 비활성화하고 복제 인스턴스 스토리지를 늘립니다. 복제 인스턴스 스토리지 증가에 대한 자세한 내용은 복제 인스턴스 수정 섹션을 참조하세요.

두 번째 메시지와 유사한 메시지가 로그에 있는 경우 다음을 수행합니다.

- 트랜잭션이 많거나 장시간 실행 중인 DML 작업이 있는 테이블이 태스크의 다른 테이블에 종속되지 않을 경우, 해당 테이블을 별도의 태스크로 이동합니다.
- `MemoryLimitTotal` 및 `MemoryKeepTime` 설정을 늘려 트랜잭션을 메모리에 보관하는 기간을 늘립니다. 이렇게 하면 지연 시간이 지속될 경우에는 도움이 되지 않지만, 트랜잭션 볼륨이 단기간에 급증하는 동안에는 지연 시간을 줄이는 데 도움이 될 수 있습니다. 이러한 태스크 설정에 대한 내용은 변경 처리 튜닝 설정 섹션을 참조하세요.
- `BatchApplyEnabled`를 `true`로 설정하여 트랜잭션에 일괄 적용을 사용할 수 있는지 평가합니다. `BatchApplyEnabled` 설정에 대한 내용은 대상 메타데이터 작업 설정 섹션을 참조하세요.

데이터베이스 잠금

AWS DMS가 복제 대상으로 사용 중인 데이터베이스에 애플리케이션이 액세스할 경우, 해당 애플리케이션 때문에 DMS가 액세스하려는 테이블을 잠길 수 있습니다. 이 경우 잠금 경합이 발생합니다. DMS는 소스에서 발생한 순서대로 대상 데이터베이스에 변경 사항을 작성하므로, 잠금 경합으로 인해 한 테이블에 대한 쓰기 지연이 발생할 경우 모든 테이블에 대한 쓰기 작업이 지연됩니다.

이 문제를 해결하려면 대상 데이터베이스를 쿼리하여 잠금 경합 때문에 DMS 쓰기 트랜잭션이 차단되고 있는 것인지 확인합니다. 대상 데이터베이스가 DMS 쓰기 트랜잭션을 차단하고 있는 경우, 다음 중 하나 이상의 작업을 수행합니다.

- 변경 사항을 더 자주 커밋하도록 쿼리를 재구성합니다.
- 잠금 제한 시간 설정을 수정합니다.
- 테이블을 분할하여 잠금 경합을 최소화합니다.

참고로, 잠금 경합을 최적화하는 절차는 데이터베이스 엔진에 따라 다릅니다. 잠금 경합을 튜닝할 수 있는 DMS 기능은 없습니다.

느린 LOB 조회

AWS DMS는 대용량 객체(LOB) 열을 복제할 때 대상에 변경 사항을 기록하기 직전에 소스에 대한 조회를 수행합니다. 보통 이러한 조회로 인해 대상에 지연 시간이 발생하지는 않지만, 잠금으로 인해 소스 데이터베이스에서 조회가 지연되면 대상 지연 시간이 급증할 수 있습니다.

이 문제는 일반적으로 진단하기 어렵습니다. 이 문제를 해결하려면 태스크 로그에서 세부 디버깅을 활성화하고, DMS LOB 조회 호출의 타임스탬프를 비교합니다. 세부 디버깅 활성화에 대한 자세한 내용은 AWS DMS 태스크 로그 보기 및 관리 섹션을 참조하세요.

이 문제를 해결하려면 다음 작업을 수행합니다.

- 소스 데이터베이스에 대한 SELECT 쿼리 성능을 개선합니다.
- DMS LOB 설정을 튜닝합니다. LOB 설정에 대한 자세한 내용은 대용량 이진 객체(LOB) 마이그레이션 섹션을 참조하세요.

다중 AZ, 감사 로깅, 백업

Amazon RDS 대상의 경우, 다음 작업을 수행하는 동안 대상 지연 시간이 증가할 수 있습니다.

- 백업
- 다중 가용 영역(다중 AZ)을 활성화한 후
- 감사 또는 느린 쿼리 로그 등의 데이터베이스 로깅을 활성화한 후

이러한 문제는 일반적으로 진단하기 어렵습니다. 이러한 문제를 해결하려면 Amazon RDS 유지 관리 기간 또는 데이터베이스 부하가 과중한 기간에 주기적인 급증이 발생하는 지연 시간을 모니터링합니다.

이러한 문제를 해결하려면 다음 작업을 수행합니다.

- 가능하다면 단기 마이그레이션 중에는 다중 AZ, 백업 또는 로깅을 비활성화합니다.
- 유지 관리 기간을 활동이 적은 기간으로 다시 예약합니다.

AWS DMS에서 진단 지원 스크립트 작업

AWS DMS로 작업할 때 문제가 발생할 경우, 지원 엔지니어가 소스 또는 대상 데이터베이스에 대한 추가 정보를 필요로 할 수 있습니다. AWS Support는 가능한 한 가장 빠른 시간 내에 필요한 정보를 최대한 많이 확보해야 합니다. 따라서 여러 주요 관계형 데이터베이스 엔진에서 이러한 정보를 쿼리하는 스크립트를 개발했습니다.

지원 스크립트를 데이터베이스에 사용할 수 있는 경우, 아래에 설명된 해당 스크립트 주제의 링크를 사용하여 이를 다운로드할 수 있습니다. 스크립트를 확인하고 검토한 후(아래 설명 참조), 스크립트 주제에 설명된 프로시저에 따라 스크립트를 실행할 수 있습니다. 스크립트 실행이 완료되면 출력을 AWS Support 사례에 업로드할 수 있습니다(아래 설명 참조).

스크립트를 실행하기 전에, 지원 스크립트를 다운로드하거나 저장할 때 발생했을 가능성이 있는 오류를 감지할 수 있습니다. 이를 수행하려면 스크립트 파일의 체크섬을 AWS에서 제공한 값과 비교합니다. AWS는 체크섬에 SHA256 알고리즘을 사용합니다.

체크섬을 사용하여 지원 스크립트 파일을 확인하려면

1. 제공된 최신 체크섬 파일을 열어 https://d2pwp9zz55emqw.cloudfront.net/sha256Check.txt에서 이러한 지원 스크립트를 확인합니다. 예를 들어, 파일에 아래와 같은 내용이 포함되어 있을 수 있습니다.

```
MYSQL  dfafd0d511477c699f96c64693ad0b1547d47e74d5c5f2f2025b790b1422e3c8
ORACLE  6c41ebcfc99518cfa8a10cb2ce8943b153b2cc7049117183d0b5de3d551bc312
POSTGRES  6ccd274863d14f6f3146fbdbbba43f2d8d4c6a4c25380d7b41c71883aa4f9790
SQL_SERVER  971a6f2c46aec8d083d2b3b6549b1e9990af3a15fe4b922e319f4fdd358debe7
```

2. 지원 파일이 포함된 디렉터리에서 운영 체제의 SHA256 검증 명령을 실행합니다. 예를 들어, macOS 운영 체제에서는 이 주제의 뒷부분에 설명된 Oracle 지원 스크립트에서 아래의 명령을 실행할 수 있습니다.

```
shasum -a 256 awsdms_support_collector_oracle.sql
```

3. 명령 결과를 최근에 열어본 `sha256Check.txt` 파일에 표시된 값과 비교합니다. 두 값이 일치해야 합니다. 일치하지 않을 경우, 지원 엔지니어에게 불일치하는 부분을 이야기하고 깨끗한 지원 스크립트 파일을 얻을 수 있는 방법을 문의합니다.

깨끗한 지원 스크립트 파일을 보유한 경우, 스크립트를 실행하기 전에 성능 및 보안 측면에서 SQL을 읽고 이해해야 합니다. 이 스크립트에서 SQL을 실행하는 것이 마음에 걸릴 경우 문제의 SQL을 주석 처리하거나 제거할 수 있습니다. 지원 엔지니어에게 적절한 해결 방법을 문의할 수도 있습니다.

스크립트가 성공적으로 완료되고 달리 명시되지 않는 한, 스크립트는 읽기 가능한 HTML 형식으로 출력을 반환합니다. 스크립트는 업무에 영향을 미칠 수 있는 모든 데이터 또는 보안 세부 정보를 이 HTML에서 제외하도록 설계되었습니다. 또한 스크립트는 데이터베이스나 해당 환경을 수정하지 않습니다. 하지만 HTML에서 공유하기에 부적합한 정보를 발견한 경우, HTML을 업로드하기 전에 문제의 정보를 얼마든지 제거해도 됩니다. HTML이 허용되는 경우, 지원 사례의 사례 세부 정보에 있는 첨부 파일을 사용하여 업로드합니다.

아래의 각 주제에서는 지원되는 AWS DMS 데이터베이스에 사용할 수 있는 스크립트와 이를 실행하는 방법을 설명합니다. 지원 엔지니어가 아래에 설명된 특정 스크립트를 안내합니다.

주제

- Oracle 진단 지원 스크립트
- SQL Server 진단 지원 스크립트
- MySQL 호환 데이터베이스용 진단 지원 스크립트

- PostgreSQL 진단 지원 스크립트

Oracle 진단 지원 스크립트

아래에는 AWS DMS 마이그레이션 구성에서 온프레미스 또는 Amazon RDS for Oracle 데이터베이스를 분석하는 데 사용 가능한 진단 지원 스크립트가 나와 있습니다. 이러한 스크립트는 소스 또는 대상 엔드포인트에서 작동합니다. 스크립트는 모두 SQL*Plus 명령줄 유틸리티에서 실행하도록 작성되었습니다. 이 유틸리티 사용에 대한 자세한 내용은 Oracle 설명서의 A Using SQL Command Line을 참조하세요.

스크립트를 실행하기 전에, 사용하려는 사용자 계정에 Oracle 데이터베이스에 액세스하는 데 필요한 권한이 있는지 확인합니다. 표시된 권한 설정에서는 다음과 같이 사용자를 생성한 것으로 가정합니다.

```
CREATE USER script_user IDENTIFIED BY password;
```

온프레미스 데이터베이스의 경우, 다음과 같이 *script_user*에 대한 최소 권한을 설정합니다.

```
GRANT CREATE SESSION TO script_user;
GRANT SELECT on V$DATABASE to script_user;
GRANT SELECT on V$VERSION to script_user;
GRANT SELECT on GV$SGA to script_user;
GRANT SELECT on GV$INSTANCE to script_user;
GRANT SELECT on GV$DATAGUARD_CONFIG to script_user;
GRANT SELECT on GV$LOG to script_user;
GRANT SELECT on DBA_TABLESPACES to script_user;
GRANT SELECT on DBA_DATA_FILES to script_user;
GRANT SELECT on DBA_SEGMENTS to script_user;
GRANT SELECT on DBA_LOBS to script_user;
GRANT SELECT on V$ARCHIVED_LOG to script_user;
GRANT SELECT on DBA_TAB_MODIFICATIONS to script_user;
GRANT SELECT on DBA_TABLES to script_user;
GRANT SELECT on DBA_TAB_PARTITIONS to script_user;
GRANT SELECT on DBA_MVIEWS to script_user;
GRANT SELECT on DBA_OBJECTS to script_user;
GRANT SELECT on DBA_TAB_COLUMNS to script_user;
GRANT SELECT on DBA_LOG_GROUPS to script_user;
GRANT SELECT on DBA_LOG_GROUP_COLUMNS to script_user;
GRANT SELECT on V$ARCHIVE_DEST to script_user;
GRANT SELECT on DBA_SYS_PRIVS to script_user;
GRANT SELECT on DBA_TAB_PRIVS to script_user;
GRANT SELECT on DBA_TYPES to script_user;
```

```
GRANT SELECT on DBA_CONSTRAINTS to script_user;
GRANT SELECT on V$TRANSACTION to script_user;
GRANT SELECT on GV$ASM_DISK_STAT to script_user;
GRANT SELECT on GV$SESSION to script_user;
GRANT SELECT on GV$SQL to script_user;
GRANT SELECT on DBA_ENCRYPTED_COLUMNS to script_user;
GRANT SELECT on DBA_PDBS to script_user;

GRANT EXECUTE on dbms_utility to script_user;
```

Amazon RDS 데이터베이스의 경우, 다음과 같이 최소 권한을 설정합니다.

```
GRANT CREATE SESSION TO script_user;
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$DATABASE','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$VERSION','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_$SGA','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_$INSTANCE','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_
$DATAGUARD_CONFIG','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_$LOG','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TABLESPACES','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_DATA_FILES','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_SEGMENTS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_LOBS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$ARCHIVED_LOG','script_user','SELECT');
exec
 rdsadmin.rdsadmin_util.grant_sys_object('DBA_TAB_MODIFICATIONS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TABLES','script_user','SELECT');
exec
 rdsadmin.rdsadmin_util.grant_sys_object('DBA_TAB_PARTITIONS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_MVIEWS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_OBJECTS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TAB_COLUMNS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_LOG_GROUPS','script_user','SELECT');
exec
 rdsadmin.rdsadmin_util.grant_sys_object('DBA_LOG_GROUP_COLUMNS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$ARCHIVE_DEST','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_SYS_PRIVS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TAB_PRIVS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_TYPES','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_CONSTRAINTS','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('V_$TRANSACTION','script_user','SELECT');
```

```
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_
$ASM_DISK_STAT','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_$SESSION','script_user','SELECT');
exec rdsadmin.rdsadmin_util.grant_sys_object('GV_$SQL','script_user','SELECT');
exec
 rdsadmin.rdsadmin_util.grant_sys_object('DBA_ENCRYPTED_COLUMNS','script_user','SELECT');

exec rdsadmin.rdsadmin_util.grant_sys_object('DBA_PDBS','script_user','SELECT');

exec rdsadmin.rdsadmin_util.grant_sys_object('DBMS_UTILITY','script_user','EXECUTE');
```

아래에는 Oracle에서 사용할 수 있는 각 SQL*Plus 지원 스크립트를 다운로드, 검토, 실행하는 방법에 대한 설명이 나와 있습니다. 출력을 검토하고 이를 AWS Support 사례에 업로드하는 방법도 참조할 수 있습니다.

주제

- awsdms_support_collector_oracle.sql 스크립트

awsdms_support_collector_oracle.sql 스크립트

awsdms_support_collector_oracle.sql 스크립트를 다운로드합니다.

이 스크립트는 Oracle 데이터베이스 구성에 대한 정보를 수집합니다. 스크립트의 체크섬을 반드시 확인하고, 체크섬이 확인되면 스크립트의 SQL 코드를 검토하여 실행하기에 부적합한 코드는 모두 주석 처리합니다. 스크립트의 무결성 및 내용에 만족한다면 스크립트를 실행해도 됩니다.

스크립트를 실행하고 결과를 지원 사례에 업로드하려면

1. 아래의 SQL*Plus 명령줄을 사용하여 데이터베이스 환경에서 스크립트를 실행합니다.

   ```
   SQL> @awsdms_support_collector_oracle.sql
   ```

   <result>

   스크립트에 간단한 설명이 표시되며, 실행을 계속할지 중단할지 묻는 프롬프트가 표시됩니다. [Enter] 키를 눌러 계속 진행합니다.
   </result>

2. 다음에 나오는 프롬프트에서 마이그레이션할 스키마 중 하나의 이름만 입력합니다.

3. 다음에 나오는 프롬프트에서 데이터베이스에 연결하기 위해 정의한 사용자 이름(*script_user*)을 입력합니다.
4. 다음에 나오는 프롬프트에서 검사할 데이터의 일수를 입력하거나, 기본값을 수락합니다. 이렇게 하면 스크립트가 데이터베이스에서 지정된 데이터를 수집합니다.
   <result>

   스크립트가 완료되면 출력 HTML 파일의 이름(예: dms_support_oracle-2020-06-22-13-20-39-ORCL.html)이 표시됩니다. 스크립트는 이 파일을 작업 디렉터리에 저장합니다.
   </result>
5. 이 HTML 파일을 검토하고, 공유하기에 부적합한 정보는 모두 제거합니다. HTML을 공유해도 괜찮다면 AWS Support 사례에 파일을 업로드합니다. 이 파일 업로드에 대한 자세한 내용은 AWS DMS에서 진단 지원 스크립트 작업 섹션을 참조하세요.

SQL Server 진단 지원 스크립트

아래에는 AWS DMS 마이그레이션 구성에서 온프레미스 또는 Amazon RDS for SQL Server 데이터베이스를 분석하는 데 사용 가능한 진단 지원 스크립트의 설명이 나와 있습니다. 이러한 스크립트는 소스 또는 대상 엔드포인트에서 작동합니다. 온프레미스 데이터베이스의 경우, sqlcmd 명령줄 유틸리티에서 이러한 스크립트를 실행합니다. 이 유틸리티 사용에 대한 자세한 내용은 Microsoft 설명서의 sqlcmd - Use the utility 섹션을 참조하세요.

Amazon RDS 데이터베이스의 경우에는 sqlcmd 명령줄 유틸리티를 사용하여 연결할 수 없습니다. 대신 Amazon RDS SQL Server에 연결되는 클라이언트 도구를 사용하여 이러한 스크립트를 실행합니다.

스크립트를 실행하기 전에, 사용하려는 사용자 계정에 SQL Server 데이터베이스에 액세스하는 데 필요한 권한이 있는지 확인합니다. 온프레미스 및 Amazon RDS 데이터베이스의 경우, `SysAdmin` 역할 없이도 SQL Server 데이터베이스에 액세스할 때 사용하는 것과 동일한 권한을 사용할 수 있습니다.

주제

- 온프레미스 SQL Server 데이터베이스에 대한 최소 권한 설정
- Amazon RDS SQL Server 데이터베이스에 대한 최소 권한 설정
- 독립 실행형 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음
- 가용성 그룹 환경의 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음
- SQL Server 지원 스크립트

온프레미스 SQL Server 데이터베이스에 대한 최소 권한 설정

온프레미스 SQL Server 데이터베이스를 실행하기 위한 최소 권한을 설정하려면

1. SQL Server Management Studio(SSMS)를 사용하여 암호 인증이 있는 새 SQL Server 계정을 생성합니다(예: *on-prem-user*).
2. SSMS의 사용자 매핑 섹션에서 MSDB 및 MASTER 데이터베이스(공용 권한 제공)를 선택하고, 스크립트를 실행할 데이터베이스에 DB_OWNER 역할을 할당합니다.
3. 새 계정의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고, 보안을 선택한 후 명시적으로 Connect SQL 권한을 부여합니다.
4. 아래의 grant 명령을 실행합니다.

```
GRANT VIEW SERVER STATE TO on-prem-user;
USE MSDB;
GRANT SELECT ON MSDB.DBO.BACKUPSET TO on-prem-user;
GRANT SELECT ON MSDB.DBO.BACKUPMEDIAFAMILY TO on-prem-user;
GRANT SELECT ON MSDB.DBO.BACKUPFILE TO on-prem-user;
```

Amazon RDS SQL Server 데이터베이스에 대한 최소 권한 설정

Amazon RDS SQL Server 데이터베이스에 대한 최소 권한으로 실행하려면

1. SQL Server Management Studio(SSMS)를 사용하여 암호 인증이 있는 새 SQL Server 계정을 생성합니다(예: *rds-user*).
2. SSMS의 사용자 매핑 섹션에서 MSDB 데이터베이스(공용 권한 제공)를 선택하고, 스크립트를 실행할 데이터베이스에 DB_OWNER 역할을 할당합니다.
3. 새 계정의 컨텍스트(마우스 오른쪽 버튼 클릭) 메뉴를 열고, 보안을 선택한 후 명시적으로 Connect SQL 권한을 부여합니다.
4. 아래의 grant 명령을 실행합니다.

```
GRANT VIEW SERVER STATE TO rds-user;
USE MSDB;
GRANT SELECT ON MSDB.DBO.BACKUPSET TO rds-user;
GRANT SELECT ON MSDB.DBO.BACKUPMEDIAFAMILY TO rds-user;
GRANT SELECT ON MSDB.DBO.BACKUPFILE TO rds-user;
```

독립 실행형 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음

이 섹션에서는 사용자 계정에 sysadmin 권한이 필요 없는 SQL Server 데이터베이스 소스에 대해 지속적 복제를 설정하는 방법을 설명합니다.

