Amazon’s trademarks and trade dress may not be used in connection with any product or service that is not Amazon’s, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.
Table of Contents

Welcome ............................................................................................................................................. 1
Actions ............................................................................................................................................. 2
Amazon DynamoDB ............................................................................................................................................. 3
  BatchGetItem .................................................................................................................................. 5
  BatchWriteItem .......................................................................................................................... 13
  CreateBackup .......................................................................................................................... 22
  CreateGlobalTable .................................................................................................................. 25
  CreateTable .............................................................................................................................. 28
  DeleteBackup ........................................................................................................................ 37
  DeleteItem .............................................................................................................................. 40
  DeleteTable ........................................................................................................................... 49
  DescribeBackup ...................................................................................................................... 54
  DescribeContinuousBackups ................................................................................................... 57
  DescribeGlobalTable ............................................................................................................... 59
  DescribeLimits ....................................................................................................................... 61
  DescribeTable ....................................................................................................................... 64
  DescribeTimeToLive ............................................................................................................... 69
  GetItem ..................................................................................................................................... 71
  ListBackups ........................................................................................................................... 77
  ListGlobalTables .................................................................................................................... 80
  ListTables .................................................................................................................................. 82
  ListTagsOfResource .................................................................................................................. 85
  PutItem ..................................................................................................................................... 87
  Query ......................................................................................................................................... 96
  RestoreTableFromBackup ...................................................................................................... 109
  Scan ......................................................................................................................................... 113
  TagResource .......................................................................................................................... 126
  UntagResource ...................................................................................................................... 128
  UpdateGlobalTable .................................................................................................................. 130
  UpdateItem ........................................................................................................................... 133
  UpdateTable .......................................................................................................................... 145
  UpdateTimeToLive .................................................................................................................. 152
Amazon DynamoDB Accelerator ............................................................................................................. 154
  CreateCluster ......................................................................................................................... 156
  CreateParameterGroup .......................................................................................................... 162
  CreateSubnetGroup ............................................................................................................... 164
  DecreaseReplicationFactor ...................................................................................................... 166
  DeleteCluster ........................................................................................................................ 169
  DeleteParameterGroup .......................................................................................................... 172
  DeleteSubnetGroup .................................................................................................................. 174
  DescribeClusters ................................................................................................................... 176
  DescribeDefaultParameters ................................................................................................. 179
  DescribeEvents ...................................................................................................................... 181
  DescribeParameterGroups ...................................................................................................... 184
  DescribeParameters ............................................................................................................... 187
  DescribeSubnetGroups ........................................................................................................... 190
  IncreaseReplicationFactor ...................................................................................................... 192
  ListTags ................................................................................................................................... 195
  RebootNode .......................................................................................................................... 197
  TagResource .......................................................................................................................... 200
  UntagResource ...................................................................................................................... 203
  UpdateCluster ........................................................................................................................ 205
  UpdateParameterGroup ......................................................................................................... 209
  UpdateSubnetGroup ............................................................................................................... 211
Welcome

Amazon DynamoDB provides low-level API actions for managing database tables and indexes, and for creating, reading, updating and deleting data. DynamoDB also provides API actions for accessing and processing stream records.

Note
This API Reference describes the low-level API for Amazon DynamoDB. Instead of making requests to the low-level API directly from your application, we recommend that you use one of the AWS Software Development Kits (SDKs) for your programming language. The AWS SDKs take care of request authentication, serialization, and connection management. For more information, see Overview of AWS SDK Support for DynamoDB in the Amazon DynamoDB Developer Guide.

At the end of each API action description there are links to the equivalent CLI command and programming-specific language method. Similarly, at the end of each API datatype description, there are links to the equivalent programming-specific language type.
Actions

The following actions are supported by Amazon DynamoDB:

- BatchGetItem (p. 5)
- BatchWriteItem (p. 13)
- CreateBackup (p. 22)
- CreateGlobalTable (p. 25)
- CreateTable (p. 28)
- DeleteBackup (p. 37)
- DeleteItem (p. 40)
- DeleteTable (p. 49)
- DescribeBackup (p. 54)
- DescribeContinuousBackups (p. 57)
- DescribeGlobalTable (p. 59)
- DescribeLimits (p. 61)
- DescribeTable (p. 64)
- DescribeTimeToLive (p. 69)
- GetItem (p. 71)
- ListBackups (p. 77)
- ListGlobalTables (p. 80)
- ListTables (p. 82)
- ListTagsOfResource (p. 85)
- PutItem (p. 87)
- Query (p. 96)
- RestoreTableFromBackup (p. 109)
- Scan (p. 113)
- TagResource (p. 126)
- UntagResource (p. 128)
- UpdateGlobalTable (p. 130)
- UpdateItem (p. 133)
- UpdateTable (p. 145)
- UpdateTimeToLive (p. 152)

The following actions are supported by Amazon DynamoDB Accelerator:

- CreateCluster (p. 156)
- CreateParameterGroup (p. 162)
- CreateSubnetGroup (p. 164)
- DecreaseReplicationFactor (p. 166)
- DeleteCluster (p. 169)
- DeleteParameterGroup (p. 172)
- DeleteSubnetGroup (p. 174)
- DescribeClusters (p. 176)
• DescribeDefaultParameters (p. 179)
• DescribeEvents (p. 181)
• DescribeParameterGroups (p. 184)
• DescribeParameters (p. 187)
• DescribeSubnetGroups (p. 190)
• IncreaseReplicationFactor (p. 192)
• ListTags (p. 195)
• RebootNode (p. 197)
• TagResource (p. 200)
• UntagResource (p. 203)
• UpdateCluster (p. 205)
• UpdateParameterGroup (p. 209)
• UpdateSubnetGroup (p. 211)

The following actions are supported by Amazon DynamoDB Streams:

• DescribeStream (p. 214)
• GetRecords (p. 218)
• GetShardIterator (p. 223)
• ListStreams (p. 227)

Amazon DynamoDB

The following actions are supported by Amazon DynamoDB:

• BatchGetItem (p. 5)
• BatchWriteItem (p. 13)
• CreateBackup (p. 22)
• CreateGlobalTable (p. 25)
• CreateTable (p. 28)
• DeleteBackup (p. 37)
• DeleteItem (p. 40)
• DeleteTable (p. 49)
• DescribeBackup (p. 54)
• DescribeContinuousBackups (p. 57)
• DescribeGlobalTable (p. 59)
• DescribeLimits (p. 61)
• DescribeTable (p. 64)
• DescribeTimeToLive (p. 69)
• GetItem (p. 71)
• ListBackups (p. 77)
• ListGlobalTables (p. 80)
• ListTables (p. 82)
• ListTagsOfResource (p. 85)
• PutItem (p. 87)
• Query (p. 96)
• RestoreTableFromBackup (p. 109)
• Scan (p. 113)
• TagResource (p. 126)
• UntagResource (p. 128)
• UpdateGlobalTable (p. 130)
• UpdateItem (p. 133)
• UpdateTable (p. 145)
• UpdateTimeToLive (p. 152)
BatchGetItem

Service: Amazon DynamoDB

The BatchGetItem operation returns the attributes of one or more items from one or more tables. You identify requested items by primary key.

A single operation can retrieve up to 16 MB of data, which can contain as many as 100 items. BatchGetItem will return a partial result if the response size limit is exceeded, the table's provisioned throughput is exceeded, or an internal processing failure occurs. If a partial result is returned, the operation returns a value for UnprocessedKeys. You can use this value to retry the operation starting with the next item to get.

**Important**

If you request more than 100 items BatchGetItem will return a ValidationException with the message "Too many items requested for the BatchGetItem call".

For example, if you ask to retrieve 100 items, but each individual item is 300 KB in size, the system returns 52 items (so as not to exceed the 16 MB limit). It also returns an appropriate UnprocessedKeys value so you can get the next page of results. If desired, your application can include its own logic to assemble the pages of results into one data set.

If none of the items can be processed due to insufficient provisioned throughput on all of the tables in the request, then BatchGetItem will return a ProvisionedThroughputExceededException. If at least one of the items is successfully processed, then BatchGetItem completes successfully, while returning the keys of the unread items in UnprocessedKeys.

**Important**

If DynamoDB returns any unprocessed items, you should retry the batch operation on those items. However, we strongly recommend that you use an exponential backoff algorithm. If you retry the batch operation immediately, the underlying read or write requests can still fail due to throttling on the individual tables. If you delay the batch operation using exponential backoff, the individual requests in the batch are much more likely to succeed.

For more information, see Batch Operations and Error Handling in the Amazon DynamoDB Developer Guide.

By default, BatchGetItem performs eventually consistent reads on every table in the request. If you want strongly consistent reads instead, you can set ConsistentRead to true for any or all tables.

In order to minimize response latency, BatchGetItem retrieves items in parallel.

When designing your application, keep in mind that DynamoDB does not return items in any particular order. To help parse the response by item, include the primary key values for the items in your request in the ProjectionExpression parameter.

If a requested item does not exist, it is not returned in the result. Requests for nonexistent items consume the minimum read capacity units according to the type of read. For more information, see Capacity Units Calculations in the Amazon DynamoDB Developer Guide.

**Request Syntax**

```json
{
    "RequestItems": {
        "string": {
            "AttributesToGet": [ "string" ],
            "ConsistentRead": boolean,
            "ExpressionAttributeNames": {
                "string": "string"
            },
            "Keys": [
```
The request accepts the following data in JSON format.

Note
In the following list, the required parameters are described first.

RequestItems (p. 5)
A map of one or more table names and, for each table, a map that describes one or more items to retrieve from that table. Each table name can be used only once per BatchGetItem request.

Each element in the map of items to retrieve consists of the following:

- **ConsistentRead** - If true, a strongly consistent read is used; if false (the default), an eventually consistent read is used.
- **ExpressionAttributeNames** - One or more substitution tokens for attribute names in the ProjectionExpression parameter. The following are some use cases for using ExpressionAttributeNames:
  - To access an attribute whose name conflicts with a DynamoDB reserved word.
  - To create a placeholder for repeating occurrences of an attribute name in an expression.
  - To prevent special characters in an attribute name from being misinterpreted in an expression.

Use the # character in an expression to dereference an attribute name. For example, consider the following attribute name:

- **Percentile**

The name of this attribute conflicts with a reserved word, so it cannot be used directly in an expression. (For the complete list of reserved words, see Reserved Words in the Amazon DynamoDB Developer Guide). To work around this, you could specify the following for ExpressionAttributeNames:

- `{"#P":"Percentile"}`

You could then use this substitution in an expression, as in this example:

- `#P = :val`
**Note**
Tokens that begin with the : character are *expression attribute values*, which are placeholders for the actual value at runtime.

For more information on expression attribute names, see Accessing Item Attributes in the Amazon DynamoDB Developer Guide.

- **Keys** - An array of primary key attribute values that define specific items in the table. For each primary key, you must provide *all* of the key attributes. For example, with a simple primary key, you only need to provide the partition key value. For a composite key, you must provide *both* the partition key value and the sort key value.
- **ProjectionExpression** - A string that identifies one or more attributes to retrieve from the table. These attributes can include scalars, sets, or elements of a JSON document. The attributes in the expression must be separated by commas.

If no attribute names are specified, then all attributes will be returned. If any of the requested attributes are not found, they will not appear in the result.

For more information, see Accessing Item Attributes in the Amazon DynamoDB Developer Guide.

- **AttributesToGet** - This is a legacy parameter. Use ProjectionExpression instead. For more information, see AttributesToGet in the Amazon DynamoDB Developer Guide.

Type: String to KeysAndAttributes (p. 274) object map


Key Pattern: [\w\-]+

Required: Yes

**ReturnConsumedCapacity (p. 5)**

Determines the level of detail about provisioned throughput consumption that is returned in the response:

- **INDEXES** - The response includes the aggregate ConsumedCapacity for the operation, together with ConsumedCapacity for each table and secondary index that was accessed.

  Note that some operations, such as GetItem and BatchGetItem, do not access any indexes at all. In these cases, specifying INDEXES will only return ConsumedCapacity information for table(s).

- **TOTAL** - The response includes only the aggregate ConsumedCapacity for the operation.

- **NONE** - No ConsumedCapacity details are included in the response.

Type: String

Valid Values: INDEXES | TOTAL | NONE

Required: No

**Response Syntax**

```json
{
   "ConsumedCapacity": [
   {
      "CapacityUnits": number,
      "GlobalSecondaryIndexes": {
         "string" : {
            
```
"CapacityUnits": number
},
"LocalSecondaryIndexes": {
  "string": {
    "CapacityUnits": number
  }
},
"Table": {
  "CapacityUnits": number
},
"TableName": "string"
],
"Responses": {
  "string": [
  {
    "string": {
      "B": blob,
      "BOOL": boolean,
      "BS": [ blob ],
      "L": [ "AttributeValue"
        ],
      "M": {
        "string": "AttributeValue"
      },
      "N": "string",
      "NS": [ "string" ],
      "NULL": boolean,
      "S": "string",
      "SS": [ "string" ]
    }
  }
  },
  "UnprocessedKeys": {
  "string": {
    "AttributesToGet": [ "string" ],
    "ConsistentRead": boolean,
    "ExpressionAttributeNames": {
      "string": "string"
    },
    "Keys": [
    {
      "string": {
        "B": blob,
        "BOOL": boolean,
        "BS": [ blob ],
        "L": [ "AttributeValue"
          ],
        "M": {
          "string": "AttributeValue"
        },
        "N": "string",
        "NS": [ "string" ],
        "NULL": boolean,
        "S": "string",
        "SS": [ "string" ]
      }
    }
    ],
    "ProjectionExpression": "string"
  }
  }
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**ConsumedCapacity (p. 7)**

The read capacity units consumed by the entire `BatchGetItem` operation.

Each element consists of:
- `TableName` - The table that consumed the provisioned throughput.
- `CapacityUnits` - The total number of capacity units consumed.

Type: Array of `ConsumedCapacity (p. 249)` objects

**Responses (p. 7)**

A map of table name to a list of items. Each object in `Responses` consists of a table name, along with a map of attribute data consisting of the data type and attribute value.

Type: String to array of string to `AttributeValue (p. 235)` object maps map


Key Pattern: `[a-zA-Z0-9_.-]+`

Key Length Constraints: Maximum length of 65535.

**UnprocessedKeys (p. 7)**

A map of tables and their respective keys that were not processed with the current response. The `UnprocessedKeys` value is in the same form as `RequestItems`, so the value can be provided directly to a subsequent `BatchGetItem` operation. For more information, see `RequestItems` in the Request Parameters section.

Each element consists of:
- `Keys` - An array of primary key attribute values that define specific items in the table.
- `ProjectionExpression` - One or more attributes to be retrieved from the table or index. By default, all attributes are returned. If a requested attribute is not found, it does not appear in the result.
- `ConsistentRead` - The consistency of a read operation. If set to `true`, then a strongly consistent read is used; otherwise, an eventually consistent read is used.

If there are no unprocessed keys remaining, the response contains an empty `UnprocessedKeys` map.

Type: String to `KeysAndAttributes (p. 274)` object map


Key Pattern: `[a-zA-Z0-9_.-]+`

**Errors**

For information about the errors that are common to all actions, see `Common Errors (p. 341)`.
InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

ProvisionedThroughputExceededException

Your request rate is too high. The AWS SDKs for DynamoDB automatically retry requests that receive this exception. Your request is eventually successful, unless your retry queue is too large to finish. Reduce the frequency of requests and use exponential backoff. For more information, go to Error Retries and Exponential Backoff in the Amazon DynamoDB Developer Guide.

HTTP Status Code: 400

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

Example

Retrieve Items From Multiple Tables

The following sample requests attributes from two different tables.

Sample Request

```
POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>, Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.BatchGetItem

{
  "RequestItems": {
    "Forum": {
      "Keys": [
        {
          "Name": "Amazon DynamoDB"
        },
        {
          "Name": "Amazon RDS"
        },
        {
          "Name": "Amazon Redshift"
        }
      ],
      "ProjectionExpression": "Name, Threads, Messages, Views"
    },
    "Thread": {
      "Keys": [
        {
          "ForumName": "Amazon DynamoDB",
          "Subject": "Concurrent reads"
        }
      ]
    }
  }
}
```
BatchGetItem

Sample Response

HTTP/1.1 200 OK
x-amzn-RequestID: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>
{
  "Responses": {
    "Forum": [
      {
        "Name": {
          "S": "Amazon DynamoDB"
        },
        "Threads": {
          "N": "5"
        },
        "Messages": {
          "N": "19"
        },
        "Views": {
          "N": "35"
        }
      },
      {
        "Name": {
          "S": "Amazon RDS"
        },
        "Threads": {
          "N": "8"
        },
        "Messages": {
          "N": "32"
        },
        "Views": {
          "N": "38"
        }
      },
      {
        "Name": {
          "S": "Amazon Redshift"
        },
        "Threads": {
          "N": "12"
        },
        "Messages": {
          "N": "55"
        },
        "Views": {
          "N": "47"
        }
      }
    ]
  }
}
```
"Thread": [
    {
        "Tags": {
            "SS": ["Reads", "MultipleUsers"]
        },
        "Message": {
            "S": "How many users can read a single data item at a time? Are there any limits?"
        }
    }
],
"UnprocessedKeys": {
},
"ConsumedCapacity": [
    {
        "TableName": "Forum",
        "CapacityUnits": 3
    },
    {
        "TableName": "Thread",
        "CapacityUnits": 1
    }
]
```

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
**BatchWriteItem**

**Service: Amazon DynamoDB**

The BatchWriteItem operation puts or deletes multiple items in one or more tables. A single call to BatchWriteItem can write up to 16 MB of data, which can comprise as many as 25 put or delete requests. Individual items to be written can be as large as 400 KB.

**Note**

BatchWriteItem cannot update items. To update items, use the UpdateItem action.

The individual PutItem and DeleteItem operations specified in BatchWriteItem are atomic; however BatchWriteItem as a whole is not. If any requested operations fail because the table's provisioned throughput is exceeded or an internal processing failure occurs, the failed operations are returned in the UnprocessedItems response parameter. You can investigate and optionally resend the requests. Typically, you would call BatchWriteItem in a loop. Each iteration would check for unprocessed items and submit a new BatchWriteItem request with those unprocessed items until all items have been processed.

Note that if none of the items can be processed due to insufficient provisioned throughput on all of the tables in the request, then BatchWriteItem will return a ProvisionedThroughputExceededException.

**Important**

If DynamoDB returns any unprocessed items, you should retry the batch operation on those items. However, we strongly recommend that you use an exponential backoff algorithm. If you retry the batch operation immediately, the underlying read or write requests can still fail due to throttling on the individual tables. If you delay the batch operation using exponential backoff, the individual requests in the batch are much more likely to succeed.

For more information, see Batch Operations and Error Handling in the Amazon DynamoDB Developer Guide.

With BatchWriteItem, you can efficiently write or delete large amounts of data, such as from Amazon Elastic MapReduce (EMR), or copy data from another database into DynamoDB. In order to improve performance with these large-scale operations, BatchWriteItem does not behave in the same way as individual PutItem and DeleteItem calls would. For example, you cannot specify conditions on individual put and delete requests, and BatchWriteItem does not return deleted items in the response.

If you use a programming language that supports concurrency, you can use threads to write items in parallel. Your application must include the necessary logic to manage the threads. With languages that don't support threading, you must update or delete the specified items one at a time. In both situations, BatchWriteItem performs the specified put and delete operations in parallel, giving you the power of the thread pool approach without having to introduce complexity into your application.

Parallel processing reduces latency, but each specified put and delete request consumes the same number of write capacity units whether it is processed in parallel or not. Delete operations on nonexistent items consume one write capacity unit.

If one or more of the following is true, DynamoDB rejects the entire batch write operation:

- One or more tables specified in the BatchWriteItem request does not exist.
- Primary key attributes specified on an item in the request do not match those in the corresponding table's primary key schema.
- You try to perform multiple operations on the same item in the same BatchWriteItem request. For example, you cannot put and delete the same item in the same BatchWriteItem request.
- There are more than 25 requests in the batch.
- Any individual item in a batch exceeds 400 KB.
- The total request size exceeds 16 MB.
Request Syntax

```json
{
  "RequestItems": {
    "string": [
      {
        "DeleteRequest": {
          "Key": {
            "string": {
              "B": blob,
              "BOOL": boolean,
              "BS": [ blob ],
              "L": [               
                "AttributeValue"
              ],
              "M": {               
                "string": "AttributeValue"
              },
              "N": "string",
              "NS": [ "string" ],
              "NULL": boolean,
              "S": "string",
              "SS": [ "string" ]
          }
        },
        "PutRequest": {
          "Item": {
            "string": {
              "B": blob,
              "BOOL": boolean,
              "BS": [ blob ],
              "L": [               
                "AttributeValue"
              ],
              "M": {               
                "string": "AttributeValue"
              },
              "N": "string",
              "NS": [ "string" ],
              "NULL": boolean,
              "S": "string",
              "SS": [ "string" ]
          }
        }
      }
    }
  },
  "ReturnConsumedCapacity": "string",
  "ReturnItemCollectionMetrics": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**

In the following list, the required parameters are described first.
RequestItems (p. 14)

A map of one or more table names and, for each table, a list of operations to be performed (DeleteRequest or PutRequest). Each element in the map consists of the following:

- **DeleteRequest** - Perform a DeleteItem operation on the specified item. The item to be deleted is identified by a **Key** subelement:
  - **Key** - A map of primary key attribute values that uniquely identify the item. Each entry in this map consists of an attribute name and an attribute value. For each primary key, you must provide all of the key attributes. For example, with a simple primary key, you only need to provide a value for the partition key. For a composite primary key, you must provide values for both the partition key and the sort key.

- **PutRequest** - Perform a PutItem operation on the specified item. The item to be put is identified by an **Item** subelement:
  - **Item** - A map of attributes and their values. Each entry in this map consists of an attribute name and an attribute value. Attribute values must not be null; string and binary type attributes must have lengths greater than zero; and set type attributes must not be empty. Requests that contain empty values will be rejected with a **ValidationException** exception.

If you specify any attributes that are part of an index key, then the data types for those attributes must match those of the schema in the table's attribute definition.

Type: String to array of WriteRequest (p. 306) objects map


Key Pattern: [a-zA-Z0-9_.-]+

Array Members: Minimum number of 1 item. Maximum number of 25 items.

Required: Yes

ReturnConsumedCapacity (p. 14)

Determines the level of detail about provisioned throughput consumption that is returned in the response:

- **INDEXES** - The response includes the aggregate ConsumedCapacity for the operation, together with ConsumedCapacity for each table and secondary index that was accessed.

  Note that some operations, such as GetItem and BatchGetItem, do not access any indexes at all. In these cases, specifying INDEXES will only return ConsumedCapacity information for table(s).

- **TOTAL** - The response includes only the aggregate ConsumedCapacity for the operation.

- **NONE** - No ConsumedCapacity details are included in the response.

Type: String

Valid Values: INDEXES | TOTAL | NONE

Required: No

ReturnItemCollectionMetrics (p. 14)

Determines whether item collection metrics are returned. If set to **SIZE**, the response includes statistics about item collections, if any, that were modified during the operation are returned in the response. If set to **NONE** (the default), no statistics are returned.

Type: String
Valid Values: SIZE | NONE

Required: No

Response Syntax

```
{
  "ConsumedCapacity": [
    {
      "CapacityUnits": number,
      "GlobalSecondaryIndexes": {
        "string": {
          "CapacityUnits": number
        }
      },
      "LocalSecondaryIndexes": {
        "string": {
          "CapacityUnits": number
        }
      },
      "Table": {
        "CapacityUnits": number
      },
      "TableName": "string"
    }
  ],
  "ItemCollectionMetrics": {
    "string": [
      {
        "ItemCollectionKey": {
          "string": {
            "B": blob,
            "BOOL": boolean,
            "BS": [ blob ],
            "L": [
              "AttributeValue"
            ],
            "M": {
              "string": "AttributeValue"
            },
            "N": "string",
            "NS": [ "string" ],
            "NULL": boolean,
            "S": "string",
            "SS": [ "string" ]
          }
        },
        "SizeEstimateRangeGB": [ number ]
      }
    ]
  }
},
"UnprocessedItems": {
  "string": [
    {
      "DeleteRequest": {
        "Key": {
          "string": {
            "B": blob,
            "BOOL": boolean,
            "BS": [ blob ],
            "L": [
              "AttributeValue"
            ],
            "M": {
              "string": "AttributeValue"
            },
            "N": "string",
            "NS": [ "string" ],
            "NULL": boolean,
            "S": "string",
            "SS": [ "string" ]
          }
        }
      }
    }
  ]
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ConsumedCapacity (p. 16)

The capacity units consumed by the entire BatchWriteItem operation.

Each element consists of:
- TableName - The table that consumed the provisioned throughput.
- CapacityUnits - The total number of capacity units consumed.

Type: Array of ConsumedCapacity (p. 249) objects

ItemCollectionMetrics (p. 16)

A list of tables that were processed by BatchWriteItem and, for each table, information about any item collections that were affected by individual DeleteItem or PutItem operations.

Each entry consists of the following subelements:
- ItemCollectionKey - The partition key value of the item collection. This is the same as the partition key value of the item.
- SizeEstimateRangeGB - An estimate of item collection size, expressed in GB. This is a two-element array containing a lower bound and an upper bound for the estimate. The estimate
includes the size of all the items in the table, plus the size of all attributes projected into all of the local secondary indexes on the table. Use this estimate to measure whether a local secondary index is approaching its size limit.

The estimate is subject to change over time; therefore, do not rely on the precision or accuracy of the estimate.

Type: String to array of ItemCollectionMetrics (p. 273) objects map


Key Pattern: [a-zA-Z0-9_.-]+

UnprocessedItems (p. 16)

A map of tables and requests against those tables that were not processed. The UnprocessedItems value is in the same form as RequestItems, so you can provide this value directly to a subsequent BatchGetItem operation. For more information, see RequestItems in the Request Parameters section.

Each UnprocessedItems entry consists of a table name and, for that table, a list of operations to perform (DeleteRequest or PutRequest).

- **DeleteRequest** - Perform a DeleteItem operation on the specified item. The item to be deleted is identified by a Key subelement:
  - **Key** - A map of primary key attribute values that uniquely identify the item. Each entry in this map consists of an attribute name and an attribute value.

- **PutRequest** - Perform a PutItem operation on the specified item. The item to be put is identified by an Item subelement:
  - **Item** - A map of attributes and their values. Each entry in this map consists of an attribute name and an attribute value. Attribute values must not be null; string and binary type attributes must have lengths greater than zero; and set type attributes must not be empty. Requests that contain empty values will be rejected with a ValidationException exception.

  If you specify any attributes that are part of an index key, then the data types for those attributes must match those of the schema in the table's attribute definition.

If there are no unprocessed items remaining, the response contains an empty UnprocessedItems map.

Type: String to array of WriteRequest (p. 306) objects map


Key Pattern: [a-zA-Z0-9_.-]+

Array Members: Minimum number of 1 item. Maximum number of 25 items.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

**InternalServerError**

An error occurred on the server side.

HTTP Status Code: 500

**ItemCollectionSizeLimitExceeded**

An item collection is too large. This exception is only returned for tables that have one or more local secondary indexes.
HTTP Status Code: 400

ProvisionedThroughputExceededException

Your request rate is too high. The AWS SDKs for DynamoDB automatically retry requests that receive this exception. Your request is eventually successful, unless your retry queue is too large to finish. Reduce the frequency of requests and use exponential backoff. For more information, go to Error Retries and Exponential Backoff in the Amazon DynamoDB Developer Guide.

HTTP Status Code: 400

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

Example

Multiple Operations on One Table

This example writes several items to the Forum table. The response shows that the final put operation failed, possibly because the application exceeded the provisioned throughput on the table. The UnprocessedItems object shows the unsuccessful put request. The application can call BatchWriteItem again to address such unprocessed requests.

Sample Request

```json
POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.BatchWriteItem

{
    "RequestItems": {
        "Forum": [
            {
                "PutRequest": {
                    "Item": {
                        "Name": { "S": "Amazon DynamoDB" }
                    }
                }
            },
            {
                "PutRequest": {
                    "Item": {
                        "Name": { "S": "Amazon RDS" }
                    }
                }
            }
        ]
    }
}
```
"Category": {  
  "S": "Amazon Web Services"  
}  
},  
"PutRequest": {  
  "Item": {  
    "Name": {  
      "S": "Amazon Redshift"  
    },  
    "Category": {  
      "S": "Amazon Web Services"  
    }  
  } 
},  
"PutRequest": {  
  "Item": {  
    "Name": {  
      "S": "Amazon ElastiCache"  
    },  
    "Category": {  
      "S": "Amazon Web Services"  
    }  
  } 
}  
],  
"ReturnConsumedCapacity": "TOTAL"  
}
"TableName": "Forum",
"CapacityUnits": 3
]
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateBackup
Service: Amazon DynamoDB

Creates a backup for an existing table.

Each time you create an On-Demand Backup, the entire table data is backed up. There is no limit to the number of on-demand backups that can be taken.

When you create an On-Demand Backup, a time marker of the request is cataloged, and the backup is created asynchronously, by applying all changes until the time of the request to the last full table snapshot. Backup requests are processed instantaneously and become available for restore within minutes.

You can call CreateBackup at a maximum rate of 50 times per second.

All backups in DynamoDB work without consuming any provisioned throughput on the table.

If you submit a backup request on 2018-12-14 at 14:25:00, the backup is guaranteed to contain all data committed to the table up to 14:24:00, and data committed after 14:26:00 will not be. The backup may or may not contain data modifications made between 14:24:00 and 14:26:00. On-Demand Backup does not support causal consistency.

Along with data, the following are also included on the backups:
- Global secondary indexes (GSIs)
- Local secondary indexes (LSIs)
- Streams
- Provisioned read and write capacity

Request Syntax

```
{
  "BackupName": "string",
  "TableName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**BackupName (p. 22)**

Specified name for the backup.

Type: String


Required: Yes

**TableName (p. 22)**

The name of the table.
Type: String


Pattern: \[a-zA-Z0-9_.-]+\]

Required: Yes

Response Syntax

```
{
  "BackupDetails": {
    "BackupArn": "string",
    "BackupCreationDateTime": number,
    "BackupName": "string",
    "BackupSizeBytes": number,
    "BackupStatus": "string"
  }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**BackupDetails (p. 23)**

Contains the details of the backup created for the table.

Type: BackupDetails (p. 241) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

**BackupInUseException**

There is another ongoing conflicting backup control plane operation on the table. The backups is either being created, deleted or restored to a table.

HTTP Status Code: 400

**ContinuousBackupsUnavailableException**

Backups have not yet been enabled for this table.

HTTP Status Code: 400

**InternalServerError**

An error occurred on the server side.

HTTP Status Code: 500

**LimitExceededException**

Up to 50 CreateBackup operations are allowed per second, per account. There is no limit to the number of daily on-demand backups that can be taken.
Up to 10 simultaneous table operations are allowed per account. These operations include `CreateTable`, `UpdateTable`, `DeleteTable`, `UpdateTimeToLive`, and `RestoreTableFromBackup`.

For tables with secondary indexes, only one of those tables can be in the `CREATING` state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the `ACTIVE` state is 250.

For tables with secondary indexes, only one of those tables can be in the `CREATING` state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the `ACTIVE` state is 250.

HTTP Status Code: 400

`TableInUseException`

A table by that name is either being created or deleted.

HTTP Status Code: 400

`TableNotFoundException`

A table with the name `TableName` does not currently exist within the subscriber's account.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateGlobalTable
Service: Amazon DynamoDB

Creates a global table from an existing table. A global table creates a replication relationship between two or more DynamoDB tables with the same table name in the provided regions.

Tables can only be added as the replicas of a global table group under the following conditions:

- The tables must have the same name.
- The tables must contain no items.
- The tables must have the same hash key and sort key (if present).
- The tables must have DynamoDB Streams enabled (NEW_AND_OLD_IMAGES).

Request Syntax

```
{
  "GlobalTableName": "string",
  "ReplicationGroup": [
    {
      "RegionName": "string"
    }
  ]
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**GlobalTableName (p. 25)**

The global table name.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: Yes

**ReplicationGroup (p. 25)**

The regions where the global table needs to be created.

Type: Array of Replica (p. 289) objects

Required: Yes

Response Syntax

```
{
  "GlobalTableDescription": {
    "CreationDateTime": number,
    "GlobalTableArn": "string",
    "GlobalTableName": "string",
    "ReplicationGroup": [
      {
        "RegionName": "string"
      }
    ]
  }
}
```
CreateGlobalTable

```json

"GlobalTableName": "string",
"GlobalTableStatus": "string",
"ReplicationGroup": [  
  {  
    "RegionName": "string"
  }  
]

```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**GlobalTableDescription (p. 25)**

Contains the details of the global table.

Type: `GlobalTableDescription (p. 271)` object

Errors

For information about the errors that are common to all actions, see [Common Errors (p. 341)](#).

**GlobalTableAlreadyExistsException**

The specified global table already exists.

HTTP Status Code: 400

**InternalServerException**

An error occurred on the server side.

HTTP Status Code: 500

**LimitExceededException**

Up to 50 `CreateBackup` operations are allowed per second, per account. There is no limit to the number of daily on-demand backups that can be taken.

Up to 10 simultaneous table operations are allowed per account. These operations include `CreateTable`, `UpdateTable`, `DeleteTable`, `UpdateTimeToLive`, and `RestoreTableFromBackup`.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

HTTP Status Code: 400

**TableNotFoundException**

A table with the name `TableName` does not currently exist within the subscriber's account.
HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateTable
Service: Amazon DynamoDB

The CreateTable operation adds a new table to your account. In an AWS account, table names must be unique within each region. That is, you can have two tables with same name if you create the tables in different regions.

CreateTable is an asynchronous operation. Upon receiving a CreateTable request, DynamoDB immediately returns a response with a TableStatus of CREATING. After the table is created, DynamoDB sets the TableStatus to ACTIVE. You can perform read and write operations only on an ACTIVE table.

You can optionally define secondary indexes on the new table, as part of the CreateTable operation. If you want to create multiple tables with secondary indexes on them, you must create the tables sequentially. Only one table with secondary indexes can be in the CREATING state at any given time.

You can use the DescribeTable action to check the table status.

Request Syntax

```json
{
  "AttributeDefinitions": [
    {
      "AttributeName": "string",
      "AttributeType": "string"
    }
  ],
  "GlobalSecondaryIndexes": [
    {
      "IndexName": "string",
      "KeySchema": [
        {
          "AttributeName": "string",
          "KeyType": "string"
        }
      ],
      "Projection": {
        "NonKeyAttributes": [ "string" ],
        "ProjectionType": "string"
      },
      "ProvisionedThroughput": {
        "ReadCapacityUnits": number,
        "WriteCapacityUnits": number
      }
    }
  ],
  "KeySchema": [
    {
      "AttributeName": "string",
      "KeyType": "string"
    }
  ],
  "LocalSecondaryIndexes": [
    {
      "IndexName": "string",
      "KeySchema": [
        {
          "AttributeName": "string",
          "KeyType": "string"
        }
      ],
      "Projection": {
```
CreateTable

```

"NonKeyAttributes": [ "string" ],
"ProjectionType": "string"
}

"ProvisionedThroughput": {
  "ReadCapacityUnits": number,
  "WriteCapacityUnits": number
},
"StreamSpecification": {
  "StreamEnabled": boolean,
  "StreamViewType": "string"
},
"TableName": "string"
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**AttributeDefinitions (p. 28)**

An array of attributes that describe the key schema for the table and indexes.

Type: Array of AttributeDefinition (p. 234) objects

Required: Yes

**KeySchema (p. 28)**

Specifies the attributes that make up the primary key for a table or an index. The attributes in KeySchema must also be defined in the AttributeDefinitions array. For more information, see Data Model in the Amazon DynamoDB Developer Guide.

Each KeySchemaElement in the array is composed of:
- **AttributeName** - The name of this key attribute.
- **KeyType** - The role that the key attribute will assume:
  - **HASH** - partition key
  - **RANGE** - sort key

**Note**

The partition key of an item is also known as its hash attribute. The term "hash attribute" derives from DynamoDB's usage of an internal hash function to evenly distribute data items across partitions, based on their partition key values.

The sort key of an item is also known as its range attribute. The term "range attribute" derives from the way DynamoDB stores items with the same partition key physically close together, in sorted order by the sort key value.

For a simple primary key (partition key), you must provide exactly one element with a KeyType of HASH.

For a composite primary key (partition key and sort key), you must provide exactly two elements, in this order: The first element must have a KeyType of HASH, and the second element must have a KeyType of RANGE.

For more information, see Specifying the Primary Key in the Amazon DynamoDB Developer Guide.

Type: Array of KeySchemaElement (p. 276) objects
Array Members: Minimum number of 1 item. Maximum number of 2 items.

Required: Yes

ProvisionedThroughput (p. 28)

Represents the provisioned throughput settings for a specified table or index. The settings can be modified using the UpdateTable operation.

For current minimum and maximum provisioned throughput values, see Limits in the Amazon DynamoDB Developer Guide.

Type: ProvisionedThroughput (p. 285) object

Required: Yes

TableName (p. 28)

The name of the table to create.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: Yes

GlobalSecondaryIndexes (p. 28)

One or more global secondary indexes (the maximum is five) to be created on the table. Each global secondary index in the array includes the following:

- IndexName - The name of the global secondary index. Must be unique only for this table.
- KeySchema - Specifies the key schema for the global secondary index.
- Projection - Specifies attributes that are copied (projected) from the table into the index. These are in addition to the primary key attributes and index key attributes, which are automatically projected. Each attribute specification is composed of:
  - ProjectionType - One of the following:
    - KEYS_ONLY - Only the index and primary keys are projected into the index.
    - INCLUDE - Only the specified table attributes are projected into the index. The list of projected attributes are in NonKeyAttributes.
    - ALL - All of the table attributes are projected into the index.
  - NonKeyAttributes - A list of one or more non-key attribute names that are projected into the secondary index. The total count of attributes provided in NonKeyAttributes, summed across all of the secondary indexes, must not exceed 20. If you project the same attribute into two different indexes, this counts as two distinct attributes when determining the total.
- ProvisionedThroughput - The provisioned throughput settings for the global secondary index, consisting of read and write capacity units.

Type: Array of GlobalSecondaryIndex (p. 262) objects

Required: No

LocalSecondaryIndexes (p. 28)

One or more local secondary indexes (the maximum is five) to be created on the table. Each index is scoped to a given partition key value. There is a 10 GB size limit per partition key value; otherwise, the size of a local secondary index is unconstrained.

