에서 멀티테넌트 SaaS 애플리케이션을 위한 관리형 PostgreSQL을 구현합니다. AWS

AWS 규범적 지침

AWS 규범적 지침: 에서 멀티테넌트 SaaS 애플리케이션을 위한 관리형 PostgreSQL을 구현합니다. AWS

Table of Contents

에서 멀티테넌트 SaaS 애플리케이션을 위한 관리형 PostgreSQL을 구현합니다. AWS

태비 워드와 토마스 데이비스, Amazon Web Services ()AWS

2024년 4월 (문서 기록)

운영 데이터를 저장할 데이터베이스를 선택할 때는 데이터를 어떻게 구성해야 하는지, 어떤 쿼리에 응답할지, 얼마나 빠르게 답변을 제공할 것인지, 데이터 플랫폼 자체의 복원력을 고려하는 것이 중요합니다. 이러한 일반적인 고려 사항 외에도 성능 격리, 테넌트 보안, 멀티테넌트 SaaS 애플리케이션 데이터의 일반적인 고유 특성 및 설계 패턴 등 운영 데이터에 대한 SaaS (Software as a Service) 의 영향도 포함됩니다. 이 가이드에서는 Amazon Web AWS Services () 의 PostgreSQL 데이터베이스를 멀티테넌트 SaaS 애플리케이션의 기본 운영 데이터 스토어로 사용할 때 이러한 요소가 어떻게 적용되는지 설명합니다. 구체적으로 설명하자면, 이 가이드에서는 두 가지 AWS 관리형 PostgreSQL 옵션, 즉 Amazon Aurora PostgreSQL 호환 에디션과 PostgreSQL용 Amazon Relational Database Service (Amazon RDS) 에 중점을 둡니다.

목표 비즈니스 성과

이 지침에서는 Aurora PostgreSQL과 호환되는 Aurora 및 PostgreSQL용 Amazon RDS를 사용하는 멀티테넌트 SaaS 애플리케이션의 모범 사례에 대한 자세한 분석을 제공합니다. 이 가이드에 제공된 디자인 패턴과 개념을 사용하여 멀티테넌트 SaaS 애플리케이션을 위한 Aurora PostgreSQL 호환 또는 PostgreSQL용 Amazon RDS의 구현을 알리고 표준화하는 것이 좋습니다.

이 규범적 지침은 다음과 같은 비즈니스 성과를 달성하는 데 도움이 됩니다.

- 사용 사례에 가장 적합한 AWS 관리형 PostgreSQL 옵션 선택 — 이 가이드에서는 SaaS 애플리케이션에서 데이터베이스 사용을 위한 관계형 옵션과 비관계형 옵션을 비교합니다. 또한 Aurora PostgreSQL과 호환되는 사용 사례와 PostgreSQL용 Amazon RDS에 가장 적합한 사용 사례에 대해서도 설명합니다. 이 정보는 SaaS 애플리케이션에 가장 적합한 옵션을 선택하는 데 도움이 됩니다.
- SaaS 파티셔닝 모델 채택을 통한 SaaS 모범 사례 적용 — 이 가이드에서는 PostgreSQL 데이터베이스 관리 시스템 (DBMS) 에 적용할 수 있는 세 가지 광범위한 SaaS 파티셔닝 모델, 즉 풀, 브리지, 사일로 모델 및 그 변형에 대해 설명하고 비교합니다. 이러한 접근 방식은 SaaS 모범 사례를 캡처하고 SaaS 애플리케이션을 설계할 때 유연성을 제공합니다. SaaS 파티셔닝 모델을 적용하는 것은 모범 사례를 보존하는 데 있어 중요한 부분입니다.
- 풀 SaaS 파티셔닝 모델에서 RLS의 효과적인 사용 — 행 수준 보안 (RLS) 은 사용자 또는 컨텍스트 변수를 기반으로 볼 수 있는 행을 제한하여 단일 PostgreSQL 테이블 내에서 테넌트 데이터 격리를

적용할 수 있도록 지원합니다. 풀 파티셔닝 모델을 사용하는 경우 테넌트 간 액세스를 방지하려면 RLS가 필요합니다.

SaaS 애플리케이션용 데이터베이스 선택

많은 멀티테넌트 SaaS 애플리케이션의 경우 운영 데이터베이스 선택은 관계형 데이터베이스와 비관계형 데이터베이스 중 하나를 선택하거나 이 둘을 조합하여 선택하는 것으로 요약할 수 있습니다. 결정을 내리려면 다음과 같은 높은 수준의 애플리케이션 데이터 요구 사항 및 특성을 고려하십시오.

- 애플리케이션의 데이터 모델
- 데이터 액세스 패턴
- 데이터베이스 지연 요구 사항
- 데이터 무결성 및 트랜잭션 무결성 요구 사항 (원자성, 일관성, 격리성, 내구성 또는 ACID)
- 지역 간 가용성 및 복구 요구 사항

다음 표에는 애플리케이션 데이터 요구 사항 및 특성이 나열되어 있으며 AWS 데이터베이스 오퍼링과 관련하여 이에 대해 설명합니다. Aurora PostgreSQL과 호환되고 PostgreSQL용 Amazon RDS (관계형), Amazon DynamoDB (비관계형) 등이 있습니다. 관계형 운영 데이터베이스 제품과 비관계형 운영 데이터베이스 제품 중 하나를 결정하려는 경우 이 매트릭스를 참조할 수 있습니다.

데이터베이스	SaaS 애플리케이션 데이터 요구 사항 및 특성				
	데이터 모델	액세스 패턴	지연 요구 사항	데이터 및 트랜잭션 무결성	지역 간 가용성 및 복구
관계형 (Aurora PostgreSQL과 호환되며 PostgreSQL용 Amazon RDS)	관계형이거나 고도로 표준화되어 있습니다.	사전에 철저하게 계획을 세울 필요는 없습니다.	지연 시간 허용 범위를 높이는 것이 좋습니다. 기본적으로 Aurora를 사용하고 읽기 전용 복제본, 캐싱 및 유사한 기능을 구현하여 지연	기본적으로 높은 데이터 및 트랜잭션 무결성이 유지됩니다.	Amazon RDS에서는 리전 간 확장 및 장애 조치를 위한 읽기 전용 복제본을 생성할 수 있습니다. Aurora는 이 프로세스를 대부분 자동화합니다. 여러 AWS

			시간을 줄일 수 있습니다.		리전곳에 걸친 액티브-액티브 구성의 경우 Aurora 글로벌 데이터베이스와 함께 쓰기 전달을 사용할 수 있습니다.
비관계형 (아마존 다이나모DB)	일반적으로 비정규화됩니다. 이러한 데이터베이스는 패턴을 활용하여 many-to-many관계, 대형 항목 및 시계열 데이터를 모델링합니다.	데이터 모델을 생성하기 전에 데이터에 대한 모든 액세스 패턴 (쿼리) 을 철저히 이해해야 합니다.	Amazon DynamoDB 가속기 (DAX) 와 같은 옵션을 사용하면 지연 시간이 매우 짧아 성능을 더욱 향상시킬 수 있습니다.	선택적 트랜잭션 무결성 (성능 저하). 데이터 무결성 문제는 애플리케이션으로 옮겨갑니다.	글로벌 테이블을 사용한 간편한 지역 간 복구 및 액티브-액티브 구성 (ACID 규정 준수는 단일 지역에서만 가능합니다.) AWS

일부 멀티테넌트 SaaS 애플리케이션에는 이전 표에 포함되지 않은 데이터베이스에서 더 나은 서비스를 제공하는 고유한 데이터 모델 또는 특수 상황이 있을 수 있습니다. 예를 들어 시계열 데이터 세트, 연결성이 높은 데이터 세트 또는 중앙 집중식 트랜잭션 원장을 유지 관리하려면 다른 유형의 데이터베이스를 사용해야 할 수 있습니다. 모든 가능성을 분석하는 것은 이 가이드의 범위를 벗어납니다. AWS 데이터베이스 제품의 전체 목록과 이러한 제품이 다양한 사용 사례를 개괄적으로 충족하는 방법은 Amazon Web Services 개요 백서의 데이터베이스 섹션을 참조하십시오.

이 가이드의 나머지 부분에서는 PostgreSQL을 지원하는 AWS 관계형 데이터베이스 서비스 (Amazon RDS 및 Aurora PostgreSQL과 호환됨) 에 중점을 둡니다. DynamoDB는 SaaS 애플리케이션을 최적화하기 위한 다른 접근 방식이 필요하며, 이는 이 가이드의 범위를 벗어납니다. DynamoDB에 대한 자세한 내용은 Amazon DynamoDB를 사용하여 풀링된 AWS 멀티테넌트 SaaS 데이터를 분할하는 블로그 게시물을 참조하십시오.

아마존 RDS와 Aurora 중에서 선택하기

대부분의 경우 PostgreSQL용 Amazon RDS보다 Aurora PostgreSQL과 호환되는 Aurora를 사용하는 것이 좋습니다. 다음 표에는 이러한 두 옵션 중 하나를 결정할 때 고려해야 하는 요소가 나와 있습니다.

DBMS 컴포넌트	Amazon RDS for PostgreSQL	Aurora 포스트그레SQL 호환
확장성	복제 지연 시간 (분), 읽기 전용 복제본 최대 5개	1분 미만의 복제 지연 (글로벌 데이터베이스의 경우 일반적으로 1초 미만), 최대 15개의 읽기 전용 복제본
장애 복구	체크포인트가 5분 간격 (기본값) 으로 인해 데이터베이스 성능이 저하될 수 있음	신속한 복구를 위한 병렬 스레드를 사용한 비동기 복구
페일오버	크래시 복구 시간 외에 60-120초	보통 약 30초 (크래시 복구 포함)
스토리지	최대 IOPS 256,000	Aurora 인스턴스 크기 및 용량에 의해서만 IOPS가 제한됩니다.
고가용성 및 재해 복구	대기 인스턴스가 있는 가용 영역 2개, 읽기 전용 복제본 또는 복사된 백업을 위한 지역 간 장애 조치	기본적으로 세 개의 가용 영역, Aurora 글로벌 데이터베이스를 사용한 지역 간 페일오버, 액티브-액티브 구성을 위한 쓰기 포워딩 AWS 리전
백업	백업 기간 중에는 성능에 영향을 미칠 수 있음	자동 증분 백업, 성능에 영향을 미치지 않음
데이터베이스 인스턴스 클래스	Amazon RDS 인스턴스 클래스 목록을 참조하십시오.	Aurora 인스턴스 클래스 목록 보기

위 표에 설명된 모든 범주에서는 일반적으로 Aurora PostgreSQL과 호환되는 것이 더 나은 옵션입니다. 하지만 Amazon RDS for PostgreSQL은 Aurora의 더욱 강력한 기능 세트를 희생시키면서도 더 비

용 효율적인 옵션을 제공할 수 있는 다양한 인스턴스 클래스를 갖추고 있기 때문에 중소 규모의 워크로드에는 여전히 적합할 수 있습니다.

PostgreSQL용 멀티 테넌트 SaaS 파티셔닝 모델

멀티 테넌시를 달성하는 가장 좋은 방법은 SaaS 애플리케이션의 요구 사항에 따라 다릅니다. 다음 섹션에서는 PostgreSQL에서 멀티 테넌시를 성공적으로 구현하기 위한 파티셔닝 모델을 보여줍니다.

> **Note**
>
> 이 섹션에서 설명하는 모델은 PostgreSQL용 Amazon RDS와 Aurora PostgreSQL과 호환되는 모델 모두에 적용할 수 있습니다. 이 섹션의 PostgreSQL에 대한 참조는 두 서비스 모두에 적용됩니다.

PostgreSQL에서 SaaS 파티셔닝을 위해 사용할 수 있는 상위 수준 모델은 사일로, 브리지, 풀의 세 가지입니다. 다음 이미지는 사일로 모델과 풀 모델 간의 장단점을 요약합니다. 브리지 모델은 사일로 모델과 풀 모델의 하이브리드입니다.

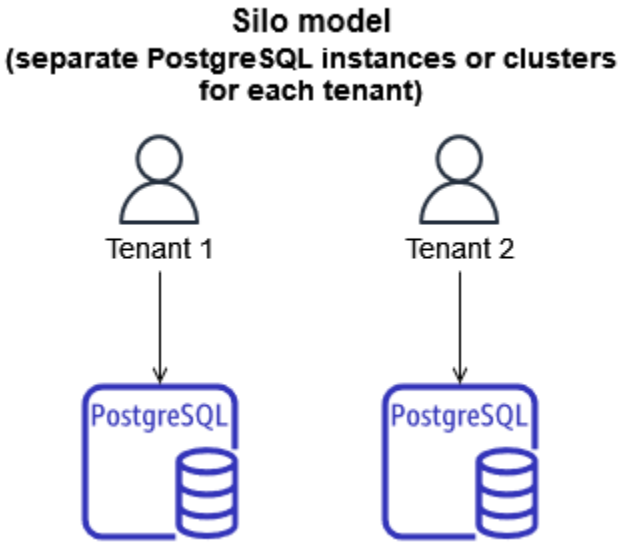
파티셔닝 모델	장점	단점
저장고	• 규정 준수 • 테넌트 간 영향 없음 • 테넌트 레벨 튜닝 • 테넌트 수준의 가용성	• 기한 경과 • 중앙 집중식 관리 없음 • 배포 복잡 • 비용
풀	• 민첩성 • 비용 최적화 • 중앙 집중식 관리 • 간소화된 배포	• 최소 테넌트 • 규정 준수 문제 • 전부 또는 전혀 가용성 없음
브릿지	• 규정 준수 일부 조정 • 민첩성 • 비용 최적화 • 중앙 집중식 관리	• 규정 준수 관련 문제 • 가용성이 전부 또는 전혀 없음 (대부분) • 최소 테넌트 • 배포 복잡성

다음 섹션에서는 각 모델에 대해 보다 자세히 설명합니다.

파티셔닝

PostgreSQL 사일로 모델

사일로 모델은 애플리케이션의 각 테넌트에 대해 PostgreSQL 인스턴스를 프로비저닝하여 구현됩니다. 사일로 모델은 테넌트 성능 및 보안 격리에 탁월하며 노이즈 인접 현상을 완전히 제거합니다. Noisy Neighbor 현상은 한 테넌트의 시스템 사용이 다른 테넌트의 성능에 영향을 미칠 때 발생합니다. 사일로 모델을 사용하면 각 테넌트에 맞게 성능을 조정하고 잠재적으로 특정 테넌트의 사일로에 대한 운영 중단을 제한할 수 있습니다. 그러나 일반적으로 사일로 모델의 채택을 주도하는 것은 엄격한 보안 및 규제 제약입니다. 이러한 제약은 SaaS 고객이 동기가 될 수 있습니다. 예를 들어 SaaS 고객은 내부 제약으로 인해 데이터를 격리하도록 요구할 수 있으며 SaaS 공급자는 추가 비용을 지불하고 이러한 서비스를 제공할 수 있습니다.

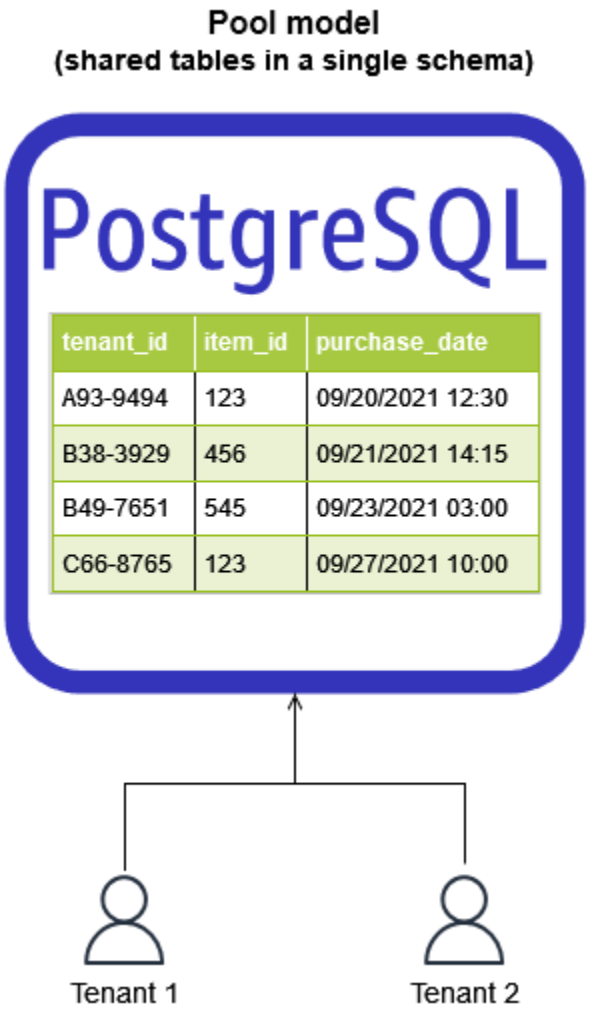


경우에 따라 사일로 모델이 필요할 수도 있지만 많은 단점이 있습니다. 여러 PostgreSQL 인스턴스에서 리소스 소비를 관리하는 것이 복잡할 수 있기 때문에 사일로 모델을 비용 효율적인 방식으로 사용하기가 어려운 경우가 많습니다. 또한 이 모델에서는 데이터베이스 워크로드의 분산된 특성으로 인해 테넌트 활동을 중앙에서 파악하기가 더 어려워집니다. 독립적으로 운영되는 워크로드를 너무 많이 관리하면 운영 및 관리 오버헤드가 증가합니다. 또한 사일로 모델은 테넌트별 리소스를 프로비저닝해야 하

기 때문에 테넌트 온보딩이 더 복잡하고 시간이 많이 걸립니다. 또한 테넌트별 PostgreSQL 인스턴스의 수가 계속 증가함에 따라 관리하는 데 더 많은 운영 시간이 필요하기 때문에 전체 SaaS 시스템을 확장하기가 더 어려울 수 있습니다. 마지막 고려 사항 중 하나는 애플리케이션 또는 데이터 액세스 계층에서 테넌트와 연결된 PostgreSQL 인스턴스의 매핑을 유지해야 한다는 점입니다. 이 때문에 이 모델을 구현하는 것이 더 복잡해집니다.