> **Note**
>
> sysadmin이 아닌 DMS 사용자는 이 섹션의 단계를 실행한 후 다음을 수행할 수 있는 권한을 갖게 됩니다.
>
> - 온라인 트랜잭션 로그 파일에서 변경 사항 읽기
> - 트랜잭션 로그 백업 파일의 변경 사항을 읽을 수 있는 디스크 액세스
> - DMS에서 사용하는 게시물 추가 또는 변경
> - 게시물에 문서 추가

1. 온프레미스 또는 Amazon EC2에서 자체 관리형 SQL Server의 데이터 변경 캡처에 설명된 대로 복제를 위해 Microsoft SQL Server를 설정합니다.
2. 소스 데이터베이스에서 MS-REPLICATION을 활성화합니다. 이 작업은 수동으로 수행하거나, sysadmin 사용자로 태스크를 한 번 실행하여 수행할 수 있습니다.
3. 아래의 스크립트를 사용하여 소스 데이터베이스에서 awsdms 스키마를 생성합니다.

```
use master
go
create schema awsdms
go


-- Create the table valued function [awsdms].[split_partition_list] on the Master
 database, as follows:
USE [master]
GO

set ansi_nulls on
go

set quoted_identifier on
go

if (object_id('[awsdms].[split_partition_list]','TF')) is not null
```

```
drop function [awsdms].[split_partition_list];

go

create function [awsdms].[split_partition_list]

(

@plist varchar(8000), –A delimited list of partitions

@dlm nvarchar(1) –Delimiting character

)

returns @partitionsTable table –Table holding the BIGINT values of the string
 fragments

(

pid bigint primary key

)

as

begin

declare @partition_id bigint;

declare @dlm_pos integer;

declare @dlm_len integer;

set @dlm_len = len(@dlm);

while (charindex(@dlm,@plist)>0)

begin

set @dlm_pos = charindex(@dlm,@plist);

set @partition_id = cast( ltrim(rtrim(substring(@plist,1,@dlm_pos-1))) as bigint);
```

```
insert into @partitionsTable (pid) values (@partition_id)

set @plist = substring(@plist,@dlm_pos+@dlm_len,len(@plist));

end

set @partition_id = cast (ltrim(rtrim(@plist)) as bigint);

insert into @partitionsTable (pid) values ( @partition_id );

return

end

GO
```

4. 아래의 스크립트를 사용하여 마스터 데이터베이스에서 [awsdms].[rtm_dump_dblog] 프로시저를 생성합니다.

```
use [MASTER]

go

if (object_id('[awsdms].[rtm_dump_dblog]','P')) is not null drop procedure
 [awsdms].[rtm_dump_dblog];
go


set ansi_nulls on
go

set quoted_identifier on
GO



CREATE procedure [awsdms].[rtm_dump_dblog]

(

@start_lsn varchar(32),

@seqno integer,
```

```
@filename varchar(260),

@partition_list varchar(8000), – A comma delimited list: P1,P2,... Pn

@programmed_filtering integer,

@minPartition bigint,

@maxPartition bigint

)

as begin

declare @start_lsn_cmp varchar(32); – Stands against the GT comparator

SET NOCOUNT ON – - Disable "rows affected display"

set @start_lsn_cmp = @start_lsn;

if (@start_lsn_cmp) is null

set @start_lsn_cmp = '00000000:00000000:0000';

if (@partition_list is null)

begin

RAISERROR ('Null partition list waspassed',16,1);

return

end

if (@start_lsn) is not null

set @start_lsn = '0x'+@start_lsn;

if (@programmed_filtering=0)



SELECT
```

```
[Current LSN],

[operation],

[Context],

[Transaction ID],

[Transaction Name],

[Begin Time],

[End Time],

[Flag Bits],

[PartitionID],

[Page ID],

[Slot ID],

[RowLog Contents 0],

[Log Record],

[RowLog Contents 1]

FROM

fn_dump_dblog (

@start_lsn, NULL, N'DISK', @seqno, @filename,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default,
```

```
default, default, default, default, default, default, default,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default)

where [Current LSN] collate SQL_Latin1_General_CP1_CI_AS > @start_lsn_cmp collate
 SQL_Latin1_General_CP1_CI_AS

and

(

( [operation] in ('LOP_BEGIN_XACT','LOP_COMMIT_XACT','LOP_ABORT_XACT') )

or

( [operation] in ('LOP_INSERT_ROWS','LOP_DELETE_ROWS','LOP_MODIFY_ROW')

and

( ( [context] in ('LCX_HEAP','LCX_CLUSTERED','LCX_MARK_AS_GHOST') ) or ([context] =
 'LCX_TEXT_MIX' and (datalength([RowLog Contents 0]) in (0,1))))

and [PartitionID] in ( select * from master.awsdms.split_partition_list
 (@partition_list,','))

)

or

([operation] = 'LOP_HOBT_DDL')

)


else


SELECT
```

```
[Current LSN],

[operation],

[Context],

[Transaction ID],

[Transaction Name],

[Begin Time],

[End Time],

[Flag Bits],

[PartitionID],

[Page ID],

[Slot ID],

[RowLog Contents 0],

[Log Record],

[RowLog Contents 1] — After Image

FROM

fn_dump_dblog (

@start_lsn, NULL, N'DISK', @seqno, @filename,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default,
```

```
default, default, default, default, default, default, default,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default,

default, default, default, default, default, default, default)

where [Current LSN] collate SQL_Latin1_General_CP1_CI_AS > @start_lsn_cmp collate
 SQL_Latin1_General_CP1_CI_AS

and

(

( [operation] in ('LOP_BEGIN_XACT','LOP_COMMIT_XACT','LOP_ABORT_XACT') )

or

( [operation] in ('LOP_INSERT_ROWS','LOP_DELETE_ROWS','LOP_MODIFY_ROW')

and

( ( [context] in ('LCX_HEAP','LCX_CLUSTERED','LCX_MARK_AS_GHOST') ) or ([context] =
 'LCX_TEXT_MIX' and (datalength([RowLog Contents 0]) in (0,1))))

and ([PartitionID] is not null) and ([PartitionID] >= @minPartition and
 [PartitionID]<=@maxPartition)

)

or

([operation] = 'LOP_HOBT_DDL')

)



SET NOCOUNT OFF – Re-enable "rows affected display"

end
```

```
GO
```

5. 아래의 스크립트를 사용하여 마스터 데이터베이스에서 인증서를 생성합니다.

```
Use [master]
Go

CREATE CERTIFICATE [awsdms_rtm_dump_dblog_cert] ENCRYPTION BY PASSWORD =
 N'@5trongpassword'

WITH SUBJECT = N'Certificate for FN_DUMP_DBLOG Permissions';
```

6. 아래의 스크립트를 사용하여 인증서에서 로그인을 생성합니다.

```
Use [master]
Go

CREATE LOGIN awsdms_rtm_dump_dblog_login FROM CERTIFICATE
 [awsdms_rtm_dump_dblog_cert];
```

7. 아래의 스크립트를 사용하여 sysadmin 서버 역할에 로그인을 추가합니다.

```
ALTER SERVER ROLE [sysadmin] ADD MEMBER [awsdms_rtm_dump_dblog_login];
```

8. 아래의 스크립트를 사용하여 [master].[awsdms].[rtm_dump_dblog]에 서명을 추가합니다.

```
Use [master]
GO
ADD SIGNATURE
TO [master].[awsdms].[rtm_dump_dblog] BY CERTIFICATE [awsdms_rtm_dump_dblog_cert]
 WITH PASSWORD = '@5trongpassword';
```

> **Note**
>
> 저장된 프로시저를 다시 생성할 경우 서명을 다시 추가해야 합니다.

9. 아래의 스크립트를 사용하여 마스터 데이터베이스에 [awsdms].[rtm_position_1st_timestamp]를 생성합니다.

```
use [master]
    if object_id('[awsdms].[rtm_position_1st_timestamp]','P') is not null
```

```
   DROP PROCEDURE [awsdms].[rtm_position_1st_timestamp];
   go
   create procedure [awsdms].[rtm_position_1st_timestamp]
   (
   @dbname                sysname,      -- Database name
   @seqno                 integer,      -- Backup set sequence/position number
within file
   @filename              varchar(260), -- The backup filename
   @1stTimeStamp          varchar(40)   -- The timestamp to position by
   )
   as begin

   SET NOCOUNT ON       -- Disable "rows affected display"

   declare @firstMatching table
   (
   cLsn varchar(32),
   bTim datetime
   )

   declare @sql nvarchar(4000)
   declare @nl                      char(2)
   declare @tb                      char(2)
   declare @fnameVar                nvarchar(254) = 'NULL'

   set @nl  = char(10); -- New line
   set @tb  = char(9)   -- Tab separator

   if (@filename is not null)
   set @fnameVar = ''''+@filename +''''

   set @sql='use ['+@dbname+'];'+@nl+
   'select top 1 [Current LSN],[Begin Time]'+@nl+
   'FROM fn_dump_dblog (NULL, NULL, NULL, '+ cast(@seqno as varchar(10))+','+
@fnameVar+','+@nl+
   @tb+'default, default, default, default, default, default, default,'+@nl+
   @tb+'default, default, default, default, default, default, default,'+@nl+
   @tb+'default, default, default, default, default, default, default,'+@nl+
   @tb+'default, default, default, default, default, default, default,'+@nl+
   @tb+'default, default, default, default, default, default, default,'+@nl+
   @tb+'default, default, default, default, default, default, default,'+@nl+
   @tb+'default, default, default, default, default, default, default,'+@nl+
   @tb+'default, default, default, default, default, default, default,'+@nl+
   @tb+'default, default, default, default, default, default, default)'+@nl+
```

```
    'where operation=''LOP_BEGIN_XACT''' +@nl+
    'and [Begin Time]>= cast('+''''+@1stTimeStamp+''''+' as datetime)'+@nl

    --print @sql
    delete from  @firstMatching
    insert into @firstMatching  exec sp_executesql @sql    -- Get them all

    select top 1 cLsn as [matching LSN],convert(varchar,bTim,121) as [matching
 Timestamp] from @firstMatching;

    SET NOCOUNT OFF      -- Re-enable "rows affected display"

    end
    GO
```

10. 아래의 스크립트를 사용하여 마스터 데이터베이스에서 인증서를 생성합니다.

```
Use [master]
Go
CREATE CERTIFICATE [awsdms_rtm_position_1st_timestamp_cert]
ENCRYPTION BY PASSWORD = '@5trongpassword'
WITH SUBJECT = N'Certificate for FN_POSITION_1st_TIMESTAMP Permissions';
```

11. 아래의 스크립트를 사용하여 인증서에서 로그인을 생성합니다.

```
Use [master]
Go
CREATE LOGIN awsdms_rtm_position_1st_timestamp_login FROM CERTIFICATE
 [awsdms_rtm_position_1st_timestamp_cert];
```

12. 아래의 스크립트를 사용하여 sysadmin 역할에 로그인을 추가합니다.

```
ALTER SERVER ROLE [sysadmin] ADD MEMBER [awsdms_rtm_position_1st_timestamp_login];
```

13. 아래의 스크립트를 사용하여 인증서를 통해 [master].[awsdms].[rtm_position_1st_timestamp]에 서명을 추가합니다.

```
Use [master]
    GO
    ADD SIGNATURE
    TO [master].[awsdms].[rtm_position_1st_timestamp]
    BY CERTIFICATE [awsdms_rtm_position_1st_timestamp_cert]
```

```
    WITH PASSWORD = '@5trongpassword';
```

14. 아래의 스크립트를 사용하여 DMS 사용자에게 새 저장 프로시저에 대한 실행 액세스 권한을 부여합니다.

```
use master
go
GRANT execute on [awsdms].[rtm_position_1st_timestamp] to dms_user;
```

15. 다음 각 데이터베이스에서 다음과 같은 권한 및 역할을 보유한 사용자를 생성합니다.

> **Note**
>
> 각 복제본에서 동일한 SID를 사용하여 dmsnosysadmin 사용자 계정을 생성해야 합니다. 아래의 SQL 쿼리는 각 복제본에서 dmsnosysadmin 계정 SID 값을 확인하는 데 도움이 될 수 있습니다. 사용자 생성에 대한 자세한 내용은 Microsoft SQL 서버 설명서의 CREATE USER (Transact-SQL) 섹션을 참조하세요. Azure SQL 데이터베이스의 SQL 사용자 계정 생성에 대한 자세한 내용은 Active geo-replication 섹션을 참조하세요.

```
use master
go
grant select on sys.fn_dblog to [DMS_user]
grant view any definition to [DMS_user]
grant view server state to [DMS_user]—(should be granted to the login).
grant execute on sp_repldone to [DMS_user]
grant execute on sp_replincrementlsn to [DMS_user]
grant execute on sp_addpublication to [DMS_user]
grant execute on sp_addarticle to [DMS_user]
grant execute on sp_articlefilter to [DMS_user]
grant select on [awsdms].[split_partition_list] to [DMS_user]
grant execute on [awsdms].[rtm_dump_dblog] to [DMS_user]
```

```
use MSDB
go
grant select on msdb.dbo.backupset to [DMS_user]
grant select on msdb.dbo.backupmediafamily to [DMS_user]
grant select on msdb.dbo.backupfile to [DMS_user]
```

소스 데이터베이스에서 다음과 같은 스크립트를 실행합니다.

```
EXEC sp_addrolemember N'db_owner', N'DMS_user'
use Source_DB
go
```

16. 마지막으로, 소스 SQL Server 엔드포인트에 추가 연결 속성(ECA)을 추가합니다.

```
enableNonSysadminWrapper=true;
```

가용성 그룹 환경의 SQL Server에서 지속적 복제 설정: sysadmin 역할 없음

이 섹션에서는 사용자 계정에 sysadmin 권한이 필요 없는 가용성 그룹 환경의 SQL Server 데이터베이스 소스에 대해 지속적 복제를 설정하는 방법을 설명합니다.

> **Note**
>
> sysadmin이 아닌 DMS 사용자는 이 섹션의 단계를 실행한 후 다음을 수행할 수 있는 권한을 갖게 됩니다.
>
> - 온라인 트랜잭션 로그 파일에서 변경 사항 읽기
> - 트랜잭션 로그 백업 파일의 변경 사항을 읽을 수 있는 디스크 액세스
> - DMS에서 사용하는 게시물 추가 또는 변경
> - 게시물에 문서 추가

가용성 그룹 환경에서 sysadmin 사용자를 사용하지 않고 지속적 복제를 설정하려면

1. 온프레미스 또는 Amazon EC2에서 자체 관리형 SQL Server의 데이터 변경 캡처에 설명된 대로 복제를 위해 Microsoft SQL Server를 설정합니다.
2. 소스 데이터베이스에서 MS-REPLICATION을 활성화합니다. 이 작업은 수동으로 수행하거나, sysadmin 사용자를 통해 태스크를 한 번 실행하여 수행할 수 있습니다.

> **Note**
>
> MS-REPLICATION 배포자를 로컬로 구성하거나, 연결된 서버를 통해 sysadmin 이외의 사용자가 액세스할 수 있는 방식으로 구성해야 합니다.

3. 단일 태스크 내에서 sp_repldone을 단독으로 사용 엔드포인트 옵션이 활성화된 경우, MS-REPLICATION Log Reader 작업을 중지합니다.
4. 각각의 복제에서 다음 단계를 수행합니다.

   1. 마스터 데이터베이스에서 [awsdms][awsdms] 스키마를 생성합니다.

      ```
      CREATE SCHEMA [awsdms]
      ```

   2. 마스터 데이터베이스에서 [awsdms].[split_partition_list] 테이블 값 함수를 생성합니다.

      ```
      USE [master]
      GO

      SET ansi_nulls on
      GO

      SET quoted_identifier on
      GO

      IF (object_id('[awsdms].[split_partition_list]','TF')) is not null
        DROP FUNCTION [awsdms].[split_partition_list];
      GO

      CREATE FUNCTION [awsdms].[split_partition_list]
      (
        @plist varchar(8000),    --A delimited list of partitions
        @dlm nvarchar(1)    --Delimiting character
      )
      RETURNS @partitionsTable table --Table holding the BIGINT values of the string
       fragments
      (
        pid bigint primary key
      )
      AS
      BEGIN
        DECLARE @partition_id bigint;
        DECLARE @dlm_pos integer;
        DECLARE @dlm_len integer;
        SET @dlm_len = len(@dlm);
        WHILE (charindex(@dlm,@plist)>0)
        BEGIN
      ```

```
    SET @dlm_pos = charindex(@dlm,@plist);
    SET @partition_id = cast( ltrim(rtrim(substring(@plist,1,@dlm_pos-1))) as
 bigint);
    INSERT into @partitionsTable (pid) values (@partition_id)
    SET @plist = substring(@plist,@dlm_pos+@dlm_len,len(@plist));
  END
  SET @partition_id = cast (ltrim(rtrim(@plist)) as bigint);
  INSERT into @partitionsTable (pid) values (  @partition_id  );
  RETURN
END
GO
```

3. 마스터 데이터베이스에서 [awsdms].[rtm_dump_dblog] 프로시저를 생성합니다.

```
USE [MASTER]
GO

IF (object_id('[awsdms].[rtm_dump_dblog]','P')) is not null
  DROP PROCEDURE [awsdms].[rtm_dump_dblog];
GO

SET ansi_nulls on
GO

SET quoted_identifier on
GO

CREATE PROCEDURE [awsdms].[rtm_dump_dblog]
(
  @start_lsn            varchar(32),
  @seqno                integer,
  @filename             varchar(260),
  @partition_list       varchar(8000), -- A comma delimited list: P1,P2,... Pn
  @programmed_filtering integer,
  @minPartition         bigint,
  @maxPartition         bigint
)
AS
BEGIN

  DECLARE @start_lsn_cmp varchar(32); -- Stands against the GT comparator

  SET NOCOUNT ON  -- Disable "rows affected display"
```

```
SET @start_lsn_cmp = @start_lsn;
IF (@start_lsn_cmp) is null
  SET @start_lsn_cmp = '00000000:00000000:0000';

IF (@partition_list is null)
  BEGIN
    RAISERROR ('Null partition list was passed',16,1);
    return
    --set @partition_list = '0,';    -- A dummy which is never matched
  END

IF (@start_lsn) is not null
  SET @start_lsn = '0x'+@start_lsn;

IF (@programmed_filtering=0)
  SELECT
    [Current LSN],
    [operation],
    [Context],
    [Transaction ID],
    [Transaction Name],
    [Begin Time],
    [End Time],
    [Flag Bits],
    [PartitionID],
    [Page ID],
    [Slot ID],
    [RowLog Contents 0],
    [Log Record],
    [RowLog Contents 1] -- After Image
  FROM
    fn_dump_dblog (
      @start_lsn, NULL, N'DISK', @seqno, @filename,
      default, default, default, default, default, default, default,
      default, default, default, default, default, default, default,
      default, default, default, default, default, default, default,
      default, default, default, default, default, default, default,
      default, default, default, default, default, default, default,
      default, default, default, default, default, default, default,
      default, default, default, default, default, default, default,
      default, default, default, default, default, default, default)
  WHERE
```

```
     [Current LSN] collate SQL_Latin1_General_CP1_CI_AS > @start_lsn_cmp collate
SQL_Latin1_General_CP1_CI_AS -- This aims for implementing FN_DBLOG based on GT
comparator.
     AND
     (
       (  [operation] in ('LOP_BEGIN_XACT','LOP_COMMIT_XACT','LOP_ABORT_XACT') )
       OR
       (  [operation] in ('LOP_INSERT_ROWS','LOP_DELETE_ROWS','LOP_MODIFY_ROW')
         AND
         ( ( [context]   in ('LCX_HEAP','LCX_CLUSTERED','LCX_MARK_AS_GHOST') )
or ([context] = 'LCX_TEXT_MIX') )
         AND
         [PartitionID] in ( select * from master.awsdms.split_partition_list
(@partition_list,','))
       )
     OR
     ([operation] = 'LOP_HOBT_DDL')
   )
   ELSE
     SELECT
       [Current LSN],
       [operation],
       [Context],
       [Transaction ID],
       [Transaction Name],
       [Begin Time],
       [End Time],
       [Flag Bits],
       [PartitionID],
       [Page ID],
       [Slot ID],
       [RowLog Contents 0],
       [Log Record],
       [RowLog Contents 1] -- After Image
     FROM
       fn_dump_dblog (
         @start_lsn, NULL, N'DISK', @seqno, @filename,
         default, default, default, default, default, default, default,
         default, default, default, default, default, default, default,
         default, default, default, default, default, default, default,
         default, default, default, default, default, default, default,
         default, default, default, default, default, default, default,
         default, default, default, default, default, default, default,
         default, default, default, default, default, default, default,
```

```
          default, default, default, default, default, default, default,
          default, default, default, default, default, default, default)
     WHERE [Current LSN] collate SQL_Latin1_General_CP1_CI_AS > @start_lsn_cmp
 collate SQL_Latin1_General_CP1_CI_AS -- This aims for implementing FN_DBLOG
 based on GT comparator.
     AND
     (
       (  [operation] in ('LOP_BEGIN_XACT','LOP_COMMIT_XACT','LOP_ABORT_XACT') )
       OR
       (  [operation] in ('LOP_INSERT_ROWS','LOP_DELETE_ROWS','LOP_MODIFY_ROW')
         AND
         ( ( [context]   in ('LCX_HEAP','LCX_CLUSTERED','LCX_MARK_AS_GHOST') )
 or ([context] = 'LCX_TEXT_MIX') )
         AND ([PartitionID] is not null) and ([PartitionID] >= @minPartition and
 [PartitionID]<=@maxPartition)
       )
       OR
       ([operation] = 'LOP_HOBT_DDL')
     )
     SET NOCOUNT OFF -- Re-enable "rows affected display"
END
GO
```

4. 마스터 데이터베이스에서 인증서를 생성합니다.

```
USE [master]
GO
CREATE CERTIFICATE [awsdms_rtm_dump_dblog_cert]
  ENCRYPTION BY PASSWORD = N'@hardpassword1'
  WITH SUBJECT = N'Certificate for FN_DUMP_DBLOG Permissions'
```

5. 인증서에서 로그인을 생성합니다.

```
USE [master]
GO
CREATE LOGIN awsdms_rtm_dump_dblog_login FROM CERTIFICATE
  [awsdms_rtm_dump_dblog_cert];
```

6. sysadmin 서버 역할에 로그인을 추가합니다.

```
ALTER SERVER ROLE [sysadmin] ADD MEMBER [awsdms_rtm_dump_dblog_login];
```

7. 인증서를 사용하여 [master].[awsdms].[rtm_dump_dblog] 프로시저에 서명을 추가합니다.

```
USE [master]
GO

ADD SIGNATURE
  TO [master].[awsdms].[rtm_dump_dblog]
  BY CERTIFICATE [awsdms_rtm_dump_dblog_cert]
  WITH PASSWORD = '@hardpassword1';
```

> **Note**
>
> 저장된 프로시저를 다시 생성할 경우 서명을 다시 추가해야 합니다.

8. 마스터 데이터베이스에서 [awsdms].[rtm_position_1st_timestamp] 프로시저를 생성합니다.

```
USE [master]
IF object_id('[awsdms].[rtm_position_1st_timestamp]','P') is not null
  DROP PROCEDURE [awsdms].[rtm_position_1st_timestamp];
GO
CREATE PROCEDURE [awsdms].[rtm_position_1st_timestamp]
(
  @dbname                sysname,      -- Database name
  @seqno                 integer,      -- Backup set sequence/position number
 within file
  @filename              varchar(260), -- The backup filename
  @1stTimeStamp          varchar(40)   -- The timestamp to position by
)
AS
BEGIN
  SET NOCOUNT ON       -- Disable "rows affected display"

  DECLARE @firstMatching table
  (
    cLsn varchar(32),
    bTim datetime
  )
  DECLARE @sql nvarchar(4000)
  DECLARE @nl                        char(2)
  DECLARE @tb                        char(2)
  DECLARE @fnameVar                  sysname = 'NULL'
```

```
  SET @nl  = char(10); -- New line
  SET @tb  = char(9)   -- Tab separator

  IF (@filename is not null)
    SET @fnameVar = ''''+@filename +''''
  SET @filename = ''''+@filename +''''
  SET @sql='use ['+@dbname+'];'+@nl+
    'SELECT TOP 1 [Current LSN],[Begin Time]'+@nl+
    'FROM fn_dump_dblog (NULL, NULL, NULL, '+ cast(@seqno as varchar(10))+','+
 @filename +','+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default,'+@nl+
    @tb+'default, default, default, default, default, default, default)'+@nl+
    'WHERE operation=''LOP_BEGIN_XACT''' +@nl+
    'AND [Begin Time]>= cast('+''''+@1stTimeStamp+''''+' as datetime)'+@nl

    --print @sql
    DELETE FROM @firstMatching
    INSERT INTO @firstMatching  exec sp_executesql @sql    -- Get them all
    SELECT TOP 1 cLsn as [matching LSN],convert(varchar,bTim,121) AS[matching
 Timestamp] FROM @firstMatching;

    SET NOCOUNT OFF      -- Re-enable "rows affected display"

END
GO
```

9. 마스터 데이터베이스에서 인증서를 생성합니다.

```
USE [master]
GO
CREATE CERTIFICATE [awsdms_rtm_position_1st_timestamp_cert]
  ENCRYPTION BY PASSWORD = N'@hardpassword1'
  WITH SUBJECT = N'Certificate for FN_POSITION_1st_TIMESTAMP Permissions';
```

10.인증서에서 로그인을 생성합니다.

```
USE [master]
GO
CREATE LOGIN awsdms_rtm_position_1st_timestamp_login FROM CERTIFICATE
  [awsdms_rtm_position_1st_timestamp_cert];
```

11. sysadmin 서버 역할에 로그인을 추가합니다.

```
ALTER SERVER ROLE [sysadmin] ADD MEMBER
 [awsdms_rtm_position_1st_timestamp_login];
```

12. 인증서를 사용하여 [master].[awsdms].[rtm_position_1st_timestamp] 프로시저에 서명을 추가합니다.

```
USE [master]
GO
ADD SIGNATURE
  TO [master].[awsdms].[rtm_position_1st_timestamp]
  BY CERTIFICATE [awsdms_rtm_position_1st_timestamp_cert]
  WITH PASSWORD = '@hardpassword1';
```

> **Note**
>
> 저장된 프로시저를 다시 생성할 경우 서명을 다시 추가해야 합니다.