Each local secondary index in the array includes the following:

- IndexName - The name of the local secondary index. Must be unique only for this table.
- **KeySchema** - Specifies the key schema for the local secondary index. The key schema must begin with the same partition key as the table.
- **Projection** - Specifies attributes that are copied (projected) from the table into the index. These are in addition to the primary key attributes and index key attributes, which are automatically projected. Each attribute specification is composed of:
  - **ProjectionType** - One of the following:
    - `KEYS_ONLY` - Only the index and primary keys are projected into the index.
    - `INCLUDE` - Only the specified table attributes are projected into the index. The list of projected attributes are in `NonKeyAttributes`.
    - `ALL` - All of the table attributes are projected into the index.
  - **NonKeyAttributes** - A list of one or more non-key attribute names that are projected into the secondary index. The total count of attributes provided in `NonKeyAttributes`, summed across all of the secondary indexes, must not exceed 20. If you project the same attribute into two different indexes, this counts as two distinct attributes when determining the total.

  Type: Array of `LocalSecondaryIndex (p. 278)` objects

  Required: No

**StreamSpecification (p. 28)**

The settings for DynamoDB Streams on the table. These settings consist of:
- **StreamEnabled** - Indicates whether Streams is to be enabled (true) or disabled (false).
- **StreamViewType** - When an item in the table is modified, `StreamViewType` determines what information is written to the table's stream. Valid values for `StreamViewType` are:
  - `KEYS_ONLY` - Only the key attributes of the modified item are written to the stream.
  - `NEW_IMAGE` - The entire item, as it appears after it was modified, is written to the stream.
  - `OLD_IMAGE` - The entire item, as it appeared before it was modified, is written to the stream.
  - `NEW_AND_OLD_IMAGES` - Both the new and the old item images of the item are written to the stream.

  Type: `StreamSpecification (p. 296)` object

  Required: No

**Response Syntax**

```json
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "string",
                "AttributeType": "string"
            }
        ],
        "CreationDateTime": number,
        "GlobalSecondaryIndexes": [
            {
                "Backfilling": boolean,
                "IndexArn": "string",
                "IndexName": "string",
                "IndexSizeBytes": number,
                "IndexStatus": "string",
                "ItemCount": number,
                "KeySchema": [
```
"AttributeName": "string",
"KeyType": "string"
],
"Projection": {
  "NonKeyAttributes": [ "string" ],
  "ProjectionType": "string"
},
"ProvisionedThroughput": {
  "LastDecreaseDateTime": number,
  "LastIncreaseDateTime": number,
  "NumberOfDecreasesToday": number,
  "ReadCapacityUnits": number,
  "WriteCapacityUnits": number
}
],
"ItemCount": number,
"KeySchema": [
  {
    "AttributeName": "string",
    "KeyType": "string"
  }
],
"LatestStreamArn": "string",
"LatestStreamLabel": "string",
"LocalSecondaryIndexes": [
{
  "IndexArn": "string",
  "IndexName": "string",
  "IndexSizeBytes": number,
  "ItemCount": number,
  "KeySchema": [
    {
      "AttributeName": "string",
      "KeyType": "string"
    }
  ],
  "Projection": {
    "NonKeyAttributes": [ "string" ],
    "ProjectionType": "string"
  }
}
],
"ProvisionedThroughput": {
  "LastDecreaseDateTime": number,
  "LastIncreaseDateTime": number,
  "NumberOfDecreasesToday": number,
  "ReadCapacityUnits": number,
  "WriteCapacityUnits": number
},
"RestoreSummary": {
  "RestoreDateTime": number,
  "RestoreInProgress": boolean,
  "SourceBackupArn": "string",
  "SourceTableArn": "string"
},
"StreamSpecification": {
  "StreamEnabled": boolean,
  "StreamViewType": "string"
},
"TableArn": "string",
"TableId": "string",
"TableName": "string",
"TableSizeBytes": number,
"TableStatus": "string"
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**TableDescription (p. 31)**

Represents the properties of the table.

Type: TableDescription (p. 297) object

Errors

For information about the errors that are common to all actions, see [Common Errors (p. 341)](https://docs.aws.amazon.com/amazondynamodb/latest/APIReference/AmazonDynamoDB_API.html).

**InternalServerException**

An error occurred on the server side.

HTTP Status Code: 500

**LimitExceededException**

Up to 50 CreateBackup operations are allowed per second, per account. There is no limit to the number of daily on-demand backups that can be taken.

Up to 10 simultaneous table operations are allowed per account. These operations include CreateTable, UpdateTable, DeleteTable, UpdateTimeToLive, and RestoreTableFromBackup.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

HTTP Status Code: 400

**ResourceInUseException**

The operation conflicts with the resource’s availability. For example, you attempted to recreate an existing table, or tried to delete a table currently in the CREATING state.

HTTP Status Code: 400

Example

**Create a Table**

This example creates a table named Thread. The table primary key consists of ForumName (partition key) and Subject (sort key). A local secondary index is also created; its key consists of ForumName (partition key) and LastPostDateTime (sort key).
Sample Request

POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>, Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.CreateTable

{
    "AttributeName": [ 
    {
        "AttributeName": "ForumName",
        "AttributeType": "S"
    },
    {
        "AttributeName": "Subject",
        "AttributeType": "S"
    },
    {
        "AttributeName": "LastPostDateTime",
        "AttributeType": "S"
    }
    ],
    "TableName": "Thread",
    "KeySchema": [ 
    {
        "AttributeName": "ForumName",
        "KeyType": "HASH"
    },
    {
        "AttributeName": "Subject",
        "KeyType": "RANGE"
    }
    ],
    "LocalSecondaryIndexes": [ 
    {
        "IndexName": "LastPostIndex",
        "KeySchema": [ 
        {
            "AttributeName": "ForumName",
            "KeyType": "HASH"
        },
        {
            "AttributeName": "LastPostDateTime",
            "KeyType": "RANGE"
        }
        ],
        "Projection": {
            "ProjectionType": "KEYS_ONLY"
        }
    }
    ],
    "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 5
    }
}
Sample Response

HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>
{
    "TableDescription": {
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/Thread",
        "AttributeDefinitions": [
            {
                "AttributeName": "ForumName",
                "AttributeType": "S"
            },
            {
                "AttributeName": "LastPostDateTime",
                "AttributeType": "S"
            },
            {
                "AttributeName": "Subject",
                "AttributeType": "S"
            }
        ],
        "CreationDateTime": 1.36372808007E9,
        "ItemCount": 0,
        "KeySchema": [
            {
                "AttributeName": "ForumName",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "Subject",
                "KeyType": "RANGE"
            }
        ],
        "LocalSecondaryIndexes": [
            {
                "IndexArn": "arn:aws:dynamodb:us-west-2:123456789012:table/Thread/index/LastPostIndex",
                "IndexName": "LastPostIndex",
                "IndexSizeBytes": 0,
                "ItemCount": 0,
                "KeySchema": [
                    {
                        "AttributeName": "ForumName",
                        "KeyType": "HASH"
                    },
                    {
                        "AttributeName": "LastPostDateTime",
                        "KeyType": "RANGE"
                    }
                ],
                "Projection": {
                    "ProjectionType": "KEYS_ONLY"
                }
            }
        ],
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 5,
            "WriteCapacityUnits": 5
        }
    }
}
"TableName": "Thread",
"TableSizeBytes": 0,
"TableStatus": "CREATING"
}
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteBackup
Service: Amazon DynamoDB

Deletes an existing backup of a table.
You can call DeleteBackup at a maximum rate of 10 times per second.

Request Syntax

```json
{
    "BackupArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**BackupArn (p. 37)**

The ARN associated with the backup.

Type: String


Required: Yes

Response Syntax

```json
{
    "BackupDescription": {
        "BackupDetails": {
            "BackupArn": "string",
            "BackupCreationDateTime": number,
            "BackupName": "string",
            "BackupSizeBytes": number,
            "BackupStatus": "string"
        },
        "SourceTableDetails": {
            "ItemCount": number,
            "KeySchema": [
                {
                    "AttributeName": "string",
                    "KeyType": "string"
                }
            ],
            "ProvisionedThroughput": {
                "ReadCapacityUnits": number,
                "WriteCapacityUnits": number
            },
            "TableArn": "string",
            "TableCreationDateTime": number,
            "TableId": "string",
            "TableName": "string",
            "TableSizeBytes": number
        }
    }
}
```
"SourceTableFeatureDetails": {
  "GlobalSecondaryIndexes": [
    {
      "IndexName": "string",
      "KeySchema": [
        {
          "AttributeName": "string",
          "KeyType": "string"
        }
      ],
      "Projection": {
        "NonKeyAttributes": [ "string" ],
        "ProjectionType": "string"
      },
      "ProvisionedThroughput": {
        "ReadCapacityUnits": number,
        "WriteCapacityUnits": number
      }
    }
  ],
  "LocalSecondaryIndexes": [
    {
      "IndexName": "string",
      "KeySchema": [
        {
          "AttributeName": "string",
          "KeyType": "string"
        }
      ],
      "Projection": {
        "NonKeyAttributes": [ "string" ],
        "ProjectionType": "string"
      }
    }
  ],
  "StreamDescription": {
    "StreamEnabled": boolean,
    "StreamViewType": "string"
  },
  "TimeToLiveDescription": {
    "AttributeName": "string",
    "TimeToLiveStatus": "string"
  }
}

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

BackupDescription (p. 37)

Contains the description of the backup created for the table.

Type: BackupDescription (p. 240) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).
BackupInUseException

There is another ongoing conflicting backup control plane operation on the table. The backups is either being created, deleted or restored to a table.

HTTP Status Code: 400

BackupNotFoundException

Backup not found for the given BackupARN.

HTTP Status Code: 400

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

LimitExceeded Exception

Up to 50 CreateBackup operations are allowed per second, per account. There is no limit to the number of daily on-demand backups that can be taken.

Up to 10 simultaneous table operations are allowed per account. These operations include CreateTable, UpdateTable, DeleteTable, UpdateTimeToLive, and RestoreTableFromBackup.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteItem

Service: Amazon DynamoDB

Deletes a single item in a table by primary key. You can perform a conditional delete operation that deletes the item if it exists, or if it has an expected attribute value.

In addition to deleting an item, you can also return the item's attribute values in the same operation, using the ReturnValues parameter.

Unless you specify conditions, the DeleteItem is an idempotent operation; running it multiple times on the same item or attribute does not result in an error response.

Conditional deletes are useful for deleting items only if specific conditions are met. If those conditions are met, DynamoDB performs the delete. Otherwise, the item is not deleted.

Request Syntax

```json
{
   "ConditionalOperator": "string",
   "ConditionExpression": "string",
   "Expected": {
      "string": {
         "AttributeValueList": [
            "B": blob,
            "BOOL": boolean,
            "BS": [ blob ],
            "L": [
               "AttributeValue"
            ],
            "M": {
               "string": "AttributeValue"
            },
            "N": "string",
            "NS": [ "string" ],
            "NULL": boolean,
            "S": "string",
            "SS": [ "string" ]
         }
      },
      "ComparisonOperator": "string",
      "Exists": boolean,
      "Value": {
         "B": blob,
         "BOOL": boolean,
         "BS": [ blob ],
         "L": [
            "AttributeValue"
         ],
         "M": {
            "string": "AttributeValue"
         },
         "N": "string",
         "NS": [ "string" ],
         "NULL": boolean,
         "S": "string",
         "SS": [ "string" ]
      }},
   "ExpressionAttributeNames": {
      "string": "string"
   }
}
```
Request Parameters

The request accepts the following data in JSON format.

Note
In the following list, the required parameters are described first.

Key (p. 40)

A map of attribute names to AttributeValue objects, representing the primary key of the item to delete.

For the primary key, you must provide all of the attributes. For example, with a simple primary key, you only need to provide a value for the partition key. For a composite primary key, you must provide values for both the partition key and the sort key.

Type: String to AttributeValue (p. 235) object map

Key Length Constraints: Maximum length of 65535.

Required: Yes
TableName (p. 40)

The name of the table from which to delete the item.

Type: String
Pattern: `[a-zA-Z0-9_.-]+`
Required: Yes

ConditionalOperator (p. 40)

This is a legacy parameter. Use ConditionExpression instead. For more information, see ConditionalOperator in the Amazon DynamoDB Developer Guide.

Type: String
Valid Values: AND | OR
Required: No

ConditionExpression (p. 40)

A condition that must be satisfied in order for a conditional DeleteItem to succeed.

An expression can contain any of the following:
- Functions: attribute_exists | attribute_not_exists | attribute_type | contains | begins_with | size
  
  These function names are case-sensitive.
- Comparison operators: = | <> | < | > | <= | >= | BETWEEN | IN
- Logical operators: AND | OR | NOT

For more information on condition expressions, see Specifying Conditions in the Amazon DynamoDB Developer Guide.

Type: String
Required: No

Expected (p. 40)

This is a legacy parameter. Use ConditionExpression instead. For more information, see Expected in the Amazon DynamoDB Developer Guide.

Type: String to ExpectedAttributeValue (p. 258) object map

Key Length Constraints: Maximum length of 65535.

Required: No

ExpressionAttributeNames (p. 40)

One or more substitution tokens for attribute names in an expression. The following are some use cases for using ExpressionAttributeNames:
- To access an attribute whose name conflicts with a DynamoDB reserved word.
- To create a placeholder for repeating occurrences of an attribute name in an expression.
- To prevent special characters in an attribute name from being misinterpreted in an expression.
Use the # character in an expression to dereference an attribute name. For example, consider the following attribute name:

- Percentile

The name of this attribute conflicts with a reserved word, so it cannot be used directly in an expression. (For the complete list of reserved words, see Reserved Words in the Amazon DynamoDB Developer Guide). To work around this, you could specify the following for ExpressionAttributeNames:

- {"#P":"Percentile"}

You could then use this substitution in an expression, as in this example:

- #P = :val

**Note**
Tokens that begin with the : character are expression attribute values, which are placeholders for the actual value at runtime.

For more information on expression attribute names, see Accessing Item Attributes in the Amazon DynamoDB Developer Guide.

**Type:** String to string map

**Value Length Constraints:** Maximum length of 65535.

**Required:** No

**ExpressionAttributeValues (p. 40)**

One or more values that can be substituted in an expression.

Use the : (colon) character in an expression to dereference an attribute value. For example, suppose that you wanted to check whether the value of the ProductStatus attribute was one of the following:

- Available
- Backordered
- Discontinued

You would first need to specify ExpressionAttributeValues as follows:

```
{ "avail":{"S":"Available"}, "back":{"S":"Backordered"}, "disc":
{"S":"Discontinued"} }
```

You could then use these values in an expression, such as this:

```
ProductStatus IN (:avail, :back, :disc)
```

For more information on expression attribute values, see Specifying Conditions in the Amazon DynamoDB Developer Guide.

**Type:** String to AttributeValue (p. 235) object map

**Required:** No

**ReturnConsumedCapacity (p. 40)**

Determines the level of detail about provisioned throughput consumption that is returned in the response:

- INDEXES - The response includes the aggregate ConsumedCapacity for the operation, together with ConsumedCapacity for each table and secondary index that was accessed.

Note that some operations, such as GetItem and BatchGetItem, do not access any indexes at all. In these cases, specifying INDEXES will only return ConsumedCapacity information for table(s).
DeleteItem

- TOTAL - The response includes only the aggregate ConsumedCapacity for the operation.
- NONE - No ConsumedCapacity details are included in the response.

Type: String

Valid Values: INDEXES | TOTAL | NONE

Required: No

ReturnItemCollectionMetrics (p. 40)

Determines whether item collection metrics are returned. If set to SIZE, the response includes statistics about item collections, if any, that were modified during the operation are returned in the response. If set to NONE (the default), no statistics are returned.

Type: String

Valid Values: SIZE | NONE

Required: No

ReturnValues (p. 40)

Use ReturnValues if you want to get the item attributes as they appeared before they were deleted. For DeleteItem, the valid values are:

- NONE - If ReturnValues is not specified, or if its value is NONE, then nothing is returned. (This setting is the default for ReturnValues.)
- ALL_OLD - The content of the old item is returned.

Note

The ReturnValues parameter is used by several DynamoDB operations; however, DeleteItem does not recognize any values other than NONE or ALL_OLD.

Type: String

Valid Values: NONE | ALL_OLD | UPDATED_OLD | ALL_NEW | UPDATED_NEW

Required: No

Response Syntax

```json
{
  "Attributes": {
    "string": {
      "B": blob,
      "BOOL": boolean,
      "BS": [ blob ],
      "L": [ "AttributeValue"
      ],
      "M": {
        "string": "AttributeValue"
      },
      "N": "string",
      "NS": [ "string" ],
      "NULL": boolean,
      "S": "string",
      "SS": [ "string" ]
    }
  }
}
```
"ConsumedCapacity": {  
  "CapacityUnits": number,  
  "GlobalSecondaryIndexes": {   
    "string": {  
      "CapacityUnits": number  
    }  
  },  
  "LocalSecondaryIndexes": {   
    "string": {  
      "CapacityUnits": number  
    }  
  },  
  "Table": {  
    "CapacityUnits": number  
  },  
  "TableName": "string"  
},  
"ItemCollectionMetrics": {  
  "ItemCollectionKey": {  
    "B": blob,  
    "BOOL": boolean,  
    "BS": [ blob ],  
    "L": [  
      "AttributeValue"  
    ],  
    "M": {  
      "string": "AttributeValue"  
    },  
    "N": "string",  
    "NS": [ "string" ],  
    "NULL": boolean,  
    "S": "string",  
    "SS": [ "string" ]  
  },  
  "SizeEstimateRangeGB": [ number ]  
}  

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Attributes (p. 44)

A map of attribute names to AttributeValue objects, representing the item as it appeared before the DeleteItem operation. This map appears in the response only if ReturnValues was specified as ALL_OLD in the request.

Type: String to AttributeValue (p. 235) object map

Key Length Constraints: Maximum length of 65535.

ConsumedCapacity (p. 44)

The capacity units consumed by the DeleteItem operation. The data returned includes the total provisioned throughput consumed, along with statistics for the table and any indexes involved in the operation. ConsumedCapacity is only returned if the ReturnConsumedCapacity parameter was specified. For more information, see Provisioned Throughput in the Amazon DynamoDB Developer Guide.

45
Type: ConsumedCapacity (p. 249) object

ItemCollectionMetrics (p. 44)

Information about item collections, if any, that were affected by the DeleteItem operation. ItemCollectionMetrics is only returned if the ReturnItemCollectionMetrics parameter was specified. If the table does not have any local secondary indexes, this information is not returned in the response.

Each ItemCollectionMetrics element consists of:

- ItemCollectionKey - The partition key value of the item collection. This is the same as the partition key value of the item itself.
- SizeEstimateRangeGB - An estimate of item collection size, in gigabytes. This value is a two-element array containing a lower bound and an upper bound for the estimate. The estimate includes the size of all the items in the table, plus the size of all attributes projected into all of the local secondary indexes on that table. Use this estimate to measure whether a local secondary index is approaching its size limit.

The estimate is subject to change over time; therefore, do not rely on the precision or accuracy of the estimate.

Type: ItemCollectionMetrics (p. 273) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

ConditionalCheckFailedException

A condition specified in the operation could not be evaluated.

HTTP Status Code: 400

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

ItemCollectionSizeLimitExceededException

An item collection is too large. This exception is only returned for tables that have one or more local secondary indexes.

HTTP Status Code: 400

ProvisionedThroughputExceededException

Your request rate is too high. The AWS SDKs for DynamoDB automatically retry requests that receive this exception. Your request is eventually successful, unless your retry queue is too large to finish. Reduce the frequency of requests and use exponential backoff. For more information, go to Error Retries and Exponential Backoff in the Amazon DynamoDB Developer Guide.

HTTP Status Code: 400

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400
Example

Delete an Item

The following example deletes an item from the Thread table, but only if that item does not already have an attribute named `Replies`. Because `ReturnValues` is set to `ALL_OLD`, the response contains the item as it appeared before the delete.

Sample Request

```plaintext
POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.DeleteItem
{
    "TableName": "Thread",
    "Key": {
        "ForumName": {
            "S": "Amazon DynamoDB"
        },
        "Subject": {
            "S": "How do I update multiple items?"
        }
    },
    "ConditionExpression": "attribute_not_exists(Replies)",
    "ReturnValues": "ALL_OLD"
}
```

Sample Response

```plaintext
HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>
{
    "Attributes": {
        "LastPostedBy": {
            "S": "fred@example.com"
        },
        "ForumName": {
            "S": "Amazon DynamoDB"
        },
        "LastPostDateTime": {
            "S": "201303201023"
        },
        "Tags": {
            "SS": ["Update","Multiple Items","HelpMe"]
        },
        "Subject": {
            "S": "How do I update multiple items?"
        }
    }
}
```
"Message": {
  "S": "I want to update multiple items in a single call. What's the best way to do that?"
}
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteTable
Service: Amazon DynamoDB

The DeleteTable operation deletes a table and all of its items. After a DeleteTable request, the specified table is in the DELETING state until DynamoDB completes the deletion. If the table is in the ACTIVE state, you can delete it. If a table is in CREATING or UPDATING states, then DynamoDB returns a ResourceInUseException. If the specified table does not exist, DynamoDB returns a ResourceNotFoundException. If table is already in the DELETING state, no error is returned.

Note
DynamoDB might continue to accept data read and write operations, such as GetItem and PutItem, on a table in the DELETING state until the table deletion is complete.

When you delete a table, any indexes on that table are also deleted.

If you have DynamoDB Streams enabled on the table, then the corresponding stream on that table goes into the DISABLED state, and the stream is automatically deleted after 24 hours.

Use the DescribeTable action to check the status of the table.

Request Syntax

```json
{
  "TableName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Note
In the following list, the required parameters are described first.

**TableName** (p. 49)

The name of the table to delete.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: Yes

Response Syntax

```json
{
  "TableDescription": {
    "AttributeDefinitions": [
    {
      "AttributeName": "string",
      "AttributeType": "string"
    }
    ],
    "CreationDateTime": number,
    "GlobalSecondaryIndexes": [
```
{ "Backfilling": boolean,  
"IndexArn": "string",  
"IndexName": "string",  
"IndexSizeBytes": number,  
"IndexStatus": "string",  
"ItemCount": number,  
"KeySchema": [  
  { "AttributeName": "string",  
    "KeyType": "string"  
  },  
  "Projection": {  
    "NonKeyAttributes": [ "string" ],  
    "ProjectionType": "string"  
  },  
  "ProvisionedThroughput": {  
    "LastDecreaseDateTime": number,  
    "LastIncreaseDateTime": number,  
    "NumberOfDecreasesToday": number,  
    "ReadCapacityUnits": number,  
    "WriteCapacityUnits": number  
  }  
],  
"ItemCount": number,  
"KeySchema": [  
  { "AttributeName": "string",  
    "KeyType": "string"  
  },  
  "LatestStreamArn": "string",  
  "LatestStreamLabel": "string",  
  "LocalSecondaryIndexes": [  
    { "IndexArn": "string",  
      "IndexName": "string",  
      "IndexSizeBytes": number,  
      "ItemCount": number,  
      "KeySchema": [  
        { "AttributeName": "string",  
          "KeyType": "string"  
        },  
        "Projection": {  
          "NonKeyAttributes": [ "string" ],  
          "ProjectionType": "string"  
        }  
      ],  
      "ProvisionedThroughput": {  
        "LastDecreaseDateTime": number,  
        "LastIncreaseDateTime": number,  
        "NumberOfDecreasesToday": number,  
        "ReadCapacityUnits": number,  
        "WriteCapacityUnits": number  
      },  
      "RestoreSummary": {  
        "RestoreDateTime": number,  
        "RestoreInProgress": boolean,  
        "SourceBackupArn": "string",  
        "SourceTableArn": "string"  
      }  
    }]  
}  
}
"StreamSpecification": {
    "StreamEnabled": boolean,
    "StreamViewType": "string"
},
"TableArn": "string",
"TableId": "string",
"TableName": "string",
"TableSizeBytes": number,
"TableStatus": "string"}

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

TableDescription (p. 49)

Represents the properties of a table.

Type: TableDescription (p. 297) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

LimitExceededException

Up to 50 CreateBackup operations are allowed per second, per account. There is no limit to the number of daily on-demand backups that can be taken.

Up to 10 simultaneous table operations are allowed per account. These operations include CreateTable, UpdateTable, DeleteTable, UpdateTimeToLive, and RestoreTableFromBackup.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

HTTP Status Code: 400

ResourceInUseException

The operation conflicts with the resource's availability. For example, you attempted to recreate an existing table, or tried to delete a table currently in the CREATING state.

HTTP Status Code: 400
ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

Example

Delete a Table

This example deletes the Reply table.

Sample Request

```plaintext
POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.DeleteTable

{
    "TableName": "Reply"
}
```

Sample Response

```plaintext
HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>

{
    "TableDescription": {
        "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/Reply",
        "ItemCount": 0,
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 5,
            "WriteCapacityUnits": 5
        },
        "TableName": "Reply",
        "TableSizeBytes": 0,
        "TableStatus": "DELETING"
    }
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:
• AWS Command Line Interface
• AWS SDK for .NET
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
DescribeBackup

Service: Amazon DynamoDB

Describes an existing backup of a table.

You can call DescribeBackup at a maximum rate of 10 times per second.

Request Syntax

```
{
  "BackupArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Note
In the following list, the required parameters are described first.

BackupArn (p. 54)

The ARN associated with the backup.

Type: String


Required: Yes

Response Syntax

```
{
  "BackupDescription": {
    "BackupDetails": {
      "BackupArn": "string",
      "BackupCreationDateTime": number,
      "BackupName": "string",
      "BackupSizeBytes": number,
      "BackupStatus": "string"
    },
    "SourceTableDetails": {
      "ItemCount": number,
      "KeySchema": [
        {
          "AttributeName": "string",
          "KeyType": "string"
        }
      ],
      "ProvisionedThroughput": {
        "ReadCapacityUnits": number,
        "WriteCapacityUnits": number
      },
      "TableArn": "string",
      "TableCreationDateTime": number,
      "TableId": "string",
      "TableName": "string",
      "TableSizeBytes": number
    }
  }
}
```
"SourceTableFeatureDetails": {
  "GlobalSecondaryIndexes": [
    {
      "IndexName": "string",
      "KeySchema": [
        {
          "AttributeName": "string",
          "KeyType": "string"
        }
      ],
      "Projection": {
        "NonKeyAttributes": [ "string" ],
        "ProjectionType": "string"
      },
      "ProvisionedThroughput": {
        "ReadCapacityUnits": number,
        "WriteCapacityUnits": number
      }
    }
  ],
  "LocalSecondaryIndexes": [
    {
      "IndexName": "string",
      "KeySchema": [
        {
          "AttributeName": "string",
          "KeyType": "string"
        }
      ],
      "Projection": {
        "NonKeyAttributes": [ "string" ],
        "ProjectionType": "string"
      }
    }
  ],
  "StreamDescription": {
    "StreamEnabled": boolean,
    "StreamViewType": "string"
  },
  "TimeToLiveDescription": {
    "AttributeName": "string",
    "TimeToLiveStatus": "string"
  }
}

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

BackupDescription (p. 54)

Contains the description of the backup created for the table.

Type: BackupDescription (p. 240) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).
BackupNotFoundException

Backup not found for the given BackupARN.

HTTP Status Code: 400

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeContinuousBackups

Service: Amazon DynamoDB

Checks the status of the backup restore settings on the specified table. If backups are enabled, ContinuousBackupsStatus will be set to ENABLED.

You can call DescribeContinuousBackups at a maximum rate of 10 times per second.

Request Syntax

```
{
   "TableName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**

In the following list, the required parameters are described first.

**TableName (p. 57)**

Name of the table for which the customer wants to check the backup and restore settings.

Type: String


Pattern: `[a-zA-Z0-9_.-]+`

Required: Yes

Response Syntax

```
{
   "ContinuousBackupsDescription": {
      "ContinuousBackupsStatus": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**ContinuousBackupsDescription (p. 57)**

ContinuousBackupsDescription can be one of the following: ENABLED, DISABLED.

Type: ContinuousBackupsDescription (p. 251) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).
InternalServerError
An error occurred on the server side.
HTTP Status Code: 500

TableNotFoundException
A table with the name TableName does not currently exist within the subscriber's account.
HTTP Status Code: 400

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeGlobalTable
Service: Amazon DynamoDB
Returns information about the specified global table.

Request Syntax

```
{
   "GlobalTableName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**GlobalTableName (p. 59)**
The name of the global table.
Type: String
Pattern: [a-zA-Z0-9_.-]+
Required: Yes

Response Syntax

```
{
   "GlobalTableDescription": {
      "CreationDateTime": number,
      "GlobalTableArn": "string",
      "GlobalTableName": "string",
      "GlobalTableStatus": "string",
      "ReplicationGroup": [
         {
            "RegionName": "string"
         }
      ]
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.
The following data is returned in JSON format by the service.

**GlobalTableDescription (p. 59)**
Contains the details of the global table.
Type: GlobalTableDescription (p. 271) object
Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

GlobalTableNotFoundException

The specified global table does not exist.

HTTP Status Code: 400

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeLimits

Service: Amazon DynamoDB

Returns the current provisioned-capacity limits for your AWS account in a region, both for the region as a whole and for any one DynamoDB table that you create there.

When you establish an AWS account, the account has initial limits on the maximum read capacity units and write capacity units that you can provision across all of your DynamoDB tables in a given region. Also, there are per-table limits that apply when you create a table there. For more information, see Limits page in the Amazon DynamoDB Developer Guide.

Although you can increase these limits by filing a case at AWS Support Center, obtaining the increase is not instantaneous. The DescribeLimits action lets you write code to compare the capacity you are currently using to those limits imposed by your account so that you have enough time to apply for an increase before you hit a limit.

For example, you could use one of the AWS SDKs to do the following:

1. Call DescribeLimits for a particular region to obtain your current account limits on provisioned capacity there.
2. Create a variable to hold the aggregate read capacity units provisioned for all your tables in that region, and one to hold the aggregate write capacity units. Zero them both.
3. Call ListTables to obtain a list of all your DynamoDB tables.
4. For each table name listed by ListTables, do the following:
   • Call DescribeTable with the table name.
   • Use the data returned by DescribeTable to add the read capacity units and write capacity units provisioned for the table itself to your variables.
   • If the table has one or more global secondary indexes (GSIs), loop over these GSIs and add their provisioned capacity values to your variables as well.
5. Report the account limits for that region returned by DescribeLimits, along with the total current provisioned capacity levels you have calculated.

This will let you see whether you are getting close to your account-level limits.

The per-table limits apply only when you are creating a new table. They restrict the sum of the provisioned capacity of the new table itself and all its global secondary indexes.

For existing tables and their GSIs, DynamoDB will not let you increase provisioned capacity extremely rapidly, but the only upper limit that applies is that the aggregate provisioned capacity over all your tables and GSIs cannot exceed either of the per-account limits.

**Note**

DescribeLimits should only be called periodically. You can expect throttling errors if you call it more than once in a minute.

The DescribeLimits Request element has no content.

**Response Syntax**

```json
{
   "AccountMaxReadCapacityUnits": number,
   "AccountMaxWriteCapacityUnits": number,
   "TableMaxReadCapacityUnits": number,
   "TableMaxWriteCapacityUnits": number
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

AccountMaxReadCapacityUnits (p. 61)

The maximum total read capacity units that your account allows you to provision across all of your tables in this region.

Type: Long

Valid Range: Minimum value of 1.

AccountMaxWriteCapacityUnits (p. 61)

The maximum total write capacity units that your account allows you to provision across all of your tables in this region.

Type: Long

Valid Range: Minimum value of 1.

TableMaxReadCapacityUnits (p. 61)

The maximum read capacity units that your account allows you to provision for a new table that you are creating in this region, including the read capacity units provisioned for its global secondary indexes (GSIs).

Type: Long

Valid Range: Minimum value of 1.

TableMaxWriteCapacityUnits (p. 61)

The maximum write capacity units that your account allows you to provision for a new table that you are creating in this region, including the write capacity units provisioned for its global secondary indexes (GSIs).

Type: Long

Valid Range: Minimum value of 1.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

Example

DescribeLimits

Sample Request
POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.DescribeLimits
{
}

Sample Response

HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Date: <Date>
{
  "AccountMaxReadCapacityUnits": 20000,
  "AccountMaxWriteCapacityUnits": 20000,
  "TableMaxReadCapacityUnits": 10000,
  "TableMaxWriteCapacityUnits": 10000
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeTable
Service: Amazon DynamoDB

Returns information about the table, including the current status of the table, when it was created, the primary key schema, and any indexes on the table.

**Note**
If you issue a DescribeTable request immediately after a CreateTable request, DynamoDB might return a ResourceNotFoundException. This is because DescribeTable uses an eventually consistent query, and the metadata for your table might not be available at that moment. Wait for a few seconds, and then try the DescribeTable request again.

**Request Syntax**

```json
{
  "TableName": "string"
}
```

**Request Parameters**

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**TableName (p. 64)**

The name of the table to describe.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: Yes

**Response Syntax**

```json
{
  "Table": {
    "AttributeDefinitions": [
      {
        "AttributeName": "string",
        "AttributeType": "string"
      }
    ],
    "CreationDateTime": number,
    "GlobalSecondaryIndexes": [
      {
        "Backfilling": boolean,
        "IndexArn": "string",
        "IndexName": "string",
        "IndexSizeBytes": number,
        "IndexStatus": "string",
        "ItemCount": number,
        "KeySchema": [
          {
            "AttributeName": "string",
            "AttributeType": "string"
          }
        ]
      }
    ]
  }
}
```
Amazon DynamoDB API Reference

DescribeTable

"KeyType": "string"
",
"Projection": {
  "NonKeyAttributes": [ "string" ],
  "ProjectionType": "string"
},
"ProvisionedThroughput": {
  "LastDecreaseDateTime": number,
  "LastIncreaseDateTime": number,
  "NumberOfDecreasesToday": number,
  "ReadCapacityUnits": number,
  "WriteCapacityUnits": number
}
",
"ItemCount": number,
"KeySchema": [ {
  "AttributeName": "string",
  "KeyType": "string"
} ],
"LatestStreamArn": "string",
"LatestStreamLabel": "string",
"LocalSecondaryIndexes": [ {
  "IndexArn": "string",
  "IndexName": "string",
  "IndexSizeBytes": number,
  "ItemCount": number,
  "KeySchema": [ {
    "AttributeName": "string",
    "KeyType": "string"
  } ],
  "Projection": { "NonKeyAttributes": [ "string" ],
    "ProjectionType": "string"
  }
} ],
"ProvisionedThroughput": {
  "LastDecreaseDateTime": number,
  "LastIncreaseDateTime": number,
  "NumberOfDecreasesToday": number,
  "ReadCapacityUnits": number,
  "WriteCapacityUnits": number
},
"RestoreSummary": { "RestoreDateTime": number,
  "RestoreInProgress": boolean,
  "SourceBackupArn": "string",
  "SourceTableArn": "string"
},
"StreamSpecification": { "StreamEnabled": boolean,
  "StreamViewType": "string"
},
"TableArn": "string",
"TableId": "string",
"TableName": "string",
"TableSizeBytes": number,
"TableStatus": "string"
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Table (p. 64)

The properties of the table.

Type: TableDescription (p. 297) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

Example

Describe a Table

This example describes the Thread table.

Sample Request

POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.DescribeTable
{
    "TableName":"Thread"
}

Sample Response
HTTP/1.1 200 OK
x-amzn-RequestId: <RequestID>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>
{
  "Table": {
    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/Thread",
    "AttributeDefinitions": [
      {
        "AttributeName": "ForumName",
        "AttributeType": "S"
      },
      {
        "AttributeName": "LastPostDateTime",
        "AttributeType": "S"
      },
      {
        "AttributeName": "Subject",
        "AttributeType": "S"
      }
    ],
    "CreationDateTime": 1.363729002358E9,
    "ItemCount": 0,
    "KeySchema": [
      {
        "AttributeName": "ForumName",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "Subject",
        "KeyType": "RANGE"
      }
    ],
    "LocalSecondaryIndexes": [
      {"IndexName": "LastPostIndex",
       "IndexSizeBytes": 0,
       "ItemCount": 0,
       "KeySchema": [
         {"AttributeName": "ForumName",
          "KeyType": "HASH"},
         {"AttributeName": "LastPostDateTime",
          "KeyType": "RANGE"}
       ],
       "Projection": {
         "ProjectionType": "KEYS_ONLY"
       }
      },
      "ProvisionedThroughput": {
        "NumberOfDecreasesToday": 0,
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 5
      },
      "TableName": "Thread",
      "TableSizeBytes": 0,
      "TableStatus": "ACTIVE"}
}
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeTimeToLive
Service: Amazon DynamoDB

Gives a description of the Time to Live (TTL) status on the specified table.

Request Syntax

```
{
    "TableName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**TableName (p. 69)**

The name of the table to be described.