PostgreSQL 풀 모델

풀 모델은 단일 PostgreSQL 인스턴스 (Amazon RDS 또는 Aurora) 를 프로비저닝하고 행 수준 보안 (RLS) 을 사용하여 테넌트 데이터 격리를 유지함으로써 구현됩니다. RLS 정책은`SELECT` 쿼리에서 반환되는 테이블 행 또는`INSERT`,`UPDATE`, 및`DELETE` 명령의 영향을 받는 행을 제한합니다. 풀 모델은 모든 테넌트 데이터를 단일 PostgreSQL 스키마로 중앙 집중화하므로 훨씬 더 비용 효율적이고 유지 관리에 필요한 운영 오버헤드가 줄어듭니다. 이 솔루션의 모니터링도 중앙 집중식으로 인해 훨씬 더 간단해졌습니다. 그러나 수영장 모델에서 테넌트별 영향을 모니터링하려면 일반적으로 애플리케이션에 몇 가지 추가 계측기가 필요합니다. 이는 PostgreSQL이 기본적으로 어떤 테넌트가 리소스를 소비하는지 인식하지 못하기 때문입니다. 새로운 인프라가 필요하지 않기 때문에 테넌트 온보딩이 간소화됩니다. 이러한 민첩성을 통해 빠르고 자동화된 테넌트 온보딩 워크플로를 더 쉽게 달성할 수 있습니다.

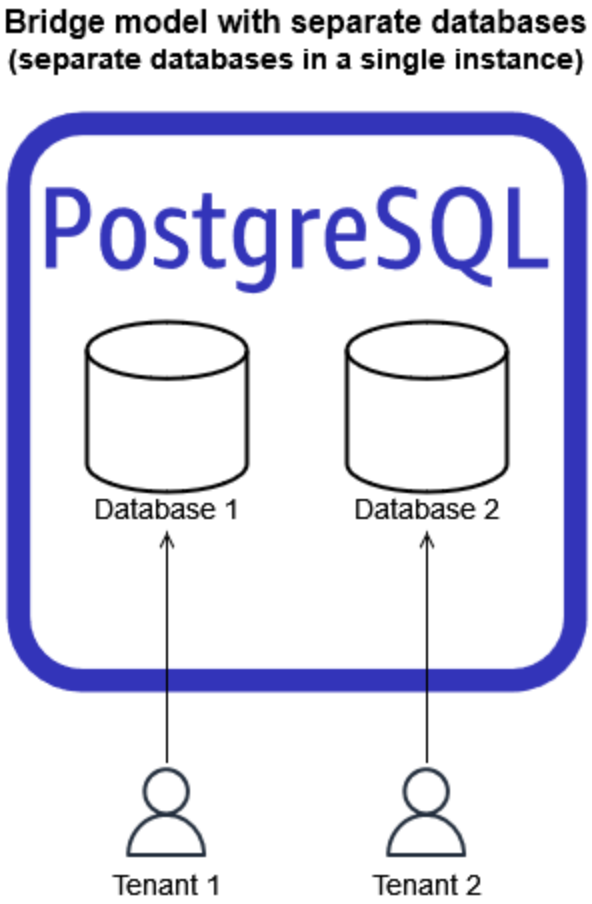


풀 모델은 일반적으로 비용 효율적이고 관리가 더 간단하지만 몇 가지 단점이 있습니다. 풀 모델에서는 노이즈 인접 현상을 완전히 제거할 수 없습니다. 그러나 PostgreSQL 인스턴스에서 적절한 리소스를 사용할 수 있도록 하고 읽기 전용 복제본이나 Amazon으로 쿼리를 오프로드하는 등 PostgreSQL 부하를 줄이는 전략을 사용하면 문제를 완화할 수 ElastiCache 있습니다. 애플리케이션 계측이 테넌트별 활동을 기록하고 모니터링할 수 있기 때문에 효과적인 모니터링은 테넌트 성능 격리 문제에 대응하는 데에도 중요한 역할을 합니다. 마지막으로, 일부 SaaS 고객은 RLS에서 제공하는 논리적 분리가 충분하지 않아 추가 격리 조치를 요청할 수 있습니다.

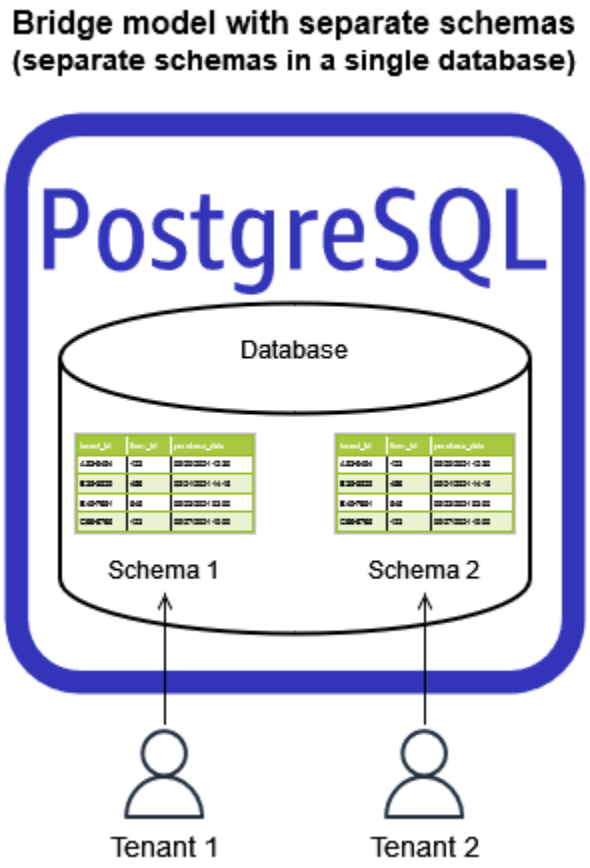
PostgreSQL 브리지 모델

PostgreSQL 브리지 모델은 풀링된 접근 방식과 사일로화된 접근 방식의 조합입니다. 풀링 모델과 마찬가지로 각 테넌트에 대해 단일 PostgreSQL 인스턴스를 프로비저닝합니다. 테넌트 데이터 격리를 유지하려면 PostgreSQL 논리적 구조를 사용합니다. 다음 다이어그램에서 PostgreSQL 데이터베이스는 데이터를 논리적으로 분리하는 데 사용됩니다.

> **Note**
>
> PostgreSQL 데이터베이스는 PostgreSQL용 Amazon RDS for PostgreSQL 또는 Aurora PostgreSQL 호환 DB 인스턴스를 참조하지 않습니다. 대신 데이터를 분리하는 PostgreSQL 데이터베이스 관리 시스템의 논리적 구조를 말합니다.



다음 다이어그램에 나와 있는 것처럼 각 데이터베이스에 테넌트별 스키마가 있는 단일 PostgreSQL 데이터베이스를 사용하여 브리지 모델을 구현할 수도 있습니다.



브리지 모델은 풀 모델과 동일한 노이즈 인접 및 테넌트 성능 격리 문제를 겪습니다. 또한 테넌트별로 별도의 데이터베이스나 스키마를 프로비저닝해야 하므로 운영 및 프로비저닝 오버헤드가 추가로 발생합니다. 테넌트 성능 문제에 신속하게 대응하려면 효과적인 모니터링이 필요합니다. 또한 테넌트별 사용량을 모니터링하기 위한 애플리케이션 계측이 필요합니다. 전반적으로 브리지 모델은 새로운 PostgreSQL 데이터베이스 또는 스키마를 요구하여 테넌트 온보딩 노력을 약간 강화하는 RLS의 대안으로 볼 수 있습니다. 사일로 모델과 마찬가지로 애플리케이션 또는 데이터 액세스 계층은 관련 PostgreSQL 데이터베이스 또는 스키마에 대한 테넌트의 매핑을 유지해야 합니다.

의사 결정 매트릭스

PostgreSQL과 함께 사용해야 하는 멀티 테넌트 SaaS 파티셔닝 모델을 결정하려면 다음 의사 결정 매트릭스를 참조하십시오. 매트릭스는 다음 네 가지 파티셔닝 옵션을 분석합니다.

- 사일로 (Silo) - 테넌트별로 별도의 PostgreSQL 인스턴스 또는 클러스터
- 개별 데이터베이스와의 브리지 - 단일 PostgreSQL 인스턴스 또는 클러스터의 각 테넌트에 대한 별도의 데이터베이스입니다.

- 별도의 스키마가 있는 브리지 - 단일 PostgreSQL 인스턴스 또는 클러스터의 단일 PostgreSQL 데이터베이스의 각 테넌트에 대한 별도의 스키마입니다.
- 풀 - 단일 인스턴스 및 스키마의 테넌트용 공유 테이블입니다.

	저장고	별도의 데이터베이스가 있는 브리지	별도의 스키마가 있는 브리지	풀
사용 사례	리소스 사용을 완벽하게 제어하여 데이터를 격리하는 것이 핵심 요구 사항이며, 규모가 매우 크고 성능에 매우 민감한 테넌트가 있는 경우	데이터 격리는 핵심 요구 사항이며 테넌트 데이터의 상호 참조는 제한적이거나 전혀 필요하지 않습니다.	적당한 수의 테넌트가 있고 데이터 양이 적당합니다. 테넌트의 데이터를 상호 참조해야 하는 경우 이 모델이 선호되는 모델입니다.	테넌트당 데이터 수가 적은 다수의 테넌트
새로운 테넌트 온보딩 민첩성	매우 느느립니다. (각 테넌트마다 새 인스턴스 또는 클러스터가 필요합니다.)	약간 느느립니다. (스키마 객체를 저장하려면 각 테넌트에 대해 새 데이터베이스를 만들어야 합니다.)	약간 느느립니다. (각 테넌트가 객체를 저장하려면 새 스키마를 만들어야 합니다.)	가장 빠른 옵션. (최소 설정이 필요합니다.)
데이터베이스 연결 풀 구성 노력 및 효율성	상당한 노력이 필요합니다. (테넌트당 하나의 연결 풀) 효율성이 떨어집니다. (테넌트 간 데이터베이스 연결 공유는 없습니다.)	상당한 노력이 필요합니다. (Amazon RDS 프록시를 사용하지 않는 한 테넌트당 하나의 연결 풀 구성) 효율성이 떨어집니다. (테넌트 및 총 연결 수 간 데	더 적은 노력이 필요합니다. (모든 테넌트에 대한 하나의 연결 풀 구성) 적당히 효율적입니다. (세션 풀 모드에서만SET ROLE 또는SET	최소한의 노력이 필요합니다. 가장 효율적입니다. (모든 테넌트를 위한 하나의 연결 풀과 모든 테넌트에서 효율적인 연결 재사용. DB 인스턴스

	저장고	별도의 데이터베이스가 있는 브리지	별도의 스키마가 있는 브리지	풀
		이터베이스 연결 공유가 없습니다. DB 인스턴스 클래스를 기반으로 모든 테넌트별로 제한됩니다.)	SCHEMA 명령을 통한 연결 재사용이 가능합니다. SET또한 명령은 Amazon RDS Proxy를 사용할 때 세션 피닝을 유발하지만 효율성을 위해 각 요청에 대해 클라이언트 연결 풀을 제거하고 각 요청에 대해 직접 연결할 수 있습니다.)	클래스를 기반으로 제한됩니다.)

	저장고	별도의 데이터베이스가 있는 브리지	별도의 스키마가 있는 브리지	풀
데이터베이스 유지 관리 (진공 관리) 및 리소스 사용	간단한 관리	중간 수준의 복잡 (이후 각 데이터베이스에 대해 진공 작업자를 시작해야 하기 때문에 리소스 소비가 높아질 수 있으며`vacuum_naptime` , 이로 인해 autovacuum Launcher CPU 사용량이 높아질 수 있습니다. 또한 각 데이터베이스의 PostgreSQL 시스템 카탈로그 테이블을 정리하는 것과 관련된 추가 오버헤드가 발생할 수 있습니다.)	대형 PostgreSQL 시스템 카탈로그 테이블 (테넌트 수와 관계에 따라 층`pg_catalog` 크기는 수십 GB 단위입니다. 테이블 팽창을 제어하려면 진공 청소기 관련 매개 변수를 수정해야 할 수 있습니다.)	테넌트 수와 테넌트당 데이터에 따라 테이블이 클 수 있습니다. (테이블 팽창을 제어하려면 진공 청소기 관련 매개 변수를 수정해야 할 수 있습니다.)
확장 프로그램 관리 노력	상당한 노력 (개별 인스턴스의 각 데이터베이스에 대해)	상당한 노력 (각 데이터베이스 수준에서)	최소한의 노력 (공통 데이터베이스에서 한 번)	최소한의 노력 (공통 데이터베이스에서 한 번)
배포 작업 변경	중요한 작업 (각 개별 인스턴스에 Connect 변경 사항을 롤아웃하십시오.)	중요한 작업 (각 데이터베이스 및 스키마에 Connect 변경 사항을 롤아웃하십시오.)	적당한 노력. (공통 데이터베이스에 Connect 각 스키마에 대한 변경 내용을 롤아웃합니다.)	최소한의 노력. (공통 데이터베이스에 Connect 변경 사항을 적용하십시오.)

	저장고	별도의 데이터베이스가 있는 브리지	별도의 스키마가 있는 브리지	풀
변경 배포 — 영향 범위	최소화 (단일 테넌트가 영향을 받습니다.)	최소화 (단일 테넌트가 영향을 받습니다.)	최소화 (단일 테넌트가 영향을 받습니다.)	매우 큽니다. (모든 임차인이 영향을 받습니다.)
쿼리 성능 관리 및 작업	관리 가능한 쿼리 성능.	관리 가능한 쿼리 성능.	관리 가능한 쿼리 성능.	쿼리 성능을 유지하려면 상당한 노력이 필요할 수 있습니다. (시간이 지남에 따라 테이블 크기 증가로 인해 쿼리가 더 느리게 실행될 수 있습니다. 테이블 파티셔닝 및 데이터베이스 샤딩을 사용하여 성능을 유지할 수 있습니다.)
테넌트 간 리소스 영향	영향 (테넌트 간 리소스 공유 없음)	중간 영향 (테넌트는 인스턴스 CPU 및 메모리와 같은 공통 리소스를 공유합니다.)	중간 영향 (테넌트는 인스턴스 CPU 및 메모리와 같은 공통 리소스를 공유합니다.)	큰 영향 (테넌트는 리소스, 잠금 충돌 등의 측면에서 서로 영향을 미칩니다.)
테넌트 수준 조정 (예: 테넌트당 추가 인덱스 생성 또는 특정 테넌트에 대한 DB 매개변수 조정)	가능한.	다소 가능한 (각 테넌트에 대해 스키마 수준 변경이 가능하지만 데이터베이스 매개변수는 모든 테넌트에서 전역적으로 적용됩니다.)	다소 가능한 (각 테넌트에 대해 스키마 수준 변경이 가능하지만 데이터베이스 매개변수는 모든 테넌트에서 전역적으로 적용됩니다.)	가능한 한 경과. (테이블은 모든 테넌트가 공유합니다.)

	저장고	별도의 데이터베이스가 있는 브리지	별도의 스키마가 있는 브리지	풀
성능에 민감한 테넌트를 위한 노력 재조정	최소화 (밸런스를 조정할 필요가 없습니다. 이 시나리오를 처리하기 위해 서버 및 I/O 리소스를 확장하십시오.)	보통 (논리적 복제를 사용하거나 데이터베이스를 내보내는pg_dump 데 사용하지만 데이터 크기에 따라 가동 중지 시간이 길어질 수 있습니다. Amazon RDS for PostgreSQL RDS의 전송 가능한 데이터베이스 기능을 사용하여 인스턴스 간에 데이터베이스를 더 빠르게 복사할 수 있습니다.)	보통이지만 다운타임이 오래 걸릴 수 있습니다. (논리적 복제를 사용하거나 스키마를 내보내는pg_dump 데 사용하지만 데이터 크기에 따라 가동 중지 시간이 길어질 수 있습니다.)	모든 테넌트가 동일한 테이블을 공유하기 때문에 중요합니다. (데이터베이스를 분할하려면 모든 항목을 다른 인스턴스에 복사하고 테넌트 데이터를 정리하기 위한 추가 단계가 필요합니다.) 애플리케이션 로직의 변경이 필요할 가능성이 높습니다.
메이저 버전 업그레이드를 위한 데이터베이스 다운타임	표준 가동 중지 중지 중지 (PostgreSQL 시스템 카탈로그 크기에 따라 다릅니다.)	다운타임이 더 길어질 수 있습니다. (시스템 카탈로그 크기에 따라 시간이 달라집니다. PostgreSQL 시스템 카탈로그 테이블도 데이터베이스 간에 복제됩니다.)	다운타임이 더 길어질 수 있습니다. (PostgreSQL 시스템 카탈로그 크기에 따라 시간이 달라집니다.)	표준 가동 중지 중지 중지 (PostgreSQL 시스템 카탈로그 크기에 따라 다릅니다.)

	저장고	별도의 데이터베이스가 있는 브리지	별도의 스키마가 있는 브리지	풀
관리 오버헤드(예: 데이터베이스 로그 분석 또는 백업 작업 모니터링용)	중요한 작업	최소한의 노력.	최소한의 노력.	최소한의 노력.
테넌트 수준의 가용성	최하위. (각 테넌트에 장애가 발생하고 독립적으로 복구됩니다.)	더 높은 영향 범위. (하드웨어 또는 리소스 문제가 발생할 경우 모든 테넌트에 장애가 발생하고 함께 복구됩니다.)	더 높은 영향 범위. (하드웨어 또는 리소스 문제가 발생할 경우 모든 테넌트에 장애가 발생하고 함께 복구됩니다.)	더 높은 영향 범위. (하드웨어 또는 리소스 문제가 발생할 경우 모든 테넌트에 장애가 발생하고 함께 복구됩니다.)
테넌트 수준의 백업 및 복구 노력	최소 (각 테넌트별로 개별적으로 백업 및 복원할 수 있습니다.)	적당한 노력. (각 테넌트에 대해 논리적 내보내기 및 가져오기를 사용하십시오. 일부 코딩 및 자동화가 필요합니다.)	적당한 노력. (각 테넌트에 대해 논리적 내보내기 및 가져오기를 사용하십시오. 일부 코딩 및 자동화가 필요합니다.)	중요한 작업 (모든 임차인이 동일한 테이블을 공유합니다.)
테넌트 수준의 point-in-time 복구 노력	최소한의 노력. (스냅샷을 사용하여 특정 시점 복구를 사용하거나 Amazon Aurora Aurora에서 백트래킹을 사용하십시오.)	적당한 노력. (스냅샷 복원을 사용한 다음 내보내기/가져오기를 사용합니다. 하지만 이렇게 하면 작업 속도가 느려질 수 있습니다.)	적당한 노력. (스냅샷 복원을 사용한 다음 내보내기/가져오기를 사용합니다. 하지만 이렇게 하면 작업 속도가 느려질 수 있습니다.)	상당한 노력과 복잡성.
동일한 스키마	테넌트별로 동일한 스키마 이름.	테넌트별로 동일한 스키마 이름.	테넌트별로 서로 다른 스키마.	공통 스키마입니다.