13. 다음 각 데이터베이스에서 다음과 같은 권한/역할을 보유한 사용자를 생성합니다.

> **Note**
>
> 각 복제본에서 동일한 SID를 사용하여 dmsnosysadmin 사용자 계정을 생성해야 합니다. 아래의 SQL 쿼리는 각 복제본에서 dmsnosysadmin 계정 SID 값을 확인하는 데 도움이 될 수 있습니다. 사용자 생성에 대한 자세한 내용은 Microsoft SQL 서버 설명서의 CREATE USER (Transact-SQL) 섹션을 참조하세요. Azure SQL 데이터베이스의 SQL 사용자 계정 생성에 대한 자세한 내용은 Active geo-replication 섹션을 참조하세요.

```
SELECT @@servername servername, name, sid, create_date, modify_date
  FROM sys.server_principals
  WHERE name = 'dmsnosysadmin';
```

14.각 복제에서 마스터 데이터베이스에 대한 권한을 부여합니다.

```
USE master
GO

GRANT select on sys.fn_dblog to dmsnosysadmin;
GRANT view any definition to dmsnosysadmin;
GRANT view server state to dmsnosysadmin -- (should be granted to the login).
GRANT execute on sp_repldone to dmsnosysadmin;
GRANT execute on sp_replincrementlsn to dmsnosysadmin;
GRANT execute on sp_addpublication to dmsnosysadmin;
GRANT execute on sp_addarticle to dmsnosysadmin;
GRANT execute on sp_articlefilter to dmsnosysadmin;
GRANT select on [awsdms].[split_partition_list] to dmsnosysadmin;
GRANT execute on [awsdms].[rtm_dump_dblog] to dmsnosysadmin;
GRANT execute on [awsdms].[rtm_position_1st_timestamp] to dmsnosysadmin;
```

15.각 복제에서 msdb 데이터베이스에 대한 권한을 부여합니다.

```
USE msdb
GO
GRANT select on msdb.dbo.backupset to dmsnosysadmin
GRANT select on msdb.dbo.backupmediafamily to dmsnosysadmin
GRANT select on msdb.dbo.backupfile to dmsnosysadmin
```

16.소스 데이터베이스에서 db_owner 역할을 dmsnosysadmin에 추가합니다. 데이터베이스가 동기화되었으므로, 기본 복제에만 역할을 추가할 수 있습니다.

```
use <source DB>
GO
EXEC sp_addrolemember N'db_owner', N'dmsnosysadmin'
```

SQL Server 지원 스크립트

다음 주제에서는 SQL Server에 사용할 수 있는 각 지원 스크립트를 다운로드, 검토, 실행하는 방법을 설명합니다. 스크립트 출력을 검토하고 이를 AWS Support 사례에 업로드하는 방법도 설명합니다.

주제

- awsdms_support_collector_sql_server.sql 스크립트

awsdms_support_collector_sql_server.sql 스크립트

awsdms_support_collector_sql_server.sql 스크립트를 다운로드합니다.

> **Note**
>
> 이 SQL Server 진단 지원 스크립트는 SQL Server 2014 이상 버전에서만 실행합니다.

이 스크립트는 SQL Server 데이터베이스 구성에 대한 정보를 수집합니다. 스크립트의 체크섬을 반드시 확인하고, 체크섬이 확인되면 스크립트의 SQL 코드를 검토하여 실행하기에 부적합한 코드는 모두 주석 처리합니다. 스크립트의 무결성 및 내용에 만족한다면 스크립트를 실행해도 됩니다.

온프레미스 SQL Server 데이터베이스용 스크립트를 실행하려면

1. 아래의 sqlcmd 명령줄을 사용하여 스크립트를 실행합니다.

   ```
   sqlcmd -Uon-prem-user -Ppassword -SDMS-SQL17AG-N1 -y 0
   -iC:\Users\admin\awsdms_support_collector_sql_server.sql -oC:\Users\admin
   \DMS_Support_Report_SQLServer.html -dsqlserverdb01
   ```

   지정된 sqlcmd 명령 파라미터에는 다음 내용이 포함됩니다.

   - -U – 데이터베이스 사용자 이름.
   - -P – 데이터베이스 사용자 암호.
   - -S – SQL Server 데이터베이스 서버 이름.
   - -y – sqlcmd 유틸리티의 최대 출력 열 너비. 값이 0이면 너비가 무제한인 열이 지정됩니다.
   - -i – 실행할 지원 스크립트의 경로(이 경우 awsdms_support_collector_sql_server.sql).
   - -o – 수집된 데이터베이스 구성 정보가 포함된 출력 HTML 파일의 경로(지정한 파일 이름 포함).
   - -d – SQL Server 데이터베이스 이름.
2. 스크립트가 완료되면 출력 HTML 파일을 검토하고, 공유하기에 부적합한 정보는 모두 제거합니다. HTML을 공유해도 괜찮다면 AWS Support 사례에 파일을 업로드합니다. 이 파일 업로드에 대한 자세한 내용은 AWS DMS에서 진단 지원 스크립트 작업 섹션을 참조하세요.

Amazon RDS for SQL Server에서는 sqlcmd 명령줄 유틸리티를 사용하여 연결할 수 없으므로, 다음 프로시저를 사용합니다.

온프레미스 RDS SQL Server 데이터베이스용 스크립트를 실행하려면

1. RDS SQL Server에 `Master` 사용자로 연결하고 출력을 HTML 파일로 저장할 수 있는 클라이언트 도구를 사용하여 스크립트를 실행합니다.
2. 출력 HTML 파일을 검토하고, 공유하기에 부적합한 정보는 모두 제거합니다. HTML을 공유해도 괜찮다면 AWS Support 사례에 파일을 업로드합니다. 이 파일 업로드에 대한 자세한 내용은 AWS DMS에서 진단 지원 스크립트 작업 섹션을 참조하세요.

MySQL 호환 데이터베이스용 진단 지원 스크립트

아래에는 AWS DMS 마이그레이션 구성에서 온프레미스 또는 Amazon RDS for MySQL 데이터베이스를 분석하는 데 사용 가능한 진단 지원 스크립트가 나와 있습니다. 이러한 스크립트는 소스 또는 대상 엔드포인트에서 작동합니다. 스크립트는 모두 MySQL SQL 명령줄 유틸리티에서 실행하도록 작성되었습니다.

MySQL 클라이언트 설치에 대한 자세한 내용은 MySQL 설명서의 Installing MySQL Shell 섹션을 참조하세요. MySQL 클라이언트 사용에 대한 자세한 내용은 MySQL 설명서의 Using MySQL Shell Commands 섹션을 참조하세요.

스크립트를 실행하기 전에, 사용하려는 사용자 계정에 MySQL 호환 데이터베이스에 액세스하는 데 필요한 권한이 있는지 확인합니다. 아래의 절차를 사용하여 사용자 계정을 생성하고, 이 스크립트를 실행하는 데 필요한 최소 권한을 제공합니다.

이러한 스크립트를 실행할 수 있는 최소 권한으로 사용자 계정을 설정하려면

1. 스크립트를 실행할 사용자를 생성합니다.

```
create user 'username'@'hostname' identified by password;
```

2. 데이터베이스에서 `select` 명령을 부여하여 데이터베이스를 분석합니다.

```
grant select on database-name.* to username;
grant replication client on *.* to username;
```

3.

```
grant execute on procedure mysql.rds_show_configuration to username;
```

다음 주제에서는 MySQL 호환 데이터베이스에 사용할 수 있는 각 지원 스크립트를 다운로드, 검토, 실행하는 방법을 설명합니다. 스크립트 출력을 검토하고 이를 AWS Support 사례에 업로드하는 방법도 설명합니다.

주제

- awsdms_support_collector_MySQL.sql 스크립트

awsdms_support_collector_MySQL.sql 스크립트

awsdms_support_collector_MySQL.sql 스크립트를 다운로드합니다.

이 스크립트는 MySQL 호환 데이터베이스 구성에 대한 정보를 수집합니다. 스크립트의 체크섬을 반드시 확인하고, 체크섬이 확인되면 스크립트의 SQL 코드를 검토하여 실행하기에 부적합한 코드는 모두 주석 처리합니다. 스크립트의 무결성 및 내용에 만족한다면 스크립트를 실행해도 됩니다.

명령줄을 사용하여 데이터베이스 환경에 연결한 후 스크립트를 실행합니다.

이 스크립트를 실행하고 결과를 지원 사례에 업로드하려면

1. 아래의 mysql 명령을 사용하여 데이터베이스에 연결합니다.

```
mysql -h hostname -P port -u username database-name
```

2. 아래의 mysql source 명령을 사용하여 스크립트를 실행합니다.

```
mysql> source awsdms_support_collector_MySQL_compatible_DB.sql
```

생성된 보고서를 검토하고, 공유하기에 부적합한 정보는 모두 제거합니다. 내용을 공유해도 괜찮다면 AWS Support 사례에 파일을 업로드합니다. 이 파일 업로드에 대한 자세한 내용은 AWS DMS에서 진단 지원 스크립트 작업 섹션을 참조하세요.

> **Note**
>
> - MySQL 호환 데이터베이스용 진단 지원 스크립트 에 설명된 필수 권한을 보유한 사용자 계정이 이미 있는 경우, 기존 사용자 계정을 사용하여 스크립트를 실행할 수도 있습니다.
> - 스크립트를 실행하기 전에 데이터베이스에 연결해야 합니다.
> - 스크립트는 텍스트 형식으로 출력을 생성합니다.

- 보안 모범 사례를 고려한다면, 이 MySQL 진단 지원 스크립트를 실행하기 위한 용도로만 새 사용자 계정을 생성할 경우 스크립트를 실행한 후에는 이 사용자 계정을 삭제하는 것이 좋습니다.

PostgreSQL 진단 지원 스크립트

아래에는 AWS DMS 마이그레이션 구성에서 PostgreSQL RDBMS(온프레미스, Amazon RDS, Aurora PostgreSQL)를 분석하는 데 사용 가능한 진단 지원 스크립트가 나와 있습니다. 이러한 스크립트는 소스 또는 대상 엔드포인트에서 작동합니다. 스크립트는 모두 psql 명령줄 유틸리티에서 실행하도록 작성되었습니다.

이러한 스크립트를 실행하기 전에, 사용하려는 사용자 계정에 PostgreSQL RDBMS에 액세스하는 데 필요한 다음과 같은 권한이 있는지 확인합니다.

- PostgreSQL 10.x 이상 버전 – `pg_catalog.pg_ls_waldir` 함수에 대한 실행 권한이 있는 사용자 계정.
- PostgreSQL 9.x 이전 버전 – 기본 권한이 있는 사용자 계정.

이러한 스크립트를 실행하려면 적절한 권한이 있는 기존 계정을 사용하는 것이 좋습니다.

새 사용자 계정을 생성하거나 기존 계정에 이러한 스크립트를 실행할 수 있는 권한을 부여해야 하는 경우, PostgreSQL 버전을 기반으로 하는 모든 PostgreSQL RDBMS에 대해 다음과 같은 SQL 명령을 실행할 수 있습니다.

PostgreSQL 데이터베이스 버전 10.x 이상에서 이러한 스크립트를 실행할 수 있는 권한을 계정에 부여하려면

- 다음 중 하나를 수행합니다.
  - 새 사용자 계정의 경우 다음을 실행합니다.

```
CREATE USER script_user WITH PASSWORD 'password';
GRANT EXECUTE ON FUNCTION pg_catalog.pg_ls_waldir TO script_user;
```

  - 새 사용자 계정의 경우 다음을 실행합니다.

```
GRANT EXECUTE ON FUNCTION pg_catalog.pg_ls_waldir TO script_user;
```

PostgreSQL 9.x 이하 버전의 데이터베이스에서 이러한 스크립트를 실행할 수 있는 권한을 계정에 부여하려면

- 다음 중 하나를 수행합니다.
  - 새 사용자 계정의 경우 기본 권한으로 다음을 실행합니다.

```
CREATE USER script_user WITH PASSWORD password;
```

  - 기존 사용자 계정의 경우 기존 권한을 사용합니다.

> **Note**
>
> 이러한 스크립트는 PostgreSQL 9.x 이하 버전 데이터베이스의 WAL 크기 확인과 관련된 특정 기능을 지원하지 않습니다. 자세한 내용은 AWS Support에 문의하세요.

다음 주제에서는 PostgreSQL에 사용할 수 있는 각 지원 스크립트를 다운로드, 검토, 실행하는 방법을 설명합니다. 스크립트 출력을 검토하고 이를 AWS Support 사례에 업로드하는 방법도 설명합니다.

주제

- awsdms_support_collector_postgres.sql 스크립트

awsdms_support_collector_postgres.sql 스크립트

awsdms_support_collector_postgres.sql 스크립트를 다운로드합니다.

이 스크립트는 PostgreSQL 데이터베이스 구성에 대한 정보를 수집합니다. 스크립트에서 체크섬을 반드시 확인해야 합니다. 체크섬이 확인되면 스크립트의 SQL 코드를 검토하여 실행하기에 부적합한 코드는 모두 주석 처리합니다. 스크립트의 무결성 및 내용에 만족한다면 스크립트를 실행해도 됩니다.

> **Note**
>
> psql 클라이언트 버전 10 이상에서 이 스크립트를 실행할 수 있습니다.

다음 절차에 따라 데이터베이스 환경 또는 명령줄에서 이 스크립트를 실행할 수 있습니다. 둘 중 어떤 경우든 나중에 AWS Support에 파일을 업로드할 수 있습니다.

이 스크립트를 실행하고 결과를 지원 사례에 업로드하려면

1. 다음 중 하나를 수행합니다.

   - 아래의 psql 명령줄을 사용하여 데이터베이스 환경에서 스크립트를 실행합니다.

     ```
     dbname=# \i awsdms_support_collector_postgres.sql
     ```

     다음에 나오는 프롬프트에서 마이그레이션할 스키마 중 하나의 이름만 입력합니다.

     다음에 나오는 프롬프트에서 데이터베이스에 연결하기 위해 정의한 사용자 이름(*script_user*)을 입력합니다.

   - 명령줄에서 직접 아래의 스크립트를 실행합니다. 이 옵션을 사용하면 스크립트 실행 전에 프롬프트가 표시되지 않습니다.

     ```
     psql -h database-hostname -p port -U script_user -d database-name -f
      awsdms_support_collector_postgres.sql
     ```

2. 출력 HTML 파일을 검토하고, 공유하기에 부적합한 정보는 모두 제거합니다. HTML을 공유해도 괜찮다면 AWS Support 사례에 파일을 업로드합니다. 이 파일 업로드에 대한 자세한 내용은 AWS DMS에서 진단 지원 스크립트 작업 섹션을 참조하세요.

AWS DMS 진단 지원 AMI 작업

AWS DMS로 작업할 때 네트워크 관련 문제가 발생할 경우, 지원 엔지니어가 네트워크 구성에 대한 추가 정보를 필요로 할 수 있습니다. AWS Support는 가능한 한 가장 빠른 시간 내에 필요한 정보를 최대한 많이 확보해야 합니다. 이에 따라 AWS DMS 네트워킹 환경을 테스트하기 위한 진단 도구를 갖춘 사전 구축된 Amazon EC2 AMI를 개발했습니다.

Amazon Machine Image(AMI)에 설치된 진단 테스트에는 다음 사항이 포함됩니다.

- Virtual Private Cloud(VPC)
- 네트워크 패킷 손실
- 네트워크 지연 시간
- 최대 전송 단위(MTU) 크기

주제

- 새 AWS DMS 진단 Amazon EC2 인스턴스 시작
- IAM 역할 생성
- 진단 테스트 실행
- 다음 단계
- 리전별 AMI ID

> **Note**
>
> Oracle 소스에서 성능 문제가 발생할 경우, Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능을 평가하여 성능을 개선할 방법을 찾을 수 있습니다. 자세한 내용은 Oracle 재실행 로그 또는 아카이브 로그의 읽기 성능 평가 섹션을 참조하세요.

새 AWS DMS 진단 Amazon EC2 인스턴스 시작

이 섹션에서는 새 Amazon EC2 인스턴스를 시작합니다. Amazon EC2 인스턴스를 시작하는 방법에 대한 내용은 Amazon EC2 사용 설명서의 Amazon EC2 Linux 인스턴스 시작 섹션을 참조하세요.

아래의 설정을 사용하여 Amazon EC2 인스턴스를 시작합니다.

- 애플리케이션 및 OS 이미지(Amazon Machine Image)의 경우, DMS-DIAG-AMI를 검색합니다. 콘솔에 로그온한 경우 이 쿼리로 AMI를 검색할 수 있습니다. 해당 리전의 AWS 진단 AMI의 AMI ID는 아래에서 리전별 AMI ID 섹션을 참조하세요.
- 인스턴스 유형으로는 t2.micro를 선택하는 것을 권장합니다.
- 네트워크 설정에서 복제 인스턴스가 사용하는 것과 동일한 VPC를 선택합니다.

인스턴스가 활성화되면 인스턴스에 연결합니다. Amazon EC2 Linux 인스턴스에 연결하는 방법에 대한 내용은 Linux 인스턴스에 연결합니다 섹션을 참조하세요.

IAM 역할 생성

필요한 최소한의 권한을 사용하여 복제 인스턴스에서 진단 테스트를 실행하려면 다음과 같은 권한 정책을 사용하는 IAM 역할을 생성합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "dms:DescribeEndpoints",
                "dms:DescribeTableStatistics",
                "dms:DescribeReplicationInstances",
                "dms:DescribeReplicationTasks",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": "*"
        }
    ]
}
```

새 IAM 사용자에게 역할을 연결합니다. IAM 역할, 정책, 사용자 생성에 대한 자세한 내용은 IAM 사용 설명서의 다음 섹션을 참조하세요.

- IAM 시작하기
- IAM 역할 생성
- IAM 정책 생성

진단 테스트 실행

Amazon EC2 인스턴스를 생성하고 연결한 후, 다음을 수행하여 복제 인스턴스에서 진단 테스트를 실행합니다.

1. AWS CLI를 구성합니다.

   ```
   $ aws configure
   ```

   진단 테스트를 실행하는 데 사용할 AWS 사용자 계정의 액세스 보안 인증을 제공합니다. VPC 및 복제 인스턴스의 리전을 입력합니다.

2. 해당 리전에서 사용 가능한 AWS DMS 태스크를 표시합니다. 샘플 리전을 현재 리전으로 바꿉니다.

   ```
   $ dms-report -r us-east-1 -l
   ```

이 명령은 사용자의 태스크 상태를 표시합니다.

3. 태스크 엔드포인트 및 설정을 표시합니다. *<DMS-Task-ARN>*을 Amazon 리소스 이름(ARN) 태스크로 바꿉니다.

```
$ dms-report -t <DMS-Task-ARN>
```

이 명령은 태스크의 엔드포인트와 설정을 표시합니다.

4. 진단 테스트를 실행합니다. *<DMS-Task-ARN>*을 태스크 ARN으로 바꿉니다.

```
$ dms-report -t <DMS-Task-ARN> -n y
```

이 명령은 복제 인스턴스의 VPC, 네트워크 패킷 전송, 네트워크 지연 시간, 네트워크 최대 전송 단위(MTU) 크기에 대한 진단 데이터를 표시합니다.