Type: String


Pattern: [a-zA-Z0-9_-.]+

Required: Yes

Response Syntax

```
{
    "TimeToLiveDescription": {
        "AttributeName": "string",
        "TimeToLiveStatus": "string"
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**TimeToLiveDescription (p. 69)**

Type: TimeToLiveDescription (p. 303) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).
InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
**GetItem**

Service: Amazon DynamoDB

The GetItem operation returns a set of attributes for the item with the given primary key. If there is no matching item, GetItem does not return any data and there will be no Item element in the response.

GetItem provides an eventually consistent read by default. If your application requires a strongly consistent read, set ConsistentRead to true. Although a strongly consistent read might take more time than an eventually consistent read, it always returns the last updated value.

**Request Syntax**

```json
{
    "AttributesToGet": [ "string" ],
    "ConsistentRead": boolean,
    "ExpressionAttributeNames": {
        "string": "string"
    },
    "Key": {
        "string": {
            "B": blob,
            "BOOL": boolean,
            "BS": [ blob ],
            "L": [
                "AttributeValue"
            ],
            "M": {
                "string": "AttributeValue"
            },
            "N": "string",
            "NS": [ "string" ],
            "NULL": boolean,
            "S": "string",
            "SS": [ "string" ]
        }
    },
    "ProjectionExpression": "string",
    "ReturnConsumedCapacity": "string",
    "TableName": "string"
}
```

**Request Parameters**

The request accepts the following data in JSON format.

*Note*  
In the following list, the required parameters are described first.

**Key (p. 71)**

A map of attribute names to AttributeValue objects, representing the primary key of the item to retrieve.

For the primary key, you must provide all of the attributes. For example, with a simple primary key, you only need to provide a value for the partition key. For a composite primary key, you must provide values for both the partition key and the sort key.

Type: String to AttributeValue (p. 235) object map

Key Length Constraints: Maximum length of 65535.
REQUIRED: Yes

**TableName (p. 71)**

The name of the table containing the requested item.

**Type:** String

**Length Constraints:** Minimum length of 3. Maximum length of 255.

**Pattern:** [a-zA-Z0-9_.-]+  

**Required:** Yes

**AttributesToGet (p. 71)**

This is a legacy parameter. Use `ProjectionExpression` instead. For more information, see `AttributesToGet` in the *Amazon DynamoDB Developer Guide*.

**Type:** Array of strings

**Array Members:** Minimum number of 1 item.

**Length Constraints:** Maximum length of 65535.

**Required:** No

**ConsistentRead (p. 71)**

Determines the read consistency model: If set to `true`, then the operation uses strongly consistent reads; otherwise, the operation uses eventually consistent reads.

**Type:** Boolean

**Required:** No

**ExpressionAttributeNames (p. 71)**

One or more substitution tokens for attribute names in an expression. The following are some use cases for using `ExpressionAttributeNames`:

- To access an attribute whose name conflicts with a DynamoDB reserved word.
- To create a placeholder for repeating occurrences of an attribute name in an expression.
- To prevent special characters in an attribute name from being misinterpreted in an expression.

Use the `#` character in an expression to dereference an attribute name. For example, consider the following attribute name:

- Percentile

The name of this attribute conflicts with a reserved word, so it cannot be used directly in an expression. (For the complete list of reserved words, see `Reserved Words` in the *Amazon DynamoDB Developer Guide*). To work around this, you could specify the following for `ExpressionAttributeNames`:

- `{"#P":"Percentile"}`

You could then use this substitution in an expression, as in this example:

- `#P = :val`

**Note**  
Tokens that begin with the `:` character are *expression attribute values*, which are placeholders for the actual value at runtime.

For more information on expression attribute names, see `Accessing Item Attributes` in the *Amazon DynamoDB Developer Guide*. 

---

**72**
Type: String to string map

Value Length Constraints: Maximum length of 65535.

Required: No

**ProjectionExpression (p. 71)**

A string that identifies one or more attributes to retrieve from the table. These attributes can include scalars, sets, or elements of a JSON document. The attributes in the expression must be separated by commas.

If no attribute names are specified, then all attributes will be returned. If any of the requested attributes are not found, they will not appear in the result.

For more information, see *Accessing Item Attributes* in the *Amazon DynamoDB Developer Guide*.

Type: String

Required: No

**ReturnConsumedCapacity (p. 71)**

Determines the level of detail about provisioned throughput consumption that is returned in the response:

- **INDEXES** - The response includes the aggregate `ConsumedCapacity` for the operation, together with `ConsumedCapacity` for each table and secondary index that was accessed.

  Note that some operations, such as `GetItem` and `BatchGetItem`, do not access any indexes at all. In these cases, specifying `INDEXES` will only return `ConsumedCapacity` information for table(s).

- **TOTAL** - The response includes only the aggregate `ConsumedCapacity` for the operation.

- **NONE** - No `ConsumedCapacity` details are included in the response.

Type: String

Valid Values: INDEXES | TOTAL | NONE

Required: No

**Response Syntax**

```json
{
    "ConsumedCapacity": {
        "CapacityUnits": number,
        "GlobalSecondaryIndexes": {
            "string": {
                "CapacityUnits": number
            }
        },
        "LocalSecondaryIndexes": {
            "string": {
                "CapacityUnits": number
            }
        },
        "Table": {
            "CapacityUnits": number
        },
        "TableName": "string"
    },
    "Item": {
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**ConsumedCapacity (p. 73)**

The capacity units consumed by the GetItem operation. The data returned includes the total provisioned throughput consumed, along with statistics for the table and any indexes involved in the operation. ConsumedCapacity is only returned if the ReturnConsumedCapacity parameter was specified. For more information, see Provisioned Throughput in the Amazon DynamoDB Developer Guide.

Type: ConsumedCapacity (p. 249) object

**Item (p. 73)**

A map of attribute names to AttributeValue objects, as specified by ProjectionExpression.

Type: String to AttributeValue (p. 235) object map

Key Length Constraints: Maximum length of 65535.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

**InternalServer**

An error occurred on the server side.

HTTP Status Code: 500

**ProvisionedThroughputExceededException**

Your request rate is too high. The AWS SDKs for DynamoDB automatically retry requests that receive this exception. Your request is eventually successful, unless your retry queue is too large to finish. Reduce the frequency of requests and use exponential backoff. For more information, go to Error Retries and Exponential Backoff in the Amazon DynamoDB Developer Guide.

HTTP Status Code: 400
ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

Example

Retrieve Item Attributes

The following example retrieves three attributes from the Thread table. In the response, the ConsumedCapacityUnits value is 1, because ConsistentRead is set to true. If ConsistentRead had been set to false (or not specified) for the same request, an eventually consistent read would have been used and ConsumedCapacityUnits would have been 0.5.

Sample Request

POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.GetItem

{
  "TableName": "Thread",
  "Key": {
    "ForumName": {
      "S": "Amazon DynamoDB"
    },
    "Subject": {
      "S": "How do I update multiple items?"
    }
  },
  "ProjectionExpression":"LastPostDateTime, Message, Tags",
  "ConsistentRead": true,
  "ReturnConsumedCapacity": "TOTAL"
}

Sample Response

HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amzn-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>

{
  "ConsumedCapacity": {
    "CapacityUnits": 1,
    "TableName": "Thread"
  },
  "Item": {

"Tags": {  
  "SS": ["Update","Multiple Items","HelpMe"]
},  
"LastPostDateTime": {  
  "S": "201303190436"
},  
"Message": {  
  "S": "I want to update multiple items in a single call. What's the best way to do that?"
}  
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ListBackups

Service: Amazon DynamoDB

List backups associated with an AWS account. To list backups for a given table, specify TableName. ListBackups returns a paginated list of results with at most 1MB worth of items in a page. You can also specify a limit for the maximum number of entries to be returned in a page.

In the request, start time is inclusive but end time is exclusive. Note that these limits are for the time at which the original backup was requested.

You can call ListBackups a maximum of 5 times per second.

Request Syntax

```
{
   "ExclusiveStartBackupArn": "string",
   "Limit": number,
   "TableName": "string",
   "TimeRangeLowerBound": number,
   "TimeRangeUpperBound": number
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**

In the following list, the required parameters are described first.

**ExclusiveStartBackupArn (p. 77)**

LastEvaluatedBackupARN returned by the previous ListBackups call.

Type: String


Required: No

**Limit (p. 77)**

Maximum number of backups to return at once.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

**TableName (p. 77)**

The backups from the table specified by TableName are listed.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: No
TimeRangeLowerBound (p. 77)
   Only backups created after this time are listed. TimeRangeLowerBound is inclusive.
   Type: Timestamp
   Required: No
TimeRangeUpperBound (p. 77)
   Only backups created before this time are listed. TimeRangeUpperBound is exclusive.
   Type: Timestamp
   Required: No

Response Syntax

```
{  
   "BackupSummaries": [  
      {  
         "BackupArn": "string",  
         "BackupCreationDateTime": number,  
         "BackupName": "string",  
         "BackupSizeBytes": number,  
         "BackupStatus": "string",  
         "TableArn": "string",  
         "TableId": "string",  
         "TableName": "string"  
      }  
   ],  
   "LastEvaluatedBackupArn": "string"  
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

BackupSummaries (p. 78)
   List of BackupSummary objects.
   Type: Array of BackupSummary (p. 243) objects

LastEvaluatedBackupArn (p. 78)
   Last evaluated BackupARN.
   Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InternalServerError
   An error occurred on the server side.
HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ListGlobalTables
Service: Amazon DynamoDB
Lists all global tables that have a replica in the specified region.

Request Syntax

```
{
    "ExclusiveStartGlobalTableName": "string",
    "Limit": number,
    "RegionName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ExclusiveStartGlobalTableName (p. 80)**

The first global table name that this operation will evaluate.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: No

**Limit (p. 80)**

The maximum number of table names to return.

Type: Integer

Valid Range: Minimum value of 1.

Required: No

**RegionName (p. 80)**

Lists the global tables in a specific region.

Type: String

Required: No

Response Syntax

```
{
    "GlobalTables": [
        {
            "GlobalTableName": "string",
            "ReplicationGroup": [
                {
                    "RegionName": "string"
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

GlobalTables (p. 80)

List of global table names.

Type: Array of GlobalTable (p. 270) objects

LastEvaluatedGlobalTableName (p. 80)

Last evaluated global table name.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InternalServerErro

An error occurred on the server side.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ListTables
Service: Amazon DynamoDB

Returns an array of table names associated with the current account and endpoint. The output from ListTables is paginated, with each page returning a maximum of 100 table names.

Request Syntax

```json
{
  "ExclusiveStartTableName": "string",
  "Limit": number
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**

In the following list, the required parameters are described first.

**ExclusiveStartTableName (p. 82)**

The first table name that this operation will evaluate. Use the value that was returned for LastEvaluatedTableName in a previous operation, so that you can obtain the next page of results.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: No

**Limit (p. 82)**

A maximum number of table names to return. If this parameter is not specified, the limit is 100.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

Response Syntax

```json
{
  "LastEvaluatedTableName": "string",
  "TableNames": [ "string" ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.
**LastEvaluatedTableName (p. 82)**

The name of the last table in the current page of results. Use this value as the ExclusiveStartTableName in a new request to obtain the next page of results, until all the table names are returned.

If you do not receive a LastEvaluatedTableName value in the response, this means that there are no more table names to be retrieved.

Type: String


Pattern: [a-zA-Z0-9_.-]+

**TableNames (p. 82)**

The names of the tables associated with the current account at the current endpoint. The maximum size of this array is 100.

If LastEvaluatedTableName also appears in the output, you can use this value as the ExclusiveStartTableName parameter in a subsequent ListTables request and obtain the next page of results.

Type: Array of strings


Pattern: [a-zA-Z0-9_.-]+

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 341).

**InternalServerError**

An error occurred on the server side.

HTTP Status Code: 500

**Example**

**List Tables**

This example requests a list of tables, starting with a table named *Forum* and ending after three table names have been returned.

**Sample Request**

```
POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
```
X-Amz-Target: DynamoDB_20120810.ListTables

{
    "ExclusiveStartTableName": "Forum",
    "Limit": 3
}

Sample Response

HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>

{
    "LastEvaluatedTableName": "Thread",
    "TableNames": ["Forum","Reply","Thread"]
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ListTagsOfResource

Service: Amazon DynamoDB

List all tags on an Amazon DynamoDB resource. You can call ListTagsOfResource up to 10 times per second, per account.

For an overview on tagging DynamoDB resources, see Tagging for DynamoDB in the Amazon DynamoDB Developer Guide.

Request Syntax

```
{
   "NextToken": "string",
   "ResourceArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ResourceArn (p. 85)**

The Amazon DynamoDB resource with tags to be listed. This value is an Amazon Resource Name (ARN).

Type: String


Required: Yes

**NextToken (p. 85)**

An optional string that, if supplied, must be copied from the output of a previous call to ListTagOfResource. When provided in this manner, this API fetches the next page of results.

Type: String

Required: No

Response Syntax

```
{
   "NextToken": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.
The following data is returned in JSON format by the service.

**NextToken (p. 85)**

If this value is returned, there are additional results to be displayed. To retrieve them, call ListTagsOfResource again, with NextToken set to this value.

Type: String

**Tags (p. 85)**

The tags currently associated with the Amazon DynamoDB resource.

Type: Array of Tag (p. 302) objects

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 341).

**InternalServer**

An error occurred on the server side.

HTTP Status Code: 500

**ResourceNotFoundException**

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
PutItem

Service: Amazon DynamoDB

Creates a new item, or replaces an old item with a new item. If an item that has the same primary key as the new item already exists in the specified table, the new item completely replaces the existing item. You can perform a conditional put operation (add a new item if one with the specified primary key doesn't exist), or replace an existing item if it has certain attribute values. You can return the item's attribute values in the same operation, using the ReturnValues parameter.

Important
This topic provides general information about the PutItem API. For information on how to call the PutItem API using the AWS SDK in specific languages, see the following:

- PutItem in the AWS Command Line Interface
- PutItem in the AWS SDK for .NET
- PutItem in the AWS SDK for C++
- PutItem in the AWS SDK for Go
- PutItem in the AWS SDK for Java
- PutItem in the AWS SDK for JavaScript
- PutItem in the AWS SDK for PHP V3
- PutItem in the AWS SDK for Python
- PutItem in the AWS SDK for Ruby V2

When you add an item, the primary key attribute(s) are the only required attributes. Attribute values cannot be null. String and Binary type attributes must have lengths greater than zero. Set type attributes cannot be empty. Requests with empty values will be rejected with a ValidationException exception.

Note
To prevent a new item from replacing an existing item, use a conditional expression that contains the attribute_not_exists function with the name of the attribute being used as the partition key for the table. Since every record must contain that attribute, the attribute_not_exists function will only succeed if no matching item exists.

For more information about PutItem, see Working with Items in the Amazon DynamoDB Developer Guide.

Request Syntax

```
{
  "ConditionalOperator": "string",
  "ConditionExpression": "string",
  "Expected": {
    "string" : {
      "AttributeValueList": [
        {
          "S": blob,
          "BOOL": boolean,
          "BS": [ blob ],
          "L": [
            "AttributeValue",
          ],
          "M": {
            "string" : "AttributeValue"
          },
          "N": "string",
          "NS": [ "string" ]
        }
      ]
    }
  }
}
```
"NULL": boolean,
"S": "string",
"SS": [ "string" ]
}
],
"ComparisonOperator": "string",
"Exists": boolean,
"Value": {
  "B": blob,
  "BOOL": boolean,
  "BS": [ blob ],
  "L": [
    "AttributeValue"
  ],
  "M": {
    "string" : "AttributeValue"
  },
  "N": "string",
  "NS": [ "string" ],
  "NULL": boolean,
  "S": "string",
  "SS": [ "string" ]
}
}
,"ExpressionAttributeNames": {
  "string" : "string"
},
"ExpressionAttributeValues": {
  "string" : {
    "B": blob,
    "BOOL": boolean,
    "BS": [ blob ],
    "L": [
      "AttributeValue"
    ],
    "M": {
      "string" : "AttributeValue"
    },
    "N": "string",
    "NS": [ "string" ],
    "NULL": boolean,
    "S": "string",
    "SS": [ "string" ]
  }
},
"Item": {
  "string" : {
    "B": blob,
    "BOOL": boolean,
    "BS": [ blob ],
    "L": [
      "AttributeValue"
    ],
    "M": {
      "string" : "AttributeValue"
    },
    "N": "string",
    "NS": [ "string" ],
    "NULL": boolean,
    "S": "string",
    "SS": [ "string" ]
  }
},
"ReturnConsumedCapacity": "string",
"ReturnItemCollectionMetrics": "string"
"ReturnValues": "string",
"TableName": "string"
}

Request Parameters

The request accepts the following data in JSON format.

Note
In the following list, the required parameters are described first.

Item (p. 87)

A map of attribute name/value pairs, one for each attribute. Only the primary key attributes are required; you can optionally provide other attribute name-value pairs for the item.

You must provide all of the attributes for the primary key. For example, with a simple primary key, you only need to provide a value for the partition key. For a composite primary key, you must provide both values for both the partition key and the sort key.

If you specify any attributes that are part of an index key, then the data types for those attributes must match those of the schema in the table's attribute definition.

For more information about primary keys, see Primary Key in the Amazon DynamoDB Developer Guide.

Each element in the Item map is an AttributeValue object.

Type: String to AttributeValue (p. 235) object map

Key Length Constraints: Maximum length of 65535.

Required: Yes

TableName (p. 87)

The name of the table to contain the item.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: Yes

ConditionalOperator (p. 87)

This is a legacy parameter. Use ConditionExpression instead. For more information, see ConditionalOperator in the Amazon DynamoDB Developer Guide.

Type: String

Valid Values: AND | OR

Required: No

ConditionExpression (p. 87)

A condition that must be satisfied in order for a conditional PutItem operation to succeed.

An expression can contain any of the following:

- Functions: attribute_exists | attribute_not_exists | attribute_type | contains | begins_with | size
These function names are case-sensitive.

- Comparison operators: = | < > | < | > | <= | >= | BETWEEN | IN
- Logical operators: AND | OR | NOT

For more information on condition expressions, see Specifying Conditions in the Amazon DynamoDB Developer Guide.

Type: String

Required: No

**Expected (p. 87)**

This is a legacy parameter. Use ConditionExpression instead. For more information, see Expected in the Amazon DynamoDB Developer Guide.

Type: String to ExpectedAttributeValue (p. 258) object map

Key Length Constraints: Maximum length of 65535.

Required: No

**ExpressionAttributeNames (p. 87)**

One or more substitution tokens for attribute names in an expression. The following are some use cases for using ExpressionAttributeNames:

- To access an attribute whose name conflicts with a DynamoDB reserved word.
- To create a placeholder for repeating occurrences of an attribute name in an expression.
- To prevent special characters in an attribute name from being misinterpreted in an expression.

Use the # character in an expression to dereference an attribute name. For example, consider the following attribute name:

- **Percentile**

The name of this attribute conflicts with a reserved word, so it cannot be used directly in an expression. (For the complete list of reserved words, see Reserved Words in the Amazon DynamoDB Developer Guide). To work around this, you could specify the following for ExpressionAttributeNames:

- {"#P":"Percentile"}

You could then use this substitution in an expression, as in this example:

- **#P = :val**

  **Note**
  Tokens that begin with the : character are expression attribute values, which are placeholders for the actual value at runtime.

  For more information on expression attribute names, see Accessing Item Attributes in the Amazon DynamoDB Developer Guide.

Type: String to string map

Value Length Constraints: Maximum length of 65535.

Required: No

**ExpressionAttributeValues (p. 87)**

One or more values that can be substituted in an expression.
Use the : (colon) character in an expression to dereference an attribute value. For example, suppose that you wanted to check whether the value of the `ProductStatus` attribute was one of the following:

Available | Backordered | Discontinued

You would first need to specify `ExpressionAttributeValues` as follows:

```
{ "avail":{"S":"Available"}, "back":{"S":"Backordered"}, "disc":
{"S":"Discontinued"} }
```

You could then use these values in an expression, such as this:

```
ProductStatus IN (:avail, :back, :disc)
```

For more information on expression attribute values, see Specifying Conditions in the Amazon DynamoDB Developer Guide.

Type: String to `AttributeValue (p. 235)` object map

Required: No

**ReturnConsumedCapacity (p. 87)**

Determines the level of detail about provisioned throughput consumption that is returned in the response:

- **INDEXES** - The response includes the aggregate `ConsumedCapacity` for the operation, together with `ConsumedCapacity` for each table and secondary index that was accessed.

  Note that some operations, such as `GetItem` and `BatchGetItem`, do not access any indexes at all. In these cases, specifying INDEXES will only return `ConsumedCapacity` information for table(s).

- **TOTAL** - The response includes only the aggregate `ConsumedCapacity` for the operation.

- **NONE** - No `ConsumedCapacity` details are included in the response.

Type: String

Valid Values: INDEXES | TOTAL | NONE

Required: No

**ReturnItemCollectionMetrics (p. 87)**

Determines whether item collection metrics are returned. If set to `SIZE`, the response includes statistics about item collections, if any, that were modified during the operation are returned in the response. If set to `NONE` (the default), no statistics are returned.

Type: String

Valid Values: SIZE | NONE

Required: No

**ReturnValues (p. 87)**

Use `ReturnValues` if you want to get the item attributes as they appeared before they were updated with the `PutItem` request. For `PutItem`, the valid values are:

- **NONE** - If `ReturnValues` is not specified, or if its value is `NONE`, then nothing is returned. (This setting is the default for `ReturnValues`.)

- **ALL_OLD** - If `PutItem` overwrote an attribute name-value pair, then the content of the old item is returned.
Note
The ReturnValues parameter is used by several DynamoDB operations; however, PutItem does not recognize any values other than NONE or ALL_OLD.

Type: String
Valid Values: NONE | ALL_OLD | UPDATED_OLD | ALL_NEW | UPDATED_NEW
Required: No

Response Syntax

```json
{
    "Attributes": {
        "string": {
            "B": blob,
            "BOOL": boolean,
            "BS": [ blob ],
            "L": [ "AttributeValue" ],
            "M": {
                "string": "AttributeValue"
            },
            "N": "string",
            "NS": [ "string" ],
            "NULL": boolean,
            "S": "string",
            "SS": [ "string" ]
        }
    },
    "ConsumedCapacity": {
        "CapacityUnits": number,
        "GlobalSecondaryIndexes": {
            "string": {
                "CapacityUnits": number
            }
        },
        "LocalSecondaryIndexes": {
            "string": {
                "CapacityUnits": number
            }
        },
        "Table": {
            "CapacityUnits": number
        },
        "TableName": "string"
    },
    "ItemCollectionMetrics": {
        "ItemCollectionKey": {
            "string": {
                "B": blob,
                "BOOL": boolean,
                "BS": [ blob ],
                "L": [ "AttributeValue" ],
                "M": {
                    "string": "AttributeValue"
                },
                "N": "string",
                "NS": [ "string" ],
                "NULL": boolean,
                "S": "string",
                "SS": [ "string" ]
            }
        }
    }
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Attributes (p. 92)

The attribute values as they appeared before the `PutItem` operation, but only if `ReturnValues` is specified as `ALL_OLD` in the request. Each element consists of an attribute name and an attribute value.

Type: String to `AttributeValue (p. 235)` object map

Key Length Constraints: Maximum length of 65535.

ConsumedCapacity (p. 92)

The capacity units consumed by the `PutItem` operation. The data returned includes the total provisioned throughput consumed, along with statistics for the table and any indexes involved in the operation. `ConsumedCapacity` is only returned if the `ReturnConsumedCapacity` parameter was specified. For more information, see `Provisioned Throughput` in the `Amazon DynamoDB Developer Guide`.

Type: `ConsumedCapacity (p. 249)` object

ItemCollectionMetrics (p. 92)

Information about item collections, if any, that were affected by the `PutItem` operation. `ItemCollectionMetrics` is only returned if the `ReturnItemCollectionMetrics` parameter was specified. If the table does not have any local secondary indexes, this information is not returned in the response.

Each `ItemCollectionMetrics` element consists of:

- `ItemCollectionKey` - The partition key value of the item collection. This is the same as the partition key value of the item itself.
- `SizeEstimateRangeGB` - An estimate of item collection size, in gigabytes. This value is a two-element array containing a lower bound and an upper bound for the estimate. The estimate includes the size of all the items in the table, plus the size of all attributes projected into all of the local secondary indexes on that table. Use this estimate to measure whether a local secondary index is approaching its size limit.

The estimate is subject to change over time; therefore, do not rely on the precision or accuracy of the estimate.

Type: `ItemCollectionMetrics (p. 273)` object

Errors

For information about the errors that are common to all actions, see `Common Errors (p. 341)`. 
ConditionalCheckFailedException

A condition specified in the operation could not be evaluated.

HTTP Status Code: 400

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

ItemCollectionSizeLimitExceededException

An item collection is too large. This exception is only returned for tables that have one or more local secondary indexes.

HTTP Status Code: 400

ProvisionedThroughputExceededException

Your request rate is too high. The AWS SDKs for DynamoDB automatically retry requests that receive this exception. Your request is eventually successful, unless your retry queue is too large to finish. Reduce the frequency of requests and use exponential backoff. For more information, go to Error Retries and Exponential Backoff in the Amazon DynamoDB Developer Guide.

HTTP Status Code: 400

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

Example

Put an Item

The following example puts a new item into the Thread table, but only if there is not already an item in the table with the same key.

Sample Request

```json
POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>, Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.PutItem

{
  "TableName": "Thread",
  "Item": {
    "LastPostDateTime": {
      "S": "201303190422"
    },
    "Tags": {
```
"SS": ["Update","Multiple Items","HelpMe"]
},
"ForumName": {
    "S": "Amazon DynamoDB"
},
"Message": {
    "S": "I want to update multiple items in a single call. What's the best way to do that?"
},
"Subject": {
    "S": "How do I update multiple items?"
},
"LastPostedBy": {
    "S": "fred@example.com"
},
"ConditionExpression": "ForumName <> :f and Subject <> :s",
"ExpressionAttributeValues": {
    ":f": {"S": "Amazon DynamoDB"},
    ":s": {"S": "How do I update multiple items?"}
}

Sample Response

HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>
{
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
Query
Service: Amazon DynamoDB

The Query operation finds items based on primary key values. You can query any table or secondary index that has a composite primary key (a partition key and a sort key).

Use the KeyConditionExpression parameter to provide a specific value for the partition key. The Query operation will return all of the items from the table or index with that partition key value. You can optionally narrow the scope of the Query operation by specifying a sort key value and a comparison operator in KeyConditionExpression. To further refine the Query results, you can optionally provide a FilterExpression. A FilterExpression determines which items within the results should be returned to you. All of the other results are discarded.

A Query operation always returns a result set. If no matching items are found, the result set will be empty. Queries that do not return results consume the minimum number of read capacity units for that type of read operation.

Note
DynamoDB calculates the number of read capacity units consumed based on item size, not on the amount of data that is returned to an application. The number of capacity units consumed will be the same whether you request all of the attributes (the default behavior) or just some of them (using a projection expression). The number will also be the same whether or not you use a FilterExpression.

Query results are always sorted by the sort key value. If the data type of the sort key is Number, the results are returned in numeric order; otherwise, the results are returned in order of UTF-8 bytes. By default, the sort order is ascending. To reverse the order, set the ScanIndexForward parameter to false.

A single Query operation will read up to the maximum number of items set (if using the Limit parameter) or a maximum of 1 MB of data and then apply any filtering to the results using FilterExpression. If LastEvaluatedKey is present in the response, you will need to paginate the result set. For more information, see Paginating the Results in the Amazon DynamoDB Developer Guide.

FilterExpression is applied after a Query finishes, but before the results are returned. A FilterExpression cannot contain partition key or sort key attributes. You need to specify those attributes in the KeyConditionExpression.

Note
A Query operation can return an empty result set and a LastEvaluatedKey if all the items read for the page of results are filtered out.

You can query a table, a local secondary index, or a global secondary index. For a query on a table or on a local secondary index, you can set the ConsistentRead parameter to true and obtain a strongly consistent result. Global secondary indexes support eventually consistent reads only, so do not specify ConsistentRead when querying a global secondary index.

Request Syntax

```
{
    "AttributesToGet": [ "string" ],
    "ConditionalOperator": "string",
    "ConsistentRead": boolean,
    "ExclusiveStartKey": {
        "string": {
            "B": blob,
            "BOOL": boolean,
            "BS": [ blob ],
            "L": [ "AttributeValue"
```
Amazon DynamoDB API Reference
Query

"M": {
  "string": "AttributeValue"
},
"N": "string",
"NS": [ "string" ],
"NULL": boolean,
"S": "string",
"SS": [ "string" ]
},
"ExpressionAttributeNames": {
  "string": "string"
},
"ExpressionAttributeValues": {
  "string": {
    "B": blob,
    "BOOL": boolean,
    "BS": [ blob ],
    "L": [ "AttributeValue"
    ],
    "M": {
      "string": "AttributeValue"
    },
    "N": "string",
    "NS": [ "string" ],
    "NULL": boolean,
    "S": "string",
    "SS": [ "string" ]
  }
},
"FilterExpression": "string",
"IndexName": "string",
"KeyConditionExpression": "string",
"KeyConditions": {
  "string": {
    "AttributeValueList": [
    {
      "B": blob,
      "BOOL": boolean,
      "BS": [ blob ],
      "L": [ "AttributeValue"
    },
    "M": {
      "string": "AttributeValue"
    },
    "N": "string",
    "NS": [ "string" ],
    "NULL": boolean,
    "S": "string",
    "SS": [ "string" ]
    }
    ],
    "ComparisonOperator": "string"
  }
},
"Limit": number,
"ProjectionExpression": "string",
"QueryFilter": {
  "string": {
    "AttributeValueList": [
    {
      "B": blob,
      "BOOL": boolean,
      "BS": [ blob ],
      "L": [ "AttributeValue"
    },
    "M": {
      "string": "AttributeValue"
    },
    "N": "string",
    "NS": [ "string" ],
    "NULL": boolean,
    "S": "string",
    "SS": [ "string" ]
    }
    ],
    "ComparisonOperator": "string"
  }
},
Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**TableName (p. 96)**

The name of the table containing the requested items.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: Yes

**AttributesToGet (p. 96)**

This is a legacy parameter. Use ProjectionExpression instead. For more information, see AttributesToGet in the Amazon DynamoDB Developer Guide.

Type: Array of strings

Array Members: Minimum number of 1 item.

Length Constraints: Maximum length of 65535.

Required: No

**ConditionalOperator (p. 96)**

This is a legacy parameter. Use FilterExpression instead. For more information, see ConditionalOperator in the Amazon DynamoDB Developer Guide.

Type: String

Valid Values: AND | OR

Required: No
ConsistentRead (p. 96)

Determines the read consistency model: If set to true, then the operation uses strongly consistent reads; otherwise, the operation uses eventually consistent reads.

Strongly consistent reads are not supported on global secondary indexes. If you query a global secondary index with ConsistentRead set to true, you will receive a ValidationException.

Type: Boolean

Required: No

ExclusiveStartKey (p. 96)

The primary key of the first item that this operation will evaluate. Use the value that was returned for LastEvaluatedKey in the previous operation.

The data type for ExclusiveStartKey must be String, Number or Binary. No set data types are allowed.

Type: String to AttributeValue (p. 235) object map

Key Length Constraints: Maximum length of 65535.

Required: No

ExpressionAttributeNames (p. 96)

One or more substitution tokens for attribute names in an expression. The following are some use cases for using ExpressionAttributeNames:

• To access an attribute whose name conflicts with a DynamoDB reserved word.

• To create a placeholder for repeating occurrences of an attribute name in an expression.

• To prevent special characters in an attribute name from being misinterpreted in an expression.

Use the # character in an expression to dereference an attribute name. For example, consider the following attribute name:

• Percentile

The name of this attribute conflicts with a reserved word, so it cannot be used directly in an expression. (For the complete list of reserved words, see Reserved Words in the Amazon DynamoDB Developer Guide). To work around this, you could specify the following for ExpressionAttributeNames:

• {"#P":"Percentile"}

You could then use this substitution in an expression, as in this example:

• #P = :val

Note

Tokens that begin with the : character are expression attribute values, which are placeholders for the actual value at runtime.

For more information on expression attribute names, see Accessing Item Attributes in the Amazon DynamoDB Developer Guide.

Type: String to string map

Value Length Constraints: Maximum length of 65535.

Required: No

ExpressionAttributeValues (p. 96)

One or more values that can be substituted in an expression.
Use the colon character in an expression to dereference an attribute value. For example, suppose that you wanted to check whether the value of the `ProductStatus` attribute was one of the following:

Available | Backordered | Discontinued

You would first need to specify `ExpressionAttributeValues` as follows:

```json
{ "avail":{"S":"Available"}, "back":{"S":"Backordered"}, "disc":
{"S":"Discontinued"} }
```

You could then use these values in an expression, such as this:

`ProductStatus IN (:avail, :back, :disc)`

For more information on expression attribute values, see Specifying Conditions in the Amazon DynamoDB Developer Guide.

**Type:** String to `AttributeValue (p. 235)` object map  
**Required:** No

### FilterExpression (p. 96)

A string that contains conditions that DynamoDB applies after the `Query` operation, but before the data is returned to you. Items that do not satisfy the `FilterExpression` criteria are not returned.

A `FilterExpression` does not allow key attributes. You cannot define a filter expression based on a partition key or a sort key.

**Note**  
A `FilterExpression` is applied after the items have already been read; the process of filtering does not consume any additional read capacity units.

For more information, see Filter Expressions in the Amazon DynamoDB Developer Guide.

**Type:** String  
**Required:** No

### IndexName (p. 96)

The name of an index to query. This index can be any local secondary index or global secondary index on the table. Note that if you use the `IndexName` parameter, you must also provide `TableName`.

**Type:** String  
**Length Constraints:** Minimum length of 3. Maximum length of 255.  
**Pattern:** `[a-zA-Z0-9_.-]+`  
**Required:** No

### KeyConditionExpression (p. 96)

The condition that specifies the key value(s) for items to be retrieved by the `Query` action.

The condition must perform an equality test on a single partition key value.

The condition can optionally perform one of several comparison tests on a single sort key value. This allows `Query` to retrieve one item with a given partition key value and sort key value, or several items that have the same partition key value but different sort key values.

The partition key equality test is required, and must be specified in the following format:
partitionKeyName = :partitionkeyval

If you also want to provide a condition for the sort key, it must be combined using AND with the condition for the sort key. Following is an example, using the = comparison operator for the sort key:

partitionKeyName = :partitionkeyval AND sortKeyName = :sortkeyval

Valid comparisons for the sort key condition are as follows:

- sortKeyName = :sortkeyval - true if the sort key value is equal to :sortkeyval.
- sortKeyName < :sortkeyval - true if the sort key value is less than :sortkeyval.
- sortKeyName <= :sortkeyval - true if the sort key value is less than or equal to :sortkeyval.
- sortKeyName > :sortkeyval - true if the sort key value is greater than :sortkeyval.
- sortKeyName >= :sortkeyval - true if the sort key value is greater than or equal to :sortkeyval.
- sortKeyName BETWEEN :sortkeyval1 AND :sortkeyval2 - true if the sort key value is greater than or equal to :sortkeyval1, and less than or equal to :sortkeyval2.
- begins_with ( sortKeyName, :sortkeyval ) - true if the sort key value begins with a particular operand. (You cannot use this function with a sort key that is of type Number.) Note that the function name begins_with is case-sensitive.

Use the ExpressionAttributeValue parameter to replace tokens such as :partitionval and :sortval with actual values at runtime.

You can optionally use the ExpressionAttributeName parameter to replace the names of the partition key and sort key with placeholder tokens. This option might be necessary if an attribute name conflicts with a DynamoDB reserved word. For example, the following KeyConditionExpression parameter causes an error because Size is a reserved word:

- Size = :myval

To work around this, define a placeholder (such a #S) to represent the attribute name Size.

KeyConditionExpression then is as follows:

- #S = :myval

For a list of reserved words, see Reserved Words in the Amazon DynamoDB Developer Guide.

For more information on ExpressionAttributeValue and ExpressionAttributeName, see Using Placeholders for Attribute Names and Values in the Amazon DynamoDB Developer Guide.

Type: String

Required: No

KeyConditions (p. 96)

This is a legacy parameter. Use KeyConditionExpression instead. For more information, see KeyConditions in the Amazon DynamoDB Developer Guide.

Type: String to Condition (p. 246) object map

Key Length Constraints: Maximum length of 65535.

Required: No

Limit (p. 96)

The maximum number of items to evaluate (not necessarily the number of matching items). If DynamoDB processes the number of items up to the limit while processing the results, it stops the operation and returns the matching values up to that point, and a key in LastEvaluatedKey to apply in a subsequent operation, so that you can pick up where you left off. Also, if the processed data set size exceeds 1 MB before DynamoDB reaches this limit, it stops the operation and returns
the matching values up to the limit, and a key in LastEvaluatedKey to apply in a subsequent operation to continue the operation. For more information, see Query and Scan in the Amazon DynamoDB Developer Guide.

Type: Integer
Valid Range: Minimum value of 1.
Required: No

ProjectionExpression (p. 96)

A string that identifies one or more attributes to retrieve from the table. These attributes can include scalars, sets, or elements of a JSON document. The attributes in the expression must be separated by commas.

If no attribute names are specified, then all attributes will be returned. If any of the requested attributes are not found, they will not appear in the result.

For more information, see Accessing Item Attributes in the Amazon DynamoDB Developer Guide.

Type: String
Required: No

QueryFilter (p. 96)

This is a legacy parameter. Use FilterExpression instead. For more information, see QueryFilter in the Amazon DynamoDB Developer Guide.

Type: String to Condition (p. 246) object map

Key Length Constraints: Maximum length of 65535.
Required: No

ReturnConsumedCapacity (p. 96)

Determines the level of detail about provisioned throughput consumption that is returned in the response:

- INDEXES - The response includes the aggregate ConsumedCapacity for the operation, together with ConsumedCapacity for each table and secondary index that was accessed.
  
  Note that some operations, such as GetItem and BatchGetItem, do not access any indexes at all. In these cases, specifying INDEXES will only return ConsumedCapacity information for table(s).
  
  TOTAL - The response includes only the aggregate ConsumedCapacity for the operation.
  
  NONE - No ConsumedCapacity details are included in the response.

Type: String
Valid Values: INDEXES | TOTAL | NONE
Required: No

ScanIndexForward (p. 96)

Specifies the order for index traversal: If true (default), the traversal is performed in ascending order; if false, the traversal is performed in descending order.

Items with the same partition key value are stored in sorted order by sort key. If the sort key data type is Number, the results are stored in numeric order. For type String, the results are stored in order of ASCII character code values. For type Binary, DynamoDB treats each byte of the binary data as unsigned.
If `ScanIndexForward` is `true`, DynamoDB returns the results in the order in which they are stored (by sort key value). This is the default behavior. If `ScanIndexForward` is `false`, DynamoDB reads the results in reverse order by sort key value, and then returns the results to the client.

Type: Boolean

Required: No

**Select (p. 96)**

The attributes to be returned in the result. You can retrieve all item attributes, specific item attributes, the count of matching items, or in the case of an index, some or all of the attributes projected into the index.

- **ALL_ATTRIBUTES** - Returns all of the item attributes from the specified table or index. If you query a local secondary index, then for each matching item in the index DynamoDB will fetch the entire item from the parent table. If the index is configured to project all item attributes, then all of the data can be obtained from the local secondary index, and no fetching is required.
- **ALL_PROJECTED_ATTRIBUTES** - Allowed only when querying an index. Retrieves all attributes that have been projected into the index. If the index is configured to project all attributes, this return value is equivalent to specifying ALL_ATTRIBUTES.
- **COUNT** - Returns the number of matching items, rather than the matching items themselves.
- **SPECIFIC_ATTRIBUTES** - Returns only the attributes listed in `AttributesToGet`. This return value is equivalent to specifying `AttributesToGet` without specifying any value for `Select`.

If you query or scan a local secondary index and request only attributes that are projected into that index, the operation will read only the index and not the table. If any of the requested attributes are not projected into the local secondary index, DynamoDB will fetch each of these attributes from the parent table. This extra fetching incurs additional throughput cost and latency.

If you query or scan a global secondary index, you can only request attributes that are projected into the index. Global secondary index queries cannot fetch attributes from the parent table.

If neither `Select` nor `AttributesToGet` are specified, DynamoDB defaults to `ALL_ATTRIBUTES` when accessing a table, and `ALL_PROJECTED_ATTRIBUTES` when accessing an index. You cannot use both `Select` and `AttributesToGet` together in a single request, unless the value for `Select` is `SPECIFIC_ATTRIBUTES`. (This usage is equivalent to specifying `AttributesToGet` without any value for `Select`.)

**Note**

If you use the `ProjectionExpression` parameter, then the value for `Select` can only be `SPECIFIC_ATTRIBUTES`. Any other value for `Select` will return an error.

Type: String

Valid Values: ALL_ATTRIBUTES | ALL_PROJECTED_ATTRIBUTES | SPECIFIC_ATTRIBUTES | COUNT

Required: No

**Response Syntax**

```json
{
  "ConsumedCapacity": {
    "CapacityUnits": number,
    "GlobalSecondaryIndexes": {
      "string": {
        "CapacityUnits": number
      }
    }
  }
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ConsumedCapacity (p. 103)

The capacity units consumed by the `query` operation. The data returned includes the total provisioned throughput consumed, along with statistics for the table and any indexes involved in the operation. ConsumedCapacity is only returned if the ReturnConsumedCapacity parameter was
specified For more information, see Provisioned Throughput in the Amazon DynamoDB Developer Guide.