	저장고	별도의 데이터베이스가 있는 브리지	별도의 스키마가 있는 브리지	풀
테넌트별 사용자 지정 (예: 특정 테넌트에 대한 추가 테이블 열)	가능한.	가능한.	가능한.	복잡합니다 (모든 테넌트가 동일한 테이블을 공유하기 때문에).
객체 관계형 매핑 (ORM) 계층 (예: Ruby) 에서의 카탈로그 관리 효율성	효율적입니다 (클라이언트 연결이 테넌트별로 다르기 때문에).	효율적입니다 (클라이언트 연결이 데이터베이스에만 해당되기 때문에).	적당히 효율적입니다. (사용된 ORM, 사용자/역할 보안 모델 및`search_path` 구성에 따라 클라이언트가 모든 테넌트의 메타데이터를 캐시하는 경우가 있어 DB 연결 메모리 사용량이 높아지는 경우가 있습니다.)	효율적입니다 (모든 테넌트가 동일한 테이블을 공유하기 때문에).
통합된 테넌트 보고 작업	중요한 작업 (모든 테넌트의 데이터를 통합하거나 [ETL] 을 추출, 변환 및 다른 보고 데이터베이스로 로드하려면 외부 데이터 래퍼 [FDW] 를 사용해야 합니다.)	중요한 작업 (FDW를 사용하여 모든 테넌트 또는 ETL의 데이터를 다른 보고 데이터베이스로 통합해야 합니다.)	적당한 노력. (유니온을 사용하여 모든 스키마의 데이터를 집계할 수 있습니다.)	최소한의 노력. (모든 테넌트 데이터는 동일한 테이블에 있으므로 보고가 간단합니다.)

	저장고	별도의 데이터베이스가 있는 브리지	별도의 스키마가 있는 브리지	풀
보고용 테넌트별 읽기 전용 인스턴스 (예: 구독 기반)	최소 (읽기 전용 복제본을 생성합니다.)	적당한 노력. (논리적 복제 또는 AWS Database Migration Service [AWS DMS] 를 사용하여 구성할 수 있습니다.)	적당한 노력. (논리적 복제를 사용하거나 구성할 AWS DMS 수 있습니다.)	복잡합니다 (모든 테넌트가 동일한 테이블을 공유하기 때문에).
데이터 격리	최하위.	상급. (PostgreSQL 역할을 사용하여 데이터베이스 수준 권한을 관리할 수 있습니다.)	상급. (PostgreSQL 역할을 사용하여 스키마 수준 권한을 관리할 수 있습니다.)	더 나쁜. (모든 테넌트가 동일한 테이블을 공유하므로 테넌트 격리를 위한 행 수준 보안 [RLS] 와 같은 기능을 구현해야 합니다.)
테넌트별 스토리지 암호화 키	가능한. (각 PostgreSQL 클러스터에는 스토리지 암호화를 위한 자체AWS Key Management Service [AWS KMS] 키가 있을 수 있습니다.)	가능한 한 경과. (모든 테넌트는 스토리지 암호화를 위해 동일한 KMS 키를 공유합니다.)	가능한 한 경과. (모든 테넌트는 스토리지 암호화를 위해 동일한 KMS 키를 공유합니다.)	가능한 한 경과. (모든 테넌트는 스토리지 암호화를 위해 동일한 KMS 키를 공유합니다.)

	저장고	별도의 데이터베이스가 있는 브리지	별도의 스키마가 있는 브리지	풀
각 테넌트의 데이터베이스 인증에AWS Identity and Access Management (IAM) 사용	가능한.	가능한.	가능 (각 스키마에 대해 별도의 PostgreSQL 사용자를 사용하는 경우).	불가능합니다 (모든 테넌트가 테이블을 공유하기 때문에).
인프라	가장 높음 (아무것도 공유되지 않기 때문에)	보통	보통	최하위.
데이터 복제 및 스토리지 사용	모든 테넌트 중 가장 높은 집계. (PostgreSQL 시스템 카탈로그 테이블과 애플리케이션의 정적 및 공통 데이터는 모든 테넌트에서 중복됩니다.)	모든 테넌트 중 가장 높은 집계. (PostgreSQL 시스템 카탈로그 테이블과 애플리케이션의 정적 및 공통 데이터는 모든 테넌트에서 중복됩니다.)	보통 (애플리케이션의 정적 및 공통 데이터는 공통 스키마에 있고 다른 테넌트가 액세스할 수 있습니다.)	최소화 (데이터 중복 없음. 애플리케이션의 정적 데이터와 공통 데이터는 동일한 스키마에 있을 수 있습니다.)
테넌트 중심 모니터링 (문제를 일으키는 테넌트를 빠르게 파악)	최소 (각 테넌트가 개별적으로 모니터링되므로 특정 테넌트의 활동을 쉽게 확인할 수 있습니다.)	적당한 노력. (모든 테넌트가 동일한 물리적 리소스를 공유하므로 특정 테넌트의 활동을 확인하려면 추가 필터링을 적용해야 합니다.)	적당한 노력. (모든 테넌트가 동일한 물리적 리소스를 공유하므로 특정 테넌트의 활동을 확인하려면 추가 필터링을 적용해야 합니다.)	중요한 작업 (모든 테넌트가 테이블을 비롯한 모든 리소스를 공유하므로 바인드 변수 캡처를 사용하여 특정 SQL 쿼리가 속한 테넌트를 확인해야 합니다.)

	저장고	별도의 데이터베이스가 있는 브리지	별도의 스키마가 있는 브리지	풀
중앙 집중식 관리 및 상태/활동 모니터링	상당한 노력 (중앙 모니터링 및 중앙 지휘 센터 설치).	모든 테넌트가 동일한 인스턴스를 공유하기 때문에 작업량이 적당합니다.	모든 테넌트가 동일한 인스턴스를 공유하기 때문에 작업량이 적당합니다.	최소한의 노력 (모든 테넌트가 스키마를 포함하여 동일한 리소스를 공유하기 때문)
객체 식별자 (OID) 및 트랜잭션 ID (XID) 를 한눈에 파악할 수 있는 기회	최소화	높음. (OID, XID는 단일 PostgreSQL 클러스터 전체 카운터이기 때문에 물리적 데이터베이스를 효과적으로 정리하는 데 문제가 있을 수 있습니다.)	보통 (OID, XID는 단일 PostgreSQL 클러스터 전체 카운터이기 때문입니다).	높음. (예를 들어 단일 테이블은 out-of-line 열 수에 따라 TOAST OID 제한인 40억 개에 도달할 수 있습니다.)

행 수준 보안 권장 사항

PostgreSQL을 사용하는 풀링된 모델에서 테넌트 데이터 격리를 유지하려면 행 수준 보안 (RLS) 이 필요합니다. RLS는 격리 정책을 데이터베이스 수준에서 중앙 집중화하여 소프트웨어 개발자가 이러한 격리를 유지해야 하는 부담을 덜어줍니다. RLS를 구현하는 가장 일반적인 방법은 PostgreSQL DBMS에서 이 기능을 활성화하는 것입니다. RLS에는 지정된 열의 값을 기준으로 데이터 행에 대한 액세스를 필터링하는 작업이 포함됩니다. 다음 두 가지 방법을 사용하여 데이터에 대한 액세스를 필터링할 수 있습니다.

- 테이블의 지정된 데이터 열을 현재 PostgreSQL 사용자의 값과 비교합니다. 해당 사용자는 로그인한 PostgreSQL 사용자와 동일한 열의 값에 액세스할 수 있습니다.
- 테이블의 지정된 데이터 열을 응용 프로그램에서 설정한 런타임 변수 값과 비교합니다. 해당 세션 중에 런타임 변수와 동일한 열의 값에 액세스할 수 있습니다.

첫 번째 옵션을 사용하려면 각 테넌트에 대해 새 PostgreSQL 사용자를 만들어야 하므로 두 번째 옵션을 사용하는 것이 좋습니다. 대신 PostgreSQL을 사용하는 SaaS 애플리케이션은 PostgreSQL을 쿼리할 때 런타임에 테넌트별 컨텍스트를 설정해야 합니다. 이는 RLS를 시행하는 효과가 있습니다. RLS를 table-by-table 기반으로 활성화할 수도 있습니다. 가장 좋은 방법은 테넌트 데이터가 포함된 모든 테이블에서 RLS를 사용하도록 설정하는 것입니다.

다음 예제에서는 테이블 두 개를 만들고 RLS를 활성화합니다. 이 예제에서는 데이터 열을 런타임 변수 값과`app.current_tenant` 비교합니다.

```
-- Create a table for our tenants with indexes on the primary key and the tenant’s name
CREATE TABLE tenant (
    tenant_id UUID DEFAULT uuid_generate_v4() PRIMARY KEY,
    name VARCHAR(255) UNIQUE,
    status VARCHAR(64) CHECK (status IN ('active', 'suspended', 'disabled')),
    tier VARCHAR(64) CHECK (tier IN ('gold', 'silver', 'bronze'))
);

-- Create a table for users of a tenant
CREATE TABLE tenant_user (
    user_id UUID DEFAULT uuid_generate_v4() PRIMARY KEY,
    tenant_id UUID NOT NULL REFERENCES tenant (tenant_id) ON DELETE RESTRICT,
    email VARCHAR(255) NOT NULL UNIQUE,
    given_name VARCHAR(255) NOT NULL CHECK (given_name <> ''),
    family_name VARCHAR(255) NOT NULL CHECK (family_name <> '')
);
```

```
-- Turn on RLS
ALTER TABLE tenant ENABLE ROW LEVEL SECURITY;

-- Restrict read and write actions so tenants can only see their rows
-- Cast the UUID value in tenant_id to match the type current_setting
-- This policy implies a WITH CHECK that matches the USING clause
CREATE POLICY tenant_isolation_policy ON tenant
USING (tenant_id = current_setting('app.current_tenant')::UUID);

-- And do the same for the tenant users
ALTER TABLE tenant_user ENABLE ROW LEVEL SECURITY;

CREATE POLICY tenant_user_isolation_policy ON tenant_user
USING (tenant_id = current_setting('app.current_tenant')::UUID);
```

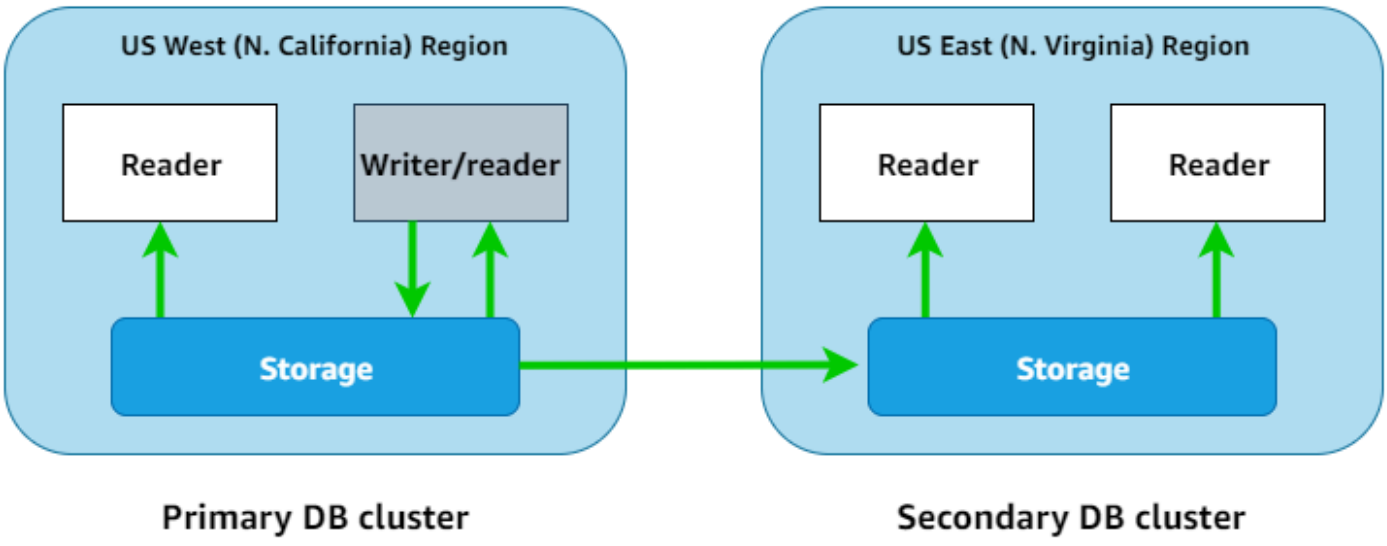
자세한 내용은 블로그 게시물 PostgreSQL 행 수준 보안을 사용한 다중 테넌트 데이터 격리를 참조하세요. AWSSaaS 팩토리 팀에는 RLS GitHub 구현을 지원하는 몇 가지 예도 있습니다.

풀 모델을 위한 PostgreSQL 가용성

풀 모델은 특성상 PostgreSQL 인스턴스가 하나만 있습니다. 따라서 고가용성을 위해 애플리케이션을 설계하는 것이 중요합니다. 풀링된 데이터베이스에 장애가 발생하거나 중단되면 애플리케이션 성능이 저하되거나 모든 테넌트가 액세스할 수 없게 됩니다.

고가용성 기능을 활성화하여 PostgreSQL용 Amazon RDS DB 인스턴스를 두 가용 영역에 중복으로 만들 수 있습니다. 자세한 내용은 Amazon RDS 설명서의 Amazon RDS용 고가용성 (다중 AZ) 을 참조하십시오. 지역 간 장애 조치의 경우 다른 지역에 읽기 전용 복제본을 생성할 수 있습니다. AWS (이 읽기 전용 복제본은 장애 조치 프로세스의 일부로 승격되어야 합니다.) 또한 복구를 위해 여러 지역에 AWS 복제된 백업을 복제할 수 있습니다. 자세한 내용은 Amazon RDS 설명서의 자동 백업을 다른 AWS 지역으로 복제를 참조하십시오.

Aurora PostgreSQL과 호환되므로 여러 가용 영역에서 장애가 발생해도 문제가 발생하지 않도록 데이터를 자동으로 백업합니다. (Aurora 설명서의 Amazon Aurora의 고가용성을 참조하십시오.) Aurora의 복원력을 높이고 복구 속도를 높이려면 다른 가용 영역에 Aurora 읽기 전용 복제본을 만들 수 있습니다. Aurora 글로벌 데이터베이스를 사용하여 리전 간 복구 및 자동 장애 조치를 위해 데이터를 5개의 추가 AWS 지역에 복제할 수 있습니다. (Aurora 설명서의 Amazon Aurora 글로벌 데이터베이스 사용을 참조하십시오.) 또한 Aurora 글로벌 데이터베이스에서 쓰기 전달을 활성화하여 여러 데이터베이스에서 고가용성을 달성할 수 있습니다. AWS 리전



PostgreSQL용 Amazon RDS를 사용하든 Aurora PostgreSQL과 호환되는지 여부에 관계없이 풀 모델을 사용하는 모든 멀티테넌트 SaaS 애플리케이션에 대해 중단으로 인한 영향을 완화할 수 있는 고가용성 기능을 구현하는 것이 좋습니다.

모범 사례

이 섹션에는 이 가이드의 몇 가지 개략적인 내용이 나열되어 있습니다. 각 항목에 대한 자세한 내용은 해당 섹션의 링크를 참조하십시오.

관리형 PostgreSQL에 대한AWS 옵션 비교

AWS는 관리 환경에서 PostgreSQL을 실행하는 두 가지 기본 방법을 제공합니다. (이 컨텍스트에서 관리형이란 PostgreSQL 인프라 및 DBMS를AWS 서비스에서 부분적으로 또는 완전히 지원한다는 의미입니다.) 의 관리형 PostgreSQL 옵션은 PostgreSQL의 백업, 페일오버, 최적화 및 일부 관리를 자동화하는 이점이 있습니다.AWS 관리형 옵션으로 PostgreSQL용 Amazon Aurora PostgreSQL 호환 버전 및 Amazon RDS (Amazon Realtional Database Service) 를AWS 제공합니다. PostgreSQL 사용 사례를 분석하여 이 두 모델 중에서 가장 적합한 모델을 선택할 수 있습니다. 자세한 내용은 이 설명서 단원을 참조하십시오. Amazon RDS와 Aurora 중에서 선택

멀티 테넌트 SaaS 파티셔닝 모델 선택

PostgreSQL에 적용할 수 있는 세 가지 SaaS 파티셔닝 모델 (사일로, 브리지, 풀) 중에서 선택할 수 있습니다. 각 모델에는 장단점이 있으므로 사용 사례에 따라 가장 적합한 모델을 선택해야 합니다. Amazon RDS for PostgreSQL RDS와 Aurora PostgreSQL 호환은 세 가지 모델을 모두 지원합니다. SaaS 애플리케이션에서 테넌트 데이터 격리를 유지하려면 모델 선택이 중요합니다. 이러한 모델에 대한 자세한 내용은 이 가이드의 PostgreSQL용 멀티 테넌트 SaaS 파티셔닝 모델 섹션을 참조하십시오.