```
########
#
#
#       AWS DMS Diagnostic
#       Date: 07-13-2022
#
#
#       aws region: us-east-2
#
#
########
==== DMS DIAG Info ====
Public IP: 3.
Private IP: 172.30.0.240
Instance ID: i-04829b2beb8214602
Instance MAC: 02:58:04:b5:52:28
Instane Type: t2.micro
Instance Sec Group: D
Instance AWS Region: us-east-2
Instance VPC Id: vpc-08ba020355d8a952e


==== Network Packet Check ====
1.) Check DMS EC2 MetaData service
>>>>>Result: 10 packets transmitted, 10 packets received, 0% packet loss
     Looks good with no issue. <<<<<

2.) Check Source endpoint (d
>>>>>Result: 10 packets transmitted, 10 packets received, 0% packet loss
     Looks good with no issue. <<<<<

3.) Check Target endpoint (r
>>>>>Result: 10 packets transmitted, 10 packets received, 0% packet loss
     Looks good with no issue. <<<<<

==== End network packet check ====


==== Network Latency Check ====
1.) Check DMS MetaData Service
>>>>>Result: round-trip min/avg/max = 0.4/0.4/0.5 ms
     Looks good with no issue. <<<<<

2.) Check Source endpoint (d
>>>>>Result: round-trip min/avg/max = 1.0/1.1/1.2 ms
     Looks good with no issue. <<<<<

3.) Check Target endpoint (r
>>>>>Result: round-trip min/avg/max = 1.4/1.4/1.5 ms
     Looks good with no issue. <<<<<

==== End network latency check ====


==== Network MTU Check ====
1.) Check DMS MetaData Service
>>>>>Result: MTU setting looks good. Local MTU (9001) matches remote MTU (9001) <<<<<


2.) Check Source endpoint (d
>>>>>Result: MTU setting looks good. Local MTU (9001) matches remote MTU (9001) <<<<<


3.) Check Target endpoint (r
>>>>>Result: MTU setting looks good. Local MTU (9001) matches remote MTU (9001) <<<<<