Type: ConsumedCapacity (p. 249) object

Count (p. 103)
The number of items in the response.

If you used a QueryFilter in the request, then Count is the number of items returned after the filter was applied, and ScannedCount is the number of matching items before the filter was applied.

If you did not use a filter in the request, then Count and ScannedCount are the same.

Type: Integer

Items (p. 103)
An array of item attributes that match the query criteria. Each element in this array consists of an attribute name and the value for that attribute.

Type: Array of string to AttributeValue (p. 235) object maps

Key Length Constraints: Maximum length of 65535.

LastEvaluatedKey (p. 103)
The primary key of the item where the operation stopped, inclusive of the previous result set. Use this value to start a new operation, excluding this value in the new request.

If LastEvaluatedKey is empty, then the "last page" of results has been processed and there is no more data to be retrieved.

If LastEvaluatedKey is not empty, it does not necessarily mean that there is more data in the result set. The only way to know when you have reached the end of the result set is when LastEvaluatedKey is empty.

Type: String to AttributeValue (p. 235) object map

Key Length Constraints: Maximum length of 65535.

ScannedCount (p. 103)
The number of items evaluated, before any QueryFilter is applied. A high ScannedCount value with few, or no, Count results indicates an inefficient Query operation. For more information, see Count and ScannedCount in the Amazon DynamoDB Developer Guide.

If you did not use a filter in the request, then ScannedCount is the same as Count.

Type: Integer

Errors
For information about the errors that are common to all actions, see Common Errors (p. 341).

InternalServerError
An error occurred on the server side.

HTTP Status Code: 500
ProvisionedThroughputExceededException

Your request rate is too high. The AWS SDKs for DynamoDB automatically retry requests that receive this exception. Your request is eventually successful, unless your retry queue is too large to finish. Reduce the frequency of requests and use exponential backoff. For more information, go to Error Retries and Exponential Backoff in the Amazon DynamoDB Developer Guide.

HTTP Status Code: 400

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

Examples

Retrieve a Range of Items

The following example queries the Reply table for replies in a forum that were posted by particular users. There is a local secondary index on the Reply table, PostedBy-Index, to facilitate fast lookups on the these attributes.

The ProjectionExpression parameter determines which attributes are returned.

Sample Request

```json
POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.Query
{
  "TableName": "Reply",
  "IndexName": "PostedBy-Index",
  "Limit": 3,
  "ConsistentRead": true,
  "ProjectionExpression": "Id, PostedBy, ReplyDateTime",
  "KeyConditionExpression": "Id = :v1 AND PostedBy BETWEEN :v2a AND :v2b",
  "ExpressionAttributeValues": {
    ":v1": {"S": "Amazon DynamoDB#DynamoDB Thread 1"},
    ":v2a": {"S": "User A"},
    ":v2b": {"S": "User C"}
  },
  "ReturnConsumedCapacity": "TOTAL"
}
```

Sample Response

```
HTTP/1.1 200 OK
```
Count Items

The following example returns the number of items in the Thread table for a particular forum.

Sample Request

POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
User-Agent: <UserAgentString>
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.Query

{
   "TableName": "Thread",
   "ConsistentRead": true,
   "KeyConditionExpression": "ForumName = :val",
   "ExpressionAttributeValues": {":val": {"S": "Amazon DynamoDB"}}
}

Sample Response

HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>

{
  "ConsumedCapacity": {
    "CapacityUnits": 1,
    "TableName": "Reply"
  },
  "Count": 2,
  "Items": [
    {
      "ReplyDateTime": {"S": "2015-02-18T20:27:36.165Z"},
      "PostedBy": {"S": "User A"},
      "Id": {"S": "Amazon DynamoDB#DynamoDB Thread 1"}
    },
    {
      "ReplyDateTime": {"S": "2015-02-25T20:27:36.165Z"},
      "PostedBy": {"S": "User B"},
      "Id": {"S": "Amazon DynamoDB#DynamoDB Thread 1"}
    }
  ],
  "ScannedCount": 2
}
"Count": 2,
"ScannedCount": 2
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
RestoreTableFromBackup
Service: Amazon DynamoDB

Creates a new table from an existing backup. Any number of users can execute up to 10 concurrent restores in a given account.

You can call RestoreTableFromBackup at a maximum rate of 10 times per second.

You must manually set up the following on the restored table:

- Auto scaling policies
- IAM policies
- Cloudwatch metrics and alarms
- Tags
- Stream settings
- Time to Live (TTL) settings

Request Syntax

```json
{
    "BackupArn": "string",
    "TargetTableName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**BackupArn (p. 109)**

The ARN associated with the backup.

Type: String


Required: Yes

**TargetTableName (p. 109)**

The name of the new table to which the backup must be restored.

Type: String


Pattern: [a-zA-Z0-9-_.-]+

Required: Yes

Response Syntax

```json
{
    "TableDescription": {
```

109
"AttributeDefinitions": [
    {
        "AttributeName": "string",
        "AttributeType": "string"
    }
],
"CreationDateTime": number,
"GlobalSecondaryIndexes": [
    {
        "Backfilling": boolean,
        "IndexArn": "string",
        "IndexName": "string",
        "IndexSizeBytes": number,
        "IndexStatus": "string",
        "ItemCount": number,
        "KeySchema": [
            {
                "AttributeName": "string",
                "KeyType": "string"
            }
        ],
        "Projection": {
            "NonKeyAttributes": [ "string" ],
            "ProjectionType": "string"
        },
        "ProvisionedThroughput": {
            "LastDecreaseDateTime": number,
            "LastIncreaseDateTime": number,
            "NumberOfDecreasesToday": number,
            "ReadCapacityUnits": number,
            "WriteCapacityUnits": number
        }
    }
],
"ItemCount": number,
"KeySchema": [
    {
        "AttributeName": "string",
        "KeyType": "string"
    }
],
"LatestStreamArn": "string",
"LatestStreamLabel": "string",
"LocalSecondaryIndexes": [
    {
        "IndexArn": "string",
        "IndexName": "string",
        "IndexSizeBytes": number,
        "ItemCount": number,
        "KeySchema": [
            {
                "AttributeName": "string",
                "KeyType": "string"
            }
        ],
        "Projection": {
            "NonKeyAttributes": [ "string" ],
            "ProjectionType": "string"
        }
    }
],
"ProvisionedThroughput": {
    "LastDecreaseDateTime": number,
    "LastIncreaseDateTime": number,
    "NumberOfDecreasesToday": number,
    "ReadCapacityUnits": number,
    "WriteCapacityUnits": number
}
"WriteCapacityUnits": number
},
"RestoreSummary": {
  "RestoreDateTime": number,
  "RestoreInProgress": boolean,
  "SourceBackupArn": "string",
  "SourceTableArn": "string"
},
"StreamSpecification": {
  "StreamEnabled": boolean,
  "StreamViewType": "string"
},
"TableArn": "string",
"TableId": "string",
"TableName": "string",
"TableSizeBytes": number,
"TableStatus": "string"
}

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

TableDescription (p. 109)

The description of the table created from an existing backup.

Type: TableDescription (p. 297) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

BackupInUseException

There is another ongoing conflicting backup control plane operation on the table. The backups is either being created, deleted or restored to a table.

HTTP Status Code: 400

BackupNotFoundException

Backup not found for the given BackupARN.

HTTP Status Code: 400

InternalServerException

An error occurred on the server side.

HTTP Status Code: 500

LimitExceededExeception

Up to 50 CreateBackup operations are allowed per second, per account. There is no limit to the number of daily on-demand backups that can be taken.

Up to 10 simultaneous table operations are allowed per account. These operations include CreateTable, UpdateTable, DeleteTable, UpdateTimeToLive, and RestoreTableFromBackup.
For tables with secondary indexes, only one of those tables can be in the `CREATING` state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the `ACTIVE` state is 250.

For tables with secondary indexes, only one of those tables can be in the `CREATING` state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the `ACTIVE` state is 250.

HTTP Status Code: 400

`TableAlreadyExistsException`

A table with the name already exists.

HTTP Status Code: 400

`TableInUseException`

A table by that name is either being created or deleted.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
Scan

Service: Amazon DynamoDB

The `Scan` operation returns one or more items and item attributes by accessing every item in a table or a secondary index. To have DynamoDB return fewer items, you can provide a `FilterExpression` operation.

If the total number of scanned items exceeds the maximum data set size limit of 1 MB, the scan stops and results are returned to the user as a `LastEvaluatedKey` value to continue the scan in a subsequent operation. The results also include the number of items exceeding the limit. A scan can result in no table data meeting the filter criteria.

A single `Scan` operation will read up to the maximum number of items set (if using the `Limit` parameter) or a maximum of 1 MB of data and then apply any filtering to the results using `FilterExpression`. If `LastEvaluatedKey` is present in the response, you will need to paginate the result set. For more information, see Paginating the Results in the Amazon DynamoDB Developer Guide.

`Scan` operations proceed sequentially; however, for faster performance on a large table or secondary index, applications can request a parallel `Scan` operation by providing the `Segment` and `TotalSegments` parameters. For more information, see Parallel Scan in the Amazon DynamoDB Developer Guide.

`Scan` uses eventually consistent reads when accessing the data in a table; therefore, the result set might not include the changes to data in the table immediately before the operation began. If you need a consistent copy of the data, as of the time that the `Scan` begins, you can set the `ConsistentRead` parameter to `true`.

Request Syntax

```json
{
  "AttributesToGet": [ "string" ],
  "ConditionalOperator": "string",
  "ConsistentRead": boolean,
  "ExclusiveStartKey": {
    "string": {
      "B": blob,
      "BOOL": boolean,
      "BS": [ blob ],
      "L": [
        "AttributeValue"
      ],
      "M": {
        "string": "AttributeValue"
      },
      "N": "string",
      "NS": [ "string" ],
      "NULL": boolean,
      "S": "string",
      "SS": [ "string" ]
    }
  },
  "ExpressionAttributeNames": {
    "string": "string"
  },
  "ExpressionAttributeValues": {
    "string": {
      "B": blob,
      "BOOL": boolean,
      "BS": [ blob ],
      "L": [
        "AttributeValue"
      ]
    }
  }
}
```
Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**TableName (p. 113)**

The name of the table containing the requested items; or, if you provide IndexName, the name of the table to which that index belongs.

Type: String


Pattern: [a-zA-Z0-9_:-]+

Required: Yes
AttributesToGet (p. 113)

This is a legacy parameter. Use ProjectionExpression instead. For more information, see AttributesToGet in the Amazon DynamoDB Developer Guide.

Type: Array of strings
Array Members: Minimum number of 1 item.
Length Constraints: Maximum length of 65535.
Required: No

ConditionalOperator (p. 113)

This is a legacy parameter. Use FilterExpression instead. For more information, see ConditionalOperator in the Amazon DynamoDB Developer Guide.

Type: String
Valid Values: AND | OR
Required: No

ConsistentRead (p. 113)

A Boolean value that determines the read consistency model during the scan:
• If ConsistentRead is false, then the data returned from Scan might not contain the results from other recently completed write operations (PutItem, UpdateItem or DeleteItem).
• If ConsistentRead is true, then all of the write operations that completed before the Scan began are guaranteed to be contained in the Scan response.

The default setting for ConsistentRead is false.

The ConsistentRead parameter is not supported on global secondary indexes. If you scan a global secondary index with ConsistentRead set to true, you will receive a ValidationException.

Type: Boolean
Required: No

ExclusiveStartKey (p. 113)

The primary key of the first item that this operation will evaluate. Use the value that was returned for LastEvaluatedKey in the previous operation.

The data type for ExclusiveStartKey must be String, Number or Binary. No set data types are allowed.

In a parallel scan, a Scan request that includes ExclusiveStartKey must specify the same segment whose previous Scan returned the corresponding value of LastEvaluatedKey.

Type: String to AttributeValue (p. 235) object map
Key Length Constraints: Maximum length of 65535.
Required: No

ExpressionAttributeNames (p. 113)

One or more substitution tokens for attribute names in an expression. The following are some use cases for using ExpressionAttributeNames:
• To access an attribute whose name conflicts with a DynamoDB reserved word.
• To create a placeholder for repeating occurrences of an attribute name in an expression.
• To prevent special characters in an attribute name from being misinterpreted in an expression.

Use the # character in an expression to dereference an attribute name. For example, consider the following attribute name:

• Percentile

The name of this attribute conflicts with a reserved word, so it cannot be used directly in an expression. (For the complete list of reserved words, see Reserved Words in the Amazon DynamoDB Developer Guide). To work around this, you could specify the following for ExpressionAttributeNames:

• {"#P":"Percentile"}

You could then use this substitution in an expression, as in this example:

• #P = :val

**Note**

Tokens that begin with the : character are expression attribute values, which are placeholders for the actual value at runtime.

For more information on expression attribute names, see Accessing Item Attributes in the Amazon DynamoDB Developer Guide.

Type: String to string map

Value Length Constraints: Maximum length of 65535.

Required: No

**ExpressionAttributeValues (p. 113)**

One or more values that can be substituted in an expression.

Use the : (colon) character in an expression to dereference an attribute value. For example, suppose that you wanted to check whether the value of the ProductStatus attribute was one of the following:

Available | Backordered | Discontinued

You would first need to specify ExpressionAttributeValues as follows:

{ "avail":{"S":"Available"}, "back":{"S":"Backordered"}, "disc":
{"S":"Discontinued"} }

You could then use these values in an expression, such as this:

ProductStatus IN (:avail, :back, :disc)

For more information on expression attribute values, see Specifying Conditions in the Amazon DynamoDB Developer Guide.

Type: String to AttributeValue (p. 235) object map

Required: No

**FilterExpression (p. 113)**

A string that contains conditions that DynamoDB applies after the Scan operation, but before the data is returned to you. Items that do not satisfy the FilterExpression criteria are not returned.

**Note**

A FilterExpression is applied after the items have already been read; the process of filtering does not consume any additional read capacity units.
Scan

For more information, see Filter Expressions in the Amazon DynamoDB Developer Guide.

Type: String

Required: No

**IndexName (p. 113)**

The name of a secondary index to scan. This index can be any local secondary index or global secondary index. Note that if you use the IndexName parameter, you must also provide TableName.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: No

**Limit (p. 113)**

The maximum number of items to evaluate (not necessarily the number of matching items). If DynamoDB processes the number of items up to the limit while processing the results, it stops the operation and returns the matching values up to that point, and a key in LastEvaluatedKey to apply in a subsequent operation, so that you can pick up where you left off. Also, if the processed data set size exceeds 1 MB before DynamoDB reaches this limit, it stops the operation and returns the matching values up to the limit, and a key in LastEvaluatedKey to apply in a subsequent operation to continue the operation. For more information, see Query and Scan in the Amazon DynamoDB Developer Guide.

Type: Integer

Valid Range: Minimum value of 1.

Required: No

**ProjectionExpression (p. 113)**

A string that identifies one or more attributes to retrieve from the specified table or index. These attributes can include scalars, sets, or elements of a JSON document. The attributes in the expression must be separated by commas.

If no attribute names are specified, then all attributes will be returned. If any of the requested attributes are not found, they will not appear in the result.

For more information, see Accessing Item Attributes in the Amazon DynamoDB Developer Guide.

Type: String

Required: No

**ReturnConsumedCapacity (p. 113)**

Determines the level of detail about provisioned throughput consumption that is returned in the response:

- **INDEXES** - The response includes the aggregate ConsumedCapacity for the operation, together with ConsumedCapacity for each table and secondary index that was accessed.

  Note that some operations, such as GetItem and BatchGetItem, do not access any indexes at all. In these cases, specifying INDEXES will only return ConsumedCapacity information for table(s).

- **TOTAL** - The response includes only the aggregate ConsumedCapacity for the operation.
• **NONE** - No `ConsumedCapacity` details are included in the response.

  Type: String

  Valid Values: INDEXES | TOTAL | NONE

  Required: No

**ScanFilter (p. 113)**

This is a legacy parameter. Use `FilterExpression` instead. For more information, see `ScanFilter` in the *Amazon DynamoDB Developer Guide*.

  Type: String to `Condition` (p. 246) object map

  Key Length Constraints: Maximum length of 65535.

  Required: No

**Segment (p. 113)**

For a parallel `Scan` request, `Segment` identifies an individual segment to be scanned by an application worker.

Segment IDs are zero-based, so the first segment is always 0. For example, if you want to use four application threads to scan a table or an index, then the first thread specifies a `Segment` value of 0, the second thread specifies 1, and so on.

The value of `LastEvaluatedKey` returned from a parallel `Scan` request must be used as `ExclusiveStartKey` with the same segment ID in a subsequent `Scan` operation.

The value for `Segment` must be greater than or equal to 0, and less than the value provided for `TotalSegments`.

If you provide `Segment`, you must also provide `TotalSegments`.

  Type: Integer

  Valid Range: Minimum value of 0. Maximum value of 999999.

  Required: No

**Select (p. 113)**

The attributes to be returned in the result. You can retrieve all item attributes, specific item attributes, the count of matching items, or in the case of an index, some or all of the attributes projected into the index.

• **ALL_ATTRIBUTES** - Returns all of the item attributes from the specified table or index. If you query a local secondary index, then for each matching item in the index DynamoDB will fetch the entire item from the parent table. If the index is configured to project all item attributes, then all of the data can be obtained from the local secondary index, and no fetching is required.

• **ALL_PROJECTED_ATTRIBUTES** - Allowed only when querying an index. Retrieves all attributes that have been projected into the index. If the index is configured to project all attributes, this return value is equivalent to specifying `ALL_ATTRIBUTES`.

• **COUNT** - Returns the number of matching items, rather than the matching items themselves.

• **SPECIFIC_ATTRIBUTES** - Returns only the attributes listed in `AttributesToGet`. This return value is equivalent to specifying `AttributesToGet` without specifying any value for `Select`.

If you query or scan a local secondary index and request only attributes that are projected into that index, the operation will read only the index and not the table. If any of the requested attributes are not projected into the local secondary index, DynamoDB will fetch each of these attributes from the parent table. This extra fetching incurs additional throughput cost and latency.
If you query or scan a global secondary index, you can only request attributes that are projected into the index. Global secondary index queries cannot fetch attributes from the parent table.

If neither `Select` nor `AttributesToGet` are specified, DynamoDB defaults to `ALL_ATTRIBUTES` when accessing a table, and `ALL_PROJECTED_ATTRIBUTES` when accessing an index. You cannot use both `Select` and `AttributesToGet` together in a single request, unless the value for `Select` is `SPECIFIC_ATTRIBUTES`. (This usage is equivalent to specifying `AttributesToGet` without any value for `Select`.)

**Note**

If you use the `ProjectionExpression` parameter, then the value for `Select` can only be `SPECIFIC_ATTRIBUTES`. Any other value for `Select` will return an error.

Type: String

Valid Values: `ALL_ATTRIBUTES` | `ALL_PROJECTED_ATTRIBUTES` | `SPECIFIC_ATTRIBUTES` | `COUNT`

Required: No

**TotalSegments (p. 113)**

For a parallel `Scan` request, `TotalSegments` represents the total number of segments into which the `Scan` operation will be divided. The value of `TotalSegments` corresponds to the number of application workers that will perform the parallel scan. For example, if you want to use four application threads to scan a table or an index, specify a `TotalSegments` value of 4.

The value for `TotalSegments` must be greater than or equal to 1, and less than or equal to 1000000. If you specify a `TotalSegments` value of 1, the `Scan` operation will be sequential rather than parallel.

If you specify `TotalSegments`, you must also specify `Segment`.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 1000000.

Required: No

**Response Syntax**

```json
{
    "ConsumedCapacity": {
        "CapacityUnits": number,
        "GlobalSecondaryIndexes": {
            "string": {
                "CapacityUnits": number
            }
        },
        "LocalSecondaryIndexes": {
            "string": {
                "CapacityUnits": number
            }
        },
        "Table": {
            "CapacityUnits": number
        },
        "TableName": "string"
    },
    "Count": number
}
```
"Items": [
  {
    "string": {
      "B": blob,
      "BOOL": boolean,
      "BS": [ blob ],
      "L": [
        "AttributeValue"
      ],
      "M": {
        "string": "AttributeValue"
      },
      "N": "string",
      "NS": [ "string" ],
      "NULL": boolean,
      "S": "string",
      "SS": [ "string" ]
    }
  }
],
"LastEvaluatedKey": {
  "string": {
    "B": blob,
    "BOOL": boolean,
    "BS": [ blob ],
    "L": [
      "AttributeValue"
    ],
    "M": {
      "string": "AttributeValue"
    },
    "N": "string",
    "NS": [ "string" ],
    "NULL": boolean,
    "S": "string",
    "SS": [ "string" ]
  }
},
"ScannedCount": number

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**ConsumedCapacity (p. 119)**

The capacity units consumed by the *Scan* operation. The data returned includes the total provisioned throughput consumed, along with statistics for the table and any indexes involved in the operation. ConsumedCapacity is only returned if the ReturnConsumedCapacity parameter was specified. For more information, see Provisioned Throughput in the *Amazon DynamoDB Developer Guide*.

Type: ConsumedCapacity (p. 249) object

**Count (p. 119)**

The number of items in the response.

If you set *ScanFilter* in the request, then *Count* is the number of items returned after the filter was applied, and *ScannedCount* is the number of matching items before the filter was applied.
If you did not use a filter in the request, then Count is the same as ScannedCount.

Type: Integer

Items (p. 119)

An array of item attributes that match the scan criteria. Each element in this array consists of an attribute name and the value for that attribute.

Type: Array of string to AttributeValue (p. 235) object maps

Key Length Constraints: Maximum length of 65535.

LastEvaluatedKey (p. 119)

The primary key of the item where the operation stopped, inclusive of the previous result set. Use this value to start a new operation, excluding this value in the new request.

If LastEvaluatedKey is empty, then the "last page" of results has been processed and there is no more data to be retrieved.

If LastEvaluatedKey is not empty, it does not necessarily mean that there is more data in the result set. The only way to know when you have reached the end of the result set is when LastEvaluatedKey is empty.

Type: String to AttributeValue (p. 235) object map

Key Length Constraints: Maximum length of 65535.

ScannedCount (p. 119)

The number of items evaluated, before any ScanFilter is applied. A high ScannedCount value with few, or no, Count results indicates an inefficient Scan operation. For more information, see Count and ScannedCount in the Amazon DynamoDB Developer Guide.

If you did not use a filter in the request, then ScannedCount is the same as Count.

Type: Integer

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

ProvisionedThroughputExceededException

Your request rate is too high. The AWS SDKs for DynamoDB automatically retry requests that receive this exception. Your request is eventually successful, unless your retry queue is too large to finish. Reduce the frequency of requests and use exponential backoff. For more information, go to Error Retries and Exponential Backoff in the Amazon DynamoDB Developer Guide.

HTTP Status Code: 400

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.
Examples

Return All Items

The following example returns all of the items in a table. No scan filter is applied.

Sample Request

```
POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>, Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.Scan

{
   "TableName": "Reply",
   "ReturnConsumedCapacity": "TOTAL"
}
```

Sample Response

```
HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>

{
   "ConsumedCapacity": {
      "CapacityUnits": 0.5,
      "TableName": "Reply"
   },
   "Count": 4,
   "Items": [
      {
         "PostedBy": {
            "S": "joe@example.com"
         },
         "ReplyDateTime": {
            "S": "20130320115336"
         },
         "Id": {
            "S": "Amazon DynamoDB#How do I update multiple items?"
         },
         "Message": {
            "S": "Have you looked at BatchWriteItem?"
         }
      },
      {
         "PostedBy": {
            "S": "fred@example.com"
         },
         "ReplyDateTime": {
            "S": "20130320115336"
         },
         "Id": {
            "S": "Amazon DynamoDB#How do I update multiple items?"
         },
         "Message": {
            "S": "Have you looked at BatchWriteItem?"
         }
      }
   ]
}
```
Use a Filter Expression

The following example returns only those items matching specific criteria.

Sample Request

```bash
POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.Scan

{
  "TableName": "Reply",
  "ScanIndexForward": "false",
  "FilterExpression": "Attr1 eq :one AND Attr2 gt :two",
  "AttributesToGet": ["Attr1", "Attr2", "Attr3"],
  "Limit": 100
}
```
"FilterExpression": "PostedBy = :val",
"ExpressionAttributeValues": {":val": {"S": "joe@example.com"}},
"ReturnConsumedCapacity": "TOTAL"
}

Sample Response

HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>
{
  "ConsumedCapacity": {
    "CapacityUnits": 0.5,
    "TableName": "Reply"
  },
  "Count": 2,
  "Items": [
    {
      "PostedBy": {
        "S": "joe@example.com"
      },
      "ReplyDateTime": {
        "S": "20130320115336"
      },
      "Id": {
        "S": "Amazon DynamoDB#How do I update multiple items?"
      },
      "Message": {
        "S": "Have you looked at BatchWriteItem?"
      }
    },
    {
      "PostedBy": {
        "S": "joe@example.com"
      },
      "ReplyDateTime": {
        "S": "20130320115347"
      },
      "Id": {
        "S": "Amazon DynamoDB#How do I update multiple items?"
      },
      "Message": {
        "S": "BatchWriteItem is documented in the Amazon DynamoDB API Reference."
      }
    }
  ],
  "ScannedCount": 4
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
TagResource
Service: Amazon DynamoDB

Associate a set of tags with an Amazon DynamoDB resource. You can then activate these user-defined tags so that they appear on the Billing and Cost Management console for cost allocation tracking. You can call TagResource up to 5 times per second, per account.

For an overview on tagging DynamoDB resources, see Tagging for DynamoDB in the Amazon DynamoDB Developer Guide.

Request Syntax

```json
{
    "ResourceArn": "string",
    "Tags": [
        {
            "Key": "string",
            "Value": "string"
        }
    ]
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ResourceArn (p. 126)**

Identifies the Amazon DynamoDB resource to which tags should be added. This value is an Amazon Resource Name (ARN).

Type: String


Required: Yes

**Tags (p. 126)**

The tags to be assigned to the Amazon DynamoDB resource.

Type: Array of Tag (p. 302) objects

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

**InternalServerError**

An error occurred on the server side.
HTTP Status Code: 500

LimitExceededException

Up to 50 CreateBackup operations are allowed per second, per account. There is no limit to the number of daily on-demand backups that can be taken.

Up to 10 simultaneous table operations are allowed per account. These operations include CreateTable, UpdateTable, DeleteTable, UpdateTimeToLive, and RestoreTableFromBackup.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

HTTP Status Code: 400

ResourceInUseException

The operation conflicts with the resource's availability. For example, you attempted to recreate an existing table, or tried to delete a table currently in the CREATING state.

HTTP Status Code: 400

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
UntagResource

Service: Amazon DynamoDB

Removes the association of tags from an Amazon DynamoDB resource. You can call UntagResource up to 5 times per second, per account.

For an overview on tagging DynamoDB resources, see Tagging for DynamoDB in the Amazon DynamoDB Developer Guide.

Request Syntax

```
{
   "ResourceArn": "string",
   "TagKeys": [ "string" ]
}
```

Request Parameters

The request accepts the following data in JSON format.

Note

In the following list, the required parameters are described first.

**ResourceArn (p. 128)**

The Amazon DynamoDB resource the tags will be removed from. This value is an Amazon Resource Name (ARN).

Type: String


Required: Yes

**TagKeys (p. 128)**

A list of tag keys. Existing tags of the resource whose keys are members of this list will be removed from the Amazon DynamoDB resource.

Type: Array of strings


Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

**InternalServerError**

An error occurred on the server side.

HTTP Status Code: 500
**LimitExceededException**

Up to 50 `CreateBackup` operations are allowed per second, per account. There is no limit to the number of daily on-demand backups that can be taken.

Up to 10 simultaneous table operations are allowed per account. These operations include `CreateTable`, `UpdateTable`, `DeleteTable`, `UpdateTimeToLive`, and `RestoreTableFromBackup`.

For tables with secondary indexes, only one of those tables can be in the `CREATING` state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the `ACTIVE` state is 250.

For tables with secondary indexes, only one of those tables can be in the `CREATING` state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the `ACTIVE` state is 250.

HTTP Status Code: 400

**ResourceInUseException**

The operation conflicts with the resource's availability. For example, you attempted to recreate an existing table, or tried to delete a table currently in the `CREATING` state.

HTTP Status Code: 400

**ResourceNotFoundException**

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be `ACTIVE`.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
UpdateGlobalTable
Service: Amazon DynamoDB

Adds or removes replicas in the specified global table. The global table must already exist to be able to use this operation. Any replica to be added must be empty, must have the same name as the global table, must have the same key schema, must have DynamoDB Streams enabled, and cannot have any local secondary indexes (LSIs).

**Note**
Although you can use UpdateGlobalTable to add replicas and remove replicas in a single request, for simplicity we recommend that you issue separate requests for adding or removing replicas.

**Request Syntax**

```json
{
  "GlobalTableName": "string",
  "ReplicaUpdates": [
    {
      "Create": {
        "RegionName": "string"
      },
      "Delete": {
        "RegionName": "string"
      }
    }
  ]
}
```

**Request Parameters**
The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

- **GlobalTableName (p. 130)**
  The global table name.
  Type: String
  Pattern: [a-zA-Z0-9-_.-]+
  Required: Yes

- **ReplicaUpdates (p. 130)**
  A list of regions that should be added or removed from the global table.
  Type: Array of ReplicaUpdate (p. 291) objects
  Required: Yes

**Response Syntax**

```json
{
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response. The following data is returned in JSON format by the service.

**GlobalTableDescription (p. 130)**

Contains the details of the global table.

Type: *GlobalTableDescription (p. 271)* object

Errors

For information about the errors that are common to all actions, see *Common Errors (p. 341).*

**GlobalTableNotFoundException**

The specified global table does not exist.

HTTP Status Code: 400

**InternalServerError**

An error occurred on the server side.

HTTP Status Code: 500

**ReplicaAlreadyExistsException**

The specified replica is already part of the global table.

HTTP Status Code: 400

**ReplicaNotFoundException**

The specified replica is no longer part of the global table.

HTTP Status Code: 400

**TableNameNotFoundException**

A table with the name *TableName* does not currently exist within the subscriber's account.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:
• AWS Command Line Interface
• AWS SDK for .NET
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
UpdateItem

Service: Amazon DynamoDB

Edits an existing item's attributes, or adds a new item to the table if it does not already exist. You can put, delete, or add attribute values. You can also perform a conditional update on an existing item (insert a new attribute name-value pair if it doesn't exist, or replace an existing name-value pair if it has certain expected attribute values).

You can also return the item's attribute values in the same UpdateItem operation using the ReturnValues parameter.

Request Syntax

```json
{
  "AttributeUpdates": {
    "string": {
      "Action": "string",
      "Value": {
        "B": blob,
        "BOOL": boolean,
        "BS": [ blob ],
        "L": [
          "AttributeValue"
        ],
        "M": {
          "string": "AttributeValue"
        },
        "N": "string",
        "NS": [ "string" ],
        "NULL": boolean,
        "S": "string",
        "SS": [ "string" ]
      }
    }
  },
  "ConditionalOperator": "string",
  "ConditionExpression": "string",
  "Expected": {
    "string": {
      "AttributeValueList": [
        {
          "B": blob,
          "BOOL": boolean,
          "BS": [ blob ],
          "L": [
            "AttributeValue"
          ],
          "M": {
            "string": "AttributeValue"
          },
          "N": "string",
          "NS": [ "string" ],
          "NULL": boolean,
          "S": "string",
          "SS": [ "string" ]
        }
      ],
      "ComparisonOperator": "string",
      "Exists": boolean,
      "Value": {
        "B": blob,
        "BOOL": boolean,
```
Request Parameters

The request accepts the following data in JSON format.
**Note**
In the following list, the required parameters are described first.

**Key (p. 133)**

The primary key of the item to be updated. Each element consists of an attribute name and a value for that attribute.

For the primary key, you must provide all of the attributes. For example, with a simple primary key, you only need to provide a value for the partition key. For a composite primary key, you must provide values for both the partition key and the sort key.

Type: String to `AttributeValue (p. 235)` object map

Key Length Constraints: Maximum length of 65535.

Required: Yes

**TableName (p. 133)**

The name of the table containing the item to update.

Type: String


Pattern: `[a-zA-Z0-9_.-]+`

Required: Yes

**AttributeUpdates (p. 133)**

This is a legacy parameter. Use `UpdateExpression` instead. For more information, see `AttributeUpdates` in the *Amazon DynamoDB Developer Guide*.

Type: String to `AttributeValueUpdate (p. 238)` object map

Key Length Constraints: Maximum length of 65535.

Required: No

**ConditionalOperator (p. 133)**

This is a legacy parameter. Use `ConditionExpression` instead. For more information, see `ConditionalOperator` in the *Amazon DynamoDB Developer Guide*.

Type: String

Valid Values: `AND` | `OR`

Required: No

**ConditionExpression (p. 133)**

A condition that must be satisfied in order for a conditional update to succeed.

An expression can contain any of the following:

- Functions: `attribute_exists` | `attribute_not_exists` | `attribute_type` | `contains` | `begins_with` | `size`
  
  These function names are case-sensitive.

- Comparison operators: `=` | `<>` | `<` | `>` | `<=` | `>=` | `BETWEEN` | `IN`

- Logical operators: `AND` | `OR` | `NOT`
For more information on condition expressions, see Specifying Conditions in the Amazon DynamoDB Developer Guide.

Type: String
Required: No

Expected (p. 133)

This is a legacy parameter. Use ConditionExpression instead. For more information, see Expected in the Amazon DynamoDB Developer Guide.

Type: String to ExpectedAttributeValue (p. 258) object map
Key Length Constraints: Maximum length of 65535.
Required: No

ExpressionAttributeNames (p. 133)

One or more substitution tokens for attribute names in an expression. The following are some use cases for using ExpressionAttributeNames:
• To access an attribute whose name conflicts with a DynamoDB reserved word.
• To create a placeholder for repeating occurrences of an attribute name in an expression.
• To prevent special characters in an attribute name from being misinterpreted in an expression.

Use the # character in an expression to dereference an attribute name. For example, consider the following attribute name:

• Percentile

The name of this attribute conflicts with a reserved word, so it cannot be used directly in an expression. (For the complete list of reserved words, see Reserved Words in the Amazon DynamoDB Developer Guide). To work around this, you could specify the following for ExpressionAttributeNames:
• {"#P":"Percentile"}

You could then use this substitution in an expression, as in this example:
• #P = :val

Note

Tokens that begin with the : character are expression attribute values, which are placeholders for the actual value at runtime.

For more information on expression attribute names, see Accessing Item Attributes in the Amazon DynamoDB Developer Guide.

Type: String to string map
Value Length Constraints: Maximum length of 65535.
Required: No

ExpressionAttributeValues (p. 133)

One or more values that can be substituted in an expression.

Use the : (colon) character in an expression to dereference an attribute value. For example, suppose that you wanted to check whether the value of the ProductStatus attribute was one of the following:

Available | Backordered | Discontinued

You would first need to specify ExpressionAttributeValues as follows:
{ "avail":{"S":"Available"}, "back":{"S":"Backordered"}, "disc":{"S":"Discontinued"} }

You could then use these values in an expression, such as this:

ProductStatus IN (:avail, :back, :disc)

For more information on expression attribute values, see Specifying Conditions in the Amazon DynamoDB Developer Guide.

Type: String to AttributeValue (p. 235) object map

Required: No

ReturnConsumedCapacity (p. 133)

Determines the level of detail about provisioned throughput consumption that is returned in the response:

- INDEXES - The response includes the aggregate ConsumedCapacity for the operation, together with ConsumedCapacity for each table and secondary index that was accessed.

  Note that some operations, such as GetItem and BatchGetItem, do not access any indexes at all. In these cases, specifying INDEXES will only return ConsumedCapacity information for table(s).

- TOTAL - The response includes only the aggregate ConsumedCapacity for the operation.

- NONE - No ConsumedCapacity details are included in the response.

Type: String

Valid Values: INDEXES | TOTAL | NONE

Required: No

ReturnItemCollectionMetrics (p. 133)

Determines whether item collection metrics are returned. If set to SIZE, the response includes statistics about item collections, if any, that were modified during the operation are returned in the response. If set to NONE (the default), no statistics are returned.

Type: String

Valid Values: SIZE | NONE

Required: No

ReturnValues (p. 133)

Use ReturnValues if you want to get the item attributes as they appear before or after they are updated. For UpdateItem, the valid values are:

- NONE - If ReturnValues is not specified, or if its value is NONE, then nothing is returned. (This setting is the default for ReturnValues.)

- ALL_OLD - Returns all of the attributes of the item, as they appeared before the UpdateItem operation.

- UPDATED_OLD - Returns only the updated attributes, as they appeared before the UpdateItem operation.

- ALL_NEW - Returns all of the attributes of the item, as they appear after the UpdateItem operation.

- UPDATED_NEW - Returns only the updated attributes, as they appear after the UpdateItem operation.
There is no additional cost associated with requesting a return value aside from the small network and processing overhead of receiving a larger response. No read capacity units are consumed.

The values returned are strongly consistent.

Type: String

Valid Values: NONE | ALL_OLD | UPDATED_OLD | ALL_NEW | UPDATED_NEW

Required: No

**UpdateExpression (p. 133)**

An expression that defines one or more attributes to be updated, the action to be performed on them, and new value(s) for them.

The following action values are available for UpdateExpression:

- **SET** - Adds one or more attributes and values to an item. If any of these attribute already exist, they are replaced by the new values. You can also use SET to add or subtract from an attribute that is of type Number. For example: SET myNum = myNum + :val

  SET supports the following functions:

  - **if_not_exists**(path, operand) - if the item does not contain an attribute at the specified path, then if_not_exists evaluates to operand; otherwise, it evaluates to path. You can use this function to avoid overwriting an attribute that may already be present in the item.

  - **list_append**(operand, operand) - evaluates to a list with a new element added to it. You can append the new element to the start or the end of the list by reversing the order of the operands.

  These function names are case-sensitive.

- **REMOVE** - Removes one or more attributes from an item.

- **ADD** - Adds the specified value to the item, if the attribute does not already exist. If the attribute does exist, then the behavior of ADD depends on the data type of the attribute:

  - If the existing attribute is a number, and if Value is also a number, then Value is mathematically added to the existing attribute. If Value is a negative number, then it is subtracted from the existing attribute.