풀 SaaS 파티셔닝 모델에 행 수준 보안 사용

PostgreSQL을 사용하는 풀 모델에서 테넌트 데이터 격리를 유지하려면 행 수준 보안 (RLS) 이 필요합니다. 풀 모델에서는 테넌트별로 인프라, PostgreSQL 데이터베이스 또는 스키마를 논리적으로 구분하지 않기 때문입니다. RLS는 격리 정책을 데이터베이스 수준에서 중앙 집중화하여 소프트웨어 개발자가 이러한 격리를 유지해야 하는 부담을 덜어줍니다. RLS를 사용하여 데이터베이스 작업을 특정 테넌트로 제한할 수 있습니다. 자세한 내용과 예는 이 안내서의 행 수준 보안 권장 사항 섹션을 참조하십시오.

FAQ

이 섹션에서는 멀티 테넌트 SaaS 애플리케이션에서 관리형 PostgreSQL을 구현하는 것과 관련하여 자주 제기되는 질문에 대한 답변을 제공합니다.

어떤 관리형 PostgreSQL 옵션을AWS 제공합니까?

AWSPostgreSQL용 Amazon Aurora PostgreSQL 호환 및 Amazon Relational Database Service (Amazon RDS) 를 제공합니다. AWS또한 광범위한 관리형 데이터베이스 제품 카탈로그를 제공합니다.

SaaS 애플리케이션에 가장 적합한 서비스는 무엇입니까?

Aurora PostgreSQL 호환 모델과 SaaS 애플리케이션용 PostgreSQL용 Amazon RDS 및 이 가이드에서 설명하는 모든 SaaS 파티셔닝 모델을 모두 사용할 수 있습니다. 이 두 서비스는 확장성, 장애 복구, 장애 조치, 스토리지 옵션, 고가용성, 재해 복구, 백업 및 각 옵션에서 사용할 수 있는 인스턴스 클래스에 차이가 있습니다. 최적의 선택은 특정 사용 사례에 따라 다릅니다. 이 가이드의 의사 결정 매트릭스를 사용하여 사용 사례에 가장 적합한 옵션을 선택하세요.

PostgreSQL 데이터베이스를 멀티 테넌트 SaaS 애플리케이션과 함께 사용하기로 결정한 경우 어떤 고유한 요구 사항을 고려해야 합니까?

SaaS 애플리케이션에서 사용되는 모든 데이터 스토어와 마찬가지로 가장 중요한 고려 사항은 테넌트 데이터 격리를 유지하는 방법입니다. 이 가이드에서 설명한 것처럼AWS 관리형 PostgreSQL 오퍼링을 사용하여 테넌트 데이터를 격리할 수 있는 여러 가지 방법이 있습니다. 또한 모든 PostgreSQL 구현의 경우 테넌트별로 성능 격리를 고려해야 합니다.

PostgreSQL에서 테넌트 데이터 격리를 유지하기 위해 어떤 모델을 사용할 수 있습니까?

사일로, 브리지 및 풀 모델을 SaaS 파티셔닝 전략으로 사용하여 테넌트 데이터 격리를 유지할 수 있습니다. 이러한 모델과 이를 PostgreSQL에 적용하는 방법에 대한 설명은 이 가이드의 PostgreSQL용 멀티 테넌트 SaaS 파티셔닝 모델 섹션을 참조하십시오.

여러 테넌트에서 공유되는 단일 PostgreSQL 데이터베이스를 사용하여 테넌트 데이터 격리를 유지하려면 어떻게 해야 합니까?

PostgreSQL은 단일 PostgreSQL 데이터베이스 또는 인스턴스에서 테넌트 데이터 격리를 적용하는 데 사용할 수 있는 행 수준 보안 (RLS) 기능을 지원합니다. 또한 단일 인스턴스에서 테넌트별로 별도의 PostgreSQL 데이터베이스를 프로비저닝하거나 테넌트별로 스키마를 생성하여 이 목표를 달성할 수 있습니다. 이러한 접근 방식의 장점과 단점은 이 안내서의 행 수준 보안 권장 사항 섹션을 참조하십시오.

다음 단계

AWS는 관리형 PostgreSQL을 운영하기 위한 두 가지 옵션, 즉 Aurora PostgreSQL과 호환되는 옵션과 Amazon RDS for PostgreSQL RDS를 제공합니다. 두 서비스를 평가하고 멀티테넌트 SaaS 애플리케이션의 특정 사용 사례를 가장 잘 지원하는 옵션을 선택하는 것이 좋습니다. SaaS 파티셔닝 모델을 준수하면 PostgreSQL을 사용하는 SaaS 애플리케이션이 모범 사례를 엄격하게 준수하여 테넌시를 유지할 수 있습니다. SaaS 사일로, 브리지 및 풀 파티셔닝 모델은 다양한 SaaS 사용 사례를 지원합니다. 이러한 모델은 성능 격리, 운영 오버헤드 및 테넌트 보안과 같은 요소에 따라 다양한 이점을 제공합니다.

다음 단계:

- Aurora PostgreSQL과 호환되는 Amazon RDS for PostgreSQL RDS를 평가하고 SaaS 애플리케이션에 가장 적합한 옵션을 선택하십시오.
- 애플리케이션 요구 사항 (사일로, 브리지 또는 풀) 을 충족하는 SaaS 파티셔닝 모델을 선택하십시오.
- 선택한 SaaS 파티셔닝 모델에 따라 PostgreSQL을 구현하십시오.

리소스

참조

- SaaS 스토리지 전략: 멀티테넌트 스토리지 모델 구축 (백서 AWS) AWS
- Amazon Aurora PostgreSQL용 Amazon Aurora 글로벌 데이터베이스를 사용한 지역 간 재해 복구 (블로그 게시물) AWS
- PostgreSQL 행 수준 보안을 사용한 멀티테넌트 데이터 격리 (블로그 게시물) AWS
- Amazon Aurora PostgreSQL 작업(Aurora 설명서)
- 아마존 RDS에서의 PostgreSQL (아마존 RDS 설명서)

파트너

- PostgreSQL 파트너를 위한 Amazon Aurora
- PostgreSQL 파트너를 위한 아마존 RDS

문서 기록

아래 표에 이 가이드의 주요 변경 사항이 설명되어 있습니다. 향후 업데이트에 대한 알림을 받으려면 RSS 피드를 구독하십시오.

변경 사항	설명	날짜
업데이트	Aurora에서의 쓰기 전달 기능을 반영하도록 업데이트되었습니다.	2024년 4월 29일
업데이트	Amazon RDS와 Aurora 비교표가 업데이트되었습니다.	2022년 10월 21일
—	최초 게시	2021년 9월 30일

AWS 규범적 지침 용어집

다음은 규범적 지침에서 제공하는 AWS 전략, 가이드 및 패턴에서 일반적으로 사용되는 용어입니다. 용어집 항목을 제안하려면 용어집 끝에 있는 피드백 제공 링크를 사용하십시오.

숫자

7가지 전략

애플리케이션을 클라우드로 이전하기 위한 7가지 일반적인 마이그레이션 전략 이러한 전략은 Gartner가 2011년에 파악한 5가지 전략을 기반으로 하며 다음으로 구성됩니다.

- 리팩터링/리아키텍트 - 클라우드 네이티브 기능을 최대한 활용하여 애플리케이션을 이동하고 해당 아키텍처를 수정함으로써 민첩성, 성능 및 확장성을 개선합니다. 여기에는 일반적으로 운영 체제와 데이터베이스 이식이 포함됩니다. 예: 온프레미스 Oracle 데이터베이스를 Amazon Aurora PostgreSQL 호환 에디션으로 마이그레이션하십시오.
- 리플랫포밍(리프트 앤드 리셰이프) - 애플리케이션을 클라우드로 이동하고 일정 수준의 최적화를 도입하여 클라우드 기능을 활용합니다. 예: 온프레미스 Oracle 데이터베이스를 오라클용 Amazon RDS (Amazon RDS) 로 마이그레이션합니다. AWS 클라우드
- 재구매(드롭 앤드 숍) - 일반적으로 기존 라이선스에서 SaaS 모델로 전환하여 다른 제품으로 전환합니다. 예: 고객 관계 관리 (CRM) 시스템을 Salesforce.com으로 마이그레이션하십시오.
- 리호스팅(리프트 앤드 시프트) - 애플리케이션을 변경하지 않고 클라우드로 이동하여 클라우드 기능을 활용합니다. 예: 온프레미스 Oracle 데이터베이스를 의 EC2 인스턴스에서 Oracle로 마이그레이션합니다. AWS 클라우드
- 재배치(하이퍼바이저 수준의 리프트 앤 시프트) - 새 하드웨어를 구매하거나, 애플리케이션을 다시 작성하거나, 기존 운영을 수정하지 않고도 인프라를 클라우드로 이동합니다. 온프레미스 플랫폼에서 동일한 플랫폼의 클라우드 서비스로 서버를 마이그레이션합니다. 예: Microsoft Hyper-V 애플리케이션을 다음으로 마이그레이션하십시오. AWS
- 유지(보관) - 소스 환경에 애플리케이션을 유지합니다. 대규모 리팩터링이 필요하고 해당 작업을 나중으로 연기하려는 애플리케이션과 비즈니스 차원에서 마이그레이션할 이유가 없어 유지하려는 레거시 애플리케이션이 여기에 포함될 수 있습니다.
- 사용 중지 - 소스 환경에서 더 이상 필요하지 않은 애플리케이션을 폐기하거나 제거합니다.

A

ABAC

속성 기반 액세스 제어를 참조하십시오.

추상화된 서비스

관리형 서비스를 참조하십시오.

산

원자성, 일관성, 격리성, 내구성을 참조하십시오.

능동-능동 마이그레이션

양방향 복제 도구 또는 이중 쓰기 작업을 사용하여 소스 데이터베이스와 대상 데이터베이스가 동기화된 상태로 유지되고, 두 데이터베이스 모두 마이그레이션 중 연결 애플리케이션의 트랜잭션을 처리하는 데이터베이스 마이그레이션 방법입니다. 이 방법은 일회성 전환이 필요한 대신 소규모의 제어된 배치로 마이그레이션을 지원합니다. 더 유연하지만 액티브-패시브 마이그레이션보다 더 많은 작업이 필요합니다.

능동-수동 마이그레이션

소스 데이터베이스와 대상 데이터베이스가 동기화된 상태로 유지되지만 소스 데이터베이스만 연결 애플리케이션의 트랜잭션을 처리하고 데이터는 대상 데이터베이스로 복제되는 데이터베이스 마이그레이션 방법입니다. 대상 데이터베이스는 마이그레이션 중 어떤 트랜잭션도 허용하지 않습니다.

집계 함수

행 그룹에서 연산을 수행하고 그룹에 대한 단일 반환값을 계산하는 SQL 함수입니다. 집계 함수의 예로는 및 등이 `SUM` 있습니다. `MAX`

AI

인공 지능을 참조하십시오.

AIOps

인공 지능 운영을 참조하십시오.

익명화

데이터세트에서 개인 정보를 영구적으로 삭제하는 프로세스입니다. 익명화는 개인 정보 보호에 도움이 될 수 있습니다. 익명화된 데이터는 더 이상 개인 데이터로 간주되지 않습니다.

안티 패턴

솔루션이 다른 솔루션보다 비생산적이거나 비효율적이거나 덜 효과적이어서 반복되는 문제에 자주 사용되는 솔루션입니다.

애플리케이션 제어

시스템을 멀웨어로부터 보호하기 위해 승인된 애플리케이션만 사용할 수 있는 보안 접근 방식입니다.

애플리케이션 포트폴리오

애플리케이션 구축 및 유지 관리 비용과 애플리케이션의 비즈니스 가치를 비롯하여 조직에서 사용하는 각 애플리케이션에 대한 세부 정보 모음입니다. 이 정보는 포트폴리오 검색 및 분석 프로세스의 핵심이며 마이그레이션, 현대화 및 최적화할 애플리케이션을 식별하고 우선순위를 정하는 데 도움이 됩니다.

인공 지능

컴퓨터 기술을 사용하여 학습, 문제 해결, 패턴 인식 등 일반적으로 인간과 관련된 인지 기능을 수행하는 것을 전문으로 하는 컴퓨터 과학 분야입니다. 자세한 내용은 What is Artificial Intelligence?를 참조하십시오.

인공 지능 운영(AIOps)

기계 학습 기법을 사용하여 운영 문제를 해결하고, 운영 인시던트 및 사용자 개입을 줄이고, 서비스 품질을 높이는 프로세스입니다. AWS 마이그레이션 전략에서 AIOps가 사용되는 방법에 대한 자세한 내용은 운영 통합 가이드를 참조하십시오.

비대칭 암호화

한 쌍의 키, 즉 암호화를 위한 퍼블릭 키와 복호화를 위한 프라이빗 키를 사용하는 암호화 알고리즘입니다. 퍼블릭 키는 복호화에 사용되지 않으므로 공유할 수 있지만 프라이빗 키에 대한 액세스는 엄격히 제한되어야 합니다.

원자성, 일관성, 격리성, 내구성(ACID)

오류, 정전 또는 기타 문제가 발생한 경우에도 데이터베이스의 데이터 유효성과 운영 신뢰성을 보장하는 소프트웨어 속성 세트입니다.

ABAC(속성 기반 액세스 제어)

부서, 직무, 팀 이름 등의 사용자 속성을 기반으로 세분화된 권한을 생성하는 방식입니다. 자세한 내용은 AWS Identity and Access Management (IAM) 설명서의 AWS ABAC for를 참조하십시오.

신뢰할 수 있는 데이터 소스

가장 신뢰할 수 있는 정보 소스로 간주되는 기본 버전의 데이터를 저장하는 위치입니다. 익명화, 편집 또는 가명화와 같은 데이터 처리 또는 수정의 목적으로 신뢰할 수 있는 데이터 소스의 데이터를 다른 위치로 복사할 수 있습니다.

가용 영역

다른 가용 영역의 장애로부터 격리되고 동일한 지역 내 다른 가용 영역에 저렴하고 지연 시간이 짧은 네트워크 연결을 제공하는 별도의 위치. AWS 리전

AWS 클라우드 채택 프레임워크 (AWS CAF)

조직이 클라우드로 성공적으로 AWS 전환하기 위한 효율적이고 효과적인 계획을 개발하는 데 도움이 되는 지침 및 모범 사례 프레임워크입니다. AWS CAF는 지침을 관점이라고 하는 6가지 중점 영역, 즉 비즈니스, 사람, 거버넌스, 플랫폼, 보안, 운영으로 분류합니다. 비즈니스, 사람 및 거버넌스 관점은 비즈니스 기술과 프로세스에 초점을 맞추고, 플랫폼, 보안 및 운영 관점은 전문 기술과 프로세스에 중점을 둡니다. 예를 들어, 사람 관점은 인사(HR), 직원 배치 기능 및 인력 관리를 담당하는 이해관계자를 대상으로 합니다. 이러한 관점에서 AWS CAF는 조직이 성공적인 클라우드 채택을 준비할 수 있도록 인력 개발, 교육 및 커뮤니케이션에 대한 지침을 제공합니다. 자세한 내용은 AWS CAF 웹 사이트와 AWS CAF 백서를 참조하십시오.

AWS 워크로드 검증 프레임워크 (AWS WQF)

데이터베이스 마이그레이션 워크로드를 평가하고 마이그레이션 전략을 권장하며 작업 예상치를 제공하는 도구입니다. AWS WQF는 () 에 포함됩니다. AWS Schema Conversion Tool AWS SCT데이터베이스 스키마 및 코드 객체, 애플리케이션 코드, 종속성 및 성능 특성을 분석하고 평가 보고서를 제공합니다.

B

배드 봇

개인이나 조직을 방해하거나 피해를 입히려는 의도를 가진 봇입니다.

BCP

비즈니스 연속성 계획을 참조하십시오.

동작 그래프

리소스 동작과 시간 경과에 따른 상호 작용에 대한 통합된 대화형 뷰입니다. Amazon Detective에서 동작 그래프를 사용하여 실패한 로그온 시도, 의심스러운 API 호출 및 유사한 작업을 검사할 수 있습니다. 자세한 내용은 Detective 설명서의 Data in a behavior graph를 참조하십시오.

빅 엔디안 시스템

가장 중요한 바이트를 먼저 저장하는 시스템입니다. 엔디안도 참조하십시오.

바이너리 분류

바이너리 결과(가능한 두 클래스 중 하나)를 예측하는 프로세스입니다. 예를 들어, ML 모델이 “이 이메일이 스팸인가요, 스팸이 아닌가요?”, '이 제품은 책인가요, 자동차인가요?' 등의 문제를 예측해야 할 수 있습니다.

블룸 필터

요소가 세트의 멤버인지 여부를 테스트하는 데 사용되는 메모리 효율성이 높은 확률론적 데이터 구조입니다.

블루/그린(Blue/Green) 배포

서로 다르지만 동일한 환경을 두 개 만드는 배포 전략입니다. 현재 애플리케이션 버전을 한 환경 (파란색) 에서 실행하고 다른 환경 (녹색) 에서 새 애플리케이션 버전을 실행합니다. 이 전략을 사용하면 영향을 최소화하면서 신속하게 롤백할 수 있습니다.

bot

인터넷을 통해 자동화된 작업을 실행하고 사람의 활동이나 상호 작용을 시뮬레이션하는 소프트웨어 애플리케이션입니다. 인터넷에서 정보를 인덱싱하는 웹 크롤러와 같은 일부 봇은 유용하거나 유용합니다. 배드 봇으로 알려진 일부 다른 봇은 개인이나 조직을 방해하거나 피해를 입히기 위한 것입니다.

봇넷

멀웨어에감염되어 봇 허더 또는 봇 운영자로 알려진 단일 당사자의 통제 하에 있는 봇 네트워크. 봇넷은 봇과 그 영향을 확장하는 가장 잘 알려진 메커니즘입니다.

브랜치

코드 리포지토리의 포함된 영역입니다. 리포지토리에 생성되는 첫 번째 브랜치가 기본 브랜치입니다. 기존 브랜치에서 새 브랜치를 생성한 다음 새 브랜치에서 기능을 개발하거나 버그를 수정할 수 있습니다. 기능을 구축하기 위해 생성하는 브랜치를 일반적으로 기능 브랜치라고 합니다. 기능을 출시할 준비가 되면 기능 브랜치를 기본 브랜치에 다시 병합합니다. 자세한 내용은 브랜치 정보 (문서) 를 참조하십시오. GitHub

브레이크 글래스 액세스

예외적인 상황에서 승인된 프로세스를 통해 사용자가 일반적으로 액세스 권한이 없는 데이터에 빠르게 액세스할 수 AWS 계정 있는 수단입니다. 자세한 내용은 Well-Architected AWS 지침의 브레이크 글래스 절차 구현 표시기를 참조하십시오.