==== End network MTU check ====
```

다음 단계

다음 섹션에서는 네트워크 진단 테스트 결과에 기반한 문제 해결 정보를 설명합니다.

VPC 테스트

이 테스트는 진단 Amazon EC2 인스턴스가 복제 인스턴스와 동일한 VPC에 있는지 확인합니다. 진단 Amazon EC2 인스턴스가 복제 인스턴스와 동일한 VPC에 있지 않을 경우, 해당 인스턴스를 종료하고 올바른 VPC에서 다시 생성합니다. 생성한 후에는 Amazon EC2 인스턴스의 VPC를 변경할 수 없습니다.

네트워크 패킷 손실 테스트

이 테스트는 다음과 같은 엔드포인트로 패킷 10개를 전송한 후 패킷 손실 여부를 확인합니다.

- 포트 80의 AWS DMS Amazon EC2 메타데이터 서비스
- 소스 엔드포인트
- 대상 엔드포인트

모든 패킷이 도착해야 합니다. 패킷이 손실된 경우, 네트워크 엔지니어에게 문의하여 문제를 확인하고 해결 방법을 찾습니다.

네트워크 지연 시간 테스트

이 테스트는 이전 테스트와 동일한 엔드포인트로 패킷 10개를 전송한 후 패킷 지연 시간이 발생했는지 확인합니다. 모든 패킷의 지연 시간은 100밀리초 미만이어야 합니다. 패킷에 100밀리초 이상의 지연 시간이 있을 경우, 네트워크 엔지니어에게 문의하여 문제를 확인하고 해결 방법을 찾습니다.

최대 전송 단위(MTU) 크기 테스트

이 테스트는 이전 테스트와 동일한 엔드포인트에서 경로 추적 도구를 사용하여 MTU 크기를 감지합니다. 테스트의 모든 패킷은 MTU 크기가 동일해야 합니다. MTU 크기가 다른 패킷이 있을 경우, 시스템 전문가에게 문의하여 문제를 확인하고 해결 방법을 찾습니다.

리전별 AMI ID

아래 표에는 해당 리전의 AMS 진단 AMI에 대한 AMI ID가 나와 있습니다.

리전	이미지 ID
미국 동부(오하이오)	ami-0f9c6801e8cab9399
미국 서부(오레곤)	ami-01ceeab889d966e16
미국 동부(버지니아 북부)	ami-0617285c8257e8f6e
미국 서부(캘리포니아 북부)	ami-0983b989d0a31ed55
캐나다(중부)	ami-0ba9380b30eee288e
유럽(아일랜드)	ami-05959fe4fcc448f01
유럽(런던)	ami-0de91dfa18b365c04
유럽(파리)	ami-0844ecd26f2cc9073
유럽(프랑크푸르트)	ami-09bc69e6da9f3adf4
유럽(스톡홀름)	ami-01da340b09bf5ab78
유럽(밀라노)	ami-0dcfc51c17cfe7727
유럽(스페인)	ami-0461c0b2e391c1280
아시아 태평양(싱가포르)	ami-04150ffe00fe300d6
아시아 태평양(시드니)	ami-049c1d1a9d72bafce
아시아 태평양(자카르타)	ami-02189a4b8d3b2ee23
아시아 태평양(홍콩)	ami-0543ae06eff17cb07
아시아 태평양(도쿄)	ami-07b91e8e25d7b7890
아시아 태평양(서울)	ami-0657ad8cb9b5cf71c
아시아 태평양(오사카)	ami-01ccf3e974933afdc
아시아 태평양(뭄바이)	ami-0d9fa24bbb8225e60

리전	이미지 ID
중동(UAE)	ami-0e3dcf51dfe58929b
중동(바레인)	ami-0faad6f64101607f5
남아메리카(상파울루)	ami-097dd866777097d96

AWS DMS 참조

이 참조 섹션에서 데이터 형식 변환 정보 등 AWS Database Migration Service(AWS DMS)를 사용할 때 필요한 추가 정보를 찾을 수 있습니다.

AWS DMS는 소스 및 대상 모두에서 동일한 엔진 유형을 사용하는 동종 데이터베이스 마이그레이션을 수행할 때 데이터 유형을 그대로 유지합니다. 한 가지 데이터베이스 엔진 유형에서 다른 데이터베이스 엔진으로 마이그레이션하는 이종 마이그레이션을 수행할 때는 데이터 형식이 중간 데이터 형식으로 변환됩니다. 데이터 형식이 대상 데이터베이스에서 표시되는 방식은 원본 및 대상 데이터베이스 엔진의 데이터 형식 테이블을 참조하세요.

데이터베이스를 마이그레이션할 때 데이터 형식에 대해 몇 가지 중요한 사실을 알아 두세요.

- FLOAT 데이터 형식은 본래 근사치입니다. FLOAT에 특정한 값을 삽입하면 데이터베이스에서 다르게 나타날 수 있습니다. FLOAT는 NUMBER, NUMBER(p,s) 등의 10진수 데이터 형식과 같이 정확한 데이터 형식이 아니므로 이 차이가 생깁니다. 따라서 데이터베이스에 저장된 FLOAT의 내부 값이 삽입한 값과 다를 수 있습니다. 따라서 FLOAT의 마이그레이션된 값이 원본 데이터베이스의 값과 정확히 일치하지 않을 수 있습니다.

  이 문제에 대한 자세한 내용은 다음 문서를 참조하세요.

  - Wikipedia의 IEEE 부동 소수점
  - Microsoft Learn의 IEEE 부동 소수점 표현
  - Microsoft Learn의 부동 소수점 숫자의 정밀도가 떨어지는 이유

주제

- AWS Database Migration Service에서 사용되는 데이터 형식

AWS Database Migration Service에서 사용되는 데이터 형식

AWS Database Migration Service는 내장 데이터 형식을 사용하여 소스 데이터베이스 엔진 유형에서 대상 데이터베이스 엔진 유형으로 데이터를 마이그레이션합니다. 다음 테이블에는 내장 데이터 형식과 그에 대한 설명이 나와 있습니다.

AWS DMS 데이터 형식	설명
STRING	문자열.

AWS DMS 데이터 형식	설명
WSTRING	2바이트 문자열.
BOOLEAN	부울 값.
BYTE	이진 데이터 값.
DATE	날짜 값: 연도, 월, 일.
TIME	시간 값: 시간, 분, 초.
DATETIME	타임스탬프 값: 연도, 월, 일, 시간, 분, 초, 소수부 초. 소수부 초의 최대 크기는 9자리입니다. 지원되는 형식은 다음과 같습니다. YYYY:MM:DD HH:MM:SS.F(9). Amazon S3 Select 및 Amazon S3 Glacier Select에서는 DATETIME 데이터 형식이 다릅니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 지원되는 데이터 형식에서 `timestamp` 기본 데이터 형식에 대한 설명을 참조하세요.
INT1	1바이트, 부호 있는 정수.
INT2	2바이트, 부호 있는 정수.
INT4	4바이트, 부호 있는 정수.
INT8	8바이트, 부호 있는 정수.
NUMERIC	고정 정밀도 및 크기를 사용한 정확한 숫자 값.
REAL4	단일 정밀도 부동 소수점 값.
REAL8	배정밀도 부동 소수점 값.
UINT1	1바이트, 부호 없는 정수.
UINT2	2바이트, 부호 없는 정수.
UINT4	4바이트, 부호 없는 정수.

AWS DMS 데이터 형식	설명
UINT8	8바이트, 부호 없는 정수.
BLOB	이진 대용량 객체.
CLOB	문자 대용량 객체.
NCLOB	고유 문자 대용량 객체.

> **Note**
>
> AWS DMS는 어떤 LOB 데이터 형식도 Apache Kafka 엔드포인트로 마이그레이션 할 수 없습니다.

AWS DMS 릴리스 노트

다음에서 현재 및 이전 버전의 AWS Database Migration Service (AWS DMS) 에 대한 릴리스 노트를 확인할 수 있습니다.

AWS DMS 복제 인스턴스의 자동 버전 업그레이드를 활성화하면 메이저 버전과 마이너 버전을 구분하지 않습니다. 버전이 더 이상 사용되지 않는 경우 DMS는 유지 관리 기간 동안 복제 인스턴스의 버전을 자동으로 업그레이드합니다.

참고로 API 또는 CLI를 사용하여 복제 인스턴스의 버전을 버전 3.4.x에서 3.5.x로 수동으로 업그레이드하려면 파라미터를 로 설정해야 합니다. `AllowMajorVersionUpgrade` `true` `AllowMajorVersionUpgrade`파라미터에 대한 자세한 내용은 DMS API 설명서를 참조하십시오 ModifyReplicationInstance.

> **Note**
>
> 의 현재 기본 엔진 버전은 AWS DMS 3.5.1입니다.

AWS Database Migration Service 3.5.2 릴리스 노트

3.5.2의 새로운 기능 AWS DMS

새로운 기능 또는 향상된 기능	설명
Redshift 데이터 검증	AWS DMS 이제 Redshift 타겟의 데이터 검증을 지원합니다.
Microsoft SQL Server 버전 2022를 소스와 대상으로 지원.	AWS DMS 이제 Microsoft SQL Server 버전 2022를 소스 및 타겟으로 사용할 수 있습니다.
IBM Db2 LUW를 대상으로 사용	AWS DMS 이제 IBM Db2 LUW를 타겟으로 지원합니다. 를 사용하여 AWS DMS이제 IBM Db2 LUW에서 IBM Db2 LUW로 라이브 마이그레이션을 수행할 수 있습니다.

AWS Database Migration Service 3.5.1 릴리스 노트

다음 표에서는 AWS Database Migration Service (AWS DMS) 버전 3.5.1에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
PostgreSQL 15.x 지원	AWS DMS 버전 3.5.1은 PostgreSQL 버전 15.x를 지원합니다. 자세한 내용은 PostgreSQL을 소스로 사용 및 PostgreSQL을 대상으로 사용 섹션을 참조하세요.
Amazon DocumentDB Elastic Clusters(샤딩된 컬렉션 포함) 지원	AWS DMS 버전 3.5.1은 샤딩된 컬렉션이 있는 Amazon DocumentDB 엘라스틱 클러스터를 지원합니다. 자세한 정보는 Amazon DocumentDB를 AWS Database Migration Service의 대상으로 사용을 참조하세요.
Redshift Serverless 대상	Amazon Redshift를 대상으로 사용하도록 지원합니다. 자세한 정보는 AWS Database Migration Service의 대상으로 Amazon Redshift 데이터베이스 사용을 참조하세요.
Babelfish 엔드포인트 설정	Babelfish 지원을 제공하기 위한 PostgreSQL 대상 엔드포인트 설정을 개선했습니다. 자세한 정보는 AWS Database Migration Service에서 PostgreSQL 데이터베이스를 대상으로 사용을 참조하세요.
Oracle 소스 미결 트랜잭션	AWS DMS 3.5.1은 Oracle 소스의 시작 위치에서 CDC 전용 작업을 시작할 때 미결 트랜잭션을 처리하는 방법을 개선합니다. 자세한 내용은 Oracle을 소스로 사용할 때의 엔드포인트 설정 AWS DMS 섹션의 `OpenTransactionWindow` 섹션을 참조하세요.
타겟으로서의 Amazon Timestream	Amazon Timestream을 대상 엔드포인트로 사용할 수 있도록 지원합니다. 자세한 정보는 Amazon Timestream을 AWS Database Migration Service의 대상으로 사용을 참조하세요.

AWS DMS 버전 3.5.1에는 다음과 같은 해결된 문제가 포함되어 있습니다.

해결된 문제	설명
비활성 세션이 늘어나는 소스로서의 Oracle	CDC 전용 작업의 비활성 세션이 계속 증가하여 다음 예외가 발생하는 Oracle 소스 문제를 수정했습니다. `ORA-00020: maximum number of processes exceeded on the source database`
업데이트 변경 내용을 DocumentDB에 복제하는 중	일부 시나리오에서 UPDATE 문이 제대로 복제되지 않는 DocumentDB 대상 문제를 수정했습니다.
검증 전용 작업	검증 전용 작업에 대한 데이터 유효성 검사가 비활성화된 경우 작업이 제대로 실패하도록 데이터 유효성 검사 기능에 대한 오류 처리 기능이 개선되었습니다.
연결 종료 후 Redshift 복제	연결 종료 후 대상이 0보다 크게 `ParallelApplyThreads` 설정된 경우 DMS 작업이 대상에 변경 내용을 다시 적용하려고 시도하지 않아 데이터가 손실되는 Redshift 대상 문제를 수정했습니다.
MySQL 텍스트를 중간 텍스트로 복제	전체 LOB 모드에서 중간 텍스트 데이터 유형을 MySQL에서 MySQL로 복제할 때 발생하는 문제를 수정했습니다.
암호가 회전된 경우 CDC 작업이 복제되지 않음	Secrets Manager가 암호를 교체한 후 DMS에서 데이터 복제를 중지하는 `true` 위치로 `BatchApplyEnabled` 설정된 DMS 작업 관련 문제를 수정했습니다.
몽고DB/문서DB 세그멘테이션 문제	MongoDB/DocDB 소스에서 기본 키 열에 큰 값이 포함된 경우 범위 분할이 제대로 작동하지 않는 문제를 수정했습니다.
바인딩되지 않은 숫자 값에 대한 Oracle 데이터 검증	데이터 검증 중에 DMS가 바인딩되지 않은 데이터 유형의 값을 `NUMERIC` a로 인식하는 Oracle 대상 문제를 수정했습니다. `STRING`
SQL Server 데이터 유효성 검사	DMS 데이터 유효성 검사가 잘못된 SQL 문을 생성하는 SQL Server 엔드포인트 문제를 수정했습니다.
MongoDB 자동 세그먼트화	MongoDB 대상에서 문서를 병렬로 마이그레이션할 때 데이터 자동 파티셔닝 기능을 개선했습니다.

해결된 문제	설명
Amazon S3 Apache Parquet 형식	S3 대상에 쓴 Apache Parquet 파일을 Apache Arrow C++가 설치된 Python을 사용하여 볼 수 있도록 문제를 수정했습니다.
PostgreSQL 소스 DDL 처리	PostgreSQL 소스에서 지원되지 않는 DDL 작업이 올바로 무시되지 않는 문제를 수정했습니다.
PostgreSQL `timestamptz` 데이터 오류	PostgreSQL에서 PostgreSQL로 마이그레이션할 때 CDC 중에 일괄 적용이 활성화된 상태에서 시간대 데이터가 포함된 타임스탬프가 제대로 마이그레이션되지 않는 문제를 수정했습니다.
Oracle에서 PostgreSQL로의 유효성 검사 실패	Oracle에서 PostgreSQL로 마이그레이션할 때 NUMERIC(38,30) 데이터 형식에 대한 데이터 유효성 검사가 실패하는 문제를 수정했습니다.
Oracle 확장 데이터 형식 오류	Oracle 소스에서 확장된 varchar 데이터 형식이 잘리는 문제를 수정했습니다.
필터 연산자 결합	열 필터링 기능에서 null 열 연산자를 다른 유형의 연산자와 결합할 수 없는 문제를 수정했습니다.
과도한 로깅으로 인한 CDC 지연.	PostgreSQL 소스에서 pglogical 플러그인 경고가 과도하게 로깅되어 소스 CDC 지연이 발생하는 문제를 수정했습니다.
Create Table DDL의 양방향 복제 처리	PostgreSQL에서 PostgreSQL로의 양방향 복제 시 Create Table DDL 변경 사항이 제대로 복제되지 않는 문제를 수정했습니다.
필터 사용 중 CDC 오류	필터링 기능에서 CDC 복제가 실패하는 문제를 수정했습니다.
Kafka 엔드포인트에 대한 인증 기관 호스트 이름 유효성 검사	인증 기관의 호스트 이름 유효성 검사를 비활성화하는 옵션(`SslEndpointIdentificationAlgorithm`)을 추가하여 Kafka 엔드포인트의 기능을 개선했습니다.
IBM Db2 LUW 유효성 검사	데이터 유효성 검사 중에 Db2 LUW 소스 날짜, 타임스탬프 및 시간 데이터 형식이 제대로 처리되지 않는 문제를 수정했습니다.

해결된 문제	설명
S3 유효성 검사	Db2 LUW에서 S3로의 마이그레이션 시 유효성 검사 기능이 timestamp(0) 데이터 유식을 제대로 처리하지 못하는 문제를 수정했습니다.
DMS 태스크 재시작 실패	pglogical 플러그인을 사용할 때 작업이 다시 시작되지 않고 관계형 AWS DMS 이벤트를 사용할 수 없는 PostgreSQL 소스 문제를 수정했습니다.
HIERARCHY 데이터 형식에 대한 SQL Server 유효성 검사	SQL Server 소스에서 HIERARCHY 데이터 형식 유효성 검사가 실패하는 문제를 수정했습니다.
제어 문자가 포함된 SQL Server 문자열	SQL Server 소스에서 제어 문자가 포함된 문자열이 제대로 복제되지 않는 문제를 수정했습니다.
Redshift 및 Secrets Manager	Redshift 대상에서 Secrets Manager를 사용할 때 엔드포인트 테스트가 실패하는 문제를 수정했습니다.
MySQL 설정 ParallelLoadThreads 불일치	MySQL 대상에서 태스크 설정이 변경된 후 `ParallelLoadThreads` 설정이 제대로 유지되지 않는 문제를 수정했습니다.
PostgreSQL에서 Oracle로 데이터 형식 매핑 중 오류	PostgreSQL에서 Oracle로의 마이그레이션에서 데이터 형식 TEXT에서 데이터 형식 VARCHAR2(2000)로 복제할 때 태스크가 실패하는 문제를 수정했습니다.
Oracle에서 PostgreSQL로의 데이터 유효성 검사	Oracle에서 PostgreSQL로의 마이그레이션에서 NULL 문자가 공백 문자로 복제될 때 데이터 유효성 검사가 거짓 긍정을 보고하는 문제를 수정했습니다.
구성상의 SQL 서버 소스 AlwaysOn	AlwaysOn 구성에서 복제본 이름이 실제 서버 이름과 정확히 일치하지 않을 경우 AWS DMS 작업이 실패하는 SQL Server 원본 관련 문제를 수정했습니다.
Oracle 소스 엔드포인트 테스트 실패	Oracle SID (세션 ID) 를 검색하는 동안 권한이 충분하지 않아 AWS DMS 엔드포인트 연결 테스트가 실패하는 Oracle 소스 관련 문제를 수정했습니다.

해결된 문제	설명
CDC가 새 테이블을 픽업하지 않음	CDC 전용 태스크에서 태스크가 시작된 후 소스에서 생성된 테이블이 일부 경우에서 복제되지 않는 문제를 수정했습니다.
Oracle을 소스로 사용 시 미결 트랜잭션	Oracle 소스의 시작 위치에서 CDC 전용 태스크를 시작할 때 미결 트랜잭션을 처리하는 방법을 개선했습니다.
누락 데이터 문제	캐시된 변경 사항을 적용한 후(`StopTaskCachedChangesApplied` 옵션이 true로 설정됨) 태스크를 중지하면 태스크를 재개할 때 데이터가 누락되는 문제를 수정했습니다. 원본에 많은 양의 변경 내용이 있기 때문에 AWS DMS 복제 인스턴스 디스크에 캐시된 변경 내용이 AWS DMS 지속되는 경우 이 문제가 드물게 발생할 수 있습니다.
확장 데이터 형식에 대한 데이터 유효성 검사 문제	PostgreSQL에서 Oracle로의 데이터 유효성 검사에서 확장된 데이터 형식에 대한 유효성 검사가 실패하는 문제를 수정했습니다.
일치하지 않은 문자 인코딩에 대한 데이터 유효성 검사 문제	소스와 대상 간에 문자 인코딩이 일치하지 않을 때 SQL Server에서 PostgreSQL로의 데이터 유효성 검사에 실패하는 문제를 수정했습니다.
데이터 유효성 검사 문제 ORA-01455	PostgreSQL `integer`이 Oracle `number(10)` 에 매핑될 때 유효성 검사 도중 ORA-01455 오류가 발생하는 문제를 수정했습니다.
SQL Server `IDENTITY` 지원	SQL Server에서 SQL Server로의 데이터 복제에서 대상 열에 `IDENTITY` 속성이 있는 경우 ID 열 마이그레이션이 실패하는 문제를 수정했습니다.
ALTER 문의 문자 세트 문제	MySQL에서 MySQL로의 MySQL 복제 시 CDC 중에 명령문을 마이그레이션할 때 문자 집합이 UTF16 으로 AWS DMS 변경되는 문제를 수정했습니다. `ALTER`
PostgreSQL에서 Redshift로의 공간 데이터 형식 지원	PostgreSQL에서 Amazon Redshift로 마이그레이션할 때 `spatial` 데이터 형식에 대한 지원을 추가했습니다.
.parquet 파일의 GZIP 압축	S3를 대상으로 하는 GZIP 압축을 사용하여.parquet 파일을 AWS DMS 생성하지 못하는 문제를 수정했습니다.

해결된 문제	설명
MongoDB/DocDB 소스 마이그레이션	MongoDB 소스에서 일부 파티션을 마이그레이션하지 AWS DMS 않는 문제가 해결되었습니다.
테이블 통계 문제	복제 인스턴스의 태스크 중 하나 이상에 1,001개가 넘는 테이블이 포함된 경우 테이블 통계가 표시되지 않는 문제를 수정했습니다.
IBM Db2 LUW 버전 10.1.0 이하에서 테이블 일시 중지	소스 데이터베이스 버전이 10.1.0 이하일 때 `TYPESTRINGUNITS is not valid` 오류와 함께 테이블 마이그레이션이 일시 중단되는 Db2 LUW 소스 문제를 수정했습니다.
MongoDB 파티셔닝 문제	MongoDB/DocDB에서 소스 파티션의 세그먼트가 하나 이상 누락되는 문제를 수정했습니다.
MongoDB 파티셔닝 문제	유형 변환 버그로 인해 NumberLong () 유형의 열을 기반으로 한 분할이 실패하는 문제를 수정했습니다.
MongoDB 파티셔닝 문제	MongoDB를 소스로 사용할 때 대규모 데이터 세트에 대한 자동 세그먼트화 성능을 개선했습니다.
MongoDB 드라이버 버전	MongoDB 버전 3.6 이하를 계속 지원하기 위해 MongoDB 드라이버를 1.20.0으로 다운그레이드했습니다.
Amazon S3 Apache Parquet 타임스탬프 데이터 형식	Amazon S3 파켓 대상 문제를 수정했습니다. AWS DMS 이제 형식 파라미터를 이전 버전의 AWS DMS동작과 `isAdjustedToUTC true` 일치하도록 설정합니다.
Amazon Redshift 대상 복사 명령	Amazon Redshift를 대상으로 사용할 때 Amazon S3에서 Amazon Redshift로 데이터를 복사하는 경우 대용량 테이블에 대해 복사 명령이 실패하는 문제를 수정했습니다.
PostgreSQL 지오메트리 데이터 형식	PostgreSQL에서 PostgreSQL로의 마이그레이션에서 대규모 지오메트리 데이터 형식에서 마이그레이션이 실패하는 문제를 수정했습니다.
Oracle에서 PostgreSQL로의 XML	Oracle에서 PostgreSQL로 복제할 때 마이그레이션으로 인해 XML에 공백 하나가 추가되는 문제를 해결했습니다.

해결된 문제	설명
지원되는 엔진에서 대상 체크포인트 업데이트	AWS DMS 이제 대상 데이터베이스의 `awsdms_txn_state` 테이블에서 대상 체크포인트를 업데이트합니다.
MongoDB/DocDB 레코드가 잘못된 컬렉션으로 전송됨	MongoDB/DocDB에서 데이터가 잘못된 대상 컬렉션으로 전송되는 문제를 수정했습니다.
Oracle 소스 새 테이블 선택 (EscapeCharacter 엔드포인트 설정 포함)	`EscapeCharacter` 엔드포인트 설정이 설정된 상태에서 작업을 중지했다가 다시 시작할 때 Oracle 소스에서 새 테이블만 복제를 위해 픽업하던 문제를 수정했습니다. AWS DMS
CDC 복구 체크포인트	대상 데이터 스토어와 AWS DMS 콘솔 간에 관찰된 CDC 복구 체크포인트의 불일치를 수정했습니다.
CDC 검증 전용 작업	작업의 모든 테이블에 오류가 발생해도 작업이 실패하지 않는 CDC 검증 전용 작업 관련 문제를 해결했습니다.
소스 또는 대상 연결 문제의 유효성 검사 동작	연결이 AWS DMS 끊길 때 원본 또는 대상의 테이블이 일시 중단되는 데이터 검증 관련 문제를 수정했습니다.
Oracle에서 PostgreSQL로의 데이터 유효성 검사 거짓 긍정	Oracle에서 PostgreSQL로 데이터를 검증할 때 오탐이 보고되는 문제를 수정했습니다. AWS DMS 이 문제는 대상의 소스 NULL 문자 표현의 차이가 VARCHAR 이외의 텍스트 기반 데이터 형식에서 반영되지 않기 때문에 발생합니다.
Oracle에서 PostgreSQL로 마이그레이션 시 데이터 잘림	Oracle을 소스로, PostgreSQL을 대상으로 사용할 때 Oracle `NLS_NCHAR_CHARACTERSET` 설정이 `AL16UTF16` 으로 설정된 상태에서 AWS DMS 가 NVARCHAR 열의 데이터를 자르는 문제를 수정했습니다.
데이터 유효성 검사 오류	소스 필터링과 열 추가 변환 규칙을 모두 사용할 때 `unable to create where filter clause` 오류가 발생하는 데이터 유효성 검사 문제를 수정했습니다.
Redshift 대상 오류 처리	Redshift를 대상으로 사용할 때 CDC 태스크에서 `ParallelApplyThreads` 태스크 설정이 0보다 큰 값으로 설정되면 오류 처리가 구성된 대로 작동하지 않는 문제를 수정했습니다.

해결된 문제	설명
Oracle 소스 통신 실패	Oracle을 소스로 사용할 때 태스크가 RUNNING 상태로 유지되지만 통신 실패 후 데이터를 마이그레이션할 수 없는 문제를 수정했습니다.
열 필터로 인한 CDC 테이블 일시 중단	열 필터가 적용된 CDC 단계에서 테이블이 일시 중단되는 전체 로드 + CDC 태스크 관련 문제를 수정했습니다.
S3 대상의 특수 문자에 대한 데이터 유효성 검사 실패	테이블 이름에 밑줄 이외의 특수 문자가 포함된 경우 태스크가 실패하는 S3 대상 데이터 유효성 검사 문제를 수정했습니다.
MongoDB 소스 전체 로드 및 CDC 실패	MongoDB를 소스로 사용할 때 대규모 컬렉션을 마이그레이션하면서 캐시 이벤트를 처리하는 동안 전체 로드 + CDC 태스크가 실패하는 문제를 수정했습니다.
BatchApplyEnabled 를 true로 설정 시 업그레이드 문제	작업 설정이 true로 설정된 작업이 AWS DMS 버전 3.4.6에서 3.5.1로 마이그레이션한 후 경우에 따라 실패하는 문제가 수정되었습니다. BatchApplyEnabled
대소문자를 구분하는 데이터 정렬을 사용하는 SQL Server 소스 AlwaysOn	SQL Server를 원본으로 AlwaysOn 사용할 때 대소문자를 구분하여 정렬이 실패하는 문제를 수정했습니다.
MySQL 소스 태스크 중단	MySQL을 소스로 사용할 때 소스가 제대로 구성되지 않으면 태스크가 실패하는 대신 중단되는 문제를 수정했습니다.
S3 소스 전체 로드 태스크 실패	AWS DMS 버전 3.4.6 또는 3.4.7에서 버전 3.5.1로 업그레이드한 후 작업을 재개할 때 S3를 소스로 사용할 때 작업이 실패하는 문제를 수정했습니다.
CaptureDDLs가 false로 설정된 PostgreSQL 소스	PostgreSQL을 소스로 사용할 때 CaptureDDLs 엔드포인트 설정이 false로 설정되면 DDL이 제대로 처리되지 않는 문제를 수정했습니다.
재개 시 Oracle 소스 태스크 충돌	Oracle을 소스로 사용할 때 재개 시 열 이름의 잘못된 데이터로 인해 태스크가 중단되는 문제를 수정했습니다.

해결된 문제	설명
MySQL 소스 LOB 조회 실패	MySQL을 소스로 사용할 때 `ParallelApplyThreads` 태스크 설정이 0보다 큰 값으로 설정되면 LOB 조회가 실패하는 문제를 수정했습니다.
SQL Server 소스 비논리적 LSN 오류	버전 3.4.7에서 AWS DMS 버전 3.5.1로 업그레이드한 후 `illogical LSN sequencing state error` 오류가 발생하여 작업이 실패하는 SQL Server 소스 문제를 수정했습니다.