```note
If you use ADD to increment or decrement a number value for an item that doesn't exist before the update, DynamoDB uses 0 as the initial value. Similarly, if you use ADD for an existing item to increment or decrement an attribute value that doesn't exist before the update, DynamoDB uses 0 as the initial value. For example, suppose that the item you want to update doesn't have an attribute named itemcount, but you decide to ADD the number 3 to this attribute anyway. DynamoDB will create the itemcount attribute, set its initial value to 0, and finally add 3 to it. The result will be a new itemcount attribute in the item, with a value of 3.
```

- If the existing data type is a set and if Value is also a set, then Value is added to the existing set. For example, if the attribute value is the set [1, 2], and the ADD action specified [3], then the final attribute value is [1, 2, 3]. An error occurs if an ADD action is specified for a set attribute and the attribute type specified does not match the existing set type.

Both sets must have the same primitive data type. For example, if the existing data type is a set of strings, the Value must also be a set of strings.

**Important**

The ADD action only supports Number and set data types. In addition, ADD can only be used on top-level attributes, not nested attributes.

- **DELETE** - Deletes an element from a set.
If a set of values is specified, then those values are subtracted from the old set. For example, if the attribute value was the set \([a, b, c]\) and the DELETE action specifies \([a, c]\), then the final attribute value is \([b]\). Specifying an empty set is an error.

**Important**
The DELETE action only supports set data types. In addition, DELETE can only be used on top-level attributes, not nested attributes.

You can have many actions in a single expression, such as the following: \(\text{SET} \ a=:\text{value1}, \ b=:\text{value2} \ \text{DELETE} \ :\text{value3}, \ :\text{value4}, \ :\text{value5}\)

For more information on update expressions, see Modifying Items and Attributes in the Amazon DynamoDB Developer Guide.

Type: String

Required: No

**Response Syntax**

```json
{
  "Attributes": {
    "string": {
      "B": blob,
      "BOOL": boolean,
      "BS": [ blob ],
      "L": [
        "AttributeValue"
      ],
      "M": {
        "string": "AttributeValue"
      },
      "N": "string",
      "NS": [ "string" ],
      "NULL": boolean,
      "S": "string",
      "SS": [ "string" ]
    }
  },
  "ConsumedCapacity": {
    "CapacityUnits": number,
    "GlobalSecondaryIndexes": {
      "string": {
        "CapacityUnits": number
      }
    },
    "LocalSecondaryIndexes": {
      "string": {
        "CapacityUnits": number
      }
    },
    "Table": {
      "CapacityUnits": number
    },
    "TableName": "string"
  },
  "ItemCollectionMetrics": {
    "ItemCollectionKey": {
      "B": blob,
      "BOOL": boolean,
      "BS": [ blob ],
      "L": [
        "AttributeValue"
      ],
      "M": {
        "string": "AttributeValue"
      },
      "N": "string",
      "NS": [ "string" ],
      "NULL": boolean,
      "S": "string",
      "SS": [ "string" ]
    }
  }
}
```
"BS": [ blob ],
"L": [ "AttributeValue" ],
"M": {
   "string": "AttributeValue"
},
"N": "string",
"NS": [ "string" ],
"NULL": boolean,
"S": "string",
"SS": [ "string" ]
},
"SizeEstimateRangeGB": [ number ]
}

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Attributes (p. 139)

A map of attribute values as they appear before or after the UpdateItem operation, as determined by the ReturnValues parameter.

The Attributes map is only present if ReturnValues was specified as something other than NONE in the request. Each element represents one attribute.

Type: String to AttributeValue (p. 235) object map

Key Length Constraints: Maximum length of 65535.

ConsumedCapacity (p. 139)

The capacity units consumed by the UpdateItem operation. The data returned includes the total provisioned throughput consumed, along with statistics for the table and any indexes involved in the operation. ConsumedCapacity is only returned if the ReturnConsumedCapacity parameter was specified. For more information, see Provisioned Throughput in the Amazon DynamoDB Developer Guide.

Type: ConsumedCapacity (p. 249) object

ItemCollectionMetrics (p. 139)

Information about item collections, if any, that were affected by the UpdateItem operation. ItemCollectionMetrics is only returned if the ReturnItemCollectionMetrics parameter was specified. If the table does not have any local secondary indexes, this information is not returned in the response.

Each ItemCollectionMetrics element consists of:

- ItemCollectionKey - The partition key value of the item collection. This is the same as the partition key value of the item itself.
- SizeEstimateRangeGB - An estimate of item collection size, in gigabytes. This value is a two-element array containing a lower bound and an upper bound for the estimate. The estimate includes the size of all the items in the table, plus the size of all attributes projected into all of the local secondary indexes on that table. Use this estimate to measure whether a local secondary index is approaching its size limit.
The estimate is subject to change over time; therefore, do not rely on the precision or accuracy of the estimate.

Type: ItemCollectionMetrics (p. 273) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

ConditionalCheckFailedException

A condition specified in the operation could not be evaluated.

HTTP Status Code: 400

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

ItemCollectionSizeLimitExceededException

An item collection is too large. This exception is only returned for tables that have one or more local secondary indexes.

HTTP Status Code: 400

ProvisionedThroughputExceededException

Your request rate is too high. The AWS SDKs for DynamoDB automatically retry requests that receive this exception. Your request is eventually successful, unless your retry queue is too large to finish. Reduce the frequency of requests and use exponential backoff. For more information, go to Error Retries and Exponential Backoff in the Amazon DynamoDB Developer Guide.

HTTP Status Code: 400

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

Examples

Conditional Update

This example updates the Thread table, changing the LastPostedBy attribute - but only if LastPostedBy is currently "fred@example.com". All of the item's attributes, as they appear after the update, are returned in the response.

Sample Request

POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
UpdateItem

```json
{
    "TableName": "Thread",
    "Key": {
        "ForumName": {
            "S": "Amazon DynamoDB"
        },
        "Subject": {
            "S": "Maximum number of items?"
        }
    },
    "UpdateExpression": "set LastPostedBy = :val1",
    "ConditionExpression": "LastPostedBy = :val2",
    "ExpressionAttributeValues": {
        ":val1": {"S": "alice@example.com"},
        ":val2": {"S": "fred@example.com"}
    },
    "ReturnValues": "ALL_NEW"
}
```

Sample Response

HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>

```json
{
    "Attributes": {
        "LastPostedBy": {
            "S": "alice@example.com"
        },
        "ForumName": {
            "S": "Amazon DynamoDB"
        },
        "LastPostDateTime": {
            "S": "20130320010350"
        },
        "Tags": {
            "SS": ["Update","Multiple Items","HelpMe"]
        },
        "Subject": {
            "S": "Maximum number of items?"
        },
        "Views": {
            "N": "5"
        },
        "Message": {
            "S": "I want to put 10 million data items to an Amazon DynamoDB table. Is there an upper limit?"
        }
    }
}
```
Atomic Counter

The following example increments the Replies attribute in the Thread table whenever someone posts a reply. Because ReturnValues is set to NONE, no output appears in the response payload.

Sample Request

```
POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.UpdateItem

{
  "TableName": "Thread",
  "Key": {
    "ForumName": {
      "S": "Amazon DynamoDB"
    },
    "Subject": {
      "S": "A question about updates"
    }
  },
  "UpdateExpression": "set Replies = Replies + :num",
  "ExpressionAttributeValues": {
    ":num": {"N": "1"}
  },
  "ReturnValues" : "NONE"
}
```

Sample Response

```
HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>

{
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
UpdateTable

Service: Amazon DynamoDB

Modifies the provisioned throughput settings, global secondary indexes, or DynamoDB Streams settings for a given table.

You can only perform one of the following operations at once:

- Modify the provisioned throughput settings of the table.
- Enable or disable Streams on the table.
- Remove a global secondary index from the table.
- Create a new global secondary index on the table. Once the index begins backfilling, you can use UpdateTable to perform other operations.

UpdateTable is an asynchronous operation; while it is executing, the table status changes from ACTIVE to UPDATING. While it is UPDATING, you cannot issue another UpdateTable request. When the table returns to the ACTIVE state, the UpdateTable operation is complete.

Request Syntax

```
{
    "AttributeDefinitions": [
        {
            "AttributeName": "string",
            "AttributeType": "string"
        }
    ],
    "GlobalSecondaryIndexUpdates": [
        {
            "Create": {
                "IndexName": "string",
                "KeySchema": [
                    {
                        "AttributeName": "string",
                        "KeyType": "string"
                    }
                ],
                "Projection": {
                    "NonKeyAttributes": [ "string" ],
                    "ProjectionType": "string"
                },
                "ProvisionedThroughput": {
                    "ReadCapacityUnits": number,
                    "WriteCapacityUnits": number
                }
            }
        }
    ],
    "ProvisionedThroughput": {
        "ReadCapacityUnits": number,
        "WriteCapacityUnits": number
    }
}
```
"WriteCapacityUnits": number
},
"StreamSpecification": {
    "StreamEnabled": boolean,
    "StreamViewType": "string"
},
"TableName": "string"

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**TableName (p. 145)**

The name of the table to be updated.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: Yes

**AttributeDefinitions (p. 145)**

An array of attributes that describe the key schema for the table and indexes. If you are adding a new global secondary index to the table, **AttributeDefinitions** must include the key element(s) of the new index.

Type: Array of **AttributeDefinition (p. 234)** objects

Required: No

**GlobalSecondaryIndexUpdates (p. 145)**

An array of one or more global secondary indexes for the table. For each index in the array, you can request one action:
- **Create** - add a new global secondary index to the table.
- **Update** - modify the provisioned throughput settings of an existing global secondary index.
- **Delete** - remove a global secondary index from the table.

For more information, see Managing Global Secondary Indexes in the *Amazon DynamoDB Developer Guide*.

Type: Array of **GlobalSecondaryIndexUpdate (p. 269)** objects

Required: No

**ProvisionedThroughput (p. 145)**

The new provisioned throughput settings for the specified table or index.

Type: **ProvisionedThroughput (p. 285)** object

Required: No

**StreamSpecification (p. 145)**

Represents the DynamoDB Streams configuration for the table.
Note
You will receive a ResourceInUseException if you attempt to enable a stream on a table that already has a stream, or if you attempt to disable a stream on a table which does not have a stream.

Type: StreamSpecification (p. 296) object

Required: No

Response Syntax

```json
{
    "TableDescription": {
        "AttributeDefinitions": [ {
            "AttributeName": "string",
            "AttributeType": "string"
        } ],
        "CreationDateTime": number,
        "GlobalSecondaryIndexes": [ {
            "Backfilling": boolean,
            "IndexArn": "string",
            "IndexName": "string",
            "IndexSizeBytes": number,
            "IndexStatus": "string",
            "ItemCount": number,
            "KeySchema": [ {
                "AttributeName": "string",
                "KeyType": "string"
            } ],
            "Projection": { "NonKeyAttributes": [ "string" ], "ProjectionType": "string" },
            "ProvisionedThroughput": { "LastDecreaseDateTime": number, "LastIncreaseDateTime": number, "NumberOfDecreasesToday": number, "ReadCapacityUnits": number, "WriteCapacityUnits": number }
        } ],
        "ItemCount": number,
        "KeySchema": [ { "AttributeName": "string", "KeyType": "string" } ],
        "LatestStreamArn": "string",
        "LatestStreamLabel": "string",
        "LocalSecondaryIndexes": [ { "IndexArn": "string", "IndexName": "string", "IndexSizeBytes": number, "ItemCount": number, "KeySchema": ["string", "string"] } ]
    }
}
```
{
    "AttributeName": "string",
    "KeyType": "string"
},
"Projection": {
    "NonKeyAttributes": [ "string" ],
    "ProjectionType": "string"
},
"ProvisionedThroughput": {
    "LastDecreaseDateTime": number,
    "LastIncreaseDateTime": number,
    "NumberOfDecreasesToday": number,
    "ReadCapacityUnits": number,
    "WriteCapacityUnits": number
},
"RestoreSummary": {
    "RestoreDateTime": number,
    "RestoreInProgress": boolean,
    "SourceBackupArn": "string",
    "SourceTableArn": "string"
},
"StreamSpecification": {
    "StreamEnabled": boolean,
    "StreamViewType": "string"
},
"TableArn": "string",
"TableId": "string",
"TableName": "string",
"TableSizeBytes": number,
"TableStatus": "string"
}
Up to 10 simultaneous table operations are allowed per account. These operations include CreateTable, UpdateTable, DeleteTable, UpdateTimeToLive, and RestoreTableFromBackup.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

HTTP Status Code: 400

ResourceInUseException

The operation conflicts with the resource's availability. For example, you attempted to recreate an existing table, or tried to delete a table currently in the CREATING state.

HTTP Status Code: 400

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

Example

Modify Provisioned Write Throughput

This example changes both the provisioned read and write throughput of the Thread table to 10 capacity units.

Sample Request

POST / HTTP/1.1
Host: dynamodb.<region>.<domain>;
Accept-Encoding: identity
Content-Length: <PayloadSizeBytes>
User-Agent: <UserAgentString>
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
Signature=<Signature>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDB_20120810.UpdateTable

{
  "TableName": "Thread",
  "ProvisionedThroughput": {
    "ReadCapacityUnits": 10,
    "WriteCapacityUnits": 10
  }
}

Sample Response
HTTP/1.1 200 OK
x-amzn-RequestId: <RequestId>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>
{
  "TableDescription": {
    "TableArn": "arn:aws:dynamodb:us-west-2:123456789012:table/Thread",
    "AttributeDefinitions": [
      {
        "AttributeName": "ForumName",
        "AttributeType": "S"
      },
      {
        "AttributeName": "LastPostDateTime",
        "AttributeType": "S"
      },
      {
        "AttributeName": "Subject",
        "AttributeType": "S"
      }
    ],
    "CreationDateTime": 1.363801528686E9,
    "ItemCount": 0,
    "KeySchema": [
      {
        "AttributeName": "ForumName",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "Subject",
        "KeyType": "RANGE"
      }
    ],
    "LocalSecondaryIndexes": [
      {
        "IndexName": "LastPostIndex",
        "IndexSizeBytes": 0,
        "ItemCount": 0,
        "KeySchema": [
          {
            "AttributeName": "ForumName",
            "KeyType": "HASH"
          },
          {
            "AttributeName": "LastPostDateTime",
            "KeyType": "RANGE"
          }
        ],
        "Projection": {
          "ProjectionType": "KEYS_ONLY"
        }
      }
    ],
    "ProvisionedThroughput": {
      "LastIncreaseDateTime": 1.363801701282E9,
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 5,
      "WriteCapacityUnits": 5
    },
    "TableName": "Thread",
    "TableSizeBytes": 0,
    "TableStatus": "UPDATING"
  }
}
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ToUpdateTimeToLive

Service: Amazon DynamoDB

The UpdateToTimeToLive method will enable or disable TTL for the specified table. A successful UpdateToTimeToLive call returns the current TimeToLiveSpecification; it may take up to one hour for the change to fully process. Any additional UpdateToTimeToLive calls for the same table during this one hour duration result in a ValidationException.

TTL compares the current time in epoch time format to the time stored in the TTL attribute of an item. If the epoch time value stored in the attribute is less than the current time, the item is marked as expired and subsequently deleted.

- **Note**
  The epoch time format is the number of seconds elapsed since 12:00:00 AM January 1st, 1970 UTC.

DynamoDB deletes expired items on a best-effort basis to ensure availability of throughput for other data operations.

- **Important**
  DynamoDB typically deletes expired items within two days of expiration. The exact duration within which an item gets deleted after expiration is specific to the nature of the workload. Items that have expired and not been deleted will still show up in reads, queries, and scans.

As items are deleted, they are removed from any Local Secondary Index and Global Secondary Index immediately in the same eventually consistent way as a standard delete operation.

For more information, see Time To Live in the Amazon DynamoDB Developer Guide.

**Request Syntax**

```json
{
    "TableName": "string",
    "TimeToLiveSpecification": {
        "AttributeName": "string",
        "Enabled": boolean
    }
}
```

**Request Parameters**

The request accepts the following data in JSON format.

- **Note**
  In the following list, the required parameters are described first.

**TableName (p. 152)**

  The name of the table to be configured.

  Type: String


  Pattern: [a-zA-Z0-9_.-]+

  Required: Yes

**TimeToLiveSpecification (p. 152)**

  Represents the settings used to enable or disable Time to Live for the specified table.
Type: TimeToLiveSpecification (p. 304) object

Required: Yes

Response Syntax

```
{
   "TimeToLiveSpecification": {
      "AttributeName": "string",
      "Enabled": boolean
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

TimeToLiveSpecification (p. 153)

Represents the output of an UpdateTimeToLive operation.

Type: TimeToLiveSpecification (p. 304) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

LimitExceededException

Up to 50 CreateBackup operations are allowed per second, per account. There is no limit to the number of daily on-demand backups that can be taken.

Up to 10 simultaneous table operations are allowed per account. These operations include CreateTable, UpdateTable, DeleteTable, UpdateTimeToLive, and RestoreTableFromBackup.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

For tables with secondary indexes, only one of those tables can be in the CREATING state at any point in time. Do not attempt to create more than one such table simultaneously.

The total limit of tables in the ACTIVE state is 250.

HTTP Status Code: 400

ResourceInUseException

The operation conflicts with the resource's availability. For example, you attempted to recreate an existing table, or tried to delete a table currently in the CREATING state.
HTTP Status Code: 400

ResourceNotFoundException

The operation tried to access a nonexistent table or index. The resource might not be specified correctly, or its status might not be ACTIVE.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

Amazon DynamoDB Accelerator

The following actions are supported by Amazon DynamoDB Accelerator:

- CreateCluster (p. 156)
- CreateParameterGroup (p. 162)
- CreateSubnetGroup (p. 164)
- DecreaseReplicationFactor (p. 166)
- DeleteCluster (p. 169)
- DeleteParameterGroup (p. 172)
- DeleteSubnetGroup (p. 174)
- DescribeClusters (p. 176)
- DescribeDefaultParameters (p. 179)
- DescribeEvents (p. 181)
- DescribeParameterGroups (p. 184)
- DescribeParameters (p. 187)
- DescribeSubnetGroups (p. 190)
- IncreaseReplicationFactor (p. 192)
- ListTags (p. 195)
- RebootNode (p. 197)
- TagResource (p. 200)
- UntagResource (p. 203)
- UpdateCluster (p. 205)
- UpdateParameterGroup (p. 209)
- UpdateSubnetGroup (p. 211)
CreateCluster
Service: Amazon DynamoDB Accelerator

Creates a DAX cluster. All nodes in the cluster run the same DAX caching software.

Request Syntax

```json
{
    "AvailabilityZones": [ "string" ],
    "ClusterName": "string",
    "Description": "string",
    "IamRoleArn": "string",
    "NodeType": "string",
    "NotificationTopicArn": "string",
    "ParameterGroupName": "string",
    "PreferredMaintenanceWindow": "string",
    "ReplicationFactor": number,
    "SecurityGroupIds": [ "string" ],
    "SubnetGroupName": "string",
    "Tags": [ {
        "Key": "string",
        "Value": "string"
    } ],
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**

In the following list, the required parameters are described first.

**ClusterName (p. 156)**

The cluster identifier. This parameter is stored as a lowercase string.

**Constraints:**

- A name must contain from 1 to 20 alphanumeric characters or hyphens.
- The first character must be a letter.
- A name cannot end with a hyphen or contain two consecutive hyphens.

Type: String

Required: Yes

**IamRoleArn (p. 156)**

A valid Amazon Resource Name (ARN) that identifies an IAM role. At runtime, DAX will assume this role and use the role's permissions to access DynamoDB on your behalf.

Type: String

Required: Yes

**NodeType (p. 156)**

The compute and memory capacity of the nodes in the cluster.

Type: String
Required: Yes

**ReplicationFactor (p. 156)**

The number of nodes in the DAX cluster. A replication factor of 1 will create a single-node cluster, without any read replicas. For additional fault tolerance, you can create a multiple node cluster with one or more read replicas. To do this, set `ReplicationFactor` to 2 or more.

*Note*
AWS recommends that you have at least two read replicas per cluster.

Type: Integer
Required: Yes

**AvailabilityZones (p. 156)**

The Availability Zones (AZs) in which the cluster nodes will be created. All nodes belonging to the cluster are placed in these Availability Zones. Use this parameter if you want to distribute the nodes across multiple AZs.

Type: Array of strings
Required: No

**Description (p. 156)**

A description of the cluster.

Type: String
Required: No

**NotificationTopicArn (p. 156)**

The Amazon Resource Name (ARN) of the Amazon SNS topic to which notifications will be sent.

*Note*
The Amazon SNS topic owner must be same as the DAX cluster owner.

Type: String
Required: No

**ParameterGroupName (p. 156)**

The parameter group to be associated with the DAX cluster.

Type: String
Required: No

**PreferredMaintenanceWindow (p. 156)**

Specifies the weekly time range during which maintenance on the DAX cluster is performed. It is specified as a range in the format ddd:hh24:mi-ddd:hh24:mi (24H Clock UTC). The minimum maintenance window is a 60 minute period. Valid values for ddd are:

- sun
- mon
- tue
- wed
- thu
- fri
- sat
Example: sun:05:00–sun:09:00

Note
If you don't specify a preferred maintenance window when you create or modify a cache cluster, DAX assigns a 60-minute maintenance window on a randomly selected day of the week.

Type: String
Required: No

SecurityGroupIds (p. 156)
A list of security group IDs to be assigned to each node in the DAX cluster. (Each of the security group ID is system-generated.)

If this parameter is not specified, DAX assigns the default VPC security group to each node.

Type: Array of strings
Required: No

SubnetGroupName (p. 156)
The name of the subnet group to be used for the replication group.

Important
DAX clusters can only run in an Amazon VPC environment. All of the subnets that you specify in a subnet group must exist in the same VPC.

Type: String
Required: No

Tags (p. 156)
A set of tags to associate with the DAX cluster.

Type: Array of Tag (p. 325) objects
Required: No

Response Syntax

```json
{
  "Cluster": {
    "ActiveNodes": number,
    "ClusterArn": "string",
    "ClusterDiscoveryEndpoint": {
      "Address": "string",
      "Port": number
    },
    "ClusterName": "string",
    "Description": "string",
    "IamRoleArn": "string",
    "NodeIdsToRemove": [ "string" ],
    "Nodes": [ {
      "AvailabilityZone": "string",
      "Endpoint": { "Address": "string", "Port": number }
    }, "NodeCreateTime": number,
```
"NodeId": "string",
"NodeStatus": "string",
"ParameterGroupStatus": "string"
}
],
"NodeType": "string",
"NotificationConfiguration": {
  "TopicArn": "string",
  "TopicStatus": "string"
},
"ParameterGroup": {
  "NodeIdsToReboot": [ "string" ],
  "ParameterApplyStatus": "string",
  "ParameterGroupName": "string"
},
"PreferredMaintenanceWindow": "string",
"SecurityGroups": [ {
  "SecurityGroupIdentifier": "string",
  "Status": "string"
} ],
"Status": "string",
"SubnetGroup": "string",
"TotalNodes": number
}

## Response Elements

If the action is successful, the service sends back an HTTP 200 response. The following data is returned in JSON format by the service.

**Cluster (p. 158)**

A description of the DAX cluster that you have created.

Type: [Cluster](p. 308) object

## Errors

For information about the errors that are common to all actions, see [Common Errors](p. 341).

**ClusterAlreadyExistsFault**

You already have a DAX cluster with the given identifier.

HTTP Status Code: 400

**ClusterQuotaForCustomerExceededFault**

You have attempted to exceed the maximum number of DAX clusters for your AWS account.

HTTP Status Code: 400

**InsufficientClusterCapacityFault**

There are not enough system resources to create the cluster you requested (or to resize an already-existing cluster).

HTTP Status Code: 400
InvalidClusterStateException
   The requested DAX cluster is not in the available state.
   HTTP Status Code: 400

InvalidParameterCombinationException
   Two or more incompatible parameters were specified.
   HTTP Status Code: 400

InvalidParameterGroupStateException
   One or more parameters in a parameter group are in an invalid state.
   HTTP Status Code: 400

InvalidParameterValueException
   The value for a parameter is invalid.
   HTTP Status Code: 400

InvalidVPCNetworkStateException
   The VPC network is in an invalid state.
   HTTP Status Code: 400

NodeQuotaForClusterExceededFault
   You have attempted to exceed the maximum number of nodes for a DAX cluster.
   HTTP Status Code: 400

NodeQuotaForCustomerExceededFault
   You have attempted to exceed the maximum number of nodes for your AWS account.
   HTTP Status Code: 400

ParameterGroupNotFoundException
   The specified parameter group does not exist.
   HTTP Status Code: 400

SubnetGroupNotFoundException
   The requested subnet group name does not refer to an existing subnet group.
   HTTP Status Code: 400

TagQuotaPerResourceExceeded
   You have exceeded the maximum number of tags for this DAX cluster.
   HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
CreateParameterGroup
Service: Amazon DynamoDB Accelerator

Creates a new parameter group. A parameter group is a collection of parameters that you apply to all of the nodes in a DAX cluster.

Request Syntax

```json
{
    "Description": "string",
    "ParameterGroupName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ParameterGroupName (p. 162)**

The name of the parameter group to apply to all of the clusters in this replication group.

Type: String
Required: Yes

**Description (p. 162)**

A description of the parameter group.

Type: String
Required: No

Response Syntax

```json
{
    "ParameterGroup": {
        "Description": "string",
        "ParameterGroupName": "string"
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**ParameterGroup (p. 162)**

Represents the output of a CreateParameterGroup action.

Type: ParameterGroup (p. 319) object
Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterGroupStateFault

One or more parameters in a parameter group are in an invalid state.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

ParameterGroupAlreadyExistsFault

The specified parameter group already exists.

HTTP Status Code: 400

ParameterGroupQuotaExceededFault

You have attempted to exceed the maximum number of parameter groups.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateSubnetGroup
Service: Amazon DynamoDB Accelerator

Creates a new subnet group.

Request Syntax

```
{
   "Description": "string",
   "SubnetGroupName": "string",
   "SubnetIds": [ "string" ]
}
```

Request Parameters

The request accepts the following data in JSON format.

Note
In the following list, the required parameters are described first.

SubnetGroupName (p. 164)

A name for the subnet group. This value is stored as a lowercase string.

Type: String

Required: Yes

SubnetIds (p. 164)

A list of VPC subnet IDs for the subnet group.

Type: Array of strings

Required: Yes

Description (p. 164)

A description for the subnet group

Type: String

Required: No

Response Syntax

```
{
   "SubnetGroup": {
      "Description": "string",
      "SubnetGroupName": "string",
      "Subnets": [
      {
         "SubnetAvailabilityZone": "string",
         "SubnetIdentifier": "string"
      }
      ],
      "VpcId": "string"
   }
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.
The following data is returned in JSON format by the service.

SubnetGroup (p. 164)

Represents the output of a CreateSubnetGroup operation.
Type: SubnetGroup (p. 324) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InvalidSubnet

An invalid subnet identifier was specified.

HTTP Status Code: 400

SubnetGroupAlreadyExistsFault

The specified subnet group already exists.

HTTP Status Code: 400

SubnetGroupQuotaExceededFault

The request cannot be processed because it would exceed the allowed number of subnets in a subnet group.

HTTP Status Code: 400

SubnetQuotaExceededFault

The request cannot be processed because it would exceed the allowed number of subnets in a subnet group.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
**DecreaseReplicationFactor**

Service: Amazon DynamoDB Accelerator

Removes one or more nodes from a DAX cluster.

**Note**
You cannot use `DecreaseReplicationFactor` to remove the last node in a DAX cluster. If you need to do this, use `DeleteCluster` instead.

**Request Syntax**

```json
{
   "AvailabilityZones": [ "string" ],
   "ClusterName": "string",
   "NewReplicationFactor": number,
   "NodeIdsToRemove": [ "string" ]
}
```

**Request Parameters**

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ClusterName (p. 166)**

The name of the DAX cluster from which you want to remove nodes.

Type: String

Required: Yes

**NewReplicationFactor (p. 166)**

The new number of nodes for the DAX cluster.

Type: Integer

Required: Yes

**AvailabilityZones (p. 166)**

The Availability Zone(s) from which to remove nodes.

Type: Array of strings

Required: No

**NodeIdsToRemove (p. 166)**

The unique identifiers of the nodes to be removed from the cluster.

Type: Array of strings

Required: No

**Response Syntax**

```json
{
}
```
"Cluster": {
  "ActiveNodes": number,
  "ClusterArn": "string",
  "ClusterDiscoveryEndpoint": {
    "Address": "string",
    "Port": number
  },
  "ClusterName": "string",
  "Description": "string",
  "IamRoleArn": "string",
  "NodeIdToRemove": [ "string" ],
  "Nodes": [
    {
      "AvailabilityZone": "string",
      "Endpoint": {
        "Address": "string",
        "Port": number
      },
      "NodeCreateTime": number,
      "NodeId": "string",
      "NodeStatus": "string",
      "ParameterGroupStatus": "string"
    }
  ],
  "NodeType": "string",
  "NotificationConfiguration": {
    "TopicArn": "string",
    "TopicStatus": "string"
  },
  "ParameterGroup": {
    "NodeIdsToReboot": [ "string" ],
    "ParameterApplyStatus": "string",
    "ParameterGroupName": "string"
  },
  "PreferredMaintenanceWindow": "string",
  "SecurityGroups": [
    {
      "SecurityGroupIdentifier": "string",
      "Status": "string"
    }
  ],
  "Status": "string",
  "SubnetGroup": "string",
  "TotalNodes": number
}

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**Cluster (p. 166)**

A description of the DAX cluster, after you have decreased its replication factor.

Type: Cluster (p. 308) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).
ClusterNotFoundFault

The requested cluster ID does not refer to an existing DAX cluster.

HTTP Status Code: 400

InvalidClusterStateFault

The requested DAX cluster is not in the available state.

HTTP Status Code: 400

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

NodeNotFoundFault

None of the nodes in the cluster have the given node ID.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteCluster
Service: Amazon DynamoDB Accelerator

Deletes a previously provisioned DAX cluster. DeleteCluster deletes all associated nodes, node endpoints and the DAX cluster itself. When you receive a successful response from this action, DAX immediately begins deleting the cluster; you cannot cancel or revert this action.

Request Syntax

```
{
  "ClusterName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ClusterName (p. 169)**

The name of the cluster to be deleted.

Type: String

Required: Yes

Response Syntax

```
{
  "Cluster": {
    "ActiveNodes": number,
    "ClusterArn": "string",
    "ClusterDiscoveryEndpoint": {
      "Address": "string",
      "Port": number
    },
    "ClusterName": "string",
    "Description": "string",
    "IamRoleArn": "string",
    "NodeIdsToRemove": ["string"],
    "Nodes": [
      {
        "AvailabilityZone": "string",
        "Endpoint": {
          "Address": "string",
          "Port": number
        },
        "NodeCreateTime": number,
        "NodeId": "string",
        "NodeStatus": "string",
        "ParameterGroupStatus": "string"
      }
    ],
    "NodeType": "string",
    "NotificationConfiguration": {
      "TopicArn": "string",
      "TopicStatus": "string"
    }
  }
}
```
### Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**Cluster (p. 169)**

A description of the DAX cluster that is being deleted.

Type: Cluster (p. 308) object

### Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

**ClusterNotFoundFault**

The requested cluster ID does not refer to an existing DAX cluster.

HTTP Status Code: 400

**InvalidClusterStateFault**

The requested DAX cluster is not in the available state.

HTTP Status Code: 400

**InvalidParameterCombinationException**

Two or more incompatible parameters were specified.

HTTP Status Code: 400

**InvalidParameterValueException**

The value for a parameter is invalid.

HTTP Status Code: 400

### See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:
• AWS Command Line Interface
• AWS SDK for .NET
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
DeleteParameterGroup
Service: Amazon DynamoDB Accelerator

Deletes the specified parameter group. You cannot delete a parameter group if it is associated with any DAX clusters.

Request Syntax

```json
{
   "ParameterGroupName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ParameterGroupName (p. 172)**

The name of the parameter group to delete.

Type: String

Required: Yes

Response Syntax

```json
{
   "DeletionMessage": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**DeletionMessage (p. 172)**

A user-specified message for this action (i.e., a reason for deleting the parameter group).

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

**InvalidParameterCombinationException**

Two or more incompatible parameters were specified.

HTTP Status Code: 400
InvalidParameterGroupStateFault

One or more parameters in a parameter group are in an invalid state.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

ParameterGroupNotFoundFault

The specified parameter group does not exist.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteSubnetGroup

Service: Amazon DynamoDB Accelerator

Deletes a subnet group.

**Note**
You cannot delete a subnet group if it is associated with any DAX clusters.

**Request Syntax**

```
{
  "SubnetGroupName": "string"
}
```

**Request Parameters**

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**SubnetGroupName (p. 174)**

The name of the subnet group to delete.

Type: String

Required: Yes

**Response Syntax**

```
{
  "DeletionMessage": "string"
}
```

**Response Elements**

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**DeletionMessage (p. 174)**

A user-specified message for this action (i.e., a reason for deleting the subnet group).

Type: String

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 341).

**SubnetGroupInUseFault**

The specified subnet group is currently in use.
HTTP Status Code: 400

SubnetGroupNotFoundFault

The requested subnet group name does not refer to an existing subnet group.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeClusters

Service: Amazon DynamoDB Accelerator

Returns information about all provisioned DAX clusters if no cluster identifier is specified, or about a specific DAX cluster if a cluster identifier is supplied.

If the cluster is in the CREATING state, only cluster level information will be displayed until all of the nodes are successfully provisioned.

If the cluster is in the DELETING state, only cluster level information will be displayed.

If nodes are currently being added to the DAX cluster, node endpoint information and creation time for the additional nodes will not be displayed until they are completely provisioned. When the DAX cluster state is available, the cluster is ready for use.

If nodes are currently being removed from the DAX cluster, no endpoint information for the removed nodes is displayed.

Request Syntax

```
{
   "ClusterNames": [ "string" ],
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**

In the following list, the required parameters are described first.

**ClusterNames (p. 176)**

The names of the DAX clusters being described.

Type: Array of strings

Required: No

**MaxResults (p. 176)**

The maximum number of results to include in the response. If more results exist than the specified MaxResults value, a token is included in the response so that the remaining results can be retrieved.

The value for MaxResults must be between 20 and 100.

Type: Integer

Required: No

**NextToken (p. 176)**

An optional token returned from a prior request. Use this token for pagination of results from this action. If this parameter is specified, the response includes only results beyond the token, up to the value specified by MaxResults.
Response Syntax

```json
{
    "Clusters": [
        {
            "ActiveNodes": number,
            "ClusterArn": "string",
            "ClusterDiscoveryEndpoint": {
                "Address": "string",
                "Port": number
            },
            "ClusterName": "string",
            "Description": "string",
            "IamRoleArn": "string",
            "NodeIdsToRemove": [ "string" ],
            "Nodes": [
                {
                    "availabilityZone": "string",
                    "Endpoint": {
                        "Address": "string",
                        "Port": number
                    },
                    "NodeCreateTime": number,
                    "NodeId": "string",
                    "NodeStatus": "string",
                    "ParameterGroupStatus": "string"
                }
            ],
            "NodeType": "string",
            "NotificationConfiguration": {
                "TopicArn": "string",
                "TopicStatus": "string"
            },
            "ParameterGroup": {
                "NodeIdsToReboot": [ "string" ],
                "ParameterApplyStatus": "string",
                "ParameterGroupName": "string"
            },
            "PreferredMaintenanceWindow": "string",
            "SecurityGroups": [
                {
                    "SecurityGroupIdentifier": "string",
                    "Status": "string"
                }
            ],
            "Status": "string",
            "SubnetGroup": "string",
            "TotalNodes": number
        }
    ],
    "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.
Clusters (p. 177)

The descriptions of your DAX clusters, in response to a DescribeClusters request.

Type: Array of Cluster (p. 308) objects

NextToken (p. 177)

Provides an identifier to allow retrieval of paginated results.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

ClusterNotFoundException

The requested cluster ID does not refer to an existing DAX cluster.

HTTP Status Code: 400

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeDefaultParameters
Service: Amazon DynamoDB Accelerator

Returns the default system parameter information for the DAX caching software.

Request Syntax

```json
{
   "MaxResults": number,
   "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**MaxResults (p. 179)**

The maximum number of results to include in the response. If more results exist than the specified `MaxResults` value, a token is included in the response so that the remaining results can be retrieved.

The value for `MaxResults` must be between 20 and 100.

Type: Integer

Required: No

**NextToken (p. 179)**

An optional token returned from a prior request. Use this token for pagination of results from this action. If this parameter is specified, the response includes only results beyond the token, up to the value specified by `MaxResults`.

Type: String

Required: No

Response Syntax

```json
{
   "NextToken": "string",
   "Parameters": [
   {
      "AllowedValues": "string",
      "ChangeType": "string",
      "DataType": "string",
      "Description": "string",
      "IsModifiable": "string",
      "NodeTypeSpecificValues": [
      {
         "NodeType": "string",
         "Value": "string"
      }
      ]
   }
]
```

179
"ParameterName": "string",
"ParameterType": "string",
"ParameterValue": "string",
"Source": "string"
}
]
}

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken (p. 179)

Provides an identifier to allow retrieval of paginated results.

Type: String

Parameters (p. 179)

A list of parameters. Each element in the list represents one parameter.

Type: Array of Parameter (p. 317) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeEvents
Service: Amazon DynamoDB Accelerator

Returns events related to DAX clusters and parameter groups. You can obtain events specific to a particular DAX cluster or parameter group by providing the name as a parameter.

By default, only the events occurring within the last hour are returned; however, you can retrieve up to 14 days' worth of events if necessary.

Request Syntax

```json
{
  "Duration": number,
  "EndTime": number,
  "MaxResults": number,
  "NextToken": "string",
  "SourceName": "string",
  "SourceType": "string",
  "StartTime": number
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**Duration (p. 181)**

The number of minutes' worth of events to retrieve.