브라운필드 전략

사용자 환경의 기존 인프라 시스템 아키텍처에 브라운필드 전략을 채택할 때는 현재 시스템 및 인프라의 제약 조건을 중심으로 아키텍처를 설계합니다. 기존 인프라를 확장하는 경우 브라운필드 전략과 그린필드 전략을 혼합할 수 있습니다.

버퍼 캐시

가장 자주 액세스하는 데이터가 저장되는 메모리 영역입니다.

사업 역량

기업이 가치를 창출하기 위해 하는 일(예: 영업, 고객 서비스 또는 마케팅)입니다. 마이크로서비스 아키텍처 및 개발 결정은 비즈니스 역량에 따라 이루어질 수 있습니다. 자세한 내용은 백서의 AWS에서 컨테이너화된 마이크로서비스 실행의 비즈니스 역량 중심의 구성화 섹션을 참조하십시오.

비즈니스 연속성 계획(BCP)

대규모 마이그레이션과 같은 중단 이벤트가 운영에 미치는 잠재적 영향을 해결하고 비즈니스가 신속하게 운영을 재개할 수 있도록 지원하는 계획입니다.

C

CAF

클라우드 채택 프레임워크를 참조하십시오AWS .

카나리아 배포

최종 사용자에게 버전을 느리고 점진적으로 릴리스하는 것입니다. 확신이 들면 새 버전을 배포하고 현재 버전을 완전히 교체합니다.

CCoE

클라우드 센터 오브 엑설런스를 참조하십시오.

CDC

변경 데이터 캡처를 참조하십시오.

변경 데이터 캡처(CDC)

데이터베이스 테이블과 같은 데이터 소스의 변경 내용을 추적하고 변경 사항에 대한 메타데이터를 기록하는 프로세스입니다. 대상 시스템의 변경 내용을 감사하거나 복제하여 동기화를 유지하는 등의 다양한 용도로 CDC를 사용할 수 있습니다.

카오스 엔지니어링

시스템의 복원력을 테스트하기 위해 의도적으로 장애나 장애를 일으키는 이벤트를 발생시키는 행위 AWS Fault Injection Service (AWS FIS) 를 사용하여 AWS 워크로드에 스트레스를 주는 실험을 수행하고 응답을 평가할 수 있습니다.

CI/CD

지속적 통합 및 지속적 전달을 참조하십시오.

분류

예측을 생성하는 데 도움이 되는 분류 프로세스입니다. 분류 문제에 대한 ML 모델은 이산 값을 예측합니다. 이산 값은 항상 서로 다릅니다. 예를 들어, 모델이 이미지에 자동차가 있는지 여부를 평가해야 할 수 있습니다.

클라이언트측 암호화

대상이 데이터를 AWS 서비스 수신하기 전에 데이터를 로컬로 암호화합니다.

클라우드 혁신 센터(CCoE)

클라우드 모범 사례 개발, 리소스 동원, 마이그레이션 타임라인 설정, 대규모 혁신을 통한 조직 선도 등 조직 전체에서 클라우드 채택 노력을 추진하는 다분야 팀입니다. 자세한 내용은 AWS 클라우드 기업 전략 블로그의 CCoE 게시물을 참조하십시오.

클라우드 컴퓨팅

원격 데이터 스토리지와 IoT 디바이스 관리에 일반적으로 사용되는 클라우드 기술 클라우드 컴퓨팅은 일반적으로 엣지 컴퓨팅 기술과 연결됩니다.

클라우드 운영 모델

IT 조직에서 하나 이상의 클라우드 환경을 구축, 성숙화 및 최적화하는 데 사용되는 운영 모델입니다. 자세한 내용은 클라우드 운영 모델 구축을 참조하십시오.

클라우드 채택 단계

조직이 마이그레이션할 때 일반적으로 거치는 4단계는 다음과 같습니다. AWS 클라우드

- 프로젝트 - 개념 증명 및 학습 목적으로 몇 가지 클라우드 관련 프로젝트 실행
- 기반 – 클라우드 채택 확장을 위한 기초 투자(예: 랜딩 존 생성, CCoE 정의, 운영 모델 구축)
- 마이그레이션 - 개별 애플리케이션 마이그레이션
- Re-invention - 제품 및 서비스 최적화와 클라우드 혁신

Stephen Orban은 기업 전략 블로그의 클라우드 우선주의를 향한 여정 및 채택 단계에 대한 블로그 게시물에서 이러한 단계를 정의했습니다. AWS 클라우드 이들이 AWS 마이그레이션 전략과 어떤 관련이 있는지에 대한 자세한 내용은 마이그레이션 준비 가이드를 참조하십시오.

CMDB

구성 관리 데이터베이스를 참조하십시오.

코드 리포지토리

소스 코드와 설명서, 샘플, 스크립트 등의 기타 자산이 버전 관리 프로세스를 통해 저장되고 업데이트되는 위치입니다. 일반 클라우드 리포지토리에는 또는 이 포함됩니다 GitHub . AWS CodeCommit코드의 각 버전을 브랜치라고 합니다. 마이크로서비스 구조에서 각 리포지토리는 단일 기능 전용입니다. 단일 CI/CD 파이프라인은 여러 리포지토리를 사용할 수 있습니다.

콜드 캐시

비어 있거나, 제대로 채워지지 않았거나, 오래되었거나 관련 없는 데이터를 포함하는 버퍼 캐시입니다. 주 메모리나 디스크에서 데이터베이스 인스턴스를 읽어야 하기 때문에 성능에 영향을 미치며, 이는 버퍼 캐시에서 읽는 것보다 느립니다.

콜드 데이터

거의 액세스되지 않고 일반적으로 과거 데이터인 데이터. 이런 종류의 데이터를 쿼리할 때는 일반적으로 느린 쿼리가 허용됩니다. 이 데이터를 성능이 낮고 비용이 저렴한 스토리지 계층 또는 클래스로 옮기면 비용을 절감할 수 있습니다.

컴퓨터 비전 (CV)

기계 학습을 사용하여 디지털 이미지 및 비디오와 같은 시각적 형식에서 정보를 분석하고 추출하는 AI 분야. 예를 들어 AWS Panorama 는 온프레미스 카메라 네트워크에 CV를 추가하는 디바이스를 제공하고, SageMaker Amazon은 CV용 이미지 처리 알고리즘을 제공합니다.

구성 드리프트

워크로드의 경우 구성이 예상 상태에서 변경됩니다. 이로 인해 워크로드가 규정을 준수하지 않게 될 수 있으며, 일반적으로 점진적이고 의도하지 않은 방식으로 진행됩니다.

구성 관리 데이터베이스(CMDB)

하드웨어 및 소프트웨어 구성 요소와 해당 구성을 포함하여 데이터베이스와 해당 IT 환경에 대한 정보를 저장하고 관리하는 리포지토리입니다. 일반적으로 마이그레이션의 포트폴리오 검색 및 분석 단계에서 CMDB의 데이터를 사용합니다.

규정 준수 팩

AWS Config 규정 준수 및 보안 검사를 사용자 지정하기 위해 조합할 수 있는 규칙 및 수정 조치 모음입니다. YAML 템플릿을 사용하여 한 AWS 계정 및 지역 또는 조직 전체에 단일 엔티티로 적합성 팩을 배포할 수 있습니다. 자세한 내용은 설명서의 적합성 팩을 참조하십시오. AWS Config

지속적 통합 및 지속적 전달(CI/CD)

소프트웨어 릴리스 프로세스의 소스, 빌드, 테스트, 스테이징 및 프로덕션 단계를 자동화하는 프로세스입니다. CI/CD는 일반적으로 파이프라인으로 설명됩니다. CI/CD를 통해 프로세스를 자동화하고, 생산성을 높이고, 코드 품질을 개선하고, 더 빠르게 제공할 수 있습니다. 자세한 내용은 지속적 전달의 이점을 참조하십시오. CD는 지속적 배포를 의미하기도 합니다. 자세한 내용은 지속적 전달(Continuous Delivery)과 지속적인 개발을 참조하십시오.

CV

컴퓨터 비전을 참조하십시오.

D

저장 데이터

스토리지에 있는 데이터와 같이 네트워크에 고정되어 있는 데이터입니다.

데이터 분류

중요도와 민감도를 기준으로 네트워크의 데이터를 식별하고 분류하는 프로세스입니다. 이 프로세스는 데이터에 대한 적절한 보호 및 보존 제어를 결정하는 데 도움이 되므로 사이버 보안 위험 관리 전략의 중요한 구성 요소입니다. 데이터 분류는 AWS Well-Architected 프레임워크의 보안 핵심 요소입니다. 자세한 내용은 데이터 분류를 참조하십시오.

데이터 드리프트

프로덕션 데이터와 ML 모델 학습에 사용된 데이터 간의 상당한 차이 또는 시간 경과에 따른 입력 데이터의 의미 있는 변화. 데이터 드리프트는 ML 모델 예측의 전반적인 품질, 정확성 및 공정성을 저하시킬 수 있습니다.

전송 중 데이터

네트워크를 통과하고 있는 데이터입니다. 네트워크 리소스 사이를 이동 중인 데이터를 예로 들 수 있습니다.

데이터 메시

중앙 집중식 관리 및 거버넌스와 함께 분산되고 분산된 데이터 소유권을 제공하는 아키텍처 프레임워크입니다.

데이터 최소화

꼭 필요한 데이터만 수집하고 처리하는 원칙입니다. 에서 데이터 최소화를 실천하면 개인 정보 보호 위험, 비용 및 분석에 따른 탄소 발자국을 줄일 AWS 클라우드 수 있습니다.

데이터 경계

신뢰할 수 있는 ID만 예상 네트워크에서 신뢰할 수 있는 리소스에 액세스하도록 하는 데 도움이 되는 AWS 환경 내 일련의 예방 가드레일입니다. 자세한 내용은 데이터 경계 구축을 참조하십시오. AWS

데이터 사전 처리

원시 데이터를 ML 모델이 쉽게 구문 분석할 수 있는 형식으로 변환하는 것입니다. 데이터를 사전 처리한다는 것은 특정 열이나 행을 제거하고 누락된 값, 일관성이 없는 값 또는 중복 값을 처리함을 의미할 수 있습니다.

데이터 출처

라이프사이클 전반에 걸쳐 데이터의 출처와 기록을 추적하는 프로세스(예: 데이터 생성, 전송, 저장 방법).

데이터 주체

데이터를 수집 및 처리하는 개인입니다.

데이터 웨어하우스

분석과 같은 비즈니스 인텔리전스를 지원하는 데이터 관리 시스템. 데이터 웨어하우스에는 일반적으로 대량의 과거 데이터가 포함되며 일반적으로 쿼리 및 분석에 사용됩니다.

데이터 정의 언어(DDL)

데이터베이스에서 테이블 및 객체의 구조를 만들거나 수정하기 위한 명령문 또는 명령입니다.

데이터베이스 조작 언어(DML)

데이터베이스에서 정보를 수정(삽입, 업데이트 및 삭제)하기 위한 명령문 또는 명령입니다.

DDL

데이터베이스 정의 언어를 참조하십시오.

딥 앙상블

예측을 위해 여러 딥 러닝 모델을 결합하는 것입니다. 딥 앙상블을 사용하여 더 정확한 예측을 얻거나 예측의 불확실성을 추정할 수 있습니다.

딥 러닝

여러 계층의 인공 신경망을 사용하여 입력 데이터와 관심 대상 변수 간의 매핑을 식별하는 ML 하위 분야입니다.

defense-in-depth

네트워크와 그 안의 데이터 기밀성, 무결성 및 가용성을 보호하기 위해 컴퓨터 네트워크 전체에 일련의 보안 메커니즘과 제어를 신중하게 계층화하는 정보 보안 접근 방식입니다. 이 전략을 채택하면 AWS Organizations 구조의 여러 계층에 AWS여러 컨트롤을 추가하여 리소스를 보호하는 데 도움이 됩니다. 예를 들어 다단계 인증, 네트워크 세분화, 암호화를 결합한 defense-in-depth 접근 방식을 사용할 수 있습니다.

위임된 관리자

에서 AWS Organizations호환 가능한 서비스는 AWS 구성원 계정을 등록하여 조직의 계정을 관리하고 해당 서비스에 대한 권한을 관리할 수 있습니다. 이러한 계정을 해당 서비스의 위임된 관리자라고 합니다. 자세한 내용과 호환되는 서비스 목록은 AWS Organizations 설명서의 AWS Organizations와 함께 사용할 수 있는 AWS 서비스를 참조하십시오.

배포

대상 환경에서 애플리케이션, 새 기능 또는 코드 수정 사항을 사용할 수 있도록 하는 프로세스입니다. 배포에는 코드 베이스의 변경 사항을 구현한 다음 애플리케이션 환경에서 해당 코드베이스를 구축하고 실행하는 작업이 포함됩니다.

개발 환경

환경을 참조하십시오.

탐지 제어

이벤트 발생 후 탐지, 기록 및 알림을 수행하도록 설계된 보안 제어입니다. 이러한 제어는 기존의 예방적 제어를 우회한 보안 이벤트를 알리는 2차 방어선입니다. 자세한 내용은 Implementing security controls on AWS의 Detective controls를 참조하십시오.

개발 가치 흐름 매핑 (DVSM)

소프트웨어 개발 라이프사이클에서 속도와 품질에 부정적인 영향을 미치는 제약 조건을 식별하고 우선 순위를 지정하는 데 사용되는 프로세스입니다. DVSM은 원래 린 제조 방식을 위해 설계된 가치 흐름 매핑 프로세스를 확장합니다. 소프트웨어 개발 프로세스를 통해 가치를 창출하고 이동하는 데 필요한 단계와 팀에 중점을 둡니다.

디지털 트윈

건물, 공장, 산업 장비 또는 생산 라인과 같은 실제 시스템을 가상으로 표현한 것입니다. 디지털 트윈은 예측 유지 보수, 원격 모니터링, 생산 최적화를 지원합니다.

치수 표

스타 스키마에서 팩트 테이블의 양적 데이터에 대한 데이터 속성을 포함하는 작은 테이블입니다. 차원 테이블 속성은 일반적으로 텍스트처럼 동작하는 텍스트 필드 또는 불연속형 숫자입니다. 이러한 속성은 일반적으로 쿼리 제한, 필터링 및 결과 집합 레이블 지정에 사용됩니다.

재해

워크로드 또는 시스템이 기본 배포 위치에서 비즈니스 목표를 달성하지 못하게 방해하는 이벤트입니다. 이러한 이벤트는 자연재해, 기술적 오류, 의도하지 않은 구성 오류 또는 멀웨어 공격과 같은 사람의 행동으로 인한 결과일 수 있습니다.

재해 복구(DR)

재해로 인한 다운타임과 데이터 손실을 최소화하기 위해 사용하는 전략과 프로세스입니다. 자세한 내용은 워크로드의 재해 복구 AWS: AWS Well-Architected 프레임워크에서의 클라우드 복구를 참조하십시오.

DML

데이터베이스 조작 언어를 참조하십시오.

도메인 기반 설계

구성 요소를 각 구성 요소가 제공하는 진화하는 도메인 또는 핵심 비즈니스 목표에 연결하여 복잡한 소프트웨어 시스템을 개발하는 접근 방식입니다. 이 개념은 에릭 에반스에 의해 그의 저서인 도메인 기반 디자인: 소프트웨어 중심의 복잡성 해결(Boston: Addison-Wesley Professional, 2003)에서 소개되었습니다. Strangler Fig 패턴과 함께 도메인 기반 설계를 사용하는 방법에 대한 자세한 내용은 컨테이너 및 Amazon API Gateway를 사용하여 기존의 Microsoft ASP.NET(ASMX) 웹 서비스를 점진적으로 현대화하는 방법을 참조하십시오.

DR

재해 복구를 참조하십시오.

드리프트 감지

기준 구성으로부터의 편차 추적. 예를 들어 시스템 리소스의 편차를 감지하는 AWS CloudFormation 데 사용하거나 거버넌스 요구 사항 준수에 영향을 미칠 수 있는 착륙 지대의 변경 사항을 탐지하는 AWS Control Tower 데 사용할 수 있습니다.

DVSM

개발 가치 흐름 매핑을 참조하십시오.

E

EDA

탐색적 데이터 분석을 참조하십시오.

엣지 컴퓨팅

IoT 네트워크의 엣지에서 스마트 디바이스의 컴퓨팅 성능을 개선하는 기술 클라우드 컴퓨팅과 비교할 때 엣지 컴퓨팅은 통신 대기 시간을 줄이고 응답 시간을 개선할 수 있습니다.

암호화

사람이 읽을 수 있는 일반 텍스트 데이터를 암호문으로 변환하는 컴퓨팅 프로세스입니다.

암호화 키

암호화 알고리즘에 의해 생성되는 무작위 비트의 암호화 문자열입니다. 키의 길이는 다양할 수 있으며 각 키는 예측할 수 없고 고유하게 설계되었습니다.

엔디안

컴퓨터 메모리에 바이트가 저장되는 순서입니다. 빅 엔디안 시스템은 가장 중요한 바이트를 먼저 저장합니다. 리틀 엔디안 시스템은 가장 덜 중요한 바이트를 먼저 저장합니다.

엔드포인트

서비스 엔드포인트를 참조하십시오.

엔드포인트 서비스

Virtual Private Cloud(VPC)에서 호스팅하여 다른 사용자와 공유할 수 있는 서비스입니다. 다른 주체 AWS 계정 또는 AWS Identity and Access Management (IAM) 보안 주체에 권한을 부여하여 엔드포인트 서비스를 생성하고 권한을 부여할 수 있습니다. AWS PrivateLink 이러한 계정 또는 보안 주체는 인터페이스 VPC 엔드포인트를 생성하여 엔드포인트 서비스에 비공개로 연결할 수 있습니

다. 자세한 내용은 Amazon Virtual Private Cloud(VPC) 설명서의 엔드포인트 서비스 생성을 참조하십시오.