pglogical을 사용하는 PostgreSQL 소스	PostgreSQL을 소스로 사용할 때 태스크를 중지하고, 테이블을 선택 규칙에서 제거하고, 태스크를 재개하고, 제거된 테이블을 변경하면 pglogical 플러그인을 사용하는 태스크가 실패하는 문제를 수정했습니다.
Aurora MySQL의 잘못된 복구 체크포인트	Aurora MySQL을 소스로 사용할 때 Aurora 장애 조치 또는 Aurora 소스 중지 및 시작의 결과로 잘못된 복구 체크포인트가 저장되는 문제를 해결했습니다.
SQL Server를 소스로 사용 시 작업 중단	SQL Server를 소스로 사용할 때 `SafeguardPolicy` 가 `RELY_ON_SQL_SERVER_REPLICATION_AGENT` 로 설정되면 작업이 중단되는 문제를 수정했습니다.
MySQL을 대상으로 사용 시 잘못된 데이터 유형 캐스팅	MySQL을 대상으로 사용할 때 배치 적용 단계에서 잘못된 데이터 유형 변환으로 인해 CDC 복제가 실패하는 문제를 해결했습니다.
PostgreSQL을 소스로 사용 시 CaptureDDL이 false로 설정되면 작업 실패	PostgreSQL을 소스로 사용할 때 `CaptureDDLs` 엔드포인트 설정이 `false`로 설정된 경우 DDL이 DML로 처리되어 작업이 실패하는 문제를 해결했습니다.
MongoDB 빈 컬렉션 중단	MongoDB를 소스로 사용할 때 빈 컬렉션으로 인해 작업이 중단되는 문제를 해결했습니다.
Redshift를 대상으로 사용 시 전체 로드 작업 중단	Redshift를 대상으로 사용할 때 복구 체크포인트 컨트롤 테이블이 활성화된 경우 전체 로드 단계에서 작업이 중단되는 문제를 해결했습니다.
S3에서 S3로 데이터 이동 없음	S3에서 S3로의 복제 시 데이터가 지정되지 않은 경우 데이터가 복제되지 않는 AWS DMS 문제를 수정했습니다`bucketFolder` .

해결된 문제	설명
GlueCatalogGeneration 이 true로 설정 시 CDC 지연 시간	S3를 대상으로 사용할 때 GlueCatalogGeneration 가 true로 설정된 경우 과도한 지연 시간이 발생하는 문제를 해결했습니다.
Oracle을 대상으로 사용 시 데이터 잘림	Oracle을 대상으로 사용할 때 VARCHAR2 열의 데이터가 AWS DMS 잘리는 문제를 수정했습니다.
PostgreSQL 언더스코어 와일드카드 동작	PostgreSQL을 소스로 사용할 때 선택 규칙의 '_' 와일드카드 동작이 문서에 설명된 대로 작동하지 않는 문제를 해결했습니다.
PostgreSQL을 소스로 사용 시 빈 WAL 헤더 문제	PostgreSQL을 소스로 사용할 때 복제 슬롯에서 수신한 빈 WAL 헤더로 인해 작업이 실패하는 문제를 해결했습니다.
MySQL 또는 MariaDB를 소스로 사용 시 압축된 바이너리 로그 관련 문제	MySQL 및 MariaDB 소스에서 BINLOG 압축이 감지되었을 때 적절한 오류 메시지가 표시되지 않는 문제를 수정했습니다. AWS DMS
S3 데이터 검증 특수 문자	프라이머리 키와 비프라이머리 키 열의 특수 문자를 처리하도록 S3 데이터 검증을 개선했습니다.
Redshift를 대상으로 사용 시 잘못된 작업 로그 항목	Redshift를 대상으로 사용할 때 UPDATES 및 DELETES에서 batch-apply 문 실패를 보고하는 잘못된 항목이 작업 로그에 표시되는 문제를 해결했습니다.
SQL Server에서 S3로의 마이그레이션 작업 중단	SQL Server에서 S3로 마이그레이션할 때 캐시된 변경 사항을 적용하는 동안 작업이 중단되는 문제를 해결했습니다.
일괄 적용 오류 시 데이터 누락	배치 적용 시 오류가 발생하면 데이터가 누락되는 배치 적용 기능 관련 문제를 해결했습니다.

AWS Database Migration Service 3.5.0 베타 릴리스 노트

> **Important**
>
> AWS DMS 3.5.0은 복제 인스턴스 엔진의 베타 버전입니다. AWS DMS 모든 이전 릴리스와 동일하게 이 버전을 지원합니다. 하지만 AWS DMS 3.5.0 베타를 프로덕션 용도로 사용하기 전에 먼저 테스트해 보는 것이 좋습니다.

다음 표에는 AWS Database Migration Service (AWS DMS) 버전 3.5.0 베타에 도입된 새로운 기능 및 개선 사항이 나와 있습니다.

새로운 기능 또는 향상된 기능	설명
Oracle 및 Microsoft SQL Server용 Time Travel	이제 DMS가 지원하는 오라클, Microsoft SQL Server 및 PostgreSQL 소스 엔드포인트와 DMS가 지원하는 PostgreSQL 및 MySQL 대상 엔드포인트가 있는 모든 AWS 지역에서 타임 트래블을 사용할 수 있습니다.
S3 유효성 검사	AWS DMS 이제 Amazon S3 대상 엔드포인트에서 복제된 데이터의 검증을 지원합니다. Amazon S3 대상 데이터 유효성 검사에 대한 자세한 내용은 Amazon S3 대상 데이터 검증 섹션을 참조하세요.
Glue Catalog 통합	AWS Glue 는 데이터를 분류하는 간단한 방법을 제공하는 서비스로서, 로 알려진 메타데이터 리포지토리로 구성되어 있습니다. AWS Glue Data Catalog이제 를 Amazon S3 대상 AWS Glue Data Catalog 엔드포인트와 통합하고 Amazon Athena와 같은 다른 AWS 서비스를 통해 Amazon S3 데이터를 쿼리할 수 있습니다. 자세한 정보는 AWS DMS의 Amazon S3 대상과 함께 AWS Glue Data Catalog을 사용을 참조하세요.
DocumentDB 대상에 대한 병렬 적용	`ParallelApply*` 새 작업 설정의 대상으로 DocumentDB를 사용하면 이제 AWS DMS CDC 복제 중에 초당 최대 5000개의 레코드를 지원합니다. 자세한 정보는 Amazon DocumentDB를 AWS Database Migration Service의 대상으로 사용을 참조하세요.

새로운 기능 또는 향상된 기능	설명
고객 중심 로깅	이제 버전 3.5.0에서 작업 로그를 보다 효과적으로 검사하고 관리할 수 있습니다 AWS DMS . AWS DMS 작업 로그 보기 및 관리에 대한 자세한 내용은 을 참조하십시오. AWS DMS 태스크 로그 보기 및 관리
Kafka 대상 엔드포인트에 대한 SASL_PLAIN 메커니즘	이제 SASL_PLAIN 인증을 사용하여 Kafka MSK 대상 엔드포인트를 지원할 수 있습니다.
MySQL에서 XA 트랜잭션 복제	이제 MySQL DMS 소스에서 XA 트랜잭션을 사용할 수 있습니다. DMS 3.5.0 이전에는 XA 트랜잭션의 일부로 적용된 DML 변경 사항이 제대로 복제되지 않았습니다.
Oracle 확장 데이터 형식	AWS DMS 이제 Oracle 버전 12.2 이상에서 확장된 데이터 유형의 복제가 지원됩니다.
Db2 LUW 환경 PureScale	AWS DMS 이제 Db2 LUW 환경에서의 복제가 지원됩니다. PureScale 이 기능은 소스 변경 위치에서 변경 사항 처리 시작 옵션을 사용해야만 지원됩니다.
`READ_COMMITTED_SNAPSHOT` 옵션이 설정된 SQL Server 소스	`READ_COMMITTED_SNAPSHOT` 옵션이 로 설정된 Microsoft SQL Server 원본 데이터베이스를 사용하는 경우 forceDataRow조회 연결 속성을 설정하여 TRUE DML 변경 내용을 올바르게 복제할 수 있습니다.

AWS DMS 3.5.0에는 다음과 같은 해결된 문제가 포함되어 있습니다.

AWS DMS 3.5.0에서 해결된 문제는 2023년 3월 17일에 시작되었습니다.

주제	해결 방법	
Oracle - 숫자에서 변환된 문자열의 특별 사례 비교	Oracle 소스에서 문자열로의 데이터 형식 변환이 있는 숫자 열에 대해 필터링 규칙이 예상대로 작동하지 않는 문제를 수정했습니다.	

주제	해결 방법
온프레미스 SQL Server AG의 향상된 기능	DMS에서 사용하지 않는 복제본에 대한 불필요한 연결을 제거하여 AlwaysOn구성에서 SQL Server 원본을 사용한 연결 처리의 효율성을 개선했습니다.
SQL Server HIERARCHYID 내부 변환	SQL Server 소스에서 HIERARCHYID 데이터 형식이 SQL Server 대상에 HIERARCHYID가 아닌 VARCHAR(250)로 복제되는 문제를 수정했습니다.
S3 대상 이동 태스크 수정	S3 대상으로의 이동 태스크가 시간이 매우 오래 걸리거나 중단된 것처럼 보이거나 완료되지 않는 문제를 수정했습니다.
Kafka의 SASL Plain 메커니즘	Kafka MSK 대상 엔드포인트에 대한 SASL Plain 인증 방법 지원을 도입했습니다.
Opensearch 2.x의 _type 파라미터로 인해 병렬 로드 및 병렬 적용 실패	Opensearch 2.x 대상에서 _type 파라미터가 지원되지 않아 병렬 로드 또는 병렬 적용이 실패하는 문제를 수정했습니다.
혼합 연산자를 사용하는 테이블 매핑 필터 지원	한 열에 하나의 필터만 적용할 수 있는 제한을 제거했습니다.
S3, Kinesis, Kafka 엔드포인트 - CDC 단계의 변경 기반 로그 열 마이그레이션	Kinesis, Kafka 및 S3 대상에서 CDC 중에 추가된 LOB 열의 데이터가 복제되지 않는 문제를 수정했습니다.
MongoDB 드라이버 업그레이드	MongoDB 드라이버를 v1.23.2로 업그레이드했습니다.
Kafka 드라이버 업데이트	Kafka 드라이버를 1.5.3에서 1.9.2로 업그레이드했습니다.

주제	해결 방법	
S3 엔드포인트 설정이 제대로 작동하지 않음	S3 대상에서 데이터에 S3 대상용 구분 기호로 지정된 문자가 포함된 경우 `AddTrailingPaddingCharacter` 엔드포인트 설정이 작동하지 않는 문제를 수정했습니다.	
Kinesis 대상 태스크가 충돌함	Kinesis 대상에서 PK 값이 비어 있고 상세 디버그가 활성화되어 있을 때 태스크가 중단되는 문제를 수정했습니다.	
S3 대상의 열 이름이 한 자리씩 이동된 경우	S3 대상에서 `AddColumnName` 이 `true`로 설정되고 `TimestampColumnName` 이 `""`로 설정되어 있을 때 열 이름이 한 자리씩 이동하는 문제를 수정했습니다.	
로깅 LOB 자르기 경고 개선	LOB를 검색하는 데 사용되는 select 문을 포함하도록 SQL Server 소스의 LOB 자르기에 대한 경고 로깅을 개선했습니다.	
TDE 암호가 잘못된 경우 DMS 태스크 충돌이 발생하지 않도록 치명적 오류 추가.	Oracle 소스에서 잘못된 TDE 암호로 인한 오류 메시지 없이 DMS 태스크가 실패하는 상황에서 의미 있는 오류 메시지를 도입하고 태스크 충돌 문제를 없앴습니다.	
CDC 중에 PostgreSQL CTAS(create table as selected) DDL의 마이그레이션 허용.	CDC 중에 DMS에서 PostgreSQL CTAS(create table as selected) DDL을 복제할 수 없는 제한을 제거했습니다.	
CDC에서 테이블 열이 삭제될 때 발생하는 pg_logical 태스크 충돌을 수정했습니다.	S3 대상을 사용하는 PostgreSQL 소스에서 LOB 지원이 비활성화되었지만 LOB가 있을 때 대상에서 열이 잘못 정렬되는 문제를 수정했습니다.	
MySQL 연결 처리에서 메모리 누수 수정	MySQL 소스에서 태스크 메모리 사용량이 지속적으로 증가하는 문제를 수정했습니다.	

주제	해결 방법	
Oracle 소스 엔드포인트 설정 - `ConvertTimestampWithZoneToUTC`	'TIMESTAMP WITH TIME ZONE' 및 'TIMESTAMP WITH LOCAL TIME ZONE' 열의 타임스탬프 값을 UTC로 변환하려면 이 속성을 `true`로 설정합니다. 기본적으로 이 속성의 값은 'false'이며, 데이터는 소스 데이터베이스 시간대를 사용하여 복제됩니다.	
Oracle 소스 - SUSPEND_TABLE을 위한 `DataTruncationErrorPolicy` 가 작동하지 않음	S3 대상을 사용하는 Oracle 소스에서 `DataTruncationErrorPolicy` 태스크 설정이 SUSPEND_TABLE로 설정되어 있는 동안 테이블이 일시 중단되지 않는 문제를 수정했습니다.	
쿼리 절을 작성하는 동안 긴 스키마 및 테이블에서 SQL Server 실패	SQL Server 소스에서 선택 규칙에 쉼표로 구분된 테이블 목록이 포함되면 태스크가 실패하거나 응답하지 않는 문제를 수정했습니다.	
MongoDB 엔드포인트를 사용하는 Secret Manager 인증	MongoDB 및 DocumentDB 엔드포인트에서 Secret Manager 기반 인증이 작동하지 않는 문제를 수정했습니다.	
NLS_NCHAR_CHARACTERSET이 UTF8로 설정된 경우 DMS가 CDC 중에 멀티바이트 varchar 열의 데이터를 자름	Oracle 대상을 사용하는 Oracle 소스에서 NLS_NCHAR_CHARACTERSET이 UTF8로 설정된 멀티바이트 VARCHAR 열의 데이터가 잘리는 문제를 수정했습니다.	
`filterTransactionsOfUser` 오라클용 ECA LogMiner	를 사용하여 Oracle에서 복제할 때 DMS가 지정된 사용자의 트랜잭션을 `filterTransactionsOfUser` 무시할 수 있도록 하는 추가 연결 속성 (ECA) 이 추가되었습니다. LogMiner	

주제	해결 방법	
백업에서 lsn이 누락된 경우 SQL Server가 복구 가능 오류 설정	SQL Server에서 LSN이 누락되어도 태스크가 실패하지 않는 문제를 수정했습니다.	

AWS Database Migration Service 3.4.7 릴리스 노트

다음 표는 AWS Database Migration Service (AWS DMS) 버전 3.4.7에 도입된 새로운 기능 및 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
Support를 대상으로 지원	AWS DMS 이제 Babelfish를 대상으로 지원합니다. 를 사용하면 AWS DMS이제 가동 중지 시간을 최소화하면서 AWS DMS 지원되는 모든 소스에서 Babelfish로 라이브 데이터를 마이그레이션할 수 있습니다. 자세한 정보는 Babelfish를 AWS Database Migration Service 대상으로 사용을 참조하세요.
전체 로드 전용 소스로서 IBM Db2 z/OS 데이터베이스 지원	AWS DMS 이제 IBM Db2 z/OS 데이터베이스를 소스로 지원합니다. 를 사용하면 AWS DMS이제 Db2 메인프레임에서 지원되는 모든 타겟으로 라이브 마이그레이션을 수행할 수 있습니다. AWS DMS 자세한 정보는 IBM Db2 for z/OS 데이터베이스를 AWS DMS 소스로 사용을 참조하세요.
SQL Server 읽기 전용 복제본을 소스로 지원	AWS DMS 이제 SQL Server 읽기 전용 복제본을 원본으로 지원합니다. 를 사용하면 AWS DMS이제 SQL Server 읽기 전용 복제본에서 지원되는 모든 AWS DMS 타겟으로 실시간 마이그레이션을 수행할 수 있습니다. 자세한 정보는 Microsoft SQL Server 데이터베이스를 원본으로 사용 AWS DMS을 참조하세요.

새로운 기능 또는 향상된 기능	설명
EventBridge DMS 이벤트 지원	AWS DMS DMS 이벤트를 사용하여 이벤트 구독을 관리할 EventBridge 수 있습니다. 자세한 정보는 AWS Database Migration Service에서 Amazon EventBridge 이벤트 및 알림 사용을 참조하세요.
VPC 소스 및 대상 엔드포인트 지원	AWS DMS 이제 Amazon VPC (가상 사설 클라우드) 엔드포인트를 소스 및 타겟으로 지원합니다. AWS DMS 이제 AWS 서비스에 대한 명시적으로 정의된 경로가 VPC에 정의되어 있으면 VPC 엔드포인트가 있는 모든 서비스에 연결할 수 있습니다. AWS DMS **Note** AWS DMS 버전 3.4.7 이상으로 업그레이드하려면 먼저 VPC 엔드포인트를 사용하거나 공개 경로를 사용하도록 AWS DMS 구성해야 합니다. 이 요구 사항은 Amazon S3, Amazon Kinesis Data Streams, Amazon DynamoDB AWS Secrets Manager, Amazon Redshift 및 Amazon 서비스의 소스 및 대상 엔드포인트에 적용됩니다. OpenSearch 자세한 정보는 VPC 엔드포인트를 AWS DMS 소스 및 대상 엔드포인트로 구성을 참조하세요.
새 PostgreSQL 버전	PostgreSQL 버전 14.x가 이제 소스 및 대상으로 지원됩니다.
Aurora Serverless v2를 대상으로 지원	AWS DMS 이제 Aurora 서버리스 v2를 타겟으로 지원합니다. 를 사용하여 AWS DMS이제 Aurora 서버리스 v2로 라이브 마이그레이션을 수행할 수 있습니다. 지원되는 AWS DMS 대상에 대한 자세한 내용은 을 참조하십시오. 마이그레이션에 적합한 대상

새로운 기능 또는 향상된 기능	설명
새 IBM Db2 for LUW 버전	AWS DMS 이제 LUW용 IBM Db2 버전 11.5.6 및 11.5.7을 소스로 지원합니다. 를 사용하여 이제 AWS DMS LUW용 IBM DB2의 최신 버전에서 라이브 마이그레이션을 수행할 수 있습니다. AWS DMS 소스에 대한 자세한 내용은 을 참조하십시오. 데이터 마이그레이션용 소스 지원되는 AWS DMS 대상에 대한 자세한 내용은 을 참조하십시오마이그레이션에 적합한 대상.

AWS DMS 3.4.7에는 다음과 같은 신규 또는 변경된 동작과 해결된 문제가 포함됩니다.

- 이제 Amazon S3를 소스로 사용할 때 테이블 정의의 날짜 형식을 사용하여 데이터 문자열을 날짜 객체로 파싱할 수 있습니다.
- 이제 새 테이블 통계 카운터를 사용할 수 있습니다. `AppliedInserts`, `AppliedDdls`, `AppliedDeletes`, `AppliedUpdates`.
- 이제 OpenSearch 타겟으로 사용할 때 기본 매핑 유형을 선택할 수 있습니다.
- Oracle, PostgreSQL 및 SQLServer 소스에 대한 새로운 `TrimSpaceInChar` 엔드포인트 설정을 통해 CHAR 및 NCHAR 데이터 형식의 데이터를 트리밍할지 여부를 지정할 수 있습니다.
- Amazon S3에 대한 새로운 `ExpectedBucket0wner` 엔드포인트 설정이 S3를 소스 또는 대상으로 사용할 때 스나이핑을 방지합니다.
- RDS SQL Server, Azure SQL Server 및 자체 관리형 SQL Server의 경우 - 이제 DMS는 프라이머리 키가 있거나 없는 마이그레이션 태스크에 선택된 모든 테이블에서 또는 프라이머리 키가 있는 자체 관리형 SQL Server 테이블에서 MS-REPLICATION의 활성화 우선 순위를 고려하는 고유 인덱스를 사용하여 MS-CDC 자동 설정을 제공합니다.
- Oracle 동종 마이그레이션 중에 Oracle 파티션 및 하위 파티션 DDL 작업의 복제에 대한 지원을 추가했습니다.
- Oracle을 소스 및 대상으로 사용할 때 복합 기본 키로 인해 데이터 유효성 검사 태스크가 충돌하는 문제를 수정했습니다.
- Redshift를 대상으로 사용할 때 대상 열이 부울로 미리 생성된 상태에서 CHARACTER VARYING 형식을 부울로 올바르게 캐스팅하는 문제를 수정했습니다.

- PostgreSQL을 대상으로 사용할 때 알려진 ODBC 문제로 인해 마이그레이션된 varchar 데이터 형식의 데이터가 잘리는 문제를 수정했습니다.
- Oracle을 대상으로 사용할 때 BatchApplyEnabled가 true로 설정되고 BatchApplyPreserveTransaction이 false로 설정되면 DELETE 작업에 대한 병렬 힌트가 준수되지 않는 문제를 수정했습니다.
- Amazon S3의 새 AddTrailingPaddingCharacter 엔드포인트 설정은 S3를 대상으로 사용할 때 문자열 데이터에 패딩을 추가합니다.
- 새 max_statement_timeout_seconds 태스크 설정은 엔드포인트 쿼리의 기본 제한 시간을 연장합니다. 이 설정은 현재 MySQL 엔드포인트 메타데이터 쿼리에 사용됩니다.
- PostgreSQL을 대상으로 사용할 때 CDC 태스크가 오류 처리 태스크 설정을 제대로 활용하지 않는 문제를 수정했습니다.
- DMS가 Redis Enterprise 인스턴스의 Redis 모드를 올바르게 식별하지 못하는 문제를 수정했습니다.
- S3 대상 Parquet 형식에 대한 includeOpForFullLoad 추가 연결 속성(ECA) 지원을 확대했습니다.
- 새로운 PostgreSQL 엔드포인트 설정 migrateBooleanAsBoolean을 도입했습니다. PostgreSQL에서 Redshift로 마이그레이션할 때 이 설정을 true로 설정하면 부울이 varchar(1)로 마이그레이션됩니다. false로 설정하면 부울이 varchar(15)로 마이그레이션되며 이는 기본 동작입니다.
- SQL Server 소스를 사용할 때 datetime 데이터 형식과 관련된 마이그레이션 문제를 수정했습니다. 이 수정 사항은 정밀도가 밀리초 단위인 경우 Null 삽입 문제를 해결합니다.
- PGLOGICAL을 사용하는 PostgreSQL 소스의 경우, pglogical을 사용하고 CDC 단계 중에 소스 테이블에서 필드를 제거할 때 제거된 필드 이후의 값이 대상 테이블로 마이그레이션되지 않는 마이그레이션 문제를 수정했습니다.
- 양방향 복제가 반복된 레코드를 가져오는 SQL Server 루프백 마이그레이션 문제를 수정했습니다.
- PostgreSQL 소스를 위한 새 ECA mapBooleanAsBoolean을 추가했습니다. 이 추가 연결 속성을 사용하여 PostgreSQL 부울의 기본 데이터 유형 매핑을 부울 데이터 유형으로 재정의할 수 있습니다. RedShift
- SQL Server를 소스로 사용할 때 ALTER DECIMAL/NUMERIC SCALE이 대상에 복제되지 않는 마이그레이션 문제를 수정했습니다.
- SQL Server 2005와의 연결 문제를 수정했습니다.
- 2022년 10월 17일부터 DMS 3.4.7은 복제 인스턴스에 대한 6세대 Amazon EC2 인스턴스 클래스를 지원합니다.

- 2022년 11월 25일부터 DMS 3.4.7에서 DMS 스키마 변환을 사용하여 데이터베이스 스키마와 코드 객체를 변환하고, DMS Fleet Advisor를 사용하여 네트워크 환경에서 마이그레이션에 적합한 데이터베이스를 검색할 수 있습니다.
- 2022년 11월 25일부터 DMS 스튜디오는 사용 중지됩니다.
- 2023년 1월 31일부터 DMS 스키마 변환은 Aurora MySQL 및 Aurora PostgreSQL을 대상 데이터 공급자로 지원합니다.
- 2023년 3월 6일부터 DMS Fleet Advisor를 사용하여 소스 데이터베이스에 적합한 크기의 대상 권장 사항을 생성할 수 있습니다.
- 2023년 3월 6일부터 Amazon에 지표 데이터 포인트를 게시할 수 있는 AWS 관리형 정책을 AWS DMS 지원합니다 CloudWatch.

2023년 5월 5일에 발표된 DMS 3.4.7 유지 관리 릴리스에서 해결된 문제

주제	해결 방법	
PostgreSQL 소스 태스크 실패	PostgreSQL 소스에서 단일 이벤트에 허용되는 최대 DDL 태스크를 초과할 경우 태스크가 실패하는 문제를 수정했습니다.	
PostgreSQL 소스 데이터 유효성 검사 거짓 긍정	Oracle 대상을 사용하는 PostgreSQL 소스에서 타임스탬프 필드를 잘못 캐스팅하면 거짓 긍정 데이터 유효성 검사 오류가 발생하는 문제를 수정했습니다.	
MySQL 소스 오류 처리	MySQL 소스에서 다음 BIN 로그를 사용할 수 없을 때 DMS 태스크가 실패하지 않는 문제를 수정했습니다.	
MySQL 소스 ROTATE_EVENT 로깅	MySQL 소스에서 ROTATE_EVENT와 관련된 로깅을 개선했습니다. 읽는 BIN 로그 이름을 포함시켰습니다.	
데이터 유효성 검사 시간 초과 문제	데이터 유효성 검사와 관련된 쿼리에 대해 `executeTimeout` 엔드포인트 설정이 준수되지 않는 데이터 유효성 검사 기능 문제를 수정했습니다.	

주제	해결 방법	
PostgreSQL 대상 병렬 전체 로드 문제	PostgreSQL 대상에서 '연결 중단' 오류로 인해 세그먼트 (병렬) 전체 로드가 실패하는 문제를 수정했습니다.	
DMS 태스크 이동 문제	S3 대상에서 DMS 태스크 이동 작업이 매우 오래 걸리거나 완료되지 않는 문제를 수정했습니다.	
PostgreSQL 소스 중복 레코드 문제	PostgreSQL 소스에서 태스크가 중지 후 재개되면 DMS 태스크가 대상에서 중복 관련 오류를 throw 하는 문제를 수정했습니다.	
Oracle 대상 데이터 유효성 검사 거짓 긍정	Oracle 대상에서 타임스탬프 필드의 시간대가 잘못 복제되어 데이터 유효성 검사가 거짓 긍정 오류를 보고하는 문제를 수정했습니다.	

2023년 2월 22일에 발표된 DMS 3.4.7 유지 관리 릴리스에서 해결된 문제

주제	해결 방법	
소스로서의 SQL Server AG 복제본	리스너 TCP 포트가 복제본 TCP 포트와 다른 AlwaysOn구성에서 SQL Server 소스에 대한 지원이 추가되었습니다.	
Amazon Redshift 대상에서 데이터 손실	Redshift 대상에서 드문 경우지만 예상치 못한 Redshift 재시작으로 인해 대상에서 데이터가 누락될 수 있었던 문제를 수정했습니다.	
SQL Server 소스 세이프가드 지원	SQL Server 소스에서 엔드포인트 설정 `"SafeguardPolicy": "EXCLUSIVE_AUTOMATIC_TRUNCATION"` 이 지정되면 DMS 태스크가 실패하고 트랜잭션 로그 백업을 읽을 수 없다는 오류가 발생하는 문제를 수정했습니다.	
Oracle 소스에 대한 데이터 유효성 검사 태스크 실패	Oracle 소스에서 잘못 식별된 프라이머리 키 값으로 인해 DMS 태스크가 데이터 유효성 검사에서 실패할 수 있는 문제를 수정했습니다.	

주제	해결 방법	
Kinesis 이미지 전 데이터 문제	스트리밍 대상(Kinesis, Kafka) 에서 "EnableBeforeImage" 태스크 설정이 문자 데이터 형식에만 작동하는 문제를 수정했습니다.	
Time Travel 로그 파일	Time Travel 기능에서 소스가 유휴 상태일 때 DMS가 0바이트 Time Travel 로그 파일을 생성하는 문제를 수정했습니다.	

2022년 12월 16일에 발표된 DMS 3.4.7 유지 관리 릴리스에서 해결된 문제

주제	해결 방법	
BatchApplyEnabled	True로 설정된 경우 과도한 로깅 문제가 해결되었습니다. BatchApplyEnabled	
새 몽고DB 엔드포인트 설정— FullLoadNoCursorTimeout	MongoDB 엔드포인트 설정은 전체 로드 NoCursorTimeout 커서를 FullLoadNoCursorTimeout 지정합니다. NoCursorTimeout 유휴 상태인 경우 서버가 커서를 닫지 못하도록 하는 MongoDB 연결 설정입니다.	