Type: Integer

Required: No

**EndTime (p. 181)**

The end of the time interval for which to retrieve events, specified in ISO 8601 format.

Type: Timestamp

Required: No

**MaxResults (p. 181)**

The maximum number of results to include in the response. If more results exist than the specified MaxResults value, a token is included in the response so that the remaining results can be retrieved.

The value for MaxResults must be between 20 and 100.

Type: Integer

Required: No

**NextToken (p. 181)**

An optional token returned from a prior request. Use this token for pagination of results from this action. If this parameter is specified, the response includes only results beyond the token, up to the value specified by MaxResults.
DescribeEvents

SourceName (p. 181)

The identifier of the event source for which events will be returned. If not specified, then all sources are included in the response.

Type: String
Required: No

SourceType (p. 181)

The event source to retrieve events for. If no value is specified, all events are returned.

Type: String
Valid Values: CLUSTER | PARAMETER_GROUP | SUBNET_GROUP
Required: No

StartTime (p. 181)

The beginning of the time interval to retrieve events for, specified in ISO 8601 format.

Type: Timestamp
Required: No

Response Syntax

```
{
  "Events": [
    {
      "Date": number,
      "Message": "string",
      "SourceName": "string",
      "SourceType": "string"
    }
  ],
  "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Events (p. 182)

An array of events. Each element in the array represents one event.

Type: Array of Event (p. 312) objects

NextToken (p. 182)

Provides an identifier to allow retrieval of paginated results.

Type: String
Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeParameterGroups
Service: Amazon DynamoDB Accelerator

Returns a list of parameter group descriptions. If a parameter group name is specified, the list will contain only the descriptions for that group.

Request Syntax

```json
{
   "MaxResults": number,
   "NextToken": "string",
   "ParameterGroupNames": [ "string" ]
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**MaxResults (p. 184)**

The maximum number of results to include in the response. If more results exist than the specified `MaxResults` value, a token is included in the response so that the remaining results can be retrieved.

The value for `MaxResults` must be between 20 and 100.

Type: Integer

Required: No

**NextToken (p. 184)**

An optional token returned from a prior request. Use this token for pagination of results from this action. If this parameter is specified, the response includes only results beyond the token, up to the value specified by `MaxResults`.

Type: String

Required: No

**ParameterGroupNames (p. 184)**

The names of the parameter groups.

Type: Array of strings

Required: No

Response Syntax

```json
{
   "NextToken": "string",
   "ParameterGroups": [
   {
      "Description": "string",
      "ParameterGroupName": "string"
   }
   ]
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response. The following data is returned in JSON format by the service.

NextToken (p. 184)

Provides an identifier to allow retrieval of paginated results.

Type: String

ParameterGroups (p. 184)

An array of parameter groups. Each element in the array represents one parameter group.

Type: Array of ParameterGroup (p. 319) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

ParameterGroupNotFoundFault

The specified parameter group does not exist.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeParameters
Service: Amazon DynamoDB Accelerator

Returns the detailed parameter list for a particular parameter group.

Request Syntax

```
{
  "MaxResults": number,
  "NextToken": "string",
  "ParameterGroupName": "string",
  "Source": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ParameterGroupName (p. 187)**

The name of the parameter group.

Type: String

Required: Yes

**MaxResults (p. 187)**

The maximum number of results to include in the response. If more results exist than the specified `MaxResults` value, a token is included in the response so that the remaining results can be retrieved.

The value for `MaxResults` must be between 20 and 100.

Type: Integer

Required: No

**NextToken (p. 187)**

An optional token returned from a prior request. Use this token for pagination of results from this action. If this parameter is specified, the response includes only results beyond the token, up to the value specified by `MaxResults`.

Type: String

Required: No

**Source (p. 187)**

How the parameter is defined. For example, `system` denotes a system-defined parameter.

Type: String

Required: No
Response Syntax

```
{
  "NextToken": "string",
  "Parameters": [
    {
      "AllowedValues": "string",
      "ChangeType": "string",
      "DataType": "string",
      "Description": "string",
      "IsModifiable": "string",
      "NodeTypeSpecificValues": [
        {
          "NodeType": "string",
          "Value": "string"
        }
      ],
      "ParameterName": "string",
      "ParameterType": "string",
      "ParameterValue": "string",
      "Source": "string"
    }
  ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**NextToken (p. 188)**

Provides an identifier to allow retrieval of paginated results.

Type: String

**Parameters (p. 188)**

A list of parameters within a parameter group. Each element in the list represents one parameter.

Type: Array of Parameter (p. 317) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

**InvalidParameterCombinationException**

Two or more incompatible parameters were specified.

HTTP Status Code: 400

**InvalidParameterValueException**

The value for a parameter is invalid.

HTTP Status Code: 400

**ParameterGroupNotFoundFault**

The specified parameter group does not exist.
HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeSubnetGroups
Service: Amazon DynamoDB Accelerator

Returns a list of subnet group descriptions. If a subnet group name is specified, the list will contain only the description of that group.

Request Syntax

```json
{
  "MaxResults": number,
  "NextToken": "string",
  "SubnetGroupNames": [ "string" ]
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

MaxResults (p. 190)

The maximum number of results to include in the response. If more results exist than the specified `MaxResults` value, a token is included in the response so that the remaining results can be retrieved.

The value for `MaxResults` must be between 20 and 100.

Type: Integer

Required: No

NextToken (p. 190)

An optional token returned from a prior request. Use this token for pagination of results from this action. If this parameter is specified, the response includes only results beyond the token, up to the value specified by `MaxResults`.

Type: String

Required: No

SubnetGroupNames (p. 190)

The name of the subnet group.

Type: Array of strings

Required: No

Response Syntax

```json
{
  "NextToken": "string",
  "SubnetGroups": [
      {
        "Description": "string",
        "SubnetGroupName": "string",
      }
  ]
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken (p. 190)

Provides an identifier to allow retrieval of paginated results.

Type: String

SubnetGroups (p. 190)

An array of subnet groups. Each element in the array represents a single subnet group.

Type: Array of SubnetGroup (p. 324) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

SubnetGroupNotFoundFault

The requested subnet group name does not refer to an existing subnet group.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
IncreaseReplicationFactor

Service: Amazon DynamoDB Accelerator

Adds one or more nodes to a DAX cluster.

Request Syntax

```json
{
  "AvailabilityZones": [ "string" ],
  "ClusterName": "string",
  "NewReplicationFactor": number
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**

In the following list, the required parameters are described first.

**ClusterName (p. 192)**

The name of the DAX cluster that will receive additional nodes.

Type: String

Required: Yes

**NewReplicationFactor (p. 192)**

The new number of nodes for the DAX cluster.

Type: Integer

Required: Yes

**AvailabilityZones (p. 192)**

The Availability Zones (AZs) in which the cluster nodes will be created. All nodes belonging to the cluster are placed in these Availability Zones. Use this parameter if you want to distribute the nodes across multiple AZs.

Type: Array of strings

Required: No

Response Syntax

```json
{
  "Cluster": {
    "ActiveNodes": number,
    "ClusterArn": "string",
    "ClusterDiscoveryEndpoint": {
      "Address": "string",
      "Port": number
    },
    "ClusterName": "string",
    "Description": "string",
    "IamRoleArn": "string",
    "NodeIdsToRemove": [ "string" ],
  }
}
```
"Nodes": [  
  {  
    "AvailabilityZone": "string",
    "Endpoint": {  
      "Address": "string",
      "Port": number
    },
    "NodeCreateTime": number,
    "NodeId": "string",
    "NodeStatus": "string",
    "ParameterGroupStatus": "string"
  },
  
  "NodeType": "string",
  "NotificationConfiguration": {  
    "TopicArn": "string",
    "TopicStatus": "string"
  },
  "ParameterGroup": {  
    "NodeIdsToReboot": [ "string" ],
    "ParameterApplyStatus": "string",
    "ParameterGroupName": "string"
  },
  
  "PreferredMaintenanceWindow": "string",
  "SecurityGroups": [  
    {  
      "SecurityGroupIdentifier": "string",
      "Status": "string"
    }
  ],
  "Status": "string",
  "SubnetGroup": "string",
  "TotalNodes": number
  
}

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Cluster (p. 192)

A description of the DAX cluster. with its new replication factor.

Type: Cluster (p. 308) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

ClusterNotFoudnFault

The requested cluster ID does not refer to an existing DAX cluster.

HTTP Status Code: 400

InsufficientClusterCapacityFault

There are not enough system resources to create the cluster you requested (or to resize an already-existing cluster).
IncreaseReplicationFactor

HTTP Status Code: 400

InvalidClusterStateException

The requested DAX cluster is not in the available state.

HTTP Status Code: 400

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

InvalidVPCNetworkStateException

The VPC network is in an invalid state.

HTTP Status Code: 400

NodeQuotaForClusterExceededFault

You have attempted to exceed the maximum number of nodes for a DAX cluster.

HTTP Status Code: 400

NodeQuotaForCustomerExceededFault

You have attempted to exceed the maximum number of nodes for your AWS account.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
**ListTags**

*Service: Amazon DynamoDB Accelerator*

List all of the tags for a DAX cluster. You can call **ListTags** up to 10 times per second, per account.

**Request Syntax**

```json
{
    "NextToken": "string",
    "ResourceName": "string"
}
```

**Request Parameters**

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ResourceName (p. 195)**

The name of the DAX resource to which the tags belong.

Type: String

Required: Yes

**NextToken (p. 195)**

An optional token returned from a prior request. Use this token for pagination of results from this action. If this parameter is specified, the response includes only results beyond the token.

Type: String

Required: No

**Response Syntax**

```json
{
    "NextToken": "string",
    "Tags": [
        {
            "Key": "string",
            "Value": "string"
        }
    ]
}
```

**Response Elements**

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**NextToken (p. 195)**

If this value is present, there are additional results to be displayed. To retrieve them, call **ListTags** again, with **NextToken** set to this value.
Type: String

**Tags (p. 195)**

A list of tags currently associated with the DAX cluster.

Type: Array of Tag (p. 325) objects

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 341).

**ClusterNotFoundFault**

The requested cluster ID does not refer to an existing DAX cluster.

HTTP Status Code: 400

**InvalidARNFault**

The Amazon Resource Name (ARN) supplied in the request is not valid.

HTTP Status Code: 400

**InvalidClusterStateFault**

The requested DAX cluster is not in the available state.

HTTP Status Code: 400

**InvalidParameterCombinationException**

Two or more incompatible parameters were specified.

HTTP Status Code: 400

**InvalidParameterValueException**

The value for a parameter is invalid.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
RebootNode

Service: Amazon DynamoDB Accelerator

Reboots a single node of a DAX cluster. The reboot action takes place as soon as possible. During the reboot, the node status is set to REBOOTING.

Request Syntax

```json
{
    "ClusterName": "string",
    "NodeId": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**

In the following list, the required parameters are described first.

ClusterName (p. 197)

The name of the DAX cluster containing the node to be rebooted.

Type: String

Required: Yes

NodeId (p. 197)

The system-assigned ID of the node to be rebooted.

Type: String

Required: Yes

Response Syntax

```json
{
    "Cluster": {
        "ActiveNodes": number,
        "ClusterArn": "string",
        "ClusterDiscoveryEndpoint": {
            "Address": "string",
            "Port": number
        },
        "ClusterName": "string",
        "Description": "string",
        "IamRoleArn": "string",
        "NodeIdsToRemove": [ "string" ],
        "Nodes": [
            {
                "AvailabilityZone": "string",
                "Endpoint": {
                    "Address": "string",
                    "Port": number
                },
                "CreateTime": number,
                "NodeId": "string",
                "NodeStatus": "string",
            }
        ],
    }
}
```
"ParameterGroupStatus": "string",

"NodeType": "string",
"NotificationConfiguration": {
    "TopicArn": "string",
    "TopicStatus": "string"
},
"ParameterGroup": {
    "NodeIdsToReboot": [ "string" ],
    "ParameterApplyStatus": "string",
    "ParameterGroupName": "string"
},
"PreferredMaintenanceWindow": "string",
"SecurityGroups": [
    {
        "SecurityGroupIdentifier": "string",
        "Status": "string"
    }
],
"Status": "string",
"SubnetGroup": "string",
"TotalNodes": number
}

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Cluster (p. 197)

A description of the DAX cluster after a node has been rebooted.

Type: Cluster (p. 308) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

ClusterNotFoundFault

The requested cluster ID does not refer to an existing DAX cluster.

HTTP Status Code: 400

InvalidClusterStateException

The requested DAX cluster is not in the available state.

HTTP Status Code: 400

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.
HTTP Status Code: 400

**NodeNotFoundFault**

None of the nodes in the cluster have the given node ID.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
TagResource
Service: Amazon DynamoDB Accelerator

Associates a set of tags with a DAX resource. You can call TagResource up to 5 times per second, per account.

Request Syntax

```
{  
  "ResourceName": "string",
  "Tags": [
    
    {  
      "Key": "string",
      "Value": "string"
    }
  ]
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ResourceName (p. 200)**

The name of the DAX resource to which tags should be added.

Type: String

Required: Yes

**Tags (p. 200)**

The tags to be assigned to the DAX resource.

Type: Array of Tag (p. 325) objects

Required: Yes

Response Syntax

```
{  
  "Tags": [
    
    {  
      "Key": "string",
      "Value": "string"
    }
  ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.
Tags (p. 200)

The list of tags that are associated with the DAX resource.

Type: Array of Tag (p. 325) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

ClusterNotFoundFault

The requested cluster ID does not refer to an existing DAX cluster.

HTTP Status Code: 400

InvalidARNFault

The Amazon Resource Name (ARN) supplied in the request is not valid.

HTTP Status Code: 400

InvalidClusterStateException

The requested DAX cluster is not in the available state.

HTTP Status Code: 400

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

TagQuotaPerResourceExceeded

You have exceeded the maximum number of tags for this DAX cluster.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
UntagResource
Service: Amazon DynamoDB Accelerator

Removes the association of tags from a DAX resource. You can call UnTagResource up to 5 times per second, per account.

Request Syntax

```json
{
  "ResourceName": "string",
  "TagKeys": [ "string" ]
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**

In the following list, the required parameters are described first.

**ResourceName (p. 203)**

The name of the DAX resource from which the tags should be removed.

Type: String

Required: Yes

**TagKeys (p. 203)**

A list of tag keys. If the DAX cluster has any tags with these keys, then the tags are removed from the cluster.

Type: Array of strings

Required: Yes

Response Syntax

```json
{
  "Tags": [
    {
      "Key": "string",
      "Value": "string"
    }
  ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**Tags (p. 203)**

The tag keys that have been removed from the cluster.
Type: Array of Tag (p. 325) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

ClusterNotFoundFault

The requested cluster ID does not refer to an existing DAX cluster.

HTTP Status Code: 400

InvalidARNFault

The Amazon Resource Name (ARN) supplied in the request is not valid.

HTTP Status Code: 400

InvalidClusterStateFault

The requested DAX cluster is not in the available state.

HTTP Status Code: 400

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

TagNotFoundFault

The tag does not exist.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
UpdateCluster
Service: Amazon DynamoDB Accelerator

Modifies the settings for a DAX cluster. You can use this action to change one or more cluster configuration parameters by specifying the parameters and the new values.

Request Syntax

```json
{
   "ClusterName": "string",
   "Description": "string",
   "NotificationTopicArn": "string",
   "NotificationTopicStatus": "string",
   "ParameterGroupName": "string",
   "PreferredMaintenanceWindow": "string",
   "SecurityGroupIds": [ "string" ]
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ClusterName (p. 205)**

The name of the DAX cluster to be modified.

Type: String

Required: Yes

**Description (p. 205)**

A description of the changes being made to the cluster.

Type: String

Required: No

**NotificationTopicArn (p. 205)**

The Amazon Resource Name (ARN) that identifies the topic.

Type: String

Required: No

**NotificationTopicStatus (p. 205)**

The current state of the topic.

Type: String

Required: No

**ParameterGroupName (p. 205)**

The name of a parameter group for this cluster.

Type: String
PreferredMaintenanceWindow (p. 205)

A range of time when maintenance of DAX cluster software will be performed. For example: sun:01:00–sun:09:00. Cluster maintenance normally takes less than 30 minutes, and is performed automatically within the maintenance window.

Type: String

Required: No

SecurityGroupIds (p. 205)

A list of user-specified security group IDs to be assigned to each node in the DAX cluster. If this parameter is not specified, DAX assigns the default VPC security group to each node.

Type: Array of strings

Required: No

Response Syntax

```
{
   "Cluster": {
      "ActiveNodes": number,
      "ClusterArn": "string",
      "ClusterDiscoveryEndpoint": {
         "Address": "string",
         "Port": number
      },
      "ClusterName": "string",
      "Description": "string",
      "IamRoleArn": "string",
      "NodeIdToRemove": [ "string" ],
      "Nodes": [ {
         "AvailabilityZone": "string",
         "Endpoint": { "Address": "string",
                        "Port": number
         },
         "NodeCreateTime": number,
         "NodeId": "string",
         "NodeStatus": "string",
         "ParameterGroupStatus": "string"
      } ],
      "NodeType": "string",
      "NotificationConfiguration": { "TopicArn": "string",
                                    "TopicStatus": "string" },
      "ParameterGroup": { "NodeIdsToReboot": [ "string" ],
                          "ParameterApplyStatus": "string",
                          "ParameterGroupName": "string" },
      "PreferredMaintenanceWindow": "string",
      "SecurityGroups": [ { "SecurityGroupIdentifier": "string",
                           "Status": "string" }
      ]
   }
}```
{
    "Status": "string",
    "SubnetGroup": "string",
    "TotalNodes": number
}

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Cluster (p. 206)

A description of the DAX cluster, after it has been modified.

Type: Cluster (p. 308) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

ClusterNotFoundFault

The requested cluster ID does not refer to an existing DAX cluster.

HTTP Status Code: 400

InvalidClusterStateException

The requested DAX cluster is not in the available state.

HTTP Status Code: 400

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterGroupStateException

One or more parameters in a parameter group are in an invalid state.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

ParameterGroupNotFoundException

The specified parameter group does not exist.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:
• AWS Command Line Interface
• AWS SDK for .NET
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
UpdateParameterGroup
Service: Amazon DynamoDB Accelerator

Modifies the parameters of a parameter group. You can modify up to 20 parameters in a single request by submitting a list parameter name and value pairs.

Request Syntax

```
{
    "ParameterGroupName": "string",
    "ParameterNameValues": [
    {
      "ParameterName": "string",
      "ParameterValue": "string"
    }
    ]
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**

In the following list, the required parameters are described first.

**ParameterGroupName (p. 209)**

The name of the parameter group.

Type: String

Required: Yes

**ParameterNameValues (p. 209)**

An array of name-value pairs for the parameters in the group. Each element in the array represents a single parameter.

Type: Array of ParameterNameValue (p. 321) objects

Required: Yes

Response Syntax

```
{
    "ParameterGroup": {
        "Description": "string",
        "ParameterGroupName": "string"
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.
ParameterGroup (p. 209)

The parameter group that has been modified.

Type: ParameterGroup (p. 319) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InvalidParameterCombinationException

Two or more incompatible parameters were specified.

HTTP Status Code: 400

InvalidParameterGroupStateFault

One or more parameters in a parameter group are in an invalid state.

HTTP Status Code: 400

InvalidParameterValueException

The value for a parameter is invalid.

HTTP Status Code: 400

ParameterGroupNotFoundFault

The specified parameter group does not exist.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
UpdateSubnetGroup
Service: Amazon DynamoDB Accelerator

Modifies an existing subnet group.

Request Syntax

```json
{
    "Description": "string",
    "SubnetGroupName": "string",
    "SubnetIds": [ "string" ]
}
```

Request Parameters

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**SubnetGroupName (p. 211)**

The name of the subnet group.

Type: String

Required: Yes

**Description (p. 211)**

A description of the subnet group.

Type: String

Required: No

**SubnetIds (p. 211)**

A list of subnet IDs in the subnet group.

Type: Array of strings

Required: No

Response Syntax

```json
{
    "SubnetGroup": {
        "Description": "string",
        "SubnetGroupName": "string",
        "Subnets": [
            {
                "SubnetAvailabilityZone": "string",
                "SubnetIdentifier": "string"
            }
        ],
        "VpcId": "string"
    }
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response. The following data is returned in JSON format by the service.

**SubnetGroup** (p. 211)

The subnet group that has been modified.

Type: SubnetGroup (p. 324) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

**InvalidSubnet**

An invalid subnet identifier was specified.

HTTP Status Code: 400

**SubnetGroupNotFoundFault**

The requested subnet group name does not refer to an existing subnet group.

HTTP Status Code: 400

**SubnetInUse**

The requested subnet is being used by another subnet group.

HTTP Status Code: 400

**SubnetQuotaExceededFault**

The request cannot be processed because it would exceed the allowed number of subnets in a subnet group.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
Amazon DynamoDB Streams

The following actions are supported by Amazon DynamoDB Streams:

- DescribeStream (p. 214)
- GetRecords (p. 218)
- GetShardIterator (p. 223)
- ListStreams (p. 227)
DescribeStream

Service: Amazon DynamoDB Streams

Returns information about a stream, including the current status of the stream, its Amazon Resource Name (ARN), the composition of its shards, and its corresponding DynamoDB table.

**Note**
You can call DescribeStream at a maximum rate of 10 times per second.

Each shard in the stream has a SequenceNumberRange associated with it. If the SequenceNumberRange has a StartingSequenceNumber but no EndingSequenceNumber, then the shard is still open (able to receive more stream records). If both StartingSequenceNumber and EndingSequenceNumber are present, then that shard is closed and can no longer receive more data.

**Request Syntax**

```
{
   "ExclusiveStartShardId": "string",
   "Limit": number,
   "StreamArn": "string"
}
```

**Request Parameters**

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**StreamArn (p. 214)**

The Amazon Resource Name (ARN) for the stream.

Type: String


Required: Yes

**ExclusiveStartShardId (p. 214)**

The shard ID of the first item that this operation will evaluate. Use the value that was returned for LastEvaluatedShardId in the previous operation.

Type: String


Required: No

**Limit (p. 214)**

The maximum number of shard objects to return. The upper limit is 100.

Type: Integer

Valid Range: Minimum value of 1.

Required: No
Response Syntax

```
{
  "StreamDescription": {
    "CreationRequestDateTime": number,
    "KeySchema": [
      {
        "AttributeName": "string",
        "KeyType": "string"
      }
    ],
    "LastEvaluatedShardId": "string",
    "Shards": [
      {
        "ParentShardId": "string",
        "SequenceNumberRange": {
          "EndingSequenceNumber": "string",
          "StartingSequenceNumber": "string"
        },
        "ShardId": "string"
      }
    ],
    "StreamArn": "string",
    "StreamLabel": "string",
    "StreamStatus": "string",
    "StreamViewType": "string",
    "TableName": "string"
  }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**StreamDescription (p. 215)**

A complete description of the stream, including its creation date and time, the DynamoDB table associated with the stream, the shard IDs within the stream, and the beginning and ending sequence numbers of stream records within the shards.

Type: StreamDescription (p. 336) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

**InternalServerError**

An error occurred on the server side.

HTTP Status Code: 500

**ResourceNotFoundException**

The operation tried to access a nonexistent stream.

HTTP Status Code: 400
Example

Describe A Stream

The following sample returns a description of a stream with a given stream ARN. All of the shards in the stream are listed in the response, along with the beginning and ending sequence numbers of stream records within the shards. Note that one of the shards is still open, because it does not have an EndingSequenceNumber.

Sample Request

```plaintext
POST / HTTP/1.1
x-amzn-RequestId: <RequestID>
x-amzn-crc32: <CRC32>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDBStreams_20120810.DescribeStream
{
}
```

Sample Response

```plaintext
HTTP/1.1 200 OK
x-amzn-RequestId: <RequestID>
x-amzn-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>
{
  "StreamDescription": {
    "StreamStatus": "ENABLED",
    "StreamViewType": "NEW_AND_OLD_IMAGES",
    "CreationRequestDateTime": "Wed May 20 13:51:10 PDT 2015",
    "TableName": "Forum",
    "KeySchema": [
      {"AttributeName": "ForumName","KeyType": "HASH"},
      {"AttributeName": "Subject","KeyType": "RANGE"}
    ],
    "Shards": [
      {
        "SequenceNumberRange": {
          "EndingSequenceNumber": "20500000000000000910398",
          "StartingSequenceNumber": "20500000000000000910398"
        },
        "ShardId": "shardId-00000001414562045508-2bac9cd2"
      },
      {
        "ParentShardId": "shardId-0000001414562045508-2bac9cd2",
        "SequenceNumberRange": {
          "EndingSequenceNumber": "82040000000000001192334",
          "StartingSequenceNumber": "82040000000000001192334"
        }
      }
    ]
  }
}
```
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
**GetRecords**  
Service: Amazon DynamoDB Streams

Retrieves the stream records from a given shard.

Specify a shard iterator using the `ShardIterator` parameter. The shard iterator specifies the position in the shard from which you want to start reading stream records sequentially. If there are no stream records available in the portion of the shard that the iterator points to, `GetRecords` returns an empty list. Note that it might take multiple calls to get to a portion of the shard that contains stream records.

**Note**  
`GetRecords` can retrieve a maximum of 1 MB of data or 1000 stream records, whichever comes first.

**Request Syntax**

```json
{
    "Limit": number,
    "ShardIterator": "string"
}
```

**Request Parameters**

The request accepts the following data in JSON format.

**Note**  
In the following list, the required parameters are described first.

**ShardIterator (p. 218)**

A shard iterator that was retrieved from a previous GetShardIterator operation. This iterator can be used to access the stream records in this shard.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2048.

Required: Yes

**Limit (p. 218)**

The maximum number of records to return from the shard. The upper limit is 1000.

Type: Integer

Valid Range: Minimum value of 1.

Required: No

**Response Syntax**

```json
{
    "NextShardIterator": "string",
    "Records": [
        {
            "awsRegion": "string",
            "dynamodb": {
                "ApproximateCreationDateTime": number,
                "Keys": {
```
"string" : {
    "B": blob,
    "BOOL": boolean,
    "BS": [ blob ],
    "L": [
        "AttributeValue"
    ],
    "M": {
        "string": "AttributeValue"
    },
    "N": "string",
    "NS": [ "string" ],
    "NULL": boolean,
    "S": "string",
    "SS": [ "string" ]
},
"NewImage": {
    "string": {
        "B": blob,
        "BOOL": boolean,
        "BS": [ blob ],
        "L": [
            "AttributeValue"
        ],
        "M": {
            "string": "AttributeValue"
        },
        "N": "string",
        "NS": [ "string" ],
        "NULL": boolean,
        "S": "string",
        "SS": [ "string" ]
    }
},
"OldImage": {
    "string": {
        "B": blob,
        "BOOL": boolean,
        "BS": [ blob ],
        "L": [
            "AttributeValue"
        ],
        "M": {
            "string": "AttributeValue"
        },
        "N": "string",
        "NS": [ "string" ],
        "NULL": boolean,
        "S": "string",
        "SS": [ "string" ]
    }
},
"SequenceNumber": "string",
"SizeBytes": number,
"StreamViewType": "string"
},
"eventID": "string",
"eventName": "string",
"eventSource": "string",
"eventVersion": "string",
"userIdentity": {
    "PrincipalId": "string",
    "Type": "string"
}
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**NextShardIterator (p. 218)**

The next position in the shard from which to start sequentially reading stream records. If set to `null`, the shard has been closed and the requested iterator will not return any more data.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2048.

**Records (p. 218)**

The stream records from the shard, which were retrieved using the shard iterator.

Type: Array of Record (p. 331) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

**ExpiredIteratorException**

The shard iterator has expired and can no longer be used to retrieve stream records. A shard iterator expires 15 minutes after it is retrieved using the GetShardIterator action.

HTTP Status Code: 400

**InternalServerError**

An error occurred on the server side.

HTTP Status Code: 500

**LimitExceededException**

Your request rate is too high. The AWS SDKs for DynamoDB automatically retry requests that receive this exception. Your request is eventually successful, unless your retry queue is too large to finish. Reduce the frequency of requests and use exponential backoff. For more information, go to Error Retries and Exponential Backoff in the *Amazon DynamoDB Developer Guide*.

HTTP Status Code: 400

**ResourceNotFoundException**

The operation tried to access a nonexistent stream.

HTTP Status Code: 400

**TrimmedDataAccessException**

The operation attempted to read past the oldest stream record in a shard.

In DynamoDB Streams, there is a 24 hour limit on data retention. Stream records whose age exceeds this limit are subject to removal (trimming) from the stream. You might receive a TrimmedDataAccessException if:
• You request a shard iterator with a sequence number older than the trim point (24 hours).
• You obtain a shard iterator, but before you use the iterator in a GetRecords request, a stream record in the shard exceeds the 24 hour period and is trimmed. This causes the iterator to access a record that no longer exists.

HTTP Status Code: 400

Example

Retrieve stream records from a shard

The following sample retrieves all the stream records from a shard. To do this, it uses a ShardIterator that was obtained from a previous GetShardIterator call.

Sample Request

```plaintext
POST / HTTP/1.1
x-amzn-RequestId: <RequestID>
x-amzn-crc32: <CRC32>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDBStreams_20120810.GetRecords
{
}
```

Sample Response

```plaintext
HTTP/1.1 200 OK
x-amzn-RequestId: <RequestID>
x-amzn-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>
{
  "Records": [
    {
      "awsRegion": "us-west-2",
      "dynamodb": {
        "ApproximateCreationDateTime": 1.46480431E9,
        "Keys": {
          "ForumName": {"S": "DynamoDB"},
          "Subject": {"S": "DynamoDB Thread 3"}
        },
        "SequenceNumber": "300000000000000499659",
        "SizeBytes": 41,
        "StreamViewType": "KEYS_ONLY"
      },
      "eventID": "e2fd9c34eff2d779b297b26f5fe4206",
```

```
"eventName": "INSERT",
"eventSource": "aws:dynamodb",
"eventVersion": "1.0"
},
{
"awsRegion": "us-west-2",
"dynamodb": {
"ApproximateCreationDateTime": 1.46480527E9,
"Keys": {
  "ForumName": {"S": "DynamoDB"},
  "Subject": {"S": "DynamoDB Thread 1"}
},
"SequenceNumber": "400000000000000499660",
"SizeBytes": 41,
"StreamViewType": "KEYS_ONLY"
},
"eventID": "4b25bd0da9a181a155114127e4837252",
"eventName": "MODIFY",
"eventSource": "aws:dynamodb",
"eventVersion": "1.0"
},
{
"awsRegion": "us-west-2",
"dynamodb": {
"ApproximateCreationDateTime": 1.46480646E9,
"Keys": {
  "ForumName": {"S": "DynamoDB"},
  "Subject": {"S": "DynamoDB Thread 2"}
},
"SequenceNumber": "500000000000000499661",
"SizeBytes": 41,
"StreamViewType": "KEYS_ONLY"
},
"eventID": "740280c73a3df7842edab3548a1b08ad",
"eventName": "REMOVE",
"eventSource": "aws:dynamodb",
"eventVersion": "1.0"
}
]

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
GetShardIterator

Service: Amazon DynamoDB Streams

Returns a shard iterator. A shard iterator provides information about how to retrieve the stream records from within a shard. Use the shard iterator in a subsequent GetRecords request to read the stream records from the shard.

**Note**
A shard iterator expires 15 minutes after it is returned to the requester.

**Request Syntax**

```json
{
    "SequenceNumber": "string",
    "ShardId": "string",
    "ShardIteratorType": "string",
    "StreamArn": "string"
}
```

**Request Parameters**

The request accepts the following data in JSON format.

**Note**
In the following list, the required parameters are described first.

**ShardId (p. 223)**

The identifier of the shard. The iterator will be returned for this shard ID.

Type: String


Required: Yes

**ShardIteratorType (p. 223)**

Determines how the shard iterator is used to start reading stream records from the shard:

- **AT_SEQUENCE_NUMBER** - Start reading exactly from the position denoted by a specific sequence number.
- **AFTER_SEQUENCE_NUMBER** - Start reading right after the position denoted by a specific sequence number.
- **TRIM_HORIZON** - Start reading at the last (untrimmed) stream record, which is the oldest record in the shard. In DynamoDB Streams, there is a 24 hour limit on data retention. Stream records whose age exceeds this limit are subject to removal (trimming) from the stream.
- **LATEST** - Start reading just after the most recent stream record in the shard, so that you always read the most recent data in the shard.

Type: String

Valid Values: TRIM_HORIZON | LATEST | AT_SEQUENCE_NUMBER | AFTER_SEQUENCE_NUMBER

Required: Yes

**StreamArn (p. 223)**

The Amazon Resource Name (ARN) for the stream.
Type: String
Required: Yes

**SequenceNumber (p. 223)**

The sequence number of a stream record in the shard from which to start reading.

Type: String
Required: No

**Response Syntax**

```
{   "ShardIterator": "string"
}
```

**Response Elements**

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**ShardIterator (p. 224)**

The position in the shard from which to start reading stream records sequentially. A shard iterator specifies this position using the sequence number of a stream record in a shard.

Type: String
Length Constraints: Minimum length of 1. Maximum length of 2048.

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 341).

**InternalServer>Error**

An error occurred on the server side.

HTTP Status Code: 500

**ResourceNotFoundException**

The operation tried to access a nonexistent stream.

HTTP Status Code: 400

**TrimmedDataAccessException**

The operation attempted to read past the oldest stream record in a shard.

In DynamoDB Streams, there is a 24 hour limit on data retention. Stream records whose age exceeds this limit are subject to removal (trimming) from the stream. You might receive a TrimmedDataAccessException if:
You request a shard iterator with a sequence number older than the trim point (24 hours).
You obtain a shard iterator, but before you use the iterator in a GetRecords request, a stream record in the shard exceeds the 24 hour period and is trimmed. This causes the iterator to access a record that no longer exists.

HTTP Status Code: 400

Example

Retrieve a Shard Iterator For a Stream

The following sample returns a shard iterator for the provided stream ARN and shard ID. The shard iterator will allow access to stream records beginning with the given sequence number.

Sample Request

POST / HTTP/1.1
x-amzn-RequestId: <RequestID>
x-amzn-crc32: <CRC32>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDBStreams_20120810.GetShardIterator

{
  "ShardId": "00000001414576573621-f55eea83",
  "ShardIteratorType": "TRIM_HORIZON"
}

Sample Response

HTTP/1.1 200 OK
x-amzn-RequestId: <RequestID>
x-amzn-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>

{
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
**ListStreams**

Service: Amazon DynamoDB Streams

Returns an array of stream ARNs associated with the current account and endpoint. If the `TableName` parameter is present, then `ListStreams` will return only the streams ARNs for that table.

**Note**

You can call `ListStreams` at a maximum rate of 5 times per second.

**Request Syntax**

```json
{
    "ExclusiveStartStreamArn": "string",
    "Limit": number,
    "TableName": "string"
}
```

**Request Parameters**

The request accepts the following data in JSON format.

**Note**

In the following list, the required parameters are described first.

- **ExclusiveStartStreamArn** (p. 227)
  
  The ARN (Amazon Resource Name) of the first item that this operation will evaluate. Use the value that was returned for `LastEvaluatedStreamArn` in the previous operation.
  
  Type: String
  
  
  Required: No

- **Limit** (p. 227)
  
  The maximum number of streams to return. The upper limit is 100.
  
  Type: Integer
  
  Valid Range: Minimum value of 1.
  
  Required: No

- **TableName** (p. 227)
  
  If this parameter is provided, then only the streams associated with this table name are returned.
  