ERP (전사적 자원 관리)

기업의 주요 비즈니스 프로세스 (예: 회계, MES, 프로젝트 관리) 를 자동화하고 관리하는 시스템입니다.

봉투 암호화

암호화 키를 다른 암호화 키로 암호화하는 프로세스입니다. 자세한 내용은 AWS Key Management Service (AWS KMS) 설명서의 봉투 암호화를 참조하십시오.

환경

실행 중인 애플리케이션의 인스턴스입니다. 다음은 클라우드 컴퓨팅의 일반적인 환경 유형입니다.

- 개발 환경 - 애플리케이션 유지 관리를 담당하는 핵심 팀만 사용할 수 있는 실행 중인 애플리케이션의 인스턴스입니다. 개발 환경은 변경 사항을 상위 환경으로 승격하기 전에 테스트하는 데 사용됩니다. 이러한 유형의 환경을 테스트 환경이라고도 합니다.
- 하위 환경 - 초기 빌드 및 테스트에 사용되는 환경을 비롯한 애플리케이션의 모든 개발 환경입니다.
- 프로덕션 환경 - 최종 사용자가 액세스할 수 있는 실행 중인 애플리케이션의 인스턴스입니다. CI/CD 파이프라인에서 프로덕션 환경이 마지막 배포 환경입니다.
- 상위 환경 - 핵심 개발 팀 이외의 사용자가 액세스할 수 있는 모든 환경입니다. 프로덕션 환경, 프로덕션 이전 환경 및 사용자 수용 테스트를 위한 환경이 여기에 포함될 수 있습니다.

에픽

애자일 방법론에서 작업을 구성하고 우선순위를 정하는 데 도움이 되는 기능적 범주입니다. 에픽은 요구 사항 및 구현 작업에 대한 개괄적인 설명을 제공합니다. 예를 들어 AWS CAF 보안 에픽에는 ID 및 액세스 관리, 탐지 제어, 인프라 보안, 데이터 보호, 사고 대응 등이 포함됩니다. AWS 마이그레이션 전략의 에픽에 대한 자세한 내용은 프로그램 구현 가이드를 참조하십시오.

ERP

엔터프라이즈 리소스 계획을 참조하십시오.

탐색 데이터 분석(EDA)

데이터 세트를 분석하여 주요 특성을 파악하는 프로세스입니다. 데이터를 수집 또는 집계한 다음 초기 조사를 수행하여 패턴을 찾고, 이상을 탐지하고, 가정을 확인합니다. EDA는 요약 통계를 계산하고 데이터 시각화를 생성하여 수행됩니다.

F

팩트 테이블

[스타 스키마의](#) 중앙 테이블. 비즈니스 운영에 대한 정량적 데이터를 저장합니다. 일반적으로 팩트 테이블에는 측정값이 포함된 열과 차원 테이블의 외부 키가 포함된 열 등 두 가지 유형의 열이 포함됩니다.

빨리 실패하세요

빈번하고 점진적인 테스트를 통해 개발 라이프사이클을 단축하는 철학. 이는 애자일 접근 방식의 중요한 부분입니다.

장애 격리 경계

장애 영향을 제한하고 워크로드의 복원력을 개선하는 데 도움이 되는 가용 영역 AWS 리전, 컨트롤 플레인 또는 데이터 플레인과 같은 경계 AWS 클라우드자세한 내용은 [AWS 장애 격리](#) 경계를 참조하십시오.

기능 브랜치

[브랜치를](#) 참조하십시오.

기능

예측에 사용하는 입력 데이터입니다. 예를 들어, 제조 환경에서 기능은 제조 라인에서 주기적으로 캡처되는 이미지일 수 있습니다.

기능 중요도

모델의 예측에 특성이 얼마나 중요한지를 나타냅니다. 이는 일반적으로 SHAP(Shapley Additive Descriptions) 및 통합 그레디언트와 같은 다양한 기법을 통해 계산할 수 있는 수치 점수로 표현됩니다. 자세한 내용은 [다음을AWS사용한 기계 학습 모델 해석 가능성을](#) 참조하십시오.

기능 변환

추가 소스로 데이터를 보강하거나, 값을 조정하거나, 단일 데이터 필드에서 여러 정보 세트를 추출하는 등 ML 프로세스를 위해 데이터를 최적화하는 것입니다. 이를 통해 ML 모델이 데이터를 활용할 수 있습니다. 예를 들어, 날짜 '2021-05-27 00:15:37'을 '2021년', '5월', '목', '15일'로 분류하면 학습 알고리즘이 다양한 데이터 구성 요소와 관련된 미묘한 패턴을 학습하는 데 도움이 됩니다.

FGAC

[세분화된 액세스 제어를](#) 참조하십시오.

세분화된 액세스 제어(FGAC)

여러 조건을 사용하여 액세스 요청을 허용하거나 거부합니다.

플래시컷 마이그레이션

단계별 접근 방식 대신 변경 데이터 캡처를 통한 지속적인 데이터 복제를 통해 최단 시간에 데이터를 마이그레이션하는 데이터베이스 마이그레이션 방법입니다. 목표는 가동 중지 시간을 최소화하는 것입니다.

G

지리적 차단

지리적 제한을 참조하십시오.

지리적 제한(지리적 차단)

CloudFrontAmazon에서는 특정 국가의 사용자가 콘텐츠 배포에 액세스하지 못하도록 하는 옵션을 제공합니다. 허용 목록 또는 차단 목록을 사용하여 승인된 국가와 차단된 국가를 지정할 수 있습니다. 자세한 내용은 설명서의 콘텐츠의 지리적 배포 제한을 참조하십시오. CloudFront

Gitflow 워크플로

하위 환경과 상위 환경이 소스 코드 리포지토리의 서로 다른 브랜치를 사용하는 방식입니다. Gitflow 워크플로는 레거시로 간주되며 트렁크 기반 워크플로는 현대적이고 선호되는 접근 방식입니다.

브라운필드 전략

새로운 환경에서 기존 인프라의 부재 시스템 아키텍처에 대한 그린필드 전략을 채택할 때 브라운필드라고도 하는 기존 인프라와의 호환성 제한 없이 모든 새로운 기술을 선택할 수 있습니다. 기존 인프라를 확장하는 경우 브라운필드 전략과 그린필드 전략을 혼합할 수 있습니다.

가드레일

조직 단위(OU) 전체에서 리소스, 정책 및 규정 준수를 관리하는 데 도움이 되는 중요 규칙입니다. 예방 가드레일은 규정 준수 표준에 부합하도록 정책을 시행하며, 서비스 제어 정책과 IAM 권한 경계를 사용하여 구현됩니다. 탐지 가드레일은 정책 위반 및 규정 준수 문제를 감지하고 해결을 위한 알림을 생성하며, 이들은, Amazon AWS Config AWS Security Hub GuardDuty AWS Trusted Advisor, Amazon Inspector 및 사용자 지정 AWS Lambda 검사를 사용하여 구현됩니다.

H

하

고가용성을 확인하세요.

이기종 데이터베이스 마이그레이션

다른 데이터베이스 엔진을 사용하는 대상 데이터베이스로 소스 데이터베이스 마이그레이션(예: Oracle에서 Amazon Aurora로) 이기종 마이그레이션은 일반적으로 리아키텍트 작업의 일부이며 스키마를 변환하는 것은 복잡한 작업일 수 있습니다.AWS 는 스키마 변환에 도움이 되는 AWS SCT를 제공합니다.

높은 가용성(HA)

문제나 재해 발생 시 개입 없이 지속적으로 운영할 수 있는 워크로드의 능력. HA 시스템은 자동으로 장애 조치되고, 지속적으로 고품질 성능을 제공하고, 성능에 미치는 영향을 최소화하면서 다양한 부하와 장애를 처리하도록 설계되었습니다.

히스토리언 현대화

제조 산업의 요구 사항을 더 잘 충족하도록 운영 기술(OT) 시스템을 현대화하고 업그레이드하는 데 사용되는 접근 방식입니다. 히스토리언은 공장의 다양한 출처에서 데이터를 수집하고 저장하는 데 사용되는 일종의 데이터베이스입니다.

동종 데이터베이스 마이그레이션

동일한 데이터베이스 엔진을 공유하는 대상 데이터베이스로 소스 데이터베이스 마이그레이션(예: Microsoft SQL Server에서 Amazon RDS for SQL Server로) 동종 마이그레이션은 일반적으로 리호스팅 또는 리플랫포밍 작업의 일부입니다. 네이티브 데이터베이스 유틸리티를 사용하여 스키마를 마이그레이션할 수 있습니다.

핫 데이터

자주 액세스하는 데이터(예: 실시간 데이터 또는 최근 번역 데이터). 일반적으로 이 데이터에는 빠른 쿼리 응답을 제공하기 위한 고성능 스토리지 계층 또는 클래스가 필요합니다.

핫픽스

프로덕션 환경의 중요한 문제를 해결하기 위한 긴급 수정입니다. 긴급성 때문에 핫픽스는 일반적으로 일반적인 DevOps 릴리스 워크플로 외부에서 만들어집니다.

하이퍼케어 기간

전환 직후 마이그레이션 팀이 문제를 해결하기 위해 클라우드에서 마이그레이션된 애플리케이션을 관리하고 모니터링하는 기간입니다. 일반적으로 이 기간은 1~4일입니다. 하이퍼케어 기간이 끝나면 마이그레이션 팀은 일반적으로 애플리케이션에 대한 책임을 클라우드 운영 팀에 넘깁니다.

I

IaC

인프라를 코드로 보세요.

자격 증명 기반 정책

환경 내에서 권한을 정의하는 하나 이상의 IAM 보안 주체에 연결된 정책입니다. AWS 클라우드

유휴 애플리케이션

90일 동안 평균 CPU 및 메모리 사용량이 5~20%인 애플리케이션입니다. 마이그레이션 프로젝트에서는 이러한 애플리케이션을 사용 중지하거나 온프레미스에 유지하는 것이 일반적입니다.

IIoT

산업용 사물 인터넷을 참조하십시오.

불변의 인프라

기존 인프라를 업데이트, 패치 또는 수정하는 대신 프로덕션 워크로드용 새 인프라를 배포하는 모델입니다. 변경 불가능한 인프라는 기본적으로 변경 가능한 인프라보다 더 일관되고 안정적이며 예측 가능합니다. 자세한 내용은 Well-Architected AWS 프레임워크의 변경 불가능한 인프라를 사용한 배포 모범 사례를 참조하십시오.

인바운드(수신) VPC

AWS 다중 계정 아키텍처에서 VPC는 애플리케이션 외부에서 네트워크 연결을 허용, 검사 및 라우팅합니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

증분 마이그레이션

한 번에 전체 전환을 수행하는 대신 애플리케이션을 조금씩 마이그레이션하는 전환 전략입니다. 예를 들어, 처음에는 소수의 마이크로서비스나 사용자만 새 시스템으로 이동할 수 있습니다. 모든 것

이 제대로 작동하는지 확인한 후에는 레거시 시스템을 폐기할 수 있을 때까지 추가 마이크로서비스 또는 사용자를 점진적으로 이동할 수 있습니다. 이 전략을 사용하면 대규모 마이그레이션과 관련된 위험을 줄일 수 있습니다.

Industry 4.0

Klaus Schwab이 연결성, 실시간 데이터, 자동화, 분석 및 AI/ML의 발전을 통한 제조 프로세스의 현대화를 지칭하기 위해 2016년 도입한 용어입니다.

인프라

애플리케이션의 환경 내에 포함된 모든 리소스와 자산입니다.

코드형 인프라(IaC)

구성 파일 세트를 통해 애플리케이션의 인프라를 프로비저닝하고 관리하는 프로세스입니다. IaC는 새로운 환경의 반복 가능성, 신뢰성 및 일관성을 위해 인프라 관리를 중앙 집중화하고, 리소스를 표준화하고, 빠르게 확장할 수 있도록 설계되었습니다.

산업용 사물 인터넷(IIoT)

제조, 에너지, 자동차, 의료, 생명과학, 농업 등의 산업 부문에서 인터넷에 연결된 센서 및 디바이스의 사용 자세한 내용은 산업용 사물 인터넷(IoT) 디지털 트랜스포메이션 전략 구축을 참조하십시오.

검사 VPC

AWS 다중 계정 아키텍처에서 VPC (동일하거나 AWS 리전다른), 인터넷 및 온프레미스 네트워크 간의 네트워크 트래픽 검사를 관리하는 중앙 집중식 VPC입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

사물 인터넷(IoT)

인터넷이나 로컬 통신 네트워크를 통해 다른 디바이스 및 시스템과 통신하는 센서 또는 프로세서가 내장된 연결된 물리적 객체의 네트워크 자세한 내용은 IoT란?을 참조하십시오.

해석력

모델의 예측이 입력에 따라 어떻게 달라지는지를 사람이 이해할 수 있는 정도를 설명하는 기계 학습 모델의 특성입니다. 자세한 내용은 Machine learning model interpretability with AWS를 참조하십시오.

IoT

사물 인터넷을 참조하십시오.

IT 정보 라이브러리(TIL)

IT 서비스를 제공하고 이러한 서비스를 비즈니스 요구 사항에 맞게 조정하기 위한 일련의 모범 사례 ITIL은 ITSM의 기반을 제공합니다.

IT 서비스 관리(TSM)

조직의 IT 서비스 설계, 구현, 관리 및 지원과 관련된 활동 클라우드 운영을 ITSM 도구와 통합하는 방법에 대한 자세한 내용은 운영 통합 가이드를 참조하십시오.

ITIL

IT 정보 라이브러리를 참조하십시오.

ITSM

IT 서비스 관리를 참조하십시오.

L

레이블 기반 액세스 제어(LBAC)

사용자 및 데이터 자체에 각각 보안 레이블 값을 명시적으로 할당하는 필수 액세스 제어(MAC)를 구현한 것입니다. 사용자 보안 레이블과 데이터 보안 레이블 간의 교차 부분에 따라 사용자가 볼 수 있는 행과 열이 결정됩니다.

랜딩 존

Landing Zone은 확장 가능하고 안전한 잘 설계된 다중 계정 AWS 환경입니다. 조직은 여기에서부터 보안 및 인프라 환경에 대한 확신을 가지고 워크로드와 애플리케이션을 신속하게 시작하고 배포할 수 있습니다. 랜딩 존에 대한 자세한 내용은 안전하고 확장 가능한 다중 계정 AWS 환경 설정을 참조하십시오.

대규모 마이그레이션

300대 이상의 서버 마이그레이션입니다.

LBAC

레이블 기반 액세스 제어를 참조하십시오.

최소 권한

작업을 수행하는 데 필요한 최소 권한을 부여하는 보안 모범 사례입니다. 자세한 내용은 IAM 설명서의 최소 권한 적용을 참조하십시오.

리프트 앤드 시프트

7 R을 참조하십시오.

리틀 엔디안 시스템

가장 덜 중요한 바이트를 먼저 저장하는 시스템입니다. 엔디안 참조.

하위 환경

환경 참조.

M

기계 학습(ML)

패턴 인식 및 학습에 알고리즘과 기법을 사용하는 인공 지능의 한 유형입니다. ML은 사물 인터넷(IoT) 데이터와 같은 기록된 데이터를 분석하고 학습하여 패턴을 기반으로 통계 모델을 생성합니다. 자세한 내용은 기계 학습을 참조하십시오.

기본 브랜치

브랜치 참조.

악성 코드

컴퓨터 보안 또는 개인 정보를 침해하도록 설계된 소프트웨어 멀웨어는 컴퓨터 시스템을 방해하거나, 민감한 정보를 유출하거나, 무단 액세스를 얻을 수 있습니다. 멀웨어의 예로는 바이러스, 웜, 랜섬웨어, 트로이 목마, 스파이웨어, 키로거 등이 있습니다.

매니지드 서비스

AWS 서비스 인프라 계층, 운영 체제 및 플랫폼을 AWS 운영하며 사용자는 엔드포인트에 액세스하여 데이터를 저장하고 검색합니다. 관리형 서비스의 예로는 아마존 심플 스토리지 서비스 (Amazon S3) 와 아마존 DynamoDB가 있습니다. 이러한 서비스를 추상화된 서비스라고도 합니다.

제조 실행 시스템 (MES)

제조 현장에서 원자재를 완제품으로 전환하는 생산 프로세스를 추적, 모니터링, 문서화 및 제어하기 위한 소프트웨어 시스템입니다.

MAP

Migration Acceleration 프로그램을 참조하십시오.

기구

도구를 만들고 도구 채택을 유도한 다음 결과를 검토하여 조정하는 전체 프로세스입니다. 메커니즘은 작동하면서 자체적으로 강화되고 개선되는 사이클입니다. 자세한 내용은 AWS Well-Architected 프레임워크에서의 메커니즘 구축을 참조하십시오.

멤버 계정

조직의 일부인 관리 계정을 AWS 계정 제외한 모든 계정 AWS Organizations하나의 계정은 한 번에 하나의 조직 멤버만 될 수 있습니다.

MES

제조 실행 시스템을 참조하십시오.

메시지 큐 텔레메트리 전송 (MQTT)

퍼블리시/구독 패턴을 기반으로 하는 리소스가 제한된 IoT 디바이스를 위한 경량 machine-to-machine (M2M) 통신 프로토콜입니다.

마이크로서비스

잘 정의된 API를 통해 통신하고 일반적으로 소규모 자체 팀이 소유하는 소규모 독립 서비스입니다. 예를 들어, 보험 시스템에는 영업, 마케팅 등의 비즈니스 역량이나 구매, 청구, 분석 등의 하위 영역에 매핑되는 마이크로 서비스가 포함될 수 있습니다. 마이크로서비스의 이점으로 민첩성, 유연한 확장, 손쉬운 배포, 재사용 가능한 코드, 복원력 등이 있습니다. 자세한 내용은 서버리스 서비스를 사용하여 마이크로서비스 통합을 참조하십시오. AWS

마이크로서비스 아키텍처

각 애플리케이션 프로세스를 마이크로서비스로 실행하는 독립 구성 요소를 사용하여 애플리케이션을 구축하는 접근 방식입니다. 이러한 마이크로서비스는 경량 API를 사용하여 잘 정의된 인터페이스를 통해 통신합니다. 애플리케이션의 특정 기능에 대한 수요에 맞게 이 아키텍처의 각 마이크로서비스를 업데이트, 배포 및 조정할 수 있습니다. 자세한 내용은 마이크로서비스 구현을 참조하십시오. AWS

Migration Acceleration Program(MAP)

조직이 클라우드로 전환하기 위한 강력한 운영 기반을 구축하고 초기 마이그레이션 비용을 상쇄할 수 있도록 컨설팅 지원, 교육 및 서비스를 제공하는 AWS 프로그램입니다. MAP에는 레거시 마이그레이션을 체계적인 방식으로 실행하기 위한 마이그레이션 방법론과 일반적인 마이그레이션 시나리오를 자동화하고 가속화하는 도구 세트가 포함되어 있습니다.