MongoDB - 단일 열 세그먼트화를 위한 필터 함수	새로운 필터 함수는 세그먼트화에 단일 열을 사용하여 MongoDB 데이터베이스 마이그레이션 성능을 개선합니다.	
MongoDB에서 Redshift로	MongoDB에서 Redshift로 마이그레이션할 때 MongoDB 컬렉션에 이진 데이터 형식이 있는 경우 DMS가 Redshift에서 대상 테이블을 생성하지 않는 문제를 수정했습니다.	
새 몽고DB SocketTimeout MS 연결 속성	새로운 MongoDB SocketTimeout MS 추가 연결 속성은 MongoDB 클라이언트의 연결 제한 시간을 밀리초 단위로 구성합니다. 값이 0보다 작거나 같으면 MongoDB 클라이언트 기본값이 사용됩니다.	
Amazon Kinesis 태스크가 충돌을 일으키는 문제를 수정했습니다.	Amazon Kinesis Data Streams 대상으로 마이그레이션할 때 테이블에 프라이머리 키가 없는 경우 null 값을 처리하는 문제를 수정했습니다.	

주제	해결 방법	
Oracle NULL PK/UK 데이터 유효성 검사 지원	NULL PK/UK 값의 데이터 유효성 검사가 지원되지 않는 제한을 제거했습니다.	
Oracle에서 Amazon S3로	Oracle에서 Amazon S3로 마이그레이션할 때 일부 레코드가 NULL로 잘못 마이그레이션되는 문제를 수정했습니다.	
Oracle Standby	Oracle Standby를 소스로 사용할 때 DMS가 미결 트랜잭션을 처리할 수 있는 기능을 추가했습니다.	
SDO_GEOMETRY 공간 데이터 형식을 사용하는 Oracle에서 Oracle로 마이그레이션	Oracle에서 Oracle로 마이그레이션할 때 테이블의 DDL에 SDO_GEOMERY 열이 있는 경우 태스크가 실패하는 문제를 수정했습니다.	
Oracle 소스	Oracle을 소스로 사용할 때 DMS가 Oracle 다시 실행 로그 시퀀스 번호를 가끔 건너뛰는 문제를 수정했습니다.	
Oracle 소스 - 아카이브/온라인 다시 실행 로그 누락	Oracle을 소스로 사용할 때 아카이브 로그가 누락되면 DMS 태스크가 실패하는 문제를 수정했습니다.	
수정 - DMS가 가끔 Oracle Standby 다시 실행 로그를 건너뛰는 문제	Oracle을 소스로 사용할 때 DMS가 Oracle 다시 실행 로그 시퀀스 번호를 가끔 건너뛰는 문제를 수정했습니다.	
수정 - CDC 중에 Oracle에서 Oracle로 공간 데이터 형식이 복제되지 않는 문제	Oracle에서 Oracle로 복제할 때 CDC 중에 공간 데이터 형식이 복제되지 않는 문제를 수정했습니다.	
Oracle 대상	Oracle을 대상으로 사용할 때 ORA-01747 오류로 인해 대상 적용이 실패하는 문제를 수정했습니다.	

주제	해결 방법
Amazon S3 - 테이블 다시 로드 데이터 손실 문제 수정	Amazon S3를 대상으로 사용할 때 테이블 다시 로드 작업에서 CDC 파일이 생성되지 않는 문제를 수정했습니다.
수정 - 프라이머리 서버를 소스로 사용할 때 SQL Server Always On 컨텍스트 초기화	SQL Server Always On을 원본으로 사용할 때 원본이 기본이고 true로 설정된 경우 가용성 그룹 (AG) 이 초기화되지 않는 문제를 수정했습니다. AlwaysOnSharedSyncedBackupIsEnabled
SQL Server 엔드포인트 설정 업데이트	원본 엔드포인트가 SQL Server Always On 가용성 그룹이고 보조 복제본인 경우 AlwaysOnSharedSynchedBackupsIsEnabled True로 설정하면 복제 작업이 실패하는 문제를 수정했습니다.
PostgreSQL 소스	CDC가 PostgreSQL 소스에서 삭제/업데이트 작업을 마이그레이션하지 못하는 문제를 수정했습니다. 이 기능은 Boolean 지원 시 3.4.7에 도입되었습니다. mapBooleanAs

AWS Database Migration Service 3.4.6 릴리스 노트

다음 표에는 AWS Database Migration Service (AWS DMS) 버전 3.4.6에 도입된 새로운 기능 및 향상된 기능이 나와 있습니다.

새로운 기능 또는 향상된 기능	설명
AWS DMS 시간 여행	AWS DMS 고객에게 로깅 기능에 유연성을 부여하고 문제 해결 경험을 향상시키는 기능인 Time Travel을 소개합니다. Time Travel을 사용하면 Amazon S3를 사용하여 AWS DMS 로그를 저장 및 암호화하고, 특정 기간 내에 로그를 보고, 다운로드하고, 난독화할 수 있습니다.
Microsoft Azure SQL 관리형 인스턴스를 소스로 지원	AWS DMS 이제 Microsoft Azure SQL 관리형 인스턴스를 소스로 지원합니다. 를 사용하면 AWS DMS이제 Microsoft Azure SQL 관리형

새로운 기능 또는 향상된 기능	설명
	인스턴스에서 AWS DMS 지원되는 모든 대상으로 실시간 마이그레이션을 수행할 수 있습니다. AWS DMS 소스에 대한 자세한 내용은 을 참조하십시오. 데이터 마이그레이션용 소스 지원되는 AWS DMS 대상에 대한 자세한 내용은 을 참조하십시오마이그레이션에 적합한 대상.
Google Cloud SQL for MySQL을 소스로 지원	AWS DMS 이제 MySQL용 구글 클라우드 SQL을 소스로 지원합니다. 를 사용하면 AWS DMS이제 MySQL용 Google Cloud SQL에서 지원되는 AWS DMS 모든 대상으로 실시간 마이그레이션을 수행할 수 있습니다. AWS DMS 소스에 대한 자세한 내용은 을 참조하십시오. 데이터 마이그레이션용 소스 지원되는 AWS DMS 대상에 대한 자세한 내용은 을 참조하십시오마이그레이션에 적합한 대상.
파티션된 데이터의 S3로 병렬 로드 지원	AWS DMS 이제 Amazon S3로 분할된 데이터에 대한 병렬 로드를 지원하여 지원되는 데이터베이스 엔진 소스 데이터에서 Amazon S3로 분할된 데이터를 마이그레이션하는 데 드는 로드 시간을 개선합니다. 이 기능은 데이터베이스 소스에 있는 테이블의 각 파티션에 대해 Amazon S3 하위 폴더를 생성하여 AWS DMS 가 병렬 프로세스를 실행하여 각 하위 폴더를 채울 수 있도록 합니다.
단일 태스크에서 여러 Apache Kafka 대상 주제 지원	AWS DMS 이제 단일 작업으로 Apache Kafka 다중 주제 대상을 지원합니다. 이제 AWS DMS를 사용하면 동일한 태스크를 사용하여 단일 데이터베이스의 여러 스키마를 다른 Apache Kafka 대상 주제로 복제할 수 있습니다. 따라서 동일한 소스 데이터베이스의 여러 테이블을 서로 다른 Kafka 대상 주제로 마이그레이션해야 하는 상황에서 여러 태스크를 따로 생성할 필요가 없습니다.

AWS DMS 3.4.6에서 해결된 문제는 다음과 같습니다.

- Amazon S3를 CSV 형식의 대상으로 사용할 때 프라이머리 키 열이 첫 번째 열이 아닌 경우 UPDATE 문의 열이 잘못된 열로 채워지는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 제한된 LOB 모드에서 BYTEA 열의 NULL 값이 있는 pglogical 플러그인을 사용하면 AWS DMS 작업이 충돌하는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 많은 수의 소스 테이블이 삭제되면 AWS DMS 작업이 중단될 수 있는 문제를 수정했습니다.
- UTC가 아닌 날짜에도 파티셔닝을 허용하는 새로운 Amazon S3 설정 `DatePartitionTimezone`을 도입하여 Amazon S3 날짜 기반 폴더 파티셔닝을 개선했습니다.
- Redshift를 대상으로 사용할 때 소스의 `TIMESTAMP WITH TIME ZONE`에서 `TIMESTAMPTZ`로 데이터 형식 간 매핑을 지원했습니다.
- MongoDB 또는 Amazon DocumentDB를 소스로 사용할 때 와일드카드 선택 규칙이 없는 태스크에 대한 CDC의 성능을 개선했습니다.
- Db2 LUW를 소스로 사용할 때 밑줄 와일드카드 및 길이가 8 미만인 스키마 이름이 AWS DMS 태스크에서 캡처되지 않는 문제를 수정했습니다.
- OpenSearch Service를 대상으로 사용할 때 데이터 볼륨이 크면 AWS DMS 인스턴스의 메모리가 부족해지던 문제를 수정했습니다.
- 전체 로드 유효성 검사만 지원하여 데이터 유효성 검사 성능을 개선했습니다.
- Sybase를 소스로 사용할 때 강제 장애 조치 후 AWS DMS 작업이 재개되지 않는 문제를 수정했습니다.
- 경고가 `Invalid BC timestamp was encountered in column` 잘못 AWS DMS 전송되던 문제를 수정했습니다.

DMS 3.4.6 유지 관리 릴리스에서 해결된 문제는 다음과 같습니다.

- Oracle을 소스 및 대상으로 사용할 때 대량 적용 모드를 활성화하면 태스크가 충돌하는 문제를 수정했습니다.
- PostgreSQL이 소스일 때 전체 로드 태스크가 `ExecuteTimeout` 엔드포인트 설정을 제대로 사용하도록 문제를 수정했습니다.
- PostgreSQL을 소스로 사용하는 동안 태스크가 제한적 LOB 모드로 설정되면 Array 데이터 형식 열을 마이그레이션할 때 발생하는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 1970년 1월 1일 이전의 시간대를 사용하는 타임스탬프를 마이그레이션할 때 발생하는 문제를 수정했습니다.
- SQL Server를 소스 및 대상으로 사용할 때 DMS가 복제 중에 빈 문자열을 null로 처리하는 문제를 수정했습니다.

- MySQL 소스 및 타겟을 사용할 때 세션 읽기 및 쓰기 제한 시간 엔드포인트 설정을 준수하도록 문제를 수정했습니다.
- Amazon S3를 소스로 사용할 때 DMS CDC 태스크가 전체 로드 관련 파일을 다운로드하는 문제를 수정했습니다.
- Amazon S3를 대상으로 사용할 때 `CdcInsertsAndUpdates` 및 `PreserveTransactions` 둘 다 `true`로 설정된 경우 로그 충돌 문제를 수정했습니다.
- ParallelApply* 기능을 활성화하면 작업이 중단되지만 Amazon Kinesis Data Streams를 소스로 사용할 때 일부 테이블에 기본 기본 키가 없는 문제가 해결되었습니다.
- Amazon Kinesis Data Streams를 소스로 사용할 StreamArn 때 잘못된 항목으로 오류가 발생하지 않던 문제를 수정했습니다.
- 기본 키 값을 빈 문자열로 지정하면 대상으로 사용할 OpenSearch 때 작업이 중단되는 문제를 수정했습니다.
- 데이터 유효성 검사에 너무 많은 디스크 공간이 사용되는 문제를 수정했습니다.

2022년 12월 13일에 발표된 DMS 3.4.6 유지 관리 릴리스에서 해결된 문제

주제	해결 방법	
SAP ASE odbc 드라이버	SAP ASE를 소스로 사용할 때 ODBC 드라이버가 문자 세트를 지원할 수 있도록 문제를 수정했습니다.	
로그 조회를 위한 Sqlserver datetime 프라이머리 키 버그	SQL Server 소스에서 프라이머리 키에 밀리초 단위 정밀도의 datetime 데이터 형식이 있으면 LOB 조회가 제대로 작동하지 않는 문제를 수정했습니다.	
SQL Server에서 Redshift로 - 'datetimeoffset'를 'timestamptz'로 매핑	SQL Server에서 Redshift로 마이그레이션할 때 SQL Server의 'datetimeoffset' 형식이 Redshift 'timestamptz' 형식에 매핑되도록 매핑을 개선했습니다.	
데이터 검증 - SkipLobColumns True	SkipLobColumns True이고, 소스에 LOB가 있고, 기본 키가 마지막 열에 있고, 검증 결과 데이터 차이가 감지되면 DMS 작업이 충돌하는 문제를 수정했습니다.	

주제	해결 방법	
MySQL 소스에서의 데이터 유효성 검사	데이터 유효성 검사가 활성화된 MySQL 소스에서 null 값이 포함된 복합 고유 키가 있는 테이블을 사용하면 DMS 태스크 충돌이 발생하는 문제를 수정했습니다.	
MySQL 소스	MySQL을 소스로 사용할 때 정밀도를 높이기 위해 열을 변경하면 오버플로 오류와 함께 테이블이 일시 중단되는 문제를 수정했습니다.	
MySQL ODBC 드라이버를 8.0.23으로 업그레이드	MySQL을 소스로 사용할 때 데이터 정렬 'utf8mb4_0900_bin'이 DMS에서 사용하는 mysql 드라이버와 호환되지 않는 문제를 수정했습니다.	
MySQL - 파티션된 테이블에 대한 DDL 변경 지원	CDC 중에 파티션 DDL skipTableSuspensionForPartitionDdl 변경으로 인한 테이블 일시 중지를 건너뛸 수 있도록 하는 새로운 MySQL 엔드포인트 설정을 도입하여 이제 DMS에서 분할된 MySQL 테이블의 DDL 변경을 지원할 수 있게 되었습니다.	
MongoDB에서 Redshift로 마이그레이션	MongoDB에서 Redshift로 마이그레이션할 때 MongoDB 컬렉션에 이진 데이터 형식이 있으면 DMS가 Redshift에서 대상 테이블을 생성하지 못하는 문제를 수정했습니다.	
Redshift 대상 - 일괄 적용에서 Time Travel Segfault	Redshift를 대상으로 사용할 때 DMS 태스크가 true로 설정되면 충돌이 발생하는 문제를 수정했습니다. BatchApplyEnabled	
Redshift 대상	Redshift를 대상으로 사용할 때 parallel-load가 type=partitions-auto로 설정되면 병렬 세그먼트가 동일한 테이블 디렉터리에 대량 CSV 파일을 쓰면서 서로 간섭하는 문제를 수정했습니다.	
Redshift 대상	Redshift를 대상으로 사용할 때 CDC 중에 대상 열은 부울 형식이고 소스는 CHARACTER VARYING 형식인 문제를 수정했습니다.	
Redshift 대상	Redshift 대상으로 복제되지 않는 DDL 변경 사항을 식별하도록 태스크 로그를 개선했습니다.	

주제	해결 방법	
PostgreSQL의 데이터 유효성 검사	PostgreSQL의 유효성 검사에서 부울 데이터 형식이 있으면 유효성 검사가 실패하는 문제를 수정했습니다.	
PostgreSQL 소스	PostgreSQL을 소스로 사용하는 ExecuteTimeout 문제를 수정하여 전체 로드 시 추가 연결 속성의 필드를 사용하도록 했습니다.	
PostgreSQL 소스	PostgreSQL을 소스로 사용할 때 요청된 태스크 재개 LSN보다 큰 LSN을 읽으면 60분 이상 동안 태스크가 실패하여 사용 중인 복제 슬롯에 문제가 있음을 나타내도록 문제를 수정했습니다.	
PostgreSQL 소스 - 1970년 1월 1일 이전의 timestamptz	PostgreSQL을 소스로 사용할 때 CDC 중에 1970년 1월 1일 이전의 timestamptz가 올바르게 마이그레이션되지 않는 문제를 수정했습니다.	
PostgreSQL 소스	PostgreSQL을 소스로 사용할 때 CDC 중에 DMS가 CHARACTER VARYING 데이터 형식 값을 자르는 문제를 수정했습니다.	
PostgreSQL 소스 - 중지된 태스크 재개	PostgreSQL 소스에서 이전에 중지된 태스크 재생을 재개할 때 CDC 중에 하나 이상의 트랜잭션이 누락되는 문제를 수정했습니다.	
Amazon S3 대상	S3를 대상으로 사용할 때, AddColumnName true이고 “" 인 경우 결과 CSV 파일 헤더가 한 열 차이로 표시되는 문제를 수정했습니다. TimestampColumnName	
Amazon S3 소스 - 태스크의 전체 로드 단계에서 메모리 사용 동작	S3를 소스로 사용할 때 전체 로드 DMS 태스크가 전체 테이블을 대상 데이터베이스에 로드한 후에만 사용된 메모리를 해제하는 문제를 수정했습니다.	
Amazon S3 대상 - 테이블 다시 로드 작업	S3를 대상으로 사용할 때 테이블 다시 로드 작업에서 CDC 파일이 생성되지 않는 문제를 수정했습니다.	

AWS 데이터베이스 마이그레이션 서비스 3.4.5 릴리스 노트

다음 표는 AWS Database Migration Service (AWS DMS) 버전 3.4.5에 도입된 새로운 기능 및 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
Redis를 대상으로 지원	AWS DMS 이제 Redis를 타겟으로 지원합니다. 를 사용하면 AWS DMS AWS DMS 지원되는 모든 소스의 라이브 데이터를 가동 중지 시간을 최소화하면서 Redis 데이터 스토어로 마이그레이션할 수 있습니다. AWS DMS 대상에 대한 자세한 내용은 을 참조하십시오. 마이그레이션에 적합한 대상
MongoDB 4.2 및 4.4를 소스로 지원	AWS DMS 이제 MongoDB 4.2 및 4.4를 소스로 지원합니다. 를 사용하면 AWS DMS이제 가동 중지 시간을 최소화하면서 MongoDB 4.2 및 4.4 클러스터에서 Amazon DocumentDB (MongoDB 호환) 를 비롯한 AWS DMS 지원되는 모든 대상으로 데이터를 마이그레이션할 수 있습니다. 소스에 대한 자세한 내용은 을 참조하십시오. AWS DMS 데이터 마이그레이션용 소스
MongoDB를 소스로 사용 시 여러 데이터베이스 지원	AWS DMS 이제 MongoDB를 소스로 사용하여 한 번의 작업으로 여러 데이터베이스를 마이그레이션할 수 있습니다. 를 사용하면 AWS DMS이제 MongoDB 클러스터의 여러 데이터베이스를 그룹화하고 한 번의 데이터베이스 마이그레이션 작업으로 마이그레이션할 수 있습니다. 가동 중지 시간을 최소화하면서 Amazon DocumentDB (MongoDB와 호환 가능) 를 포함하여 AWS DMS 지원되는 모든 대상으로 마이그레이션할 수 있습니다.
MongoDB 또는 Amazon DocumentDB(MongoDB 호환)를 소스로 사용하는 자동 세그먼트화 지원	AWS DMS 이제 MongoDB 또는 Amazon DocumentDB를 소스로 사용하는 자동 세그멘테이션을 지원합니다. 를 사용하면 AWS DMS MongoDB 또는 DocumentDB 클러스터의 컬렉션을 자동으로 분할하도록 데이터베이스 마이그레이션 작업을 구성할 수 있습니다. 그러면 가동 중지 시간을 최소화하면서 Amazon DocumentDB를 비롯한 AWS DMS 지원되는 모든 대상으로 세그먼트를 병렬로 마이그레이션할 수 있습니다.

새로운 기능 또는 향상된 기능	설명
Amazon Redshift 전체 로드 성능 개선	AWS DMS 이제 전체 로드 중에 Amazon Redshift를 타겟으로 사용할 때 병렬 스레드 사용을 지원합니다. 멀티스레드 전체 로드 작업 설정을 활용하면 AWS DMS 지원되는 모든 소스에서 Amazon Redshift로의 초기 마이그레이션 성능을 개선할 수 있습니다. 대상에 대한 AWS DMS 자세한 내용은 을 참조하십시오. 마이그레이션에 적합한 대상

AWS DMS 3.4.5에서 해결된 문제는 다음과 같습니다.

- PostgreSQL을 트랜잭션 동시성이 높은 소스로 사용할 때 재개 후 데이터가 누락되거나 중복될 수 있는 문제를 수정했습니다.
- pglogical 플러그인이 활성화된 상태에서 PostgreSQL을 소스로 사용할 때 관계 ID를 찾을 수 없습니다...라는 오류와 함께 데이터베이스 마이그레이션 태스크가 실패하는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용하고 Oracle을 대상으로 사용할 때 VARCHAR 열이 제대로 복제되지 않는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 프라이머리 키가 테이블 정의의 첫 번째 열이 아닌 경우 삭제 작업이 제대로 캡처되지 않는 문제를 수정했습니다.
- MySQL을 소스로 사용할 때 데이터베이스 마이그레이션 태스크가 특수 메타데이터 설정에서 LOB 업데이트를 누락하는 문제를 수정했습니다.
- MySQL 버전 8을 소스로 사용할 때 전체 LOB 모드에서 TIMESTAMP 열이 DATETIME으로 처리되는 문제를 수정했습니다.
- MySQL 5.6.4 이상을 소스로 사용할 때 NULL DATETIME 레코드를 파싱하면 데이터베이스 마이그레이션 태스크가 실패하는 문제를 수정했습니다.
- Amazon Redshift를 병렬 적용의 대상으로 사용할 때 스레드 종료 중 오류가 발생한 후 데이터베이스 마이그레이션 태스크가 중단되는 문제를 수정했습니다.
- CDC 일괄 적용 중에 데이터베이스 마이그레이션 태스크와 Amazon Redshift 대상 엔드포인트의 연결이 끊길 때 데이터가 손실될 수 있는 문제를 수정했습니다.
- Amazon Redshift를 대상으로 사용할 때 ACCEPTINVCHARS를 호출하여 전체 로드 성능을 개선했습니다.
- Amazon Redshift를 대상으로 one-by-one 사용하여 모드에서 병렬 적용 모드로 되돌릴 때 중복된 레코드가 복제되는 문제를 수정했습니다.

- Amazon S3를 대상으로 사용할 때 데이터베이스 마이그레이션 태스크가 Amazon S3 객체 소유권을 cannedAclForObjects=bucket_owner_full_control을 보유한 버킷 소유자로 전환하지 않는 문제를 수정했습니다.
- Oracle을 소스로 사용할 additionalArchivedLogDestId 때 ECA로 여러 아카이브 대상을 AWS DMS 지원하여 개선되었습니다.
- 전체 LOB 모드에서 LOB 열을 업데이트하는 동안 데이터베이스 마이그레이션 태스크가 OCI_INVALID_HANDLE 오류와 함께 실패하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 CDC 중에 NVARCHAR2 열이 제대로 마이그레이션되지 않는 문제를 수정했습니다.
- SQL Server용 RDS를 소스로 사용할 SafeguardPolicy 때 AWS DMS 활성화하여 개선되었습니다.
- 비 RDS SQL Server 소스를 사용할 때 데이터베이스 마이그레이션 태스크가 rdsadmin에 대한 오류를 보고하는 문제를 수정했습니다.
- SQL Server를 소스로 사용할 때 파티션 설정에서 UUID를 프라이머리 키로 사용하면 데이터 유효성 검사가 실패하는 문제를 수정했습니다.
- Db2 LUW를 소스로 사용할 때 데이터베이스 로그에서 필요한 LSN을 찾을 수 없으면 전체 로드 및 CDC 태스크가 실패할 수 있는 문제를 수정했습니다.
- MongoDB를 AWS DMS 소스로 사용할 때 사용자 지정 CDC 타임스탬프를 지원하여 개선되었습니다.
- MongoDB를 소스로 사용할 때 endSessions에서 MongoDB 드라이버 오류가 발생하면 데이터베이스 마이그레이션 태스크가 중단되는 문제를 수정했습니다.
- DynamoDB를 AWS DMS 대상으로 사용할 때 기본 필드가 아닌 필드를 업데이트하지 못하는 문제가 해결되었습니다.
- 데이터 유효성 검사가 CLOB 및 NCLOB 열에 대해 거짓 긍정 불일치를 보고하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 공백 전용 레코드에서 데이터 유효성 검사가 실패하는 문제를 수정했습니다.
- 파티션된 테이블을 자를 때 데이터베이스 마이그레이션 태스크가 충돌하는 문제를 수정했습니다.
- awsdms_apply_exceptions 제어 테이블을 생성할 때 데이터베이스 마이그레이션 태스크가 실패하는 문제를 수정했습니다.
- MySQL 버전 8을 사용할 때 caching_sha2_password 인증 플러그인 지원을 확대했습니다.

AWS Database Migration Service 3.4.4 릴리스 노트

다음 표에서는 AWS DMS 버전 3.4.4에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
Kafka를 대상으로 사용할 때 TLS 암호화 및 TLS 또는 SASL 인증 지원	AWS DMS 이제 Amazon MSK 및 온프레미스 Kafka 클러스터를 대상으로 사용하는 TLS 암호화와 TLS 또는 SASL 인증을 지원합니다. Kafka 엔드포인트에서 암호화 및 인증 사용에 대한 자세한 내용은 전송 계층 보안(TLS)을 사용하여 Kafka에 연결 섹션을 참조하세요.

3.4.4에서 해결된 문제는 다음과 같습니다. AWS DMS

- Oracle 엔드포인트를 사용할 때 작업 실패에 대한 AWS DMS 로그온을 개선했습니다.
- Oracle Data Guard 페일오버 이후 Oracle 소스 엔드포인트가 역할을 전환할 때 AWS DMS 작업 실행이 계속 처리되도록 개선되었습니다.
- Oracle 엔드포인트를 사용할 때 오류 처리가 ORA—12561을 복구 가능한 오류로 취급하도록 개선했습니다.
- Oracle을 소스로 사용할 때 `EMPTY_BLOB()` 및 `EMPTY_CLOB()` 열이 null로 마이그레이션되는 문제를 수정했습니다.
- SQL Server를 원본으로 사용할 때 열 DDL 변경 추가 후 AWS DMS 작업이 레코드를 업데이트하지 못하는 문제를 수정했습니다.
- `TIMESTAMP WITH TIME ZONE` 데이터 형식을 지원하여 PostgreSQL 소스 마이그레이션을 개선했습니다.
- PostgreSQL을 대상으로 사용할 때 전체 로드 중에 `afterConnectScript` 설정이 작동하지 않는 문제를 수정했습니다.
- PostgreSQL 엔드포인트를 사용할 때 정밀도 및 소수점 자릿수가 없는 `NUMERIC` 날짜 형식을 더 잘 처리할 수 있는 새 `mapUnboundedNumericAsString` 설정을 도입했습니다.
- PostgreSQL을 소스로 사용할 때 AWS DMS 작업을 중지했다가 다시 시작한 후 “0개의 행이 영향을 받음”으로 인해 작업이 실패하는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 `BC` 접미사가 있는 `TIMESTAMP` 데이터 유형을 AWS DMS 마이그레이션하지 못하는 문제를 수정했습니다.

- PostgreSQL을 소스로 사용할 때 “±무한대” TIMESTAMP 값을 AWS DMS 마이그레이션하지 못하는 문제가 수정되었습니다.
- csvNullValue 설정이 다른 값으로 설정된 상태에서 S3를 소스로 사용할 때 빈 문자열이 NULL로 처리되는 문제를 수정했습니다.
- S3를 대상으로 사용할 때 CDC를 사용한 전체 로드에서 timestampColumnName 추가 연결 속성을 CDC 중에 정렬할 수 있도록 개선했습니다.
- S3를 소스로 사용할 때 16진수 형식(예: BYTE, BINARY, BLOB)의 이진 데이터 형식 처리를 개선했습니다.
- S3를 대상으로 사용할 때 삭제된 레코드가 특수 문자로 마이그레이션되는 문제를 수정했습니다.
- Amazon DocumentDB(MongoDB 호환)를 대상으로 사용할 때 빈 키 값을 처리하는 문제를 수정했습니다.
- MongoDB NumberDecimal 또는 Amazon DocumentDB (MongoDB 호환) 를 소스로 사용할 때 Decimal128 열을 AWS DMS 복제하지 못하는 문제가 수정되었습니다.