  Type: String
  
  
  Pattern: [a-zA-Z0-9_.-]+
  
  Required: No

**Response Syntax**

```json
{
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

LastEvaluatedStreamArn (p. 227)

The stream ARN of the item where the operation stopped, inclusive of the previous result set. Use this value to start a new operation, excluding this value in the new request.

If LastEvaluatedStreamArn is empty, then the "last page" of results has been processed and there is no more data to be retrieved.

If LastEvaluatedStreamArn is not empty, it does not necessarily mean that there is more data in the result set. The only way to know when you have reached the end of the result set is when LastEvaluatedStream Arn is empty.

Type: String


Streams (p. 227)

A list of stream descriptors associated with the current account and endpoint.

Type: Array of Stream (p. 335) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 341).

InternalServerError

An error occurred on the server side.

HTTP Status Code: 500

ResourceNotFoundException

The operation tried to access a nonexistent stream.

HTTP Status Code: 400

Example

Retrieve All Stream ARNs

The following sample returns all of the stream ARNs.
Sample Request

POST / HTTP/1.1
x-amzn-RequestId: <RequestID>
x-amzn-crc32: <CRC32>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
X-Amz-Date: <Date>
X-Amz-Target: DynamoDBStreams_20120810.ListStreams

{}

Sample Response

HTTP/1.1 200 OK
x-amzn-RequestId: <RequestID>
x-amz-crc32: <Checksum>
Content-Type: application/x-amz-json-1.0
Content-Length: <PayloadSizeBytes>
Date: <Date>

{
    "Streams": [
    {
stream/2015-05-20T20:51:10.252",
        "TableName": "Forum",
        "StreamLabel": "2015-05-20T20:51:10.252"
    },
    {
stream/2015-05-20T20:50:02.714",
        "TableName": "Forum",
        "StreamLabel": "2015-05-20T20:50:02.714"
    },
    {
stream/2015-05-19T23:03:50.641",
        "TableName": "Forum",
        "StreamLabel": "2015-05-19T23:03:50.641"
    },
    ...
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
Data Types

The following data types are supported by Amazon DynamoDB:

- AttributeDefinition (p. 234)
- AttributeValue (p. 235)
- AttributeValueUpdate (p. 238)
- BackupDescription (p. 240)
- BackupDetails (p. 241)
- BackupSummary (p. 243)
- Capacity (p. 245)
- Condition (p. 246)
- ConsumedCapacity (p. 249)
- ContinuousBackupsDescription (p. 251)
- CreateGlobalSecondaryIndexAction (p. 252)
- CreateReplicaAction (p. 254)
- DeleteGlobalSecondaryIndexAction (p. 255)
- DeleteReplicaAction (p. 256)
- DeleteRequest (p. 257)
- ExpectedAttributeValue (p. 258)
- GlobalSecondaryIndex (p. 262)
- GlobalSecondaryIndexDescription (p. 264)
- GlobalSecondaryIndexInfo (p. 267)
- GlobalSecondaryIndexUpdate (p. 269)
- GlobalTable (p. 270)
- GlobalTableDescription (p. 271)
- ItemCollectionMetrics (p. 273)
- KeysAndAttributes (p. 274)
- KeySchemaElement (p. 276)
- LocalSecondaryIndex (p. 278)
- LocalSecondaryIndexDescription (p. 280)
- LocalSecondaryIndexInfo (p. 282)
- Projection (p. 284)
- ProvisionedThroughput (p. 285)
- ProvisionedThroughputDescription (p. 286)
- PutRequest (p. 288)
- Replica (p. 289)
- ReplicaDescription (p. 290)
- ReplicaUpdate (p. 291)
- RestoreSummary (p. 292)
- SourceTableDetails (p. 293)
- SourceTableFeatureDetails (p. 295)
- StreamSpecification (p. 296)
- TableDescription (p. 297)
• Tag (p. 302)
• TimeToLiveDescription (p. 303)
• TimeToLiveSpecification (p. 304)
• UpdateGlobalSecondaryIndexAction (p. 305)
• WriteRequest (p. 306)

The following data types are supported by Amazon DynamoDB Accelerator:

• Cluster (p. 308)
• Endpoint (p. 311)
• Event (p. 312)
• Node (p. 313)
• NodeTypeSpecificValue (p. 315)
• NotificationConfiguration (p. 316)
• Parameter (p. 317)
• ParameterGroup (p. 319)
• ParameterGroupStatus (p. 320)
• ParameterNameValue (p. 321)
• SecurityGroupMembership (p. 322)
• Subnet (p. 323)
• SubnetGroup (p. 324)
• Tag (p. 325)

The following data types are supported by Amazon DynamoDB Streams:

• AttributeValue (p. 327)
• Identity (p. 329)
• KeySchemaElement (p. 330)
• Record (p. 331)
• SequenceNumberRange (p. 333)
• Shard (p. 334)
• Stream (p. 335)
• StreamDescription (p. 336)
• StreamRecord (p. 339)
• Condition (p. 246)
• ConsumedCapacity (p. 249)
• ContinuousBackupsDescription (p. 251)
• CreateGlobalSecondaryIndexAction (p. 252)
• CreateReplicaAction (p. 254)
• DeleteGlobalSecondaryIndexAction (p. 255)
• DeleteReplicaAction (p. 256)
• DeleteRequest (p. 257)
• ExpectedAttributeValue (p. 258)
• GlobalSecondaryIndex (p. 262)
• GlobalSecondaryIndexDescription (p. 264)
• GlobalSecondaryIndexInfo (p. 267)
• GlobalSecondaryIndexUpdate (p. 269)
• GlobalTable (p. 270)
• GlobalTableDescription (p. 271)
• ItemCollectionMetrics (p. 273)
• KeysAndAttributes (p. 274)
• KeySchemaElement (p. 276)
• LocalSecondaryIndex (p. 278)
• LocalSecondaryIndexDescription (p. 280)
• LocalSecondaryIndexInfo (p. 282)
• Projection (p. 284)
• ProvisionedThroughput (p. 285)
• ProvisionedThroughputDescription (p. 286)
• PutRequest (p. 288)
• Replica (p. 289)
• ReplicaDescription (p. 290)
• ReplicaUpdate (p. 291)
• RestoreSummary (p. 292)
• SourceTableDetails (p. 293)
• SourceTableFeatureDetails (p. 295)
• StreamSpecification (p. 296)
• TableDescription (p. 297)
• Tag (p. 302)
• TimeToLiveDescription (p. 303)
• TimeToLiveSpecification (p. 304)
• UpdateGlobalSecondaryIndexAction (p. 305)
• WriteRequest (p. 306)
AttributeDefinition
Service: Amazon DynamoDB

Represents an attribute for describing the key schema for the table and indexes.

Contents

Note
In the following list, the required parameters are described first.

AttributeName
A name for the attribute.
Type: String
Length Constraints: Minimum length of 1. Maximum length of 255.
Required: Yes

AttributeType
The data type for the attribute, where:
• S - the attribute is of type String
• N - the attribute is of type Number
• B - the attribute is of type Binary
Type: String
Valid Values: S | N | B
Required: Yes

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for Ruby V2
AttributeValue
Service: Amazon DynamoDB

Represents the data for an attribute.

Each attribute value is described as a name-value pair. The name is the data type, and the value is the
data itself.

For more information, see Data Types in the Amazon DynamoDB Developer Guide.

Contents

Note
In the following list, the required parameters are described first.

B
An attribute of type Binary. For example:
"B": "dGhpcyB0ZXh0IGlzIGJhc2U2NC1lbmNvZGVk"
Type: Base64-encoded binary data object
Required: No

BOOL
An attribute of type Boolean. For example:
"BOOL": true
Type: Boolean
Required: No

BS
An attribute of type Binary Set. For example:
"BS": ["U3Vubnk=", "UmFpbnk=", "U25vd3k="]
Type: Array of Base64-encoded binary data objects
Required: No

L
An attribute of type List. For example:
"L": ["Cookies", "Coffee", 3.14159]
Type: Array of AttributeValue (p. 235) objects
Required: No

M
An attribute of type Map. For example:
"M": {"Name": {"S": "Joe"}, "Age": {"N": "35"}}
Type: String to AttributeValue (p. 235) object map
Key Length Constraints: Maximum length of 65535.
Required: No

**N**

An attribute of type Number. For example:

"N": "123.45"

Numbers are sent across the network to DynamoDB as strings, to maximize compatibility across languages and libraries. However, DynamoDB treats them as number type attributes for mathematical operations.

Type: String

Required: No

**NS**

An attribute of type Number Set. For example:

"NS": ["42.2", "-19", "7.5", "3.14"]

Numbers are sent across the network to DynamoDB as strings, to maximize compatibility across languages and libraries. However, DynamoDB treats them as number type attributes for mathematical operations.

Type: Array of strings

Required: No

**NULL**

An attribute of type Null. For example:

"NULL": true

Type: Boolean

Required: No

**S**

An attribute of type String. For example:

"S": "Hello"

Type: String

Required: No

**SS**

An attribute of type String Set. For example:

"SS": ["Giraffe", "Hippo", "Zebra"]

Type: Array of strings

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for Ruby V2
AttributeValueUpdate
Service: Amazon DynamoDB

For the UpdateItem operation, represents the attributes to be modified, the action to perform on each, and the new value for each.

**Note**
You cannot use UpdateItem to update any primary key attributes. Instead, you will need to delete the item, and then use PutItem to create a new item with new attributes.

Attribute values cannot be null; string and binary type attributes must have lengths greater than zero; and set type attributes must not be empty. Requests with empty values will be rejected with a ValidationException exception.

**Contents**

**Note**
In the following list, the required parameters are described first.

**Action**
Specifies how to perform the update. Valid values are PUT (default), DELETE, and ADD. The behavior depends on whether the specified primary key already exists in the table.

**If an item with the specified Key is found in the table:**
- **PUT** - Adds the specified attribute to the item. If the attribute already exists, it is replaced by the new value.
- **DELETE** - If no value is specified, the attribute and its value are removed from the item. The data type of the specified value must match the existing value's data type.

If a set of values is specified, then those values are subtracted from the old set. For example, if the attribute value was the set \([a, b, c]\) and the DELETE action specified \([a, c]\), then the final attribute value would be \([b]\). Specifying an empty set is an error.

- **ADD** - If the attribute does not already exist, then the attribute and its values are added to the item. If the attribute does exist, then the behavior of ADD depends on the data type of the attribute:
  - If the existing attribute is a number, and if Value is also a number, then the Value is mathematically added to the existing attribute. If Value is a negative number, then it is subtracted from the existing attribute.

  **Note**
  If you use ADD to increment or decrement a number value for an item that doesn't exist before the update, DynamoDB uses 0 as the initial value.
  In addition, if you use ADD to update an existing item, and intend to increment or decrement an attribute value which does not yet exist, DynamoDB uses 0 as the initial value. For example, suppose that the item you want to update does not yet have an attribute named itemcount, but you decide to ADD the number 3 to this attribute anyway, even though it currently does not exist. DynamoDB will create the itemcount attribute, set its initial value to 0, and finally add 3 to it. The result will be a new itemcount attribute in the item, with a value of 3.

  - If the existing data type is a set, and if the Value is also a set, then the Value is added to the existing set. (This is a set operation, not mathematical addition.) For example, if the attribute value was the set \([1, 2]\), and the ADD action specified \([3]\), then the final attribute value would be \([1, 2, 3]\). An error occurs if an Add action is specified for a set attribute and the attribute type specified does not match the existing set type.
Both sets must have the same primitive data type. For example, if the existing data type is a set of strings, the Value must also be a set of strings. The same holds true for number sets and binary sets.

This action is only valid for an existing attribute whose data type is number or is a set. Do not use ADD for any other data types.

If no item with the specified Key is found:
- PUT - DynamoDB creates a new item with the specified primary key, and then adds the attribute.
- DELETE - Nothing happens; there is no attribute to delete.
- ADD - DynamoDB creates an item with the supplied primary key and number (or set of numbers) for the attribute value. The only data types allowed are number and number set; no other data types can be specified.

Type: String

Valid Values: ADD | PUT | DELETE

Required: No

Value

Represents the data for an attribute.

Each attribute value is described as a name-value pair. The name is the data type, and the value is the data itself.

For more information, see Data Types in the Amazon DynamoDB Developer Guide.

Type: AttributeValue (p. 235) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
BackupDescription

Service: Amazon DynamoDB

Contains the description of the backup created for the table.

Contents

Note
In the following list, the required parameters are described first.

BackupDetails

Contains the details of the backup created for the table.

Type: BackupDetails (p. 241) object

Required: No

SourceTableDetails

Contains the details of the table when the backup was created.

Type: SourceTableDetails (p. 293) object

Required: No

SourceTableFeatureDetails

Contains the details of the features enabled on the table when the backup was created. For example, LSIs, GSIs, streams, TTL.

Type: SourceTableFeatureDetails (p. 295) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
BackupDetails
Service: Amazon DynamoDB

Contains the details of the backup created for the table.

Contents

**Note**
In the following list, the required parameters are described first.

**BackupArn**
ARN associated with the backup.
Type: String
Required: Yes

**BackupCreationDateTime**
Time at which the backup was created. This is the request time of the backup.
Type: Timestamp
Required: Yes

**BackupName**
Name of the requested backup.
Type: String
Required: Yes

**BackupStatus**
Backup can be in one of the following states: CREATING, ACTIVE, DELETED.
Type: String
Valid Values: CREATING | DELETED | AVAILABLE
Required: Yes

**BackupSizeBytes**
Size of the backup in bytes.
Type: Long
Valid Range: Minimum value of 0.
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
BackupSummary
Service: Amazon DynamoDB
Contains details for the backup.

Contents

Note
In the following list, the required parameters are described first.

BackupArn
ARN associated with the backup.
Type: String
Required: No

BackupCreationDateTime
Time at which the backup was created.
Type: Timestamp
Required: No

BackupName
Name of the specified backup.
Type: String
Required: No

BackupSizeBytes
Size of the backup in bytes.
Type: Long
Valid Range: Minimum value of 0.
Required: No

BackupStatus
Backup can be in one of the following states: CREATING, ACTIVE, DELETED.
Type: String
Valid Values: CREATING | DELETED | AVAILABLE
Required: No

TableArn
ARN associated with the table.
Type: String
Required: No
TableId

Unique identifier for the table.
Type: String

Pattern: [0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}

Required: No

TableName

Name of the table.
Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Capacity
Service: Amazon DynamoDB

Represents the amount of provisioned throughput capacity consumed on a table or an index.

Contents

Note
In the following list, the required parameters are described first.

CapacityUnits
The total number of capacity units consumed on a table or an index.
Type: Double
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
**Condition**

Service: Amazon DynamoDB

Represents the selection criteria for a Query or Scan operation:

- For a **Query** operation, **Condition** is used for specifying the **KeyConditions** to use when querying a table or an index. For **KeyConditions**, only the following comparison operators are supported:

  EQ | LE | LT | GE | GT | BEGINS_WITH | BETWEEN

  **Condition** is also used in a **QueryFilter**, which evaluates the query results and returns only the desired values.

- For a **Scan** operation, **Condition** is used in a **ScanFilter**, which evaluates the scan results and returns only the desired values.

**Contents**

**Note**

In the following list, the required parameters are described first.

**ComparisonOperator**

A comparator for evaluating attributes. For example, equals, greater than, less than, etc.

The following comparison operators are available:

EQ | NE | LE | LT | GE | GT | NOT_NULL | NULL | CONTAINS | NOT_CONTAINS | BEGINS_WITH | IN | BETWEEN

The following are descriptions of each comparison operator.

- **EQ**: Equal. **EQ** is supported for all data types, including lists and maps.

  AttributeValueList can contain only one AttributeValue element of type String, Number, Binary, String Set, Number Set, or Binary Set. If an item contains an AttributeValue element of a different type than the one provided in the request, the value does not match. For example, `{"S":"6"}` does not equal `{"N":"6"}`. Also, `{"N":"6"}` does not equal `{"NS": ["6", "2", "1"]}`.

- **NE**: Not equal. **NE** is supported for all data types, including lists and maps.

  AttributeValueList can contain only one AttributeValue of type String, Number, Binary, String Set, Number Set, or Binary Set. If an item contains an AttributeValue of a different type than the one provided in the request, the value does not match. For example, `{"S":"6"}` does not equal `{"N":"6"}`. Also, `{"N":"6"}` does not equal `{"NS": ["6", "2", "1"]}`.

- **LE**: Less than or equal.

  AttributeValueList can contain only one AttributeValue element of type String, Number, or Binary (not a set type). If an item contains an AttributeValue element of a different type than the one provided in the request, the value does not match. For example, `{"S":"6"}` does not equal `{"N":"6"}`. Also, `{"N":"6"}` does not compare to `{"NS": ["6", "2", "1"]}`.

- **LT**: Less than.

  AttributeValueList can contain only one AttributeValue of type String, Number, or Binary (not a set type). If an item contains an AttributeValue element of a different type than the one provided in the request, the value does not match. For example, `{"S":"6"}` does not equal `{"N":"6"}`. Also, `{"N":"6"}` does not compare to `{"NS": ["6", "2", "1"]}`.
• **GE**: Greater than or equal.

AttributeValueList can contain only one AttributeValue element of type String, Number, or Binary (not a set type). If an item contains an AttributeValue element of a different type than the one provided in the request, the value does not match. For example, \{"S": "6"\} does not equal \{"N": "6"\}. Also, \{"N": "6"\} does not compare to \{"NS": ["6", "2", "1"]\}.

• **GT**: Greater than.

AttributeValueList can contain only one AttributeValue element of type String, Number, or Binary (not a set type). If an item contains an AttributeValue element of a different type than the one provided in the request, the value does not match. For example, \{"S": "6"\} does not equal \{"N": "6"\}. Also, \{"N": "6"\} does not compare to \{"NS": ["6", "2", "1"]\}.

• **NOT_NULL**: The attribute exists. NOT_NULL is supported for all data types, including lists and maps.

  **Note**
  This operator tests for the existence of an attribute, not its data type. If the data type of attribute "a" is null, and you evaluate it using NOT_NULL, the result is a Boolean true. This result is because the attribute "a" exists; its data type is not relevant to the NOT_NULL comparison operator.

• **NULL**: The attribute does not exist. NULL is supported for all data types, including lists and maps.

  **Note**
  This operator tests for the nonexistence of an attribute, not its data type. If the data type of attribute "a" is null, and you evaluate it using NULL, the result is a Boolean false. This is because the attribute "a" exists; its data type is not relevant to the NULL comparison operator.

• **CONTAINS**: Checks for a subsequence, or value in a set.

AttributeValueList can contain only one AttributeValue element of type String, Number, or Binary (not a set type). If the target attribute of the comparison is of type String, then the operator checks for a substring match. If the target attribute of the comparison is of type Binary, then the operator looks for a subsequence of the target that matches the input. If the target attribute of the comparison is a set ("SS", "NS", or "BS"), then the operator evaluates to true if it finds an exact match with any member of the set.

CONTAINS is supported for lists: When evaluating "a CONTAINS b", "a" can be a list; however, "b" cannot be a set, a map, or a list.

• **NOT_CONTAINS**: Checks for absence of a subsequence, or absence of a value in a set.

AttributeValueList can contain only one AttributeValue element of type String, Number, or Binary (not a set type). If the target attribute of the comparison is a String, then the operator checks for the absence of a substring match. If the target attribute of the comparison is Binary, then the operator checks for the absence of a subsequence of the target that matches the input. If the target attribute of the comparison is a set ("SS", "NS", or "BS"), then the operator evaluates to true if it does not find an exact match with any member of the set.

NOT_CONTAINS is supported for lists: When evaluating "a NOT CONTAINS b", "a" can be a list; however, "b" cannot be a set, a map, or a list.

• **BEGINS_WITH**: Checks for a prefix.

AttributeValueList can contain only one AttributeValue of type String or Binary (not a Number or a set type). The target attribute of the comparison must be of type String or Binary (not a Number or a set type).
• **IN**: Checks for matching elements in a list. 

    AttributeValueList can contain one or more AttributeValue elements of type String, Number, or Binary. These attributes are compared against an existing attribute of an item. If any elements of the input are equal to the item attribute, the expression evaluates to true.

• **BETWEEN**: Greater than or equal to the first value, and less than or equal to the second value.

    AttributeValueList must contain two AttributeValue elements of the same type, either String, Number, or Binary (not a set type). A target attribute matches if the target value is greater than, or equal to, the first element and less than, or equal to, the second element. If an item contains an AttributeValue element of a different type than the one provided in the request, the value does not match. For example, `{"S":"6"}` does not compare to `{"N":"6"}`. Also, `{"N":"6"}` does not compare to `{"NS":["6", "2", "1"]}`

For usage examples of AttributeValueList and ComparisonOperator, see Legacy Conditional Parameters in the Amazon DynamoDB Developer Guide.

Type: String

Valid Values: EQ | NE | IN | LE | LT | GE | GT | BETWEEN | NOT_NULL | NULL | CONTAINS | NOT_CONTAINS | BEGINS_WITH

Required: Yes

**AttributeValueList**

One or more values to evaluate against the supplied attribute. The number of values in the list depends on the ComparisonOperator being used.

For type Number, value comparisons are numeric.

String value comparisons for greater than, equals, or less than are based on ASCII character code values. For example, a is greater than A, and a is greater than B. For a list of code values, see [http://en.wikipedia.org/wiki/ASCII#ASCII_printable_characters](http://en.wikipedia.org/wiki/ASCII#ASCII_printable_characters).

For Binary, DynamoDB treats each byte of the binary data as unsigned when it compares binary values.

Type: Array of AttributeValue (p. 235) objects

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
**ConsumedCapacity**

Service: Amazon DynamoDB

The capacity units consumed by an operation. The data returned includes the total provisioned throughput consumed, along with statistics for the table and any indexes involved in the operation. ConsumedCapacity is only returned if the request asked for it. For more information, see Provisioned Throughput in the *Amazon DynamoDB Developer Guide*.

**Contents**

**Note**

In the following list, the required parameters are described first.

**CapacityUnits**

The total number of capacity units consumed by the operation.

Type: Double

Required: No

**GlobalSecondaryIndexes**

The amount of throughput consumed on each global index affected by the operation.

Type: String to [Capacity (p. 245)] object map


Key Pattern: `[a-zA-Z0-9_.-]+`

Required: No

**LocalSecondaryIndexes**

The amount of throughput consumed on each local index affected by the operation.

Type: String to [Capacity (p. 245)] object map


Key Pattern: `[a-zA-Z0-9_.-]+`

Required: No

**Table**

The amount of throughput consumed on the table affected by the operation.

Type: [Capacity (p. 245)] object

Required: No

**TableName**

The name of the table that was affected by the operation.

Type: String


Pattern: `[a-zA-Z0-9_.-]+`

Required: No
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ContinuousBackupsDescription
Service: Amazon DynamoDB

Represents the backup and restore settings on the table when the backup was created.

Contents

Note
In the following list, the required parameters are described first.

ContinuousBackupsStatus

ContinuousBackupsStatus can be one of the following states: ENABLED, DISABLED

Type: String

Valid Values: ENABLED | DISABLED

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
CreateGlobalSecondaryIndexAction

Service: Amazon DynamoDB

Represents a new global secondary index to be added to an existing table.

Contents

**Note**

In the following list, the required parameters are described first.

**IndexName**

The name of the global secondary index to be created.

Type: String


Pattern: \[a-zA-Z0-9_.-]+\n
Required: Yes

**KeySchema**

The key schema for the global secondary index.

Type: Array of KeySchemaElement (p. 276) objects

Array Members: Minimum number of 1 item. Maximum number of 2 items.

Required: Yes

**Projection**

Represents attributes that are copied (projected) from the table into an index. These are in addition to the primary key attributes and index key attributes, which are automatically projected.

Type: Projection (p. 284) object

Required: Yes

**ProvisionedThroughput**

Represents the provisioned throughput settings for the specified global secondary index.

For current minimum and maximum provisioned throughput values, see Limits in the Amazon DynamoDB Developer Guide.

Type: ProvisionedThroughput (p. 285) object

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
CreateReplicaAction
Service: Amazon DynamoDB

Represents a replica to be added.

Contents

Note
In the following list, the required parameters are described first.

RegionName

The region of the replica to be added.

Type: String

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DeleteGlobalSecondaryIndexAction

Service: Amazon DynamoDB

Represents a global secondary index to be deleted from an existing table.

Contents

Note
In the following list, the required parameters are described first.

IndexName

The name of the global secondary index to be deleted.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DeleteReplicaAction

Service: Amazon DynamoDB

Represents a replica to be removed.

Contents

Note
In the following list, the required parameters are described first.

RegionName

The region of the replica to be removed.

Type: String
Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DeleteRequest
Service: Amazon DynamoDB

Represents a request to perform a DeleteItem operation on an item.

Contents

Note
In the following list, the required parameters are described first.

Key
A map of attribute name to attribute values, representing the primary key of the item to delete. All of the table's primary key attributes must be specified, and their data types must match those of the table's key schema.

Type: String to AttributeValue (p. 235) object map

Key Length Constraints: Maximum length of 65535.

Required: Yes

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
**ExpectedAttributeValue**

Service: Amazon DynamoDB

Represents a condition to be compared with an attribute value. This condition can be used with `DeleteItem`, `PutItem` or `UpdateItem` operations; if the comparison evaluates to true, the operation succeeds; if not, the operation fails. You can use `ExpectedAttributeValue` in one of two different ways:

- Use `AttributeValueList` to specify one or more values to compare against an attribute. Use `ComparisonOperator` to specify how you want to perform the comparison. If the comparison evaluates to true, then the conditional operation succeeds.
- Use `Value` to specify a value that DynamoDB will compare against an attribute. If the values match, then `ExpectedAttributeValue` evaluates to true and the conditional operation succeeds. Optionally, you can also set `Exists` to false, indicating that you do not expect to find the attribute value in the table. In this case, the conditional operation succeeds only if the comparison evaluates to false.

`Value` and `Exists` are incompatible with `AttributeValueList` and `ComparisonOperator`. Note that if you use both sets of parameters at once, DynamoDB will return a `ValidationException` exception.

**Contents**

**Note**
In the following list, the required parameters are described first.

**AttributeValueList**

One or more values to evaluate against the supplied attribute. The number of values in the list depends on the `ComparisonOperator` being used.

For type `Number`, value comparisons are numeric.

String value comparisons for greater than, equals, or less than are based on ASCII character code values. For example, a is greater than A, and a is greater than B. For a list of code values, see [http://en.wikipedia.org/wiki/ASCII#ASCII_printable_characters](http://en.wikipedia.org/wiki/ASCII#ASCII_printable_characters).

For `Binary`, DynamoDB treats each byte of the binary data as unsigned when it compares binary values.

For information on specifying data types in JSON, see [JSON Data Format](https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/JSONDataFormat.html) in the *Amazon DynamoDB Developer Guide*.

Type: Array of `AttributeValue` objects

Required: No

**ComparisonOperator**

A comparator for evaluating attributes in the `AttributeValueList`. For example, equals, greater than, less than, etc.

The following comparison operators are available:

- `EQ` | `NE` | `LE` | `LT` | `GE` | `GT` | `NOT_NULL` | `NULL` | `CONTAINS` | `NOT_CONTAINS` | `BEGINS_WITH` | `IN` | `BETWEEN`

The following are descriptions of each comparison operator.

- **EQ**: Equal. `EQ` is supported for all data types, including lists and maps.
**ExpectedAttributeValue**

AttributeValueList can contain only one AttributeValue element of type String, Number, Binary, String Set, Number Set, or Binary Set. If an item contains an AttributeValue element of a different type than the one provided in the request, the value does not match. For example, 