대규모 마이그레이션

애플리케이션 포트폴리오의 대다수를 웨이브를 통해 클라우드로 이동하는 프로세스로, 각 웨이브에서 더 많은 애플리케이션이 더 빠른 속도로 이동합니다. 이 단계에서는 이전 단계에서 배운 모범 사례와 교훈을 사용하여 팀, 도구 및 프로세스의 마이그레이션 팩토리를 구현하여 자동화 및 민첩한 제공을 통해 워크로드 마이그레이션을 간소화합니다. 이것은 AWS 마이그레이션 전략의 세 번째 단계입니다.

마이그레이션 팩토리

자동화되고 민첩한 접근 방식을 통해 워크로드 마이그레이션을 간소화하는 다기능 팀입니다. 마이그레이션 팩토리 팀에는 일반적으로 운영, 비즈니스 분석가 및 소유자, 마이그레이션 엔지니어, 개발자 및 스프린트에서 일하는 DevOps 전문가가 포함됩니다. 엔터프라이즈 애플리케이션 포트폴리오의 20~50%는 공장 접근 방식으로 최적화할 수 있는 반복되는 패턴으로 구성되어 있습니다. 자세한 내용은 이 콘텐츠 세트의 클라우드 마이그레이션 팩토리 가이드와 마이그레이션 팩토리에 대한 설명을 참조하십시오.

마이그레이션 메타데이터

마이그레이션을 완료하는 데 필요한 애플리케이션 및 서버에 대한 정보 각 마이그레이션 패턴에는 서로 다른 마이그레이션 메타데이터 세트가 필요합니다. 마이그레이션 메타데이터의 예로는 대상 서브넷, 보안 그룹, 계정 등이 있습니다. AWS

마이그레이션 패턴

사용되는 마이그레이션 전략, 마이그레이션 대상, 마이그레이션 애플리케이션 또는 서비스를 자세히 설명하는 반복 가능한 마이그레이션 작업입니다. 예: 애플리케이션 마이그레이션 서비스를 사용하여 Amazon EC2로 AWS 마이그레이션을 재호스팅합니다.

Migration Portfolio Assessment(MPA)

로 마이그레이션하기 위한 비즈니스 사례를 검증하기 위한 정보를 제공하는 온라인 도구입니다. AWS 클라우드 MPA는 상세한 포트폴리오 평가(서버 적정 규모 조정, 가격 책정, TCO 비교, 마이그레이션 비용 분석)와 마이그레이션 계획(애플리케이션 데이터 분석 및 데이터 수집, 애플리케이션 그룹화, 마이그레이션 우선순위 지정, 웨이브 계획)을 제공합니다. MPA 도구 (로그인 필요) 는 모든 컨설턴트와 APN 파트너 AWS 컨설턴트에게 무료로 제공됩니다.

마이그레이션 준비 상태 평가(MRA)

CAF를 사용하여 조직의 클라우드 준비 상태에 대한 통찰력을 얻고, 강점과 약점을 파악하고, 식별된 격차를 해소하기 위한 실행 계획을 수립하는 프로세스입니다. AWS 자세한 내용은 마이그레이션 준비 가이드를 참조하십시오. MRA는 AWS 마이그레이션 전략의 첫 번째 단계입니다.

마이그레이션 전략

워크로드를 로 마이그레이션하는 데 사용된 접근 방식. AWS 클라우드자세한 내용은 이 용어집의 7R 항목 및 대규모 마이그레이션 가속화를 위한 조직 동원을 참조하십시오.

ML

기계 학습을 참조하십시오.

현대화

비용을 절감하고 효율성을 높이고 혁신을 활용하기 위해 구식(레거시 또는 모놀리식) 애플리케이션과 해당 인프라를 클라우드의 민첩하고 탄력적이고 가용성이 높은 시스템으로 전환하는 것입니다. 자세한 내용은 의 AWS 클라우드애플리케이션 현대화 전략을 참조하십시오.

현대화 준비 상태 평가

조직 애플리케이션의 현대화 준비 상태를 파악하고, 이점, 위험 및 종속성을 식별하고, 조직이 해당 애플리케이션의 향후 상태를 얼마나 잘 지원할 수 있는지를 확인하는 데 도움이 되는 평가입니다. 평가 결과는 대상 아키텍처의 청사진, 현대화 프로세스의 개발 단계와 마일스톤을 자세히 설명하는 로드맵 및 파악된 격차를 해소하기 위한 실행 계획입니다. 자세한 내용은 에서 애플리케이션의 현대화 준비 상태 평가를 참조하십시오. AWS 클라우드

모놀리식 애플리케이션(모놀리식 유형)

긴밀하게 연결된 프로세스를 사용하여 단일 서비스로 실행되는 애플리케이션입니다. 모놀리식 애플리케이션에는 몇 가지 단점이 있습니다. 한 애플리케이션 기능에 대한 수요가 급증하면 전체 아키텍처 규모를 조정해야 합니다. 코드 베이스가 커지면 모놀리식 애플리케이션의 기능을 추가하거나 개선하는 것도 더 복잡해집니다. 이러한 문제를 해결하기 위해 마이크로서비스 아키텍처를 사용할 수 있습니다. 자세한 내용은 마이크로서비스로 모놀리식 유형 분해를 참조하십시오.

MPA

마이그레이션 포트폴리오 평가를 참조하십시오.

MQTT

메시지 큐 원격 분석 전송을 참조하십시오.

멀티클래스 분류

여러 클래스에 대한 예측(2개 이상의 결과 중 하나 예측)을 생성하는 데 도움이 되는 프로세스입니다. 예를 들어, ML 모델이 '이 제품은 책인가요, 자동차인가요, 휴대폰인가요?' 또는 '이 고객이 가장 관심을 갖는 제품 범주는 무엇인가요?'라고 물을 수 있습니다.

변경 가능한 인프라

프로덕션 워크로드를 위해 기존 인프라를 업데이트하고 수정하는 모델입니다. 일관성, 안정성 및 예측 가능성을 개선하기 위해 AWS Well-Architected Framework는 변경 불가능한 인프라를 모범 사례로 사용할 것을 권장합니다.

O

OAC

원본 액세스 제어를 참조하십시오.

좋아요

원본 액세스 ID를 참조하십시오.

OCM

조직 변경 관리를 참조하십시오.

오프라인 마이그레이션

마이그레이션 프로세스 중 소스 워크로드가 중단되는 마이그레이션 방법입니다. 이 방법은 가동 중지 증가를 수반하며 일반적으로 작고 중요하지 않은 워크로드에 사용됩니다.

OI

운영 통합을 참조하십시오.

안녕하세요.

운영 수준 계약을 참조하십시오.

온라인 마이그레이션

소스 워크로드를 오프라인 상태로 전환하지 않고 대상 시스템에 복사하는 마이그레이션 방법입니다. 워크로드에 연결된 애플리케이션은 마이그레이션 중에도 계속 작동할 수 있습니다. 이 방법은 가동 중지 차단 또는 최소화를 수반하며 일반적으로 중요한 프로덕션 워크로드에 사용됩니다.

OPC-UA

오픈 프로세스 커뮤니케이션 - 통합 아키텍처를 참조하십시오.

오픈 프로세스 커뮤니케이션 - 통합 아키텍처 (OPC-UA)

산업 machine-to-machine 자동화를 위한 (M2M) 통신 프로토콜. OPC-UA는 데이터 암호화, 인증 및 권한 부여 체계와 함께 상호 운용성 표준을 제공합니다.

운영 수준 협약(OLA)

서비스 수준에 관한 계약(SLA)을 지원하기 위해 직무 IT 그룹이 서로에게 제공하기로 약속한 내용을 명확히 하는 계약입니다.

운영 준비 검토 (ORR)

인시던트 및 발생 가능한 실패의 범위를 이해, 평가, 예방 또는 줄이는 데 도움이 되는 질문 및 관련 모범 사례로 구성된 체크리스트입니다. 자세한 내용은 Well-Architected AWS 프레임워크의 운영 준비 상태 검토 (ORR) 를 참조하십시오.

운영 기술 (OT)

물리적 환경과 함께 작동하여 산업 운영, 장비 및 인프라를 제어하는 하드웨어 및 소프트웨어 시스템. 제조 분야에서는 OT와 정보 기술 (IT) 시스템의 통합이 인더스트리 4.0 혁신의 핵심 초점입니다.

운영 통합(OI)

클라우드에서 운영을 현대화하는 프로세스로 준비 계획, 자동화 및 통합을 수반합니다. 자세한 내용은 운영 통합 가이드를 참조하십시오.

조직 트레일

이를 통해 AWS CloudTrail 생성되는 트레일은 조직 AWS 계정 내 모든 사용자의 모든 이벤트를 기록합니다. AWS Organizations이 트레일은 조직에 속한 각 AWS 계정 에 생성되고 각 계정의 활동을 추적합니다. 자세한 내용은 CloudTrail 설명서에서 조직을 위한 트레일 만들기를 참조하십시오.

조직 변경 관리(OCM)

사람, 문화 및 리더십 관점에서 중대하고 파괴적인 비즈니스 혁신을 관리하기 위한 프레임워크입니다. OCM은 변화 채택을 가속화하고, 과도기적 문제를 해결하고, 문화 및 조직적 변화를 주도함으로써 조직이 새로운 시스템 및 전략을 준비하고 전환할 수 있도록 지원합니다. 클라우드 채택 프로젝트에 필요한 변화 속도 때문에 AWS 마이그레이션 전략에서는 이 프레임워크를 사용자 가속화라고 합니다. 자세한 내용은 사용 가이드를 참조하십시오.

오리진 액세스 제어(OAC)

CloudFront에서는 Amazon Simple Storage Service (Amazon S3) 콘텐츠의 보안을 위해 액세스를 제한하는 향상된 옵션을 제공합니다. OAC는 모든 S3 버킷 AWS 리전, AWS KMS (SSE-KMS) 를 사용한 서버 측 암호화, S3 버킷에 대한 동적 및 요청을 모두 지원합니다. PUT DELETE

오리진 액세스 ID(OAI)

CloudFront에서는 Amazon S3 콘텐츠 보안을 위해 액세스를 제한하는 옵션입니다. OAI를 사용하면 Amazon S3가 인증할 수 있는 보안 주체를 CloudFront 생성합니다. 인증된 보안 주체는 특정 배

포를 통해서만 S3 버킷의 콘텐츠에 액세스할 수 있습니다. CloudFront 더 세분화되고 향상된 액세스 제어를 제공하는 OAC도 참조하십시오.

또는

운영 준비 상태 검토를 참조하십시오.

아니요

운영 기술을 참조하십시오.

아웃바운드(송신) VPC

AWS 다중 계정 아키텍처에서 애플리케이션 내에서 시작되는 네트워크 연결을 처리하는 VPC입니다. AWS Security Reference Architecture에서는 애플리케이션과 더 넓은 인터넷 간의 양방향 인터페이스를 보호하기 위해 인바운드, 아웃바운드 및 검사 VPC로 네트워크 계정을 설정할 것을 권장합니다.

P

권한 경계

사용자나 역할이 가질 수 있는 최대 권한을 설정하기 위해 IAM 보안 주체에 연결되는 IAM 관리 정책입니다. 자세한 내용은 IAM 설명서의 권한 경계를 참조하십시오.

개인 식별 정보(PII)

직접 보거나 다른 관련 데이터와 함께 짝을 지을 때 개인의 신원을 합리적으로 추론하는 데 사용할 수 있는 정보입니다. PII의 예로는 이름, 주소, 연락처 정보 등이 있습니다.

PII

개인 식별 정보를 참조하십시오.

플레이북

클라우드에서 핵심 운영 기능을 제공하는 등 마이그레이션과 관련된 작업을 캡처하는 일련의 사전 정의된 단계입니다. 플레이북은 스크립트, 자동화된 런북 또는 현대화된 환경을 운영하는 데 필요한 프로세스나 단계 요약의 형태를 취할 수 있습니다.

PLC

프로그래머블 로직 컨트롤러를 참조하십시오.

PLM

제품 라이프사이클 관리를 참조하십시오.

정책

권한을 정의 (ID 기반 정책 참조) 하거나, 액세스 조건을 지정 (리소스 기반 정책 참조) 하거나, 조직 내 모든 계정에 대한 최대 권한을 정의 AWS Organizations (서비스 제어 정책 참조) 할 수 있는 개체입니다.

다국어 지속성

데이터 액세스 패턴 및 기타 요구 사항을 기반으로 독립적으로 마이크로서비스의 데이터 스토리지 기술 선택. 마이크로서비스가 동일한 데이터 스토리지 기술을 사용하는 경우 구현 문제가 발생하거나 성능이 저하될 수 있습니다. 요구 사항에 가장 적합한 데이터 스토어를 사용하면 마이크로서비스를 더 쉽게 구현하고 성능과 확장성을 높일 수 있습니다. 자세한 내용은 마이크로서비스에서 데이터 지속성 활성화를 참조하십시오.

포트폴리오 평가

마이그레이션을 계획하기 위해 애플리케이션 포트폴리오를 검색 및 분석하고 우선순위를 정하는 프로세스입니다. 자세한 내용은 마이그레이션 준비 상태 평가를 참조하십시오.

조건자

일반적으로 조항에 있는 `true` `false` OR를 반환하는 쿼리 조건입니다. `WHERE`

조건부 푸시다운

전송하기 전에 쿼리의 데이터를 필터링하는 데이터베이스 쿼리 최적화 기법입니다. 이렇게 하면 관계형 데이터베이스에서 검색하고 처리해야 하는 데이터의 양이 줄어들고 쿼리 성능이 향상됩니다.

예방적 제어

이벤트 발생을 방지하도록 설계된 보안 제어입니다. 이 제어는 네트워크에 대한 무단 액세스나 원치 않는 변경을 방지하는 데 도움이 되는 1차 방어선입니다. 자세한 내용은 Implementing security controls on AWS의 Preventative controls를 참조하십시오.

보안 주체

작업을 수행하고 리소스에 액세스할 수 AWS 있는 엔티티 이 엔티티는 일반적으로 IAM 역할의 루트 사용자 또는 사용자입니다. AWS 계정자세한 내용은 IAM 설명서의 역할 용어 및 개념의 보안 주체를 참조하십시오.

개인 정보 보호 중심 설계

전체 엔지니어링 프로세스에서 개인 정보를 고려하는 시스템 엔지니어링에서의 접근 방식입니다.

프라이빗 호스팅 영역

Amazon Route 53에서 하나 이상의 VPC 내 도메인과 하위 도메인에 대한 DNS 쿼리에 응답하는 방법에 대한 정보가 담긴 컨테이너입니다. 자세한 내용은 Route 53 설명서의 프라이빗 호스팅 영역 작업을 참조하십시오.

사전 예방 제어

규정을 준수하지 않는 리소스의 배포를 방지하도록 설계된 보안 제어입니다. 이러한 컨트롤은 리소스를 프로비저닝하기 전에 리소스를 스캔합니다. 리소스가 컨트롤과 호환되지 않으면 프로비저닝되지 않습니다. 자세한 내용은 AWS Control Tower 설명서의 컨트롤 참조 안내서를 참조하고 보안 제어 구현의 사전 제어를 참조하십시오. AWS

제품 라이프사이클 관리 (PLM)

설계, 개발, 출시부터 성장 및 성숙도, 폐기 및 제거에 이르는 전체 라이프사이클에 걸쳐 제품에 대한 데이터 및 프로세스를 관리하는 것입니다.

프로덕션 환경

환경을 참조하십시오.

프로그래머블 로직 컨트롤러 (PLC)

제조 분야에서 기계를 모니터링하고 제조 프로세스를 자동화하는 매우 안정적이고 적응력이 뛰어난 컴퓨터입니다.

가명화

데이터세트의 개인 식별자를 자리 표시자 값으로 바꾸는 프로세스입니다. 가명화는 개인 정보를 보호하는 데 도움이 될 수 있습니다. 가명화된 데이터는 여전히 개인 데이터로 간주됩니다.

게시/구독 (게시/구독)

마이크로서비스 간의 비동기 통신을 통해 확장성과 응답성을 개선할 수 있는 패턴입니다. 예를 들어 마이크로서비스 기반 MES에서 마이크로서비스는 다른 마이크로서비스가 구독할 수 있는 채널에 이벤트 메시지를 게시할 수 있습니다. 시스템은 게시 서비스를 변경하지 않고도 새 마이크로서비스를 추가할 수 있습니다.

Q

쿼리 계획

SQL 관계형 데이터베이스 시스템의 데이터에 액세스하는 데 사용되는 일련의 단계 (예: 지침).

쿼리 계획 회귀

데이터베이스 서비스 최적화 프로그램이 데이터베이스 환경을 변경하기 전보다 덜 최적의 계획을 선택하는 경우입니다. 통계, 제한 사항, 환경 설정, 쿼리 파라미터 바인딩 및 데이터베이스 엔진 업데이트의 변경으로 인해 발생할 수 있습니다.

R

RACI 매트릭스

RACI (책임, 책임, 상담, 정보 제공) 를 참조하십시오.

랜섬웨어

결제가 완료될 때까지 컴퓨터 시스템이나 데이터에 대한 액세스를 차단하도록 설계된 악성 소프트웨어입니다.

RASCI 매트릭스

책임, 책임, 상담, 정보 제공 (RACI) 을 참조하십시오.

RCAC

행 및 열 액세스 제어를 참조하십시오.