- MongoDB 또는 Amazon DocumentDB(MongoDB 호환)를 소스로 사용할 때 장애 조치가 발생하면 CDC 태스크를 재시도할 수 있도록 문제를 수정했습니다.
- Kinesis, Kafka를 사용하거나 대상으로 사용할 때 데이터 RAW 유형 값에서 16진수 “0x” 접두사를 제거하는 옵션이 추가되었습니다. OpenSearch
- Db2 LUW를 소스로 사용할 때 고정 길이 문자 열에서 유효성 검사가 실패하는 문제를 수정했습니다.
- 소스 데이터 형식 또는 대상 데이터 형식만 FLOAT 또는 DOUBLE인 경우 유효성 검사가 실패하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 NULL 문자에서 유효성 검사가 실패하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 XML 열에서 유효성 검사가 실패하는 문제를 수정했습니다.
- MySQL을 소스로 사용하는 복합 키에 null이 허용되는 열이 있을 때 AWS DMS 작업이 충돌하는 문제를 수정했습니다.
- SQL Server 소스 엔드포인트의 열과 PostgreSQL 대상 엔드포인트의 UUID UNIQUEIDENTIFIER 열을 모두 AWS DMS 검증하지 못하는 문제가 수정되었습니다.
- CDC 태스크가 수정된 후 업데이트된 소스 테이블 정의를 사용하지 않는 문제를 수정했습니다.
- 잘못된 사용자 AWS DMS 이름 또는 암호로 인한 작업 실패를 복구 가능한 오류로 처리하도록 장애 조치를 개선했습니다.
- RDS for SQL Server를 소스로 사용할 때 LSN이 누락되어 AWS DMS 작업이 실패하는 문제를 수정했습니다.

AWS 데이터베이스 마이그레이션 서비스 3.4.3 릴리스 노트

다음 표에서는 AWS DMS 버전 3.4.3에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
새 Amazon DocumentDB 버전	이제 Amazon DocumentDB 버전 4.0이 소스로 지원됩니다.
새 MariaDB 버전	이제 MariaDB 버전 10.4가 소스 및 대상으로 지원됩니다.
AWS Secrets Manager 통합을 위한 지원	지원되는 엔드포인트의 데이터베이스 연결 세부 정보(사용자 보안 인증 정보)를 AWS Secrets Manager에 안전하게 저장할 수 있습니다. 그러면 엔드포인트를 만들거나 수정할 AWS DMS 때 일반 텍스트 자격 증명 대신 해당 암호를 제출할 수 있습니다. AWS DMS 그런 다음 암호를 사용하여 엔드포인트 데이터베이스에 연결합니다. AWS DMS 엔드포인트의 시크릿 생성에 대한 자세한 내용은 을 참조하십시오보안 암호를 사용하여 AWS Database Migration Service 엔드포인트에 액세스하기.
C5 및 R5 복제 인스턴스용 대용량 옵션	이제 다음과 같이 더 큰 복제 인스턴스 크기를 생성할 수 있습니다. C5 크기는 최대 96개의 vCPU 및 192GiB 메모리까지, R5 크기는 최대 96개의 vCPU 및 768GiB 메모리까지 생성할 수 있습니다.
Amazon Redshift 성능 개선	AWS DMS 이제 Redshift를 대상으로 사용하여 진행 중인 복제의 성능을 개선할 때 병렬 적용을 지원합니다. 자세한 정보는 Amazon Redshift에 대한 멀티스레드 작업 설정을 참조하세요.

AWS DMS 3.4.3에서 해결된 문제는 다음과 같습니다.

- Db2 LUW를 소스로 사용할 때 지연된 이벤트의 커밋 타임스탬프가 '1970-01-01 00:00:00'이 되는 문제를 수정했습니다.
- 전체 LOB 모드에서 SQL Server를 원본으로 사용할 때 NVARCHAR 열을 기본 키로 사용하여 AWS DMS 작업이 실패하는 문제를 수정했습니다.
- SQL Server를 소스로 사용할 때 캐시된 변경 사항 단계에서 레코드가 누락되는 문제를 수정했습니다.

- RDS for SQL Server를 원본으로 사용할 때 AWS DMS 작업이 재개된 후 레코드를 건너뛰던 문제를 수정했습니다.
- AWS DMS 어설션 로깅 구성 요소가 SQL Server에 대해 큰 로그를 생성하는 문제를 수정했습니다.
- MySQL을 소스로 사용할 때 CDC 단계에서 열 파싱 오버플로로 인해 데이터 유효성 검사가 실패하는 문제를 수정했습니다.
- PostgreSQL을 대상으로 사용할 때 데이터 유효성 검사 중 세그멘테이션 오류로 인해 AWS DMS 작업이 충돌하는 문제를 수정했습니다.
- PostgreSQL을 소스 및 대상으로 사용할 때 CDC 중에 DOUBLE 데이터 형식에서 데이터 유효성 검사가 실패하는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용하고 Redshift를 대상으로 사용할 때 복사 명령으로 삽입한 레코드가 제대로 복제되지 않는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 캐시된 변경 사항 단계 중에 발생하는 데이터 손실 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 잠재적으로 데이터 손실 또는 레코드 중복을 일으킬 수 있는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 대/소문자가 혼합된 스키마가 pglogical을 사용하여 마이그레이션되지 못하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 마지막 실패 메시지에 ORA 오류가 포함되지 않는 문제를 수정했습니다.
- Oracle을 대상으로 사용할 때 AWS DMS 작업에서 UPDATE 문을 작성하지 못하던 문제를 수정했습니다.
- ASM 및 플러그 가능 데이터베이스 구성에서 Oracle 12.2를 원본으로 사용할 때 AWS DMS 작업이 데이터를 복제하지 않는 문제를 수정했습니다.
- S3를 소스로 사용할 때 RFC 4180을 준수하도록 따옴표를 보존하여 레코드 파싱을 개선했습니다.
- 전체 로드의 열을 CDC의 열로 정렬할 수 있도록 `timestampColumnName` 처리를 개선했습니다.
- 새 엔드포인트 설정을 도입하여 `MessageMaxBytes` 1MB보다 큰 LOB 요소가 있을 때 AWS DMS 작업이 실패하는 문제를 수정했습니다.
- Redshift를 대상으로 사용할 때 세그멘테이션 오류로 인해 AWS DMS 작업이 충돌하는 문제를 수정했습니다.
- Redshift 테스트 연결에 대한 오류 로깅을 개선했습니다.
- 전체 로드 중에 MongoDB에서 DocumentDB로 모든 문서를 AWS DMS 전송하지 않는 문제가 해결되었습니다.
- 테이블 매핑 규칙에 테이블이 포함되지 않은 경우 AWS DMS 작업에서 치명적 오류가 보고되던 문제를 수정했습니다.

- MySQL을 소스로 사용할 때 AWS DMS 태스크를 다시 시작하기 전에 생성한 스키마 및 테이블이 대상에 복제되지 않는 문제를 수정했습니다.
- MySQL을 소스로 사용할 때 제외 규칙에서 와일드카드 이스케이프 [_]가 와일드카드 "_"를 이스케이프할 수 없는 문제를 수정했습니다.
- MySQL을 소스로 사용할 때 UNSIGNED BIGINT 데이터 형식의 열이 제대로 복제되지 않는 문제를 수정했습니다.

AWS 데이터베이스 마이그레이션 서비스 3.4.2 릴리스 노트

다음 표에서는 AWS DMS 버전 3.4.2에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
인터넷 게이트웨이, NAT 디바이스, VPN 연결 또는 연결 없이 Amazon VPC (가상 사설 클라우드) 를 DMS (데이터베이스 마이그레이션 서비스) 에 비공개로 연결할 수 AWS 있도록 지원합니다. AWS Direct Connect	이제 생성한 VPC 인터페이스 엔드포인트를 통해 Amazon VPC에 연결하고 AWS DMS 액세스할 수 있습니다. 이 인터페이스 엔드포인트를 사용하면 Amazon 네트워크 인프라 내에서 AWS DMS 복제 인스턴스의 모든 네트워크 활동을 격리할 수 있습니다. AWS CLI 또는 SDK AWS DMS 사용에 대한 모든 API 호출에 이 인터페이스 엔드포인트에 대한 참조를 포함하면 퍼블릭 인터넷에서는 모든 AWS DMS 활동이 보이지 않도록 할 수 있습니다. 자세한 정보는 AWS Database Migration Service에서 인프라 보안을 참조하세요. **Note** 이 기능은 지원되는 모든 AWS DMS 엔진 버전에서 사용할 수 있습니다.
Amazon S3를 대상으로 사용하는 CDC 날짜 기반 폴더 파티셔닝	AWS DMS 이제 S3를 타겟으로 사용하여 데이터를 복제할 때 날짜 기반 폴더 파티셔닝을 지원합니다. 자세한 정보는 날짜 기반 폴더 파티셔닝 사용을 참조하세요.

AWS DMS 3.4.2에서 해결된 문제는 다음과 같습니다.

- Redshift를 대상으로 사용하여 마이그레이션을 수행할 때 `STATUPDATE` 옵션을 추가했습니다.

- 새 설정을 도입하여 유효성 검사 태스크를 개선했습니다. `ValidQueryCdcDelaySecond`는 소스 및 대상 엔드포인트 모두에서 첫 번째 유효성 검사 쿼리를 지연시켜 마이그레이션 지연 시간이 높을 때 리소스 경합을 줄일 수 있습니다.
- 검증 작업을 시작하는 데 시간이 오래 걸리는 문제를 AWS DMS 수정했습니다.
- S3를 대상으로 사용하여 복제 태스크를 시작 또는 중지할 때 빈 레코드가 생성되는 문제를 수정했습니다.
- 전체 로드가 완료된 후 태스크가 중단되는 문제를 수정했습니다.
- S3를 소스로 사용할 때 소스 테이블에 데이터 오류가 있으면 태스크가 중단되는 문제를 수정했습니다.
- 시작하는 동안 소스 엔드포인트의 사용자 계정이 비활성화되어 있으면 태스크가 중단되는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 `REPLICA IDENTITY FULL`을 설정하면 태스크가 충돌하는 문제를 수정했습니다.
- PostgreSQL 소스에서 pglogical 플러그인을 사용하면 태스크가 트랜잭션을 누락하는 문제를 수정했습니다.
- Redshift를 대상으로 사용할 때 압축된 소스 파일을 삭제하지 AWS DMS 않은 문제가 해결되었습니다.
- MySQL을 `BIGINT UNSIGNED` 데이터 형식의 소스 및 대상으로 사용할 때 유효성 검사 태스크가 거짓 부정을 보고하는 문제를 수정했습니다.
- SQL Server를 프라이머리 키 열이 CHAR 형식인 소스로 사용할 때 유효성 검사 태스크가 거짓 긍정을 보고하는 문제를 수정했습니다.
- S3를 대상으로 사용하여 복제 작업을 시작하는 `start-replication` 데 사용할 때 대상 객체를 지우지 AWS DMS 않는 문제를 수정했습니다.
- Db2를 소스로 사용할 때 발생하는 데이터 유효성 검사와 관련된 몇 가지 문제를 수정했습니다.
- SQL Server를 VARCHAR 열이 프라이머리 키인 소스로 사용할 때 유효성 검사 태스크가 중단되는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 데이터 형식 TIMESTAMP WITH TIMEZONE에 대한 지원을 추가했습니다.

AWS Database Migration Service 3.4.1 베타 릴리스 노트

다음 표에서는 AWS DMS 버전 3.4.1 베타에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
새 MongoDB 버전	이제 MongoDB 버전 4.0이 소스로 지원됩니다.
SQL Server에 대한 TLS 1.2 지원	AWS DMS 이제 SQL Server 엔드포인트에 대해 TLS 1.2를 지원합니다.

AWS DMS 3.4.1 베타에서 해결된 문제는 다음과 같습니다.

- Oracle 19c TDE 지원을 개선했습니다.
- Redshift를 대상으로 사용할 때 utf8mb4 문자 세트 및 ID 데이터 형식에 대한 지원을 개선했습니다.
- MySQL을 소스로 사용하고 바이너리 로그가 없는 경우 복제 태스크 실패 처리가 개선되었습니다.
- 다양한 데이터 형식 및 문자 세트에 대한 데이터 유효성 검사 지원을 개선했습니다.
- Kinesis 및 Kafka를 대상으로 사용할 때 새로운 엔드포인트 설정 `IncludeNullAndEmpty`를 통해 null 값 처리를 개선했습니다.
- Kafka를 대상으로 사용할 때 오류 로깅 및 처리를 개선했습니다.
- SQL Server를 소스로 사용할 때 DST 시간 오프셋을 개선했습니다.
- 복제 태스크가 Oracle용 기존 테이블을 대상으로 생성하려고 시도하는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 데이터베이스 연결이 끊긴 후 복제 태스크가 중단되는 문제를 수정했습니다.
- AlwaysOn 설정에서 SQL Server를 소스로 사용할 때 복제 태스크가 새 프라이머리 서버를 검색하고 다시 연결하지 못하는 문제를 수정했습니다.
- S3를 대상으로 사용할 때 특정 조건에서 복제 태스크가 `"OP"` 열의 `"D"`를 추가하지 않는 문제를 수정했습니다.

AWS Database Migration Service 3.4.0 베타 릴리스 노트

다음 표에서는 AWS DMS 버전 3.4.0에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
새 MySQL 엔진 버전	AWS DMS 이제 트랜잭션 페이로드가 압축되는 경우를 제외하고 MySQL 버전 8.0을 소스로 지원합니다.
MySQL에 대한 TLS 1.2 지원	AWS DMS 이제 MySQL 엔드포인트에 대한 TLS 1.2를 지원합니다.
새 MariaDB 버전	AWS DMS 이제 MariaDB 버전 10.3.13을 소스로 지원합니다.
자체 관리형 Microsoft SQL Server 소스에 SysAdmin 대한 액세스 불가	AWS DMS 이제 사용자가 아닌 SysAdmin 사용자도 온프레미스 및 EC2 호스팅 SQL Server 소스 엔드포인트에 액세스할 수 있습니다. **Note** 이 기능은 현재 베타 상태입니다. 사용해 보려면 지원팀에 자세한 내용을 문의하세요 AWS .
`CREATE TABLE AS`을 사용하여 생성된 CDC 태스크 및 Oracle 소스 테이블	AWS DMS 이제 명령문을 사용하여 만든 Oracle 소스 테이블에 대해 실행되는 전체 로드 작업과 CDC 및 CDC 전용 작업을 모두 지원합니다. `CREATE TABLE AS`

AWS DMS 3.4.0에서 해결된 문제는 다음과 같습니다.

- 업그레이드 태스크 평가를 개선했습니다. 자세한 정보는 작업에 대한 마이그레이션 전 평가 활성화 및 활용을 참조하세요.
- float, real 및 double 데이터 형식에 대한 데이터 유효성 검사를 개선했습니다.
- “지정된 키가 존재하지 않습니다.” 오류를 더 잘 처리하여 대상으로서의 Amazon Redshift를 개선했습니다.
- Amazon OpenSearch Service (OpenSearch Service) 의 `ParallelApplyQueuesPerThread` 경우 `ParallelApplyThreadsParallelApplyBufferSize`, 및 를 대상으로 사용하는 등 멀티스레드 CDC 로드 작업 설정을 지원합니다.
- 복합 기본 키 사용을 지원하여 대상으로서의 OpenSearch 서비스를 개선했습니다.
- PostgreSQL을 소스로 사용할 때 암호에 특수 문자가 있는 경우 테스트 연결이 실패하는 문제를 수정했습니다.

- SQL Server를 소스로 사용할 때 일부 VARCHAR 열이 잘리는 경우 발생하는 문제를 수정했습니다.
- Amazon RDS SQL Server를 원본으로 사용할 때 열린 트랜잭션이 닫히지 AWS DMS 않는 문제를 수정했습니다. 폴링 간격 파라미터를 잘못 설정하면 데이터가 손실될 수 있습니다. 권장 폴링 간격 값을 설정하는 방법에 대한 자세한 내용은 Microsoft SQL Server 데이터베이스를 원본으로 사용 AWS DMS 섹션을 참조하세요.
- Oracle Standby를 소스로 사용할 때 Binary Reader를 사용하면 CDC 태스크가 예기치 않게 중지되는 문제를 수정했습니다.
- IBM DB2 for LUW에서 “숫자 리터럴 0은 해당 값이 범위를 벗어났으므로 유효하지 않습니다.”라는 메시지와 함께 태스크가 실패하는 문제를 수정했습니다.
- PostgreSQL에서 PostgreSQL로 마이그레이션할 때 PostgreSQL 소스에 새 열이 추가되고 해당 열이 소스에서 원래 생성된 데이터 형식과 다른 데이터 형식으로 생성되는 경우 발생하는 문제를 수정했습니다.
- MySQL 소스에서 binlog를 가져올 수 없을 때 마이그레이션 태스크가 예기치 않게 중단되는 문제를 수정했습니다.
- Oracle 대상에서 `BatchApply` 사용과 관련된 문제를 수정했습니다.
- MySQL 및 MariaDB에서 `TIME` 데이터 형식을 마이그레이션할 때 발생하는 문제를 수정했습니다.
- IBM DB2 LUW 소스에서 테이블에 프라이머리 키 또는 고유 키가 없는 경우 LOB가 포함된 테이블 마이그레이션이 실패하는 문제를 수정했습니다.

AWS Database Migration Service 3.3.4 릴리스 노트

AWS DMS 3.3.4에서 해결된 문제는 다음과 같습니다.

- PostgreSQL을 소스로 사용할 때 트랜잭션이 삭제되거나 중복되는 문제를 수정했습니다.
- 스키마 이름에 달러 기호($) 사용에 대한 지원을 개선했습니다.
- RDS SQL Server를 소스로 사용할 때 복제 인스턴스가 미결 트랜잭션을 닫지 않는 문제를 수정했습니다.
- PostgreSQL을 소스로 사용할 때 암호에 특수 문자가 있는 경우 테스트 연결이 실패하는 문제를 수정했습니다.
- “지정된 키가 존재하지 않습니다.” 오류를 더 잘 처리하여 대상으로서의 Amazon Redshift를 개선했습니다.
- 다양한 데이터 형식 및 문자 세트에 대한 데이터 유효성 검사 지원을 개선했습니다.
- 복제 태스크가 Oracle용 기존 테이블을 대상으로 생성하려고 시도하는 문제를 수정했습니다.

- Amazon S3를 대상으로 사용할 때 특정 조건에서 복제 태스크가 "OP" 열의 "D"를 추가하지 않는 문제를 수정했습니다.

AWS 데이터베이스 마이그레이션 서비스 3.3.3 릴리스 노트

다음 표에서는 AWS DMS 버전 3.3.3에 도입된 새로운 기능과 향상된 기능을 보여줍니다.

새로운 기능 또는 향상된 기능	설명
새 PostgreSQL 버전	PostgreSQL 버전 12는 이제 소스 및 대상으로 지원됩니다.
Amazon OpenSearch 서비스를 대상으로 하는 복합 기본 키 지원	AWS DMS 3.3.3부터 OpenSearch 서비스 대상은 복합 기본 키 사용을 지원합니다.
Oracle 확장 데이터 형식 지원	이제 Oracle 소스 및 대상 모두에 대한 Oracle 확장 데이터 형식이 지원됩니다.
계정당 AWS DMS 리소스 수 증가	생성할 수 있는 AWS DMS 리소스 수의 제한이 늘어났습니다. 자세한 정보는 AWS Database Migration Service에 대한 할당량을 참조하세요.

AWS DMS 3.3.3에서 해결된 문제는 다음과 같습니다.

- Amazon Kinesis에서 병렬 적용을 사용하는 특정 업데이트 문을 사용하여 태스크가 충돌하는 문제를 수정했습니다.
- Amazon S3를 대상으로 사용할 때 ALTER TABLE 문에서 태스크가 충돌하는 문제를 수정했습니다.
- Microsoft SQL Server를 소스로 사용할 때 다각형 열의 값이 잘리는 문제를 수정했습니다.
- Oracle을 소스로 사용할 때 JA16SJISTILDE 및 JA16EUCTILDE의 유니코드 변환기 관련 문제를 수정했습니다.
- MEDIUMTEXT 및 LONGTEXT 열이 MySQL에서 S3의 쉼표로 구분된 값(CSV) 형식으로 마이그레이션되지 않는 문제를 수정했습니다.
- Apache Parquet 출력을 사용할 때 부울 열이 잘못된 형식으로 변환되는 문제를 수정했습니다.
- Oracle에서 확장된 varchar 열 관련 문제를 수정했습니다.

- 특정 타임스탬프 조합으로 인해 데이터 유효성 검사 작업이 실패하는 문제를 수정했습니다.
- Sybase 데이터 정의 언어(DDL) 복제 관련 문제를 수정했습니다.
- Oracle RAC(실제 애플리케이션 클러스터) 소스가 Oracle Binary Reader와 충돌하는 문제를 수정했습니다.
- 스키마 이름의 대/소문자를 사용하는 Oracle 대상에 대한 유효성 검사 관련 문제를 수정했습니다.
- IBM Db2 버전 9.7 및 10의 유효성 검사 관련 문제를 수정했습니다.
- `StopTaskCachedChangesApplied` 및 `StopTaskCachedChangesNotApplied`기 활성화된 작업이 두 번 중지되지 않는 문제를 수정했습니다.

문서 기록

다음 표는 2018년 1월 이후 AWS Database Migration Service 사용 설명서에 적용된 중요한 변경 사항을 설명합니다.

RSS 피드를 구독하면 이 설명서에 대한 업데이트 알림을 받을 수 있습니다. AWS DMS 버전 릴리스에 관한 자세한 내용은 AWS DMS 릴리스 노트 단원을 참조하십시오.

변경 사항	설명	날짜
AWS DMS에서 대상으로 RDS IBM DB2에 대한 지원 추가	AWS DMS는 이제 Amazon RDS IBM DB2를 대상으로 사용할 수 있도록 지원합니다.	2023년 12월 4일
AWS DMS에서 대상으로 Timestream에 대한 지원 추가	AWS DMS는 이제 Timestream을 대상으로 지원합니다.	2023년 11월 17일
AWS DMS에서 Redshift 대상 데이터 검증에 대한 지원 추가	AWS DMS는 이제 Redshift 대상의 데이터 검증을 지원합니다.	2023년 11월 14일
AWS DMS에서 네 가지 새로운 엔드포인트 유형에 대한 지원 추가	AWS DMS는 이제 Microsoft Azure Database for PostgreSQL, Microsoft Azure Database for MySQL, OCI MySQL Heatwave, Google Cloud for PostgreSQL를 소스로 사용할 수 있도록 지원합니다.	2023년 10월 26일
AWS DMS에서 새로운 AWS 서비스 연결 역할에 대한 지원 추가	AWS DMS는 이제 Amazon CloudWatch에 지표 데이터 포인트를 게시하는 등 AWS DMS가 사용자 대신 리소스를 생성하고 관리할 수 있도록 AWS 서비스 연결 역할 `AWSServic`	2023년 5월 22일

	eRoleForDMSServerless 를 지원합니다.	
AWS DMS에서 새로운 AWS 관리형 정책에 대한 지원을 추가	AWS DMS는 이제 CloudWatch Logs에 서버리스 복제 로그를 게시할 수 있는 AWS 관리형 정책을 지원합니다.	2023년 5월 22일
AWS DMS에서 새로운 AWS 관리형 정책에 대한 지원을 추가	AWS DMS에서 이제 Amazon CloudWatch에 지표 데이터 포인트를 게시할 수 있는 AWS 관리형 정책을 지원합니다. 또한 AWS DMS가 AWS 관리형 정책에 대한 변경 내용 추적을 시작했습니다.	2023년 3월 6일
VPC 소스 및 대상 엔드포인트 지원	AWS DMS는 Virtual Private Cloud(VPC) 엔드포인트를 원본 및 대상으로 지원합니다. 이제 AWS DMS는 서비스에 대해 명시적으로 정의된 경로가 AWS DMS VPC에 정의되어 있을 때 VPC 엔드포인트가 있는 모든 AWS 서비스에 연결할 수 있습니다.	2022년 6월 30일
SQL Server 읽기 전용 복제본을 소스로 지원	AWS DMS는 SQL Server 읽기 전용 복제본을 소스로 지원합니다. AWS DMS를 사용하면 이제 SQL Server 읽기 전용 복제본에서 AWS DMS가 지원하는 모든 대상으로 라이브 마이그레이션을 수행할 수 있습니다.	2022년 6월 30일

전체 로드 전용 소스로서 IBM Db2 z/OS 데이터베이스 지원	AWS DMS는 이제 IBM Db2 z/OS 데이터베이스를 소스로 지원합니다. AWS DMS를 사용하면 이제 Db2 메인프레임에서 AWS DMS가 지원하는 모든 대상으로 라이브 마이그레이션을 수행할 수 있습니다.	2022년 6월 30일
EventBridge DMS 이벤트 지원	AWS DMS는 EventBridge for DMS 이벤트를 사용하여 이벤트 구독을 관리할 수 있도록 지원합니다.	2022년 6월 30일
Babelfish를 대상으로 지원	AWS DMS는 Babelfish를 대상으로 지원합니다. AWS DMS를 사용하면 이제 가동 중지 시간을 최소화하면서 AWS DMS가 지원하는 모든 소스에서 Babelfish로 라이브 데이터를 마이그레이션할 수 있습니다.	2022년 6월 30일
Aurora Serverless v2를 대상으로 지원	AWS DMS는 Aurora Serverless v2를 대상으로 지원합니다. 이제 AWS DMS를 사용하면 Aurora Serverless v2로 라이브 마이그레이션을 수행할 수 있습니다.	2022년 6월 30일
시작하기 자습서	AWS DMS 시작하기 자습서의 업데이트입니다. 이 자습서에서는 MySQL 데이터베이스를 소스로 사용하고 PostgreSQL 데이터베이스를 대상으로 사용합니다.	2021년 5월 20일

Amazon Neptune을 대상으로 지원	데이터 마이그레이션 대상으로 Amazon Neptune에 대한 지원을 추가했습니다.	2020년 6월 1일
대상으로서 Apache Kafka에 대한 지원	데이터 마이그레이션 대상으로 Apache Kafka에 대한 지원을 추가했습니다.	2020년 3월 20일
보안 콘텐츠 업데이트됨	고객 요청에 대한 응답으로 보안 콘텐츠를 업데이트하고 표준화했습니다.	2019년 12월 20일
AWS Snowball Edge을 사용하여 마이그레이션	AWS Snowball Edge를 사용하여 대규모 데이터베이스를 마이그레이션하기 위한 지원을 추가했습니다.	2019년 1월 24일
대상으로서 Amazon DocumentDB(MongoDB 호환)에 대한 지원	데이터 마이그레이션의 대상으로서 Amazon DocumentDB(MongoDB 호환)에 대한 지원이 추가되었습니다.	2019년 1월 9일
대상으로서 Amazon OpenSearch Service 및 Amazon Kinesis Data Streams에 대한 지원	데이터 마이그레이션의 대상으로서 OpenSearch Service 및 Kinesis Data Streams에 대한 지원이 추가되었습니다.	2018년 11월 15일
CDC 기본 시작 지원	변경 데이터 캡처(CDC) 사용 시 기본 시작점에 대한 지원이 추가되었습니다.	2018년 6월 28일
Db2 LUW 지원	데이터 마이그레이션 소스로서 IBM Db2 LUW에 대한 지원이 추가되었습니다.	2018년 4월 26일
대상으로서 SQL Server 지원	소스로서 Amazon RDS for Microsoft SQL Server에 대한 지원이 추가되었습니다.	2018년 2월 6일

AWS 용어집

최신 AWS 용어는 AWS 용어집 참조의 AWS 용어집을 참조하세요.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.