읽기 전용 복제본

읽기 전용 용도로 사용되는 데이터베이스의 사본입니다. 쿼리를 읽기 전용 복제본으로 라우팅하여 기본 데이터베이스의 로드를 줄일 수 있습니다.

재설계

7 R을 참조하십시오.

Recovery Point Objective(RPO)

마지막 데이터 복구 시점 이후 허용되는 최대 시간입니다. 이에 따라 마지막 복구 시점과 서비스 중단 사이에 허용되는 데이터 손실로 간주되는 범위가 결정됩니다.

Recovery Time Objective(RTO)

서비스 중단과 서비스 복원 사이의 허용 가능한 지연 시간입니다.

리팩터링

7 R을 참조하십시오.

리전

지리적 AWS 영역별 리소스 모음. AWS 리전 각각은 격리되어 있고 서로 독립적이므로 내결함성, 안정성 및 복원력을 제공합니다. 자세한 내용은 사용할 수 있는 AWS 리전 계정 지정을 참조하십시오.

회귀

숫자 값을 예측하는 ML 기법입니다. 예를 들어, '이 집은 얼마에 팔릴까?'라는 문제를 풀기 위해 ML 모델은 선형 회귀 모델을 사용하여 주택에 대해 알려진 사실(예: 면적)을 기반으로 주택의 매매 가격을 예측할 수 있습니다.

리호스팅

7 R을 참조하십시오.

release

배포 프로세스에서 변경 사항을 프로덕션 환경으로 승격시키는 행위입니다.

고쳐 놓다

7 R을 참조하십시오.

리플랫포밍

7 R을 참조하십시오.

환매

7 R을 참조하십시오.

복원력

장애를 견디거나 장애를 복구할 수 있는 애플리케이션의 능력 고가용성 및 재해 복구는 복원력을 계획할 때 일반적으로 고려해야 할 사항입니다. AWS 클라우드자세한 내용은 AWS 클라우드 복원력을 참조하십시오.

리소스 기반 정책

Amazon S3 버킷, 엔드포인트, 암호화 키 등의 리소스에 연결된 정책입니다. 이 유형의 정책은 액세스가 허용된 보안 주체, 지원되는 작업 및 충족해야 하는 기타 조건을 지정합니다.

RACI(Responsible, Accountable, Consulted, Informed) 매트릭스

마이그레이션 활동 및 클라우드 운영에 참여하는 모든 당사자의 역할과 책임을 정의하는 매트릭스입니다. 매트릭스 이름은 매트릭스에 정의된 책임 유형에서 파생됩니다. 실무 담당자 (R), 의사 결

정권자 (A), 업무 수행 조언자 (C), 결과 통보 대상자 (I). 지원자는 (S) 선택사항입니다. 지원자를 포함하면 매트릭스를 RASCI 매트릭스라고 하고, 지원자를 제외하면 RACI 매트릭스라고 합니다.

대응 제어

보안 기준에서 벗어나거나 부정적인 이벤트를 해결하도록 설계된 보안 제어입니다. 자세한 내용은 Implementing security controls on AWS의 Responsive controls를 참조하십시오.

retain

7 R을 참조하십시오.

은퇴

7 R을 참조하십시오.

회전

공격자가 자격 증명에 액세스하는 것을 더 어렵게 만들기 위해 암호를 주기적으로 업데이트하는 프로세스입니다.

행 및 열 액세스 제어(RCAC)

액세스 규칙이 정의된 기본적이고 유연한 SQL 표현식을 사용합니다. RCAC는 행 권한과 열 마스크로 구성됩니다.

RPO

복구 지점 목표를 참조하십시오.

RTO

복구 시간 목표를 참조하십시오.

런북

특정 작업을 수행하는 데 필요한 일련의 수동 또는 자동 절차입니다. 일반적으로 오류율이 높은 반복 작업이나 절차를 간소화하기 위해 런북을 만듭니다.

S

SAML 2.0

많은 ID 제공업체 (IdPs) 가 사용하는 개방형 표준입니다. 이 기능을 사용하면 페더레이션 싱글 사인온 (SSO) 이 가능하므로 조직의 모든 사용자를 위해 IAM에서 사용자를 생성하지 않고도 사용자가 AWS API 작업에 AWS Management Console 로그인하거나 API 작업을 호출할 수 있습니다.

SAML 2.0 기반 페더레이션에 대한 자세한 내용은 IAM 설명서의 SAML 2.0 기반 페더레이션 정보를 참조하십시오.

SCADA

감독 제어 및 데이터 수집을 참조하십시오.

SCP

서비스 제어 정책을 참조하십시오.

secret

에는 AWS Secrets Manager암호화된 형태로 저장하는 비밀번호나 사용자 자격 증명과 같은 기밀 또는 제한된 정보. 비밀 값과 해당 메타데이터로 구성됩니다. 비밀 값은 바이너리, 단일 문자열 또는 여러 문자열일 수 있습니다. 자세한 내용은 Secrets Manager 시크릿에는 무엇이 들어 있나요? 를 참조하십시오. Secrets Manager 설명서에서 확인할 수 있습니다.

보안 제어

위협 행위자가 보안 취약성을 악용하는 능력을 방지, 탐지 또는 감소시키는 기술적 또는 관리적 가드레일입니다. 보안 제어에는 예방적, 탐정적, 대응적, 사전 예방적 제어의 네 가지 기본 유형이 있습니다.

보안 강화

공격 표면을 줄여 공격에 대한 저항력을 높이는 프로세스입니다. 더 이상 필요하지 않은 리소스 제거, 최소 권한 부여의 보안 모범 사례 구현, 구성 파일의 불필요한 기능 비활성화 등의 작업이 여기에 포함될 수 있습니다.

보안 정보 및 이벤트 관리(SIEM) 시스템

보안 정보 관리(SIM)와 보안 이벤트 관리(SEM) 시스템을 결합하는 도구 및 서비스입니다. SIEM 시스템은 서버, 네트워크, 디바이스 및 기타 소스에서 데이터를 수집, 모니터링 및 분석하여 위협과 보안 침해를 탐지하고 알림을 생성합니다.

보안 대응 자동화

보안 이벤트에 자동으로 대응하거나 보안 이벤트를 해결하도록 설계된 사전 정의되고 프로그래밍된 조치입니다. 이러한 자동화는 보안 모범 사례를 구현하는 데 도움이 되는 탐지 또는 대응형 보안 제어 역할을 합니다. AWS 자동 응답 조치의 예로는 VPC 보안 그룹 수정, Amazon EC2 인스턴스 패치, 자격 증명 교체 등이 있습니다.

서버 측 암호화

수신자에 의한 목적지의 데이터 암호화 AWS 서비스

서비스 제어 정책(SCP)

AWS Organizations에 속한 조직의 모든 계정에 대한 권한을 중앙 집중식으로 제어하는 정책입니다. SCP는 관리자가 사용자 또는 역할에 위임할 수 있는 작업에 대해 제한을 설정하거나 가드레일을 정의합니다. SCP를 허용 목록 또는 거부 목록으로 사용하여 허용하거나 금지할 서비스 또는 작업을 지정할 수 있습니다. 자세한 내용은 AWS Organizations 설명서의 서비스 제어 정책을 참조하십시오.

서비스 엔드포인트

의 진입점 URL입니다 AWS 서비스. 엔드포인트를 사용하여 대상 서비스에 프로그래밍 방식으로 연결할 수 있습니다. 자세한 내용은 AWS 일반 참조의 AWS 서비스 엔드포인트를 참조하십시오.

서비스 수준에 관한 계약(SLA)

IT 팀이 고객에게 제공하기로 약속한 내용(예: 서비스 가동 시간 및 성능)을 명시한 계약입니다.

서비스 수준 표시기 (SLI)

오류율, 가용성 또는 처리량과 같은 서비스의 성능 측면을 측정하는 것입니다.

서비스 수준 목표 (SLO)

서비스 수준 지표로 측정되는 서비스 상태를 나타내는 대상 지표입니다.

공동 책임 모델

클라우드 보안 및 규정 준수에 AWS 대한 책임을 공유하는 것을 설명하는 모델입니다. AWS 클라우드의 보안을 책임지는 반면, 사용자는 클라우드에서의 보안을 담당합니다. 자세한 내용은 공동 책임 모델을 참조하십시오.

시엠

보안 정보 및 이벤트 관리 시스템을 참조하십시오.

단일 장애 지점 (SPOF)

응용 프로그램의 중요한 단일 구성 요소에서 발생한 오류로 인해 시스템이 중단될 수 있습니다.

SLA

SLA (서비스 수준 계약) 를 참조하십시오.

SLI

서비스 수준 표시기 참조.

SLO

서비스 수준 목표를 참조하십시오.

split-and-seed 모델

현대화 프로젝트를 확장하고 가속화하기 위한 패턴입니다. 새로운 기능과 제품 릴리스가 정의되면 핵심 팀이 분할되어 새로운 제품 팀이 만들어집니다. 이를 통해 조직의 역량과 서비스 규모를 조정하고, 개발자 생산성을 개선하고, 신속한 혁신을 지원할 수 있습니다. 자세한 내용은 의 애플리케이션 현대화를 위한 단계별 접근 방식을 참조하십시오. AWS 클라우드

SPOF

단일 장애 지점 보기.

스타 스키마

하나의 큰 팩트 테이블을 사용하여 트랜잭션 또는 측정 데이터를 저장하고 하나 이상의 작은 차원 테이블을 사용하여 데이터 속성을 저장하는 데이터베이스 구성 구조입니다. 이 구조는 데이터 웨어하우스에서 사용하거나 비즈니스 인텔리전스 용도로 설계되었습니다.

Strangler Fig 패턴

레거시 시스템을 폐기할 수 있을 때까지 시스템 기능을 점진적으로 다시 작성하고 교체하여 모놀리식 시스템을 현대화하기 위한 접근 방식. 이 패턴은 무화과 덩굴이 나무로 자라 결국 숙주를 압도하고 대체하는 것과 비슷합니다. Martin Fowler가 모놀리식 시스템을 다시 작성할 때 위험을 관리하는 방법으로 이 패턴을 도입했습니다. 이 패턴을 적용하는 방법의 예는 컨테이너 및 Amazon API Gateway를 사용하여 기존의 Microsoft ASP.NET(ASMX) 웹 서비스를 점진적으로 현대화하는 방법을 참조하십시오.

서브넷

VPC의 IP 주소 범위입니다. 서브넷은 단일 가용 영역에 상주해야 합니다.

감독 통제 및 데이터 수집 (SCADA)

제조 시 하드웨어와 소프트웨어를 사용하여 물리적 자산과 생산 작업을 모니터링하는 시스템입니다.

대칭 암호화

동일한 키를 사용하여 데이터를 암호화하고 복호화하는 암호화 알고리즘입니다.

합성 테스트

잠재적 문제를 감지하거나 성능을 모니터링하기 위해 사용자 상호 작용을 시뮬레이션하는 방식으로 시스템을 테스트합니다. Amazon CloudWatch Synthetics를 사용하여 이러한 테스트를 생성할 수 있습니다.

T

tags

리소스 구성을 위한 메타데이터 역할을 하는 키-값 쌍. AWS 태그를 사용하면 리소스를 손쉽게 관리, 식별, 정리, 검색 및 필터링할 수 있습니다. 자세한 내용은 AWS 리소스에 태그 지정을 참조하십시오.

대상 변수

지도 ML에서 예측하려는 값으로, 결과 변수라고도 합니다. 예를 들어, 제조 설정에서 대상 변수는 제품 결함일 수 있습니다.

작업 목록

런북을 통해 진행 상황을 추적하는 데 사용되는 도구입니다. 작업 목록에는 런북의 개요와 완료해야 할 일반 작업 목록이 포함되어 있습니다. 각 일반 작업에 대한 예상 소요 시간, 소유자 및 진행 상황이 작업 목록에 포함됩니다.

테스트 환경

환경을 참조하십시오.

훈련

ML 모델이 학습할 수 있는 데이터를 제공하는 것입니다. 훈련 데이터에는 정답이 포함되어야 합니다. 학습 알고리즘은 훈련 데이터에서 대상(예측하려는 답)에 입력 데이터 속성을 매핑하는 패턴을 찾고, 이러한 패턴을 캡처하는 ML 모델을 출력합니다. 그런 다음 ML 모델을 사용하여 대상을 모르는 새 데이터에 대한 예측을 할 수 있습니다.

전송 게이트웨이

VPC와 온프레미스 네트워크를 상호 연결하는 데 사용할 수 있는 네트워크 전송 허브입니다. 자세한 내용은 AWS Transit Gateway 설명서의 트랜짓 게이트웨이란 무엇입니까? 를 참조하십시오.

트렁크 기반 워크플로

개발자가 기능 브랜치에서 로컬로 기능을 구축하고 테스트한 다음 해당 변경 사항을 기본 브랜치에 병합하는 접근 방식입니다. 이후 기본 브랜치는 개발, 프로덕션 이전 및 프로덕션 환경에 순차적으로 구축됩니다.

신뢰할 수 있는 액세스

조직 내 AWS Organizations 및 해당 계정에서 사용자를 대신하여 작업을 수행하도록 지정한 서비스에 권한 부여 신뢰할 수 있는 서비스는 필요할 때 각 계정에 서비스 연결 역할을 생성하여 관

리 작업을 수행합니다. 자세한 내용은 AWS Organizations 설명서의 다른 AWS 서비스와 AWS Organizations 함께 사용을 참조하십시오.

튜닝

ML 모델의 정확도를 높이기 위해 훈련 프로세스의 측면을 여러 변경하는 것입니다. 예를 들어, 레이블링 세트를 생성하고 레이블을 추가한 다음 다양한 설정에서 이러한 단계를 여러 번 반복하여 모델을 최적화하는 방식으로 ML 모델을 훈련할 수 있습니다.

피자 두 판 팀

피자 두 판만 들고 배불리 먹을 수 있는 소규모 DevOps 팀. 피자 두 판 팀 규모는 소프트웨어 개발에 있어 가능한 최상의 공동 작업 기회를 보장합니다.

U

불확실성

예측 ML 모델의 신뢰성을 저해할 수 있는 부정확하거나 불완전하거나 알려지지 않은 정보를 나타내는 개념입니다. 불확실성에는 두 가지 유형이 있습니다. 인식론적 불확실성은 제한적이고 불완전한 데이터에 의해 발생하는 반면, 우연한 불확실성은 데이터에 내재된 노이즈와 무작위성에 의해 발생합니다. 자세한 내용은 Quantifying uncertainty in deep learning systems 가이드를 참조하십시오.

차별화되지 않은 작업

애플리케이션을 만들고 운영하는 데 필요하지만 최종 사용자에게 직접적인 가치를 제공하거나 경쟁 우위를 제공하지 못하는 작업을 헤비 리프팅이라고도 합니다. 차별화되지 않은 작업의 예로는 조달, 유지보수, 용량 계획 등이 있습니다.

상위 환경

환경을 보세요.

V

정리

스토리지를 회수하고 성능을 향상시키기 위해 증분 업데이트 후 정리 작업을 수반하는 데이터베이스 유지 관리 작업입니다.

버전 제어

리포지토리의 소스 코드 변경과 같은 변경 사항을 추적하는 프로세스 및 도구입니다.

VPC 피어링

프라이빗 IP 주소를 사용하여 트래픽을 라우팅할 수 있게 하는 두 VPC 간의 연결입니다. 자세한 내용은 Amazon VPC 설명서의 VPC 피어링이란?을 참조하십시오.

취약성

시스템 보안을 손상시키는 소프트웨어 또는 하드웨어 결함입니다.

W

웜 캐시

자주 액세스하는 최신 관련 데이터를 포함하는 버퍼 캐시입니다. 버퍼 캐시에서 데이터베이스 인스턴스를 읽을 수 있기 때문에 주 메모리나 디스크에서 읽는 것보다 빠릅니다.

웜 데이터

자주 액세스하지 않는 데이터입니다. 이런 종류의 데이터를 쿼리할 때는 일반적으로 적절히 느린 쿼리가 허용됩니다.

윈도우 함수

현재 레코드와 어떤 식으로든 관련된 행 그룹에 대해 계산을 수행하는 SQL 함수입니다. 윈도우 함수는 이동 평균을 계산하거나 현재 행의 상대적 위치를 기반으로 행 값에 액세스하는 등의 작업을 처리하는 데 유용합니다.

워크로드

고객 대면 애플리케이션이나 백엔드 프로세스 같이 비즈니스 가치를 창출하는 리소스 및 코드 모음입니다.

워크스트림

마이그레이션 프로젝트에서 특정 작업 세트를 담당하는 직무 그룹입니다. 각 워크스트림은 독립적이지만 프로젝트의 다른 워크스트림을 지원합니다. 예를 들어, 포트폴리오 워크스트림은 애플리케이션 우선순위 지정, 웨이브 계획, 마이그레이션 메타데이터 수집을 담당합니다. 포트폴리오 워크스트림은 이러한 자산을 마이그레이션 워크스트림에 전달하고, 마이그레이션 워크스트림은 서버와 애플리케이션을 마이그레이션합니다.

웜

한 번 쓰고, 많이 읽으세요.

WQF

AWS 워크로드 검증 프레임워크를 참조하십시오.

한 번 작성하고 여러 번 읽기 (WORM)

데이터를 한 번 쓰고 데이터가 삭제되거나 수정되지 않도록 하는 스토리지 모델입니다. 인증된 사용자는 필요한 만큼 데이터를 여러 번 읽을 수 있지만 변경할 수는 없습니다. 이 데이터 스토리지 인프라는 변경할 수 없는 것으로 간주됩니다.

Z

제로데이 익스플로잇

제로데이 취약점을 악용하는 공격 (일반적으로 멀웨어) 입니다.

제로데이 취약성

프로덕션 시스템의 명백한 결함 또는 취약성입니다. 위협 행위자는 이러한 유형의 취약성을 사용하여 시스템을 공격할 수 있습니다. 개발자는 공격의 결과로 취약성을 인지하는 경우가 많습니다.

좀비 애플리케이션

평균 CPU 및 메모리 사용량이 5% 미만인 애플리케이션입니다. 마이그레이션 프로젝트에서는 이러한 애플리케이션을 사용 중지하는 것이 일반적입니다.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.