```json
{"S":"6"}
```

does not equal 

```json
{"N":"6"}
```

Also, 

```json
{"N":"6"}
```

does not equal 

```json
{"NS":["6", "2", "1"]}
```

- **NE**: Not equal. NE is supported for all data types, including lists and maps.

- **LE**: Less than or equal.

- **LT**: Less than.

- **GE**: Greater than or equal.

- **GT**: Greater than.

- **NOT_NULL**: The attribute exists. NOT_NULL is supported for all data types, including lists and maps.

  **Note**
  
  This operator tests for the existence of an attribute, not its data type. If the data type of attribute "a" is null, and you evaluate it using NOT_NULL, the result is a Boolean true. This result is because the attribute "a" exists; its data type is not relevant to the NOT_NULL comparison operator.

- **NULL**: The attribute does not exist. NULL is supported for all data types, including lists and maps.

  **Note**
  
  This operator tests for the nonexistence of an attribute, not its data type. If the data type of attribute "a" is null, and you evaluate it using NULL, the result is a Boolean false. This is because the attribute "a" exists; its data type is not relevant to the NULL comparison operator.

- **CONTAINS**: Checks for a subsequence, or value in a set.
AttributeValueList can contain only one AttributeValue element of type String, Number, or Binary (not a set type). If the target attribute of the comparison is of type String, then the operator checks for a substring match. If the target attribute of the comparison is of type Binary, then the operator looks for a subsequence of the target that matches the input. If the target attribute of the comparison is a set ("SS", "NS", or "BS"), then the operator evaluates to true if it finds an exact match with any member of the set.

CONTAINS is supported for lists: When evaluating "a CONTAINS b", "a" can be a list; however, "b" cannot be a set, a map, or a list.

• NOT_CONTAINS: Checks for absence of a subsequence, or absence of a value in a set.

AttributeValueList can contain only one AttributeValue element of type String, Number, or Binary (not a set type). If the target attribute of the comparison is a String, then the operator checks for the absence of a substring match. If the target attribute of the comparison is Binary, then the operator checks for the absence of a subsequence of the target that matches the input. If the target attribute of the comparison is a set ("SS", "NS", or "BS"), then the operator evaluates to true if it does not find an exact match with any member of the set.

NOT_CONTAINS is supported for lists: When evaluating "a NOT CONTAINS b", "a" can be a list; however, "b" cannot be a set, a map, or a list.

• BEGINS_WITH: Checks for a prefix.

AttributeValueList can contain only one AttributeValue of type String or Binary (not a Number or a set type). The target attribute of the comparison must be of type String or Binary (not a Number or a set type).

• IN: Checks for matching elements in a list.

AttributeValueList can contain one or more AttributeValue elements of type String, Number, or Binary. These attributes are compared against an existing attribute of an item. If any elements of the input are equal to the item attribute, the expression evaluates to true.

• BETWEEN: Greater than or equal to the first value, and less than or equal to the second value.

AttributeValueList must contain two AttributeValue elements of the same type, either String, Number, or Binary (not a set type). A target attribute matches if the target value is greater than, or equal to, the first element and less than, or equal to, the second element. If an item contains an AttributeValue element of a different type than the one provided in the request, the value does not match. For example, {"S": "6"}) does not compare to {"N": "6"}. Also, {"N": "6"}) does not compare to {"NS": ["6", "2", "1"]}

Type: String

Valid Values: EQ | NE | IN | LE | LT | GE | GT | BETWEEN | NOT_NULL | NULL | CONTAINS | NOT_CONTAINS | BEGINS_WITH

Required: No

Exists

Causes DynamoDB to evaluate the value before attempting a conditional operation:

• If Exists is true, DynamoDB will check to see if that attribute value already exists in the table. If it is found, then the operation succeeds. If it is not found, the operation fails with a ConditionalCheckFailedException.

• If Exists is false, DynamoDB assumes that the attribute value does not exist in the table. If in fact the value does not exist, then the assumption is valid and the operation succeeds. If the value is found, despite the assumption that it does not exist, the operation fails with a ConditionalCheckFailedException.
The default setting for `Exists` is `true`. If you supply a `Value` all by itself, DynamoDB assumes the attribute exists: You don’t have to set `Exists` to `true`, because it is implied.

DynamoDB returns a `ValidationException` if:
- `Exists` is `true` but there is no `Value` to check. (You expect a value to exist, but don’t specify what that value is.)
- `Exists` is `false` but you also provide a `Value`. (You cannot expect an attribute to have a value, while also expecting it not to exist.)

Type: Boolean
Required: No

**Value**

Represents the data for the expected attribute.

Each attribute value is described as a name-value pair. The name is the data type, and the value is the data itself.

For more information, see Data Types in the Amazon DynamoDB Developer Guide.

Type: `AttributeValue` (p. 235) object
Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
GlobalSecondaryIndex

Service: Amazon DynamoDB

Represents the properties of a global secondary index.

Contents

Note
In the following list, the required parameters are described first.

IndexName

The name of the global secondary index. The name must be unique among all other indexes on this table.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: Yes

KeySchema

The complete key schema for a global secondary index, which consists of one or more pairs of attribute names and key types:

- HASH - partition key
- RANGE - sort key

Note
The partition key of an item is also known as its hash attribute. The term "hash attribute" derives from DynamoDB’s usage of an internal hash function to evenly distribute data items across partitions, based on their partition key values.

The sort key of an item is also known as its range attribute. The term "range attribute" derives from the way DynamoDB stores items with the same partition key physically close together, in sorted order by the sort key value.

Type: Array of KeySchemaElement (p. 276) objects

Array Members: Minimum number of 1 item. Maximum number of 2 items.

Required: Yes

Projection

Represents attributes that are copied (projected) from the table into the global secondary index. These are in addition to the primary key attributes and index key attributes, which are automatically projected.

Type: Projection (p. 284) object

Required: Yes

ProvisionedThroughput

Represents the provisioned throughput settings for the specified global secondary index.

For current minimum and maximum provisioned throughput values, see Limits in the Amazon DynamoDB Developer Guide.

Type: ProvisionedThroughput (p. 285) object
Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
GlobalSecondaryIndexDescription
Service: Amazon DynamoDB

Represents the properties of a global secondary index.

Contents

Note
In the following list, the required parameters are described first.

Backfilling

Indicates whether the index is currently backfilling. Backfilling is the process of reading items from the table and determining whether they can be added to the index. (Not all items will qualify: For example, a partition key cannot have any duplicate values.) If an item can be added to the index, DynamoDB will do so. After all items have been processed, the backfilling operation is complete and Backfilling is false.

Note
For indexes that were created during a CreateTable operation, the Backfilling attribute does not appear in the DescribeTable output.

Type: Boolean
Required: No

IndexArn
The Amazon Resource Name (ARN) that uniquely identifies the index.

Type: String
Required: No

IndexName
The name of the global secondary index.

Type: String


Pattern: [a-zA-Z0-9_.-]+
Required: No

IndexSizeBytes
The total size of the specified index, in bytes. DynamoDB updates this value approximately every six hours. Recent changes might not be reflected in this value.

Type: Long
Required: No

IndexStatus
The current state of the global secondary index:

- CREATING - The index is being created.
- UPDATING - The index is being updated.
- DELETING - The index is being deleted.
- ACTIVE - The index is ready for use.
Type: String

Valid Values: CREATING | UPDATING | DELETING | ACTIVE

Required: No

**ItemCount**

The number of items in the specified index. DynamoDB updates this value approximately every six hours. Recent changes might not be reflected in this value.

Type: Long

Required: No

**KeySchema**

The complete key schema for a global secondary index, which consists of one or more pairs of attribute names and key types:

- **HASH** - partition key
- **RANGE** - sort key

**Note**

The partition key of an item is also known as its hash attribute. The term "hash attribute" derives from DynamoDB's usage of an internal hash function to evenly distribute data items across partitions, based on their partition key values.

The sort key of an item is also known as its range attribute. The term "range attribute" derives from the way DynamoDB stores items with the same partition key physically close together, in sorted order by the sort key value.

Type: Array of KeySchemaElement (p. 276) objects

Array Members: Minimum number of 1 item. Maximum number of 2 items.

Required: No

**Projection**

 Represents attributes that are copied (projected) from the table into the global secondary index. These are in addition to the primary key attributes and index key attributes, which are automatically projected.

Type: Projection (p. 284) object

Required: No

**ProvisionedThroughput**

Represents the provisioned throughput settings for the specified global secondary index.

For current minimum and maximum provisioned throughput values, see Limits in the Amazon DynamoDB Developer Guide.

Type: ProvisionedThroughputDescription (p. 286) object

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
GlobalSecondaryIndexInfo
Service: Amazon DynamoDB

Represents the properties of a global secondary index for the table when the backup was created.

Contents

Note
In the following list, the required parameters are described first.

IndexName

The name of the global secondary index.

Type: String
Pattern: [a-zA-Z0-9_.-]+

Required: No

KeySchema

The complete key schema for a global secondary index, which consists of one or more pairs of attribute names and key types:
• HASH - partition key
• RANGE - sort key

Note
The partition key of an item is also known as its hash attribute. The term "hash attribute" derives from DynamoDB's usage of an internal hash function to evenly distribute data items across partitions, based on their partition key values.
The sort key of an item is also known as its range attribute. The term "range attribute" derives from the way DynamoDB stores items with the same partition key physically close together, in sorted order by the sort key value.

Type: Array of KeySchemaElement (p. 276) objects
Array Members: Minimum number of 1 item. Maximum number of 2 items.

Required: No

Projection

Represents attributes that are copied (projected) from the table into the global secondary index. These are in addition to the primary key attributes and index key attributes, which are automatically projected.

Type: Projection (p. 284) object

Required: No

ProvisionedThroughput

Represents the provisioned throughput settings for the specified global secondary index.

Type: ProvisionedThroughput (p. 285) object

Required: No
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
GlobalSecondaryIndexUpdate

Service: Amazon DynamoDB

Represents one of the following:

- A new global secondary index to be added to an existing table.
- New provisioned throughput parameters for an existing global secondary index.
- An existing global secondary index to be removed from an existing table.

Contents

**Note**
In the following list, the required parameters are described first.

**Create**

The parameters required for creating a global secondary index on an existing table:

- IndexName
- KeySchema
- AttributeDefinitions
- Projection
- ProvisionedThroughput

Type: `CreateGlobalSecondaryIndexAction (p. 252)` object

Required: No

**Delete**

The name of an existing global secondary index to be removed.

Type: `DeleteGlobalSecondaryIndexAction (p. 255)` object

Required: No

**Update**

The name of an existing global secondary index, along with new provisioned throughput settings to be applied to that index.

Type: `UpdateGlobalSecondaryIndexAction (p. 305)` object

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
GlobalTable
Service: Amazon DynamoDB

Represents the properties of a global table.

Contents

Note
In the following list, the required parameters are described first.

GlobalTableName
The global table name.
Type: String
Pattern: [a-zA-Z0-9_.-]+
Required: No

ReplicationGroup
The regions where the global table has replicas.
Type: Array of Replica (p. 289) objects
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
GlobalTableDescription

Service: Amazon DynamoDB

Contains details about the global table.

Contents

**Note**
In the following list, the required parameters are described first.

**CreationDateTime**

The creation time of the global table.

Type: Timestamp

Required: No

**GlobalTableArn**

The unique identifier of the global table.

Type: String

Required: No

**GlobalTableName**

The global table name.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: No

**GlobalTableStatus**

The current state of the global table:

- **CREATING** - The global table is being created.
- **UPDATING** - The global table is being updated.
- **DELETING** - The global table is being deleted.
- **ACTIVE** - The global table is ready for use.

Type: String

Valid Values: CREATING | ACTIVE | DELETING | UPDATING

Required: No

**ReplicationGroup**

The regions where the global table has replicas.

Type: Array of ReplicaDescription (p. 290) objects

Required: No
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
**ItemCollectionMetrics**

Service: Amazon DynamoDB

Information about item collections, if any, that were affected by the operation. ItemCollectionMetrics is only returned if the request asked for it. If the table does not have any local secondary indexes, this information is not returned in the response.

**Contents**

**Note**

In the following list, the required parameters are described first.

**ItemCollectionKey**

The partition key value of the item collection. This value is the same as the partition key value of the item.

Type: String to `AttributeValue (p. 235)` object map

Key Length Constraints: Maximum length of 65535.

Required: No

**SizeEstimateRangeGB**

An estimate of item collection size, in gigabytes. This value is a two-element array containing a lower bound and an upper bound for the estimate. The estimate includes the size of all the items in the table, plus the size of all attributes projected into all of the local secondary indexes on that table. Use this estimate to measure whether a local secondary index is approaching its size limit.

The estimate is subject to change over time; therefore, do not rely on the precision or accuracy of the estimate.

Type: Array of doubles

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
KeysAndAttributes
Service: Amazon DynamoDB

Represents a set of primary keys and, for each key, the attributes to retrieve from the table.

For each primary key, you must provide all of the key attributes. For example, with a simple primary key, you only need to provide the partition key. For a composite primary key, you must provide both the partition key and the sort key.

Contents

Note
In the following list, the required parameters are described first.

Keys
The primary key attribute values that define the items and the attributes associated with the items.

Type: Array of string to AttributeValue (p. 235) object maps
Array Members: Minimum number of 1 item. Maximum number of 100 items.

Key Length Constraints: Maximum length of 65535.

Required: Yes

AttributesToGet
This is a legacy parameter. Use ProjectionExpression instead. For more information, see Legacy Conditional Parameters in the Amazon DynamoDB Developer Guide.

Type: Array of strings
Array Members: Minimum number of 1 item.

Length Constraints: Maximum length of 65535.

Required: No

ConsistentRead
The consistency of a read operation. If set to true, then a strongly consistent read is used; otherwise, an eventually consistent read is used.

Type: Boolean

Required: No

ExpressionAttributeNames
One or more substitution tokens for attribute names in an expression. The following are some use cases for using ExpressionAttributeNames:
- To access an attribute whose name conflicts with a DynamoDB reserved word.
- To create a placeholder for repeating occurrences of an attribute name in an expression.
- To prevent special characters in an attribute name from being misinterpreted in an expression.

Use the # character in an expression to dereference an attribute name. For example, consider the following attribute name:

- Percentile

The name of this attribute conflicts with a reserved word, so it cannot be used directly in an expression. (For the complete list of reserved words, see Reserved Words in the Amazon
To work around this, you could specify the following for ExpressionAttributeNames:

- `{"#P":"Percentile"}`

You could then use this substitution in an expression, as in this example:

- `#P = :val`

**Note**

Tokens that begin with the `:` character are *expression attribute values*, which are placeholders for the actual value at runtime.

For more information on expression attribute names, see [Accessing Item Attributes](https://docs.aws.amazon.com/AmazonDynamoDB/latest/DeveloperGuide/) in the *Amazon DynamoDB Developer Guide*.

**ProjectionExpression**

A string that identifies one or more attributes to retrieve from the table. These attributes can include scalars, sets, or elements of a JSON document. The attributes in the *ProjectionExpression* must be separated by commas.

If no attribute names are specified, then all attributes will be returned. If any of the requested attributes are not found, they will not appear in the result.

For more information, see [Accessing Item Attributes](https://docs.aws.amazon.com/AmazonDynamoDB/latest/DeveloperGuide/) in the *Amazon DynamoDB Developer Guide*.

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- [AWS SDK for C++](https://docs.aws.amazon.com/AmazonDynamoDB/latest/DeveloperGuide/)
- [AWS SDK for Go](https://docs.aws.amazon.com/AmazonDynamoDB/latest/DeveloperGuide/)
- [AWS SDK for Java](https://docs.aws.amazon.com/AmazonDynamoDB/latest/DeveloperGuide/)
- [AWS SDK for Ruby V2](https://docs.aws.amazon.com/AmazonDynamoDB/latest/DeveloperGuide/)
KeySchemaElement

Service: Amazon DynamoDB

Represents a single element of a key schema. A key schema specifies the attributes that make up the primary key of a table, or the key attributes of an index.

A KeySchemaElement represents exactly one attribute of the primary key. For example, a simple primary key would be represented by one KeySchemaElement (for the partition key). A composite primary key would require one KeySchemaElement for the partition key, and another KeySchemaElement for the sort key.

A KeySchemaElement must be a scalar, top-level attribute (not a nested attribute). The data type must be one of String, Number, or Binary. The attribute cannot be nested within a List or a Map.

Contents

Note
In the following list, the required parameters are described first.

AttributeName
The name of a key attribute.

Type: String
Length Constraints: Minimum length of 1. Maximum length of 255.
Required: Yes

KeyType
The role that this key attribute will assume:
- HASH - partition key
- RANGE - sort key

Note
The partition key of an item is also known as its hash attribute. The term "hash attribute" derives from DynamoDB’s usage of an internal hash function to evenly distribute data items across partitions, based on their partition key values.
The sort key of an item is also known as its range attribute. The term "range attribute" derives from the way DynamoDB stores items with the same partition key physically close together, in sorted order by the sort key value.

Type: String
Valid Values: HASH | RANGE
Required: Yes

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
LocalSecondaryIndex

Service: Amazon DynamoDB

Represents the properties of a local secondary index.

Contents

Note
In the following list, the required parameters are described first.

IndexName

The name of the local secondary index. The name must be unique among all other indexes on this table.

Type: String
Pattern: [a-zA-Z0-9_\-]+
Required: Yes

KeySchema

The complete key schema for the local secondary index, consisting of one or more pairs of attribute names and key types:
• HASH - partition key
• RANGE - sort key

Note
The partition key of an item is also known as its hash attribute. The term "hash attribute" derives from DynamoDB's usage of an internal hash function to evenly distribute data items across partitions, based on their partition key values.
The sort key of an item is also known as its range attribute. The term "range attribute" derives from the way DynamoDB stores items with the same partition key physically close together, in sorted order by the sort key value.

Type: Array of KeySchemaElement (p. 276) objects
Array Members: Minimum number of 1 item. Maximum number of 2 items.
Required: Yes

Projection

Represents attributes that are copied (projected) from the table into the local secondary index. These are in addition to the primary key attributes and index key attributes, which are automatically projected.

Type: Projection (p. 284) object
Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:
• AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
LocalSecondaryIndexDescription

Service: Amazon DynamoDB

Represents the properties of a local secondary index.

Contents

**Note**

In the following list, the required parameters are described first.

**IndexArn**

The Amazon Resource Name (ARN) that uniquely identifies the index.

Type: String

Required: No

**IndexName**

Represents the name of the local secondary index.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: No

**IndexSizeBytes**

The total size of the specified index, in bytes. DynamoDB updates this value approximately every six hours. Recent changes might not be reflected in this value.

Type: Long

Required: No

**ItemCount**

The number of items in the specified index. DynamoDB updates this value approximately every six hours. Recent changes might not be reflected in this value.

Type: Long

Required: No

**KeySchema**

The complete key schema for the local secondary index, consisting of one or more pairs of attribute names and key types:

- **HASH** - partition key
- **RANGE** - sort key

**Note**

The partition key of an item is also known as its hash attribute. The term "hash attribute" derives from DynamoDB' usage of an internal hash function to evenly distribute data items across partitions, based on their partition key values.

The sort key of an item is also known as its range attribute. The term "range attribute" derives from the way DynamoDB stores items with the same partition key physically close together, in sorted order by the sort key value.
Type: Array of KeySchemaElement (p. 276) objects

Array Members: Minimum number of 1 item. Maximum number of 2 items.

Required: No

**Projection**

Represents attributes that are copied (projected) from the table into the global secondary index. These are in addition to the primary key attributes and index key attributes, which are automatically projected.

Type: Projection (p. 284) object

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
LocalSecondaryIndexInfo

Service: Amazon DynamoDB

Represents the properties of a local secondary index for the table when the backup was created.

Contents

**Note**
In the following list, the required parameters are described first.

**IndexName**

Represents the name of the local secondary index.

Type: String


Pattern: `^[a-zA-Z0-9_.-]+$`

Required: No

**KeySchema**

The complete key schema for a local secondary index, which consists of one or more pairs of attribute names and key types:

- `HASH` - partition key
- `RANGE` - sort key

**Note**
The partition key of an item is also known as its *hash attribute*. The term "hash attribute" derives from DynamoDB’ usage of an internal hash function to evenly distribute data items across partitions, based on their partition key values.

The sort key of an item is also known as its *range attribute*. The term "range attribute" derives from the way DynamoDB stores items with the same partition key physically close together, in sorted order by the sort key value.

Type: Array of `KeySchemaElement` (p. 276) objects

Array Members: Minimum number of 1 item. Maximum number of 2 items.

Required: No

**Projection**

Represents attributes that are copied (projected) from the table into the global secondary index. These are in addition to the primary key attributes and index key attributes, which are automatically projected.

Type: `Projection` (p. 284) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
• AWS SDK for Java
• AWS SDK for Ruby V2
Projection
Service: Amazon DynamoDB

Represents attributes that are copied (projected) from the table into an index. These are in addition to
the primary key attributes and index key attributes, which are automatically projected.

Contents

Note
In the following list, the required parameters are described first.

NonKeyAttributes

Represents the non-key attribute names which will be projected into the index.

For local secondary indexes, the total count of NonKeyAttributes summed across all of the local
secondary indexes, must not exceed 20. If you project the same attribute into two different indexes,
this counts as two distinct attributes when determining the total.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Length Constraints: Minimum length of 1. Maximum length of 255.

Required: No

ProjectionType

The set of attributes that are projected into the index:
• KEYS_ONLY - Only the index and primary keys are projected into the index.
• INCLUDE - Only the specified table attributes are projected into the index. The list of projected
  attributes are in NonKeyAttributes.
• ALL - All of the table attributes are projected into the index.

Type: String

Valid Values: ALL | KEYS_ONLY | INCLUDE

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for Ruby V2
ProvisionedThroughput

Service: Amazon DynamoDB

Represents the provisioned throughput settings for a specified table or index. The settings can be modified using the UpdateTable operation.

For current minimum and maximum provisioned throughput values, see Limits in the Amazon DynamoDB Developer Guide.

Contents

Note
In the following list, the required parameters are described first.

ReadCapacityUnits
The maximum number of strongly consistent reads consumed per second before DynamoDB returns a ThrottlingException. For more information, see Specifying Read and Write Requirements in the Amazon DynamoDB Developer Guide.

Type: Long
Valid Range: Minimum value of 1.
Required: Yes

WriteCapacityUnits
The maximum number of writes consumed per second before DynamoDB returns a ThrottlingException. For more information, see Specifying Read and Write Requirements in the Amazon DynamoDB Developer Guide.

Type: Long
Valid Range: Minimum value of 1.
Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ProvisionedThroughputDescription
Service: Amazon DynamoDB

Represents the provisioned throughput settings for the table, consisting of read and write capacity units, along with data about increases and decreases.

Contents

Note
In the following list, the required parameters are described first.

LastDecreaseDateTime
The date and time of the last provisioned throughput decrease for this table.
Type: Timestamp
Required: No

LastIncreaseDateTime
The date and time of the last provisioned throughput increase for this table.
Type: Timestamp
Required: No

NumberOfDecreasesToday
The number of provisioned throughput decreases for this table during this UTC calendar day. For current maximums on provisioned throughput decreases, see Limits in the Amazon DynamoDB Developer Guide.
Type: Long
Valid Range: Minimum value of 1.
Required: No

ReadCapacityUnits
The maximum number of strongly consistent reads consumed per second before DynamoDB returns a ThrottlingException. Eventually consistent reads require less effort than strongly consistent reads, so a setting of 50 ReadCapacityUnits per second provides 100 eventually consistent ReadCapacityUnits per second.
Type: Long
Valid Range: Minimum value of 1.
Required: No

WriteCapacityUnits
The maximum number of writes consumed per second before DynamoDB returns a ThrottlingException.
Type: Long
Valid Range: Minimum value of 1.
Required: No
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
PutRequest

Service: Amazon DynamoDB

Represents a request to perform a PutItem operation on an item.

Contents

Note
In the following list, the required parameters are described first.

Item
A map of attribute name to attribute values, representing the primary key of an item to be processed by PutItem. All of the table's primary key attributes must be specified, and their data types must match those of the table's key schema. If any attributes are present in the item which are part of an index key schema for the table, their types must match the index key schema.

Type: String to AttributeValue (p. 235) object map

Key Length Constraints: Maximum length of 65535.

Required: Yes

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Replica
Service: Amazon DynamoDB

Represents the properties of a replica.

Contents

Note
In the following list, the required parameters are described first.

RegionName
The region where the replica needs to be created.
Type: String
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ReplicaDescription
Service: Amazon DynamoDB

Contains the details of the replica.

Contents

Note
In the following list, the required parameters are described first.

RegionName
The name of the region.
Type: String
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ReplicaUpdate
Service: Amazon DynamoDB

Represents one of the following:
- A new replica to be added to an existing global table.
- New parameters for an existing replica.
- An existing replica to be removed from an existing global table.

Contents

Note
In the following list, the required parameters are described first.

Create
The parameters required for creating a replica on an existing global table.

Type: CreateReplicaAction (p. 254) object

Required: No

Delete
The name of the existing replica to be removed.

Type: DeleteReplicaAction (p. 256) object

Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
RestoreSummary
Service: Amazon DynamoDB
Contains details for the restore.

Contents

Note
In the following list, the required parameters are described first.

RestoreDateTime
Point in time or source backup time.
Type: Timestamp
Required: Yes

RestoreInProgress
Indicates if a restore is in progress or not.
Type: Boolean
Required: Yes

SourceBackupArn
ARN of the backup from which the table was restored.
Type: String
Required: No

SourceTableArn
ARN of the source table of the backup that is being restored.
Type: String
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for Ruby V2
SourceTableDetails
Service: Amazon DynamoDB

Contains the details of the table when the backup was created.

Contents

Note
In the following list, the required parameters are described first.

KeySchema
Schema of the table.
Type: Array of KeySchemaElement (p. 276) objects
Array Members: Minimum number of 1 item. Maximum number of 2 items.
Required: Yes

ProvisionedThroughput
Read IOPs and Write IOPS on the table when the backup was created.
Type: ProvisionedThroughput (p. 285) object
Required: Yes

TableCreationDateTime
Time when the source table was created.
Type: Timestamp
Required: Yes

TableId
Unique identifier for the table for which the backup was created.
Type: String
Pattern: [0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}
Required: Yes

TableName
The name of the table for which the backup was created.
Type: String
Pattern: [a-zA-Z0-9-.]+
Required: Yes

ItemCount
Number of items in the table. Please note this is an approximate value.
Type: Long
Valid Range: Minimum value of 0.
Required: No

**TableArn**

ARN of the table for which backup was created.

Type: String

Required: No

**TableSizeBytes**

Size of the table in bytes. Please note this is an approximate value.

Type: Long

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
SourceTableFeatureDetails
Service: Amazon DynamoDB

Contains the details of the features enabled on the table when the backup was created. For example, LSIs, GSIs, streams, TTL.

Contents

Note
In the following list, the required parameters are described first.

GlobalSecondaryIndexes

Represents the GSI properties for the table when the backup was created. It includes the IndexName, KeySchema, Projection and ProvisionedThroughput for the GSIs on the table at the time of backup.

Type: Array of GlobalSecondaryIndexInfo (p. 267) objects

Required: No

LocalSecondaryIndexes

Represents the LSI properties for the table when the backup was created. It includes the IndexName, KeySchema and Projection for the LSIs on the table at the time of backup.

Type: Array of LocalSecondaryIndexInfo (p. 282) objects

Required: No

StreamDescription

Stream settings on the table when the backup was created.

Type: StreamSpecification (p. 296) object

Required: No

TimeToLiveDescription

Time to Live settings on the table when the backup was created.

Type: TimeToLiveDescription (p. 303) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
StreamSpecification

Service: Amazon DynamoDB

Represents the DynamoDB Streams configuration for a table in DynamoDB.

Contents

**Note**
In the following list, the required parameters are described first.

**StreamEnabled**
Indicates whether DynamoDB Streams is enabled (true) or disabled (false) on the table.

Type: Boolean
Required: No

**StreamViewType**
When an item in the table is modified, StreamViewType determines what information is written to the stream for this table. Valid values for StreamViewType are:

- `KEYS_ONLY` - Only the key attributes of the modified item are written to the stream.
- `NEW_IMAGE` - The entire item, as it appears after it was modified, is written to the stream.
- `OLD_IMAGE` - The entire item, as it appeared before it was modified, is written to the stream.
- `NEW_AND_OLD_IMAGES` - Both the new and the old item images of the item are written to the stream.

Type: String

Valid Values: `NEW_IMAGE` | `OLD_IMAGE` | `NEW_AND_OLD_IMAGES` | `KEYS_ONLY`

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
TableDescription
Service: Amazon DynamoDB

Represents the properties of a table.

Contents

Note
In the following list, the required parameters are described first.

AttributeDefinitions

An array of AttributeDefinition objects. Each of these objects describes one attribute in the table and index key schema.

Each AttributeDefinition object in this array is composed of:
- AttributeName - The name of the attribute.
- AttributeType - The data type for the attribute.

Type: Array of AttributeDefinition (p. 234) objects
Required: No

CreationDateTime

The date and time when the table was created, in UNIX epoch time format.

Type: Timestamp
Required: No

GlobalSecondaryIndexes

The global secondary indexes, if any, on the table. Each index is scoped to a given partition key value. Each element is composed of:
- Backfilling - If true, then the index is currently in the backfilling phase. Backfilling occurs only when a new global secondary index is added to the table; it is the process by which DynamoDB populates the new index with data from the table. (This attribute does not appear for indexes that were created during a CreateTable operation.)
- IndexName - The name of the global secondary index.
- IndexSizeBytes - The total size of the global secondary index, in bytes. DynamoDB updates this value approximately every six hours. Recent changes might not be reflected in this value.
- IndexStatus - The current status of the global secondary index:
  - CREATING - The index is being created.
  - UPDATING - The index is being updated.
  - DELETING - The index is being deleted.
  - ACTIVE - The index is ready for use.
- ItemCount - The number of items in the global secondary index. DynamoDB updates this value approximately every six hours. Recent changes might not be reflected in this value.
- KeySchema - Specifies the complete index key schema. The attribute names in the key schema must be between 1 and 255 characters (inclusive). The key schema must begin with the same partition key as the table.
- Projection - Specifies attributes that are copied (projected) from the table into the index. These are in addition to the primary key attributes and index key attributes, which are automatically projected. Each attribute specification is composed of:
TableDescription

- **ProjectionType** - One of the following:
  - **KEYS_ONLY** - Only the index and primary keys are projected into the index.
  - **INCLUDE** - Only the specified table attributes are projected into the index. The list of projected attributes are in **NonKeyAttributes**.
  - **ALL** - All of the table attributes are projected into the index.
- **NonKeyAttributes** - A list of one or more non-key attribute names that are projected into the secondary index. The total count of attributes provided in **NonKeyAttributes**, summed across all of the secondary indexes, must not exceed 20. If you project the same attribute into two different indexes, this counts as two distinct attributes when determining the total.
- **ProvisionedThroughput** - The provisioned throughput settings for the global secondary index, consisting of read and write capacity units, along with data about increases and decreases.

If the table is in the **DELETING** state, no information about indexes will be returned.

**Type:** Array of **GlobalSecondaryIndexDescription (p. 264)** objects

**ItemCount**

The number of items in the specified table. DynamoDB updates this value approximately every six hours. Recent changes might not be reflected in this value.

**Type:** Long

**Required:** No

**KeySchema**

The primary key structure for the table. Each **KeySchemaElement** consists of:

- **AttributeName** - The name of the attribute.
- **KeyType** - The role of the attribute:
  - **HASH** - partition key
  - **RANGE** - sort key

**Note**

The partition key of an item is also known as its **hash attribute**. The term "hash attribute" derives from DynamoDB’ usage of an internal hash function to evenly distribute data items across partitions, based on their partition key values.

The sort key of an item is also known as its **range attribute**. The term "range attribute" derives from the way DynamoDB stores items with the same partition key physically close together, in sorted order by the sort key value.

For more information about primary keys, see [Primary Key](https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GlobalSecondaryIndexStructure.html) in the *Amazon DynamoDB Developer Guide*.

**Type:** Array of **KeySchemaElement (p. 276)** objects

**Array Members:** Minimum number of 1 item. Maximum number of 2 items.

**Required:** No

**LatestStreamArn**

The Amazon Resource Name (ARN) that uniquely identifies the latest stream for this table.

**Type:** String

**Length Constraints:** Minimum length of 37. Maximum length of 1024.
Required: No

**LatestStreamLabel**

A timestamp, in ISO 8601 format, for this stream.

Note that LatestStreamLabel is not a unique identifier for the stream, because it is possible that a stream from another table might have the same timestamp. However, the combination of the following three elements is guaranteed to be unique:

- the AWS customer ID.
- the table name.
- the StreamLabel.

Type: String

Required: No

**LocalSecondaryIndexes**

Represents one or more local secondary indexes on the table. Each index is scoped to a given partition key value. Tables with one or more local secondary indexes are subject to an item collection size limit, where the amount of data within a given item collection cannot exceed 10 GB. Each element is composed of:

- **IndexName** - The name of the local secondary index.
- **KeySchema** - Specifies the complete index key schema. The attribute names in the key schema must be between 1 and 255 characters (inclusive). The key schema must begin with the same partition key as the table.
- **Projection** - Specifies attributes that are copied (projected) from the table into the index. These are in addition to the primary key attributes and index key attributes, which are automatically projected. Each attribute specification is composed of:
  - **ProjectionType** - One of the following:
    - KEYS_ONLY - Only the index and primary keys are projected into the index.
    - INCLUDE - Only the specified table attributes are projected into the index. The list of projected attributes are in NonKeyAttributes.
    - ALL - All of the table attributes are projected into the index.
  - **NonKeyAttributes** - A list of one or more non-key attribute names that are projected into the secondary index. The total count of attributes provided in NonKeyAttributes, summed across all of the secondary indexes, must not exceed 20. If you project the same attribute into two different indexes, this counts as two distinct attributes when determining the total.
- **IndexSizeBytes** - Represents the total size of the index, in bytes. DynamoDB updates this value approximately every six hours. Recent changes might not be reflected in this value.
- **ItemCount** - Represents the number of items in the index. DynamoDB updates this value approximately every six hours. Recent changes might not be reflected in this value.

If the table is in the **DELETING** state, no information about indexes will be returned.

Type: Array of LocalSecondaryIndexDescription (p. 280) objects

Required: No

**ProvisionedThroughput**

The provisioned throughput settings for the table, consisting of read and write capacity units, along with data about increases and decreases.

Type: ProvisionedThroughputDescription (p. 286) object

Required: No
**TableDescription**

**RestoreSummary**
Contains details for the restore.
Type: RestoreSummary (p. 292) object
Required: No

**StreamSpecification**
The current DynamoDB Streams configuration for the table.
Type: StreamSpecification (p. 296) object
Required: No

**TableArn**
The Amazon Resource Name (ARN) that uniquely identifies the table.
Type: String
Required: No

**TableId**
Unique identifier for the table for which the backup was created.
Type: String
Pattern: [0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}
Required: No

**TableName**
The name of the table.
Type: String
Pattern: [a-zA-Z0-9_.-]+
Required: No

**TableSizeBytes**
The total size of the specified table, in bytes. DynamoDB updates this value approximately every six hours. Recent changes might not be reflected in this value.
Type: Long
Required: No

**TableStatus**
The current state of the table:
- CREATING - The table is being created.
- UPDATING - The table is being updated.
- DELETING - The table is being deleted.
- ACTIVE - The table is ready for use.
Type: String
Valid Values: CREATING | UPDATING | DELETING | ACTIVE

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Tag
Service: Amazon DynamoDB

Describes a tag. A tag is a key-value pair. You can add up to 50 tags to a single DynamoDB table.

AWS-assigned tag names and values are automatically assigned the aws: prefix, which the user cannot assign. AWS-assigned tag names do not count towards the tag limit of 50. User-assigned tag names have the prefix user: in the Cost Allocation Report. You cannot backdate the application of a tag.

For an overview on tagging DynamoDB resources, see Tagging for DynamoDB in the *Amazon DynamoDB Developer Guide*.

Contents

**Note**
In the following list, the required parameters are described first.

**Key**

The key of the tag. Tag keys are case sensitive. Each DynamoDB table can only have up to one tag with the same key. If you try to add an existing tag (same key), the existing tag value will be updated to the new value.

Type: String


Required: Yes

**Value**

The value of the tag. Tag values are case-sensitive and can be null.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 256.

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
TimeToLiveDescription
Service: Amazon DynamoDB

The description of the Time to Live (TTL) status on the specified table.

Contents

Note
In the following list, the required parameters are described first.

AttributeName

The name of the Time to Live attribute for items in the table.
Type: String
Length Constraints: Minimum length of 1. Maximum length of 255.
Required: No

TimeToLiveStatus

The Time to Live status for the table.
Type: String
Valid Values: ENABLING | DISABLING | ENABLED | DISABLED
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
TimeToLiveSpecification
Service: Amazon DynamoDB

Represents the settings used to enable or disable Time to Live for the specified table.

Contents

Note
In the following list, the required parameters are described first.

AttributeName
The name of the Time to Live attribute used to store the expiration time for items in the table.
Type: String
Length Constraints: Minimum length of 1. Maximum length of 255.
Required: Yes

Enabled
Indicates whether Time To Live is to be enabled (true) or disabled (false) on the table.
Type: Boolean
Required: Yes

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
UpdateGlobalSecondaryIndexAction

Service: Amazon DynamoDB

Represents the new provisioned throughput settings to be applied to a global secondary index.

Contents

Note

In the following list, the required parameters are described first.

IndexName

The name of the global secondary index to be updated.

Type: String


Pattern: \[a-zA-Z0-9_.-]+\]

Required: Yes

ProvisionedThroughput

Represents the provisioned throughput settings for the specified global secondary index.

For current minimum and maximum provisioned throughput values, see Limits in the Amazon DynamoDB Developer Guide.

Type: ProvisionedThroughput (p. 285) object

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
WriteRequest
Service: Amazon DynamoDB

Represents an operation to perform - either DeleteItem or PutItem. You can only request one of these operations, not both, in a single WriteRequest. If you do need to perform both of these operations, you will need to provide two separate WriteRequest objects.

Contents

Note
In the following list, the required parameters are described first.

DeleteRequest
A request to perform a DeleteItem operation.
Type: DeleteRequest (p. 257) object
Required: No

PutRequest
A request to perform a PutItem operation.
Type: PutRequest (p. 288) object
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Amazon DynamoDB Accelerator

The following data types are supported by Amazon DynamoDB Accelerator:

- Cluster (p. 308)
- Endpoint (p. 311)
- Event (p. 312)
- Node (p. 313)
- NodeTypeSpecificValue (p. 315)
- NotificationConfiguration (p. 316)
- Parameter (p. 317)
- ParameterGroup (p. 319)
- ParameterGroupStatus (p. 320)
- ParameterNameValue (p. 321)
- SecurityGroupMembership (p. 322)
• Subnet (p. 323)
• SubnetGroup (p. 324)
• Tag (p. 325)
**Cluster**

Service: Amazon DynamoDB Accelerator

Contains all of the attributes of a specific DAX cluster.

**Contents**

*Note*

In the following list, the required parameters are described first.

**ActiveNodes**

The number of nodes in the cluster that are active (i.e., capable of serving requests).

Type: Integer

Required: No

**ClusterArn**

The Amazon Resource Name (ARN) that uniquely identifies the cluster.

Type: String

Required: No

**ClusterDiscoveryEndpoint**

The configuration endpoint for this DAX cluster, consisting of a DNS name and a port number. Client applications can specify this endpoint, rather than an individual node endpoint, and allow the DAX client software to intelligently route requests and responses to nodes in the DAX cluster.

Type: `Endpoint (p. 311)` object

Required: No

**ClusterName**

The name of the DAX cluster.

Type: String

Required: No

**Description**

The description of the cluster.

Type: String

Required: No

**IamRoleArn**

A valid Amazon Resource Name (ARN) that identifies an IAM role. At runtime, DAX will assume this role and use the role's permissions to access DynamoDB on your behalf.

Type: String

Required: No

**NodeIdsToRemove**

A list of nodes to be removed from the cluster.
Nodes
A list of nodes that are currently in the cluster.
Type: Array of Node (p. 313) objects
Required: No

NodeType
The node type for the nodes in the cluster. (All nodes in a DAX cluster are of the same type.)
Type: String
Required: No

NotificationConfiguration
Describes a notification topic and its status. Notification topics are used for publishing DAX events to subscribers using Amazon Simple Notification Service (SNS).
Type: NotificationConfiguration (p. 316) object
Required: No

ParameterGroup
The parameter group being used by nodes in the cluster.
Type: ParameterGroupStatus (p. 320) object
Required: No

PreferredMaintenanceWindow
A range of time when maintenance of DAX cluster software will be performed. For example: sun:01:00–sun:09:00. Cluster maintenance normally takes less than 30 minutes, and is performed automatically within the maintenance window.
Type: String
Required: No

SecurityGroups
A list of security groups, and the status of each, for the nodes in the cluster.
Type: Array of SecurityGroupMembership (p. 322) objects
Required: No

Status
The current status of the cluster.
Type: String
Required: No

SubnetGroup
The subnet group where the DAX cluster is running.
Type: String
Cluster

Required: No

**TotalNodes**

The total number of nodes in the cluster.

Type: Integer

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Endpoint
Service: Amazon DynamoDB Accelerator

Represents the information required for client programs to connect to the configuration endpoint for a DAX cluster, or to an individual node within the cluster.

Contents

Note
In the following list, the required parameters are described first.

Address
The DNS hostname of the endpoint.
Type: String
Required: No

Port
The port number that applications should use to connect to the endpoint.
Type: Integer
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Event
Service: Amazon DynamoDB Accelerator

 Represents a single occurrence of something interesting within the system. Some examples of events are creating a DAX cluster, adding or removing a node, or rebooting a node.

Contents

Note
In the following list, the required parameters are described first.

Date
The date and time when the event occurred.
Type: Timestamp
Required: No

Message
A user-defined message associated with the event.
Type: String
Required: No

SourceName
The source of the event. For example, if the event occurred at the node level, the source would be the node ID.
Type: String
Required: No

SourceType
Specifies the origin of this event - a cluster, a parameter group, a node ID, etc.
Type: String
Valid Values: CLUSTER | PARAMETER_GROUP | SUBNET_GROUP
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Node
Service: Amazon DynamoDB Accelerator

Represents an individual node within a DAX cluster.

Contents

Note
In the following list, the required parameters are described first.

AvailabilityZone
The Availability Zone (AZ) in which the node has been deployed.
Type: String
Required: No

Endpoint
The endpoint for the node, consisting of a DNS name and a port number. Client applications can connect directly to a node endpoint, if desired (as an alternative to allowing DAX client software to intelligently route requests and responses to nodes in the DAX cluster.
Type: Endpoint (p. 311) object
Required: No

NodeCreateTime
The date and time (in UNIX epoch format) when the node was launched.
Type: Timestamp
Required: No

NodeId
A system-generated identifier for the node.
Type: String
Required: No

NodeStatus
The current status of the node. For example: available.
Type: String
Required: No

ParameterGroupStatus
The status of the parameter group associated with this node. For example, in-sync.
Type: String
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
NodeTypeSpecificValue
Service: Amazon DynamoDB Accelerator

Represents a parameter value that is applicable to a particular node type.

Contents

Note
In the following list, the required parameters are described first.

NodeType
A node type to which the parameter value applies.
Type: String
Required: No

Value
The parameter value for this node type.
Type: String
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
NotificationConfiguration

Service: Amazon DynamoDB Accelerator

Describes a notification topic and its status. Notification topics are used for publishing DAX events to subscribers using Amazon Simple Notification Service (SNS).

Contents

Note
In the following list, the required parameters are described first.

TopicArn
The Amazon Resource Name (ARN) that identifies the topic.

Type: String
Required: No

TopicStatus
The current state of the topic.

Type: String
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Parameter
Service: Amazon DynamoDB Accelerator

Describes an individual setting that controls some aspect of DAX behavior.

Contents

Note
In the following list, the required parameters are described first.

AllowedValues
A range of values within which the parameter can be set.
Type: String
Required: No

ChangeType
The conditions under which changes to this parameter can be applied. For example, requires-reboot indicates that a new value for this parameter will only take effect if a node is rebooted.
Type: String
Valid Values: IMMEDIATE | REQUIRES_REBOOT
Required: No

DataType
The data type of the parameter. For example, integer:
Type: String
Required: No

Description
A description of the parameter
Type: String
Required: No

IsModifiable
Whether the customer is allowed to modify the parameter.
Type: String
Valid Values: TRUE | FALSE | CONDITIONAL
Required: No

NodeTypeSpecificValues
A list of node types, and specific parameter values for each node.
Type: Array of NodeTypeSpecificValue (p. 315) objects
Required: No

ParameterName
The name of the parameter.
Parameter

Type: String
Required: No

**ParameterType**

Determines whether the parameter can be applied to any nodes, or only nodes of a particular type.

Type: String

Valid Values: DEFAULT | NODE_TYPE_SPECIFIC
Required: No

**ParameterValue**

The value for the parameter.

Type: String
Required: No

**Source**

How the parameter is defined. For example, system denotes a system-defined parameter.

Type: String
Required: No

### See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ParameterGroup
Service: Amazon DynamoDB Accelerator

A named set of parameters that are applied to all of the nodes in a DAX cluster.

Contents

Note
In the following list, the required parameters are described first.

Description

A description of the parameter group.

Type: String
Required: No

ParameterGroupName

The name of the parameter group.

Type: String
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ParameterGroupStatus
Service: Amazon DynamoDB Accelerator

The status of a parameter group.

Contents

Note
In the following list, the required parameters are described first.

NodeIdToReboot
The node IDs of one or more nodes to be rebooted.
Type: Array of strings
Required: No

ParameterApplyStatus
The status of parameter updates.
Type: String
Required: No

ParameterGroupName
The name of the parameter group.
Type: String
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ParameterNameValue

Service: Amazon DynamoDB Accelerator

An individual DAX parameter.

Contents

Note
In the following list, the required parameters are described first.

ParameterName

The name of the parameter.

Type: String
Required: No

ParameterValue

The value of the parameter.

Type: String
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
SecurityGroupMembership
Service: Amazon DynamoDB Accelerator
An individual VPC security group and its status.

Contents

Note
In the following list, the required parameters are described first.

SecurityGroupIdentifier
The unique ID for this security group.
Type: String
Required: No

Status
The status of this security group.
Type: String
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Subnet
Service: Amazon DynamoDB Accelerator

Represents the subnet associated with a DAX cluster. This parameter refers to subnets defined in Amazon Virtual Private Cloud (Amazon VPC) and used with DAX.

Contents

Note
In the following list, the required parameters are described first.

SubnetAvailabilityZone

The Availability Zone (AZ) for subnet subnet.

Type: String

Required: No

SubnetIdentifier

The system-assigned identifier for the subnet.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
SubnetGroup
Service: Amazon DynamoDB Accelerator

Represents the output of one of the following actions:

- CreateSubnetGroup
- ModifySubnetGroup

Contents

Note
In the following list, the required parameters are described first.

Description
The description of the subnet group.
Type: String
Required: No

SubnetGroupName
The name of the subnet group.
Type: String
Required: No

Subnets
A list of subnets associated with the subnet group.
Type: Array of Subnet (p. 323) objects
Required: No

VpcId
The Amazon Virtual Private Cloud identifier (VPC ID) of the subnet group.
Type: String
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Tag

Service: Amazon DynamoDB Accelerator

A description of a tag. Every tag is a key-value pair. You can add up to 50 tags to a single DAX cluster.

AWS-assigned tag names and values are automatically assigned the `aws:` prefix, which the user cannot assign. AWS-assigned tag names do not count towards the tag limit of 50. User-assigned tag names have the prefix `user:`.

You cannot backdate the application of a tag.

Contents

Note

In the following list, the required parameters are described first.

Key

The key for the tag. Tag keys are case sensitive. Every DAX cluster can only have one tag with the same key. If you try to add an existing tag (same key), the existing tag value will be updated to the new value.

Type: String

Required: No

Value

The value of the tag. Tag values are case-sensitive and can be null.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Amazon DynamoDB Streams

The following data types are supported by Amazon DynamoDB Streams:

- AttributeValue (p. 327)
- Identity (p. 329)
- KeySchemaElement (p. 330)
- Record (p. 331)
- SequenceNumberRange (p. 333)
- Shard (p. 334)
- Stream (p. 335)
- StreamDescription (p. 336)
- StreamRecord (p. 339)
AttributeValue

Service: Amazon DynamoDB Streams

Represents the data for an attribute. You can set one, and only one, of the elements.

Each attribute in an item is a name-value pair. An attribute can be single-valued or multi-valued set. For example, a book item can have title and authors attributes. Each book has one title but can have many authors. The multi-valued attribute is a set; duplicate values are not allowed.

Contents

Note
In the following list, the required parameters are described first.

B
A Binary data type.
Type: Base64-encoded binary data object
Required: No

BOOL
A Boolean data type.
Type: Boolean
Required: No

BS
A Binary Set data type.
Type: Array of Base64-encoded binary data objects
Required: No

L
A List data type.
Type: Array of AttributeValue (p. 327) objects
Required: No

M
A Map data type.
Type: String to AttributeValue (p. 327) object map
Key Length Constraints: Maximum length of 65535.
Required: No

N
A Number data type.
Type: String
Required: No
NS

A Number Set data type.
Type: Array of strings
Required: No

NULL

A Null data type.
Type: Boolean
Required: No

S

A String data type.
Type: String
Required: No

SS

A String Set data type.
Type: Array of strings
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Identity
Service: Amazon DynamoDB Streams

Contains details about the type of identity that made the request.

Contents

Note
In the following list, the required parameters are described first.

PrincipalId

A unique identifier for the entity that made the call. For Time To Live, the principalId is "dynamodb.amazonaws.com".

Type: String
Required: No

Type

The type of the identity. For Time To Live, the type is "Service".

Type: String
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
KeySchemaElement
Service: Amazon DynamoDB Streams

Represents a single element of a key schema. A key schema specifies the attributes that make up the primary key of a table, or the key attributes of an index.

A KeySchemaElement represents exactly one attribute of the primary key. For example, a simple primary key (partition key) would be represented by one KeySchemaElement. A composite primary key (partition key and sort key) would require one KeySchemaElement for the partition key, and another KeySchemaElement for the sort key.

Note
The partition key of an item is also known as its hash attribute. The term "hash attribute" derives from DynamoDB's usage of an internal hash function to evenly distribute data items across partitions, based on their partition key values.
The sort key of an item is also known as its range attribute. The term "range attribute" derives from the way DynamoDB stores items with the same partition key physically close together, in sorted order by the sort key value.

Contents

Note
In the following list, the required parameters are described first.

AttributeName

The name of a key attribute.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 255.

Required: Yes

KeyType

The attribute data, consisting of the data type and the attribute value itself.

Type: String

Valid Values: HASH | RANGE

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Record
Service: Amazon DynamoDB Streams

A description of a unique event within a stream.

Contents

Note
In the following list, the required parameters are described first.

awsRegion
The region in which the GetRecords request was received.
Type: String
Required: No
dynamodb
The main body of the stream record, containing all of the DynamoDB-specific fields.
Type: StreamRecord (p. 339) object
Required: No
eventID
A globally unique identifier for the event that was recorded in this stream record.
Type: String
Required: No
eventName
The type of data modification that was performed on the DynamoDB table:
• INSERT - a new item was added to the table.
• MODIFY - one or more of an existing item's attributes were modified.
• REMOVE - the item was deleted from the table
Type: String
Valid Values: INSERT | MODIFY | REMOVE
Required: No
eventSource
The AWS service from which the stream record originated. For DynamoDB Streams, this is aws:dynamodb.
Type: String
Required: No
eventVersion
The version number of the stream record format. This number is updated whenever the structure of Record is modified.
Client applications must not assume that eventVersion will remain at a particular value, as this number is subject to change at any time. In general, eventVersion will only increase as the low-level DynamoDB Streams API evolves.
Type: String
Required: No

**userIdentity**

Items that are deleted by the Time to Live process after expiration have the following fields:
- Records[].userIdentity.type
  - "Service"
- Records[].userIdentity.principalId
  - "dynamodb.amazonaws.com"

Type: Identity (p. 329) object
Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
SequenceNumberRange

Service: Amazon DynamoDB Streams

The beginning and ending sequence numbers for the stream records contained within a shard.

Contents

Note
In the following list, the required parameters are described first.

EndingSequenceNumber
The last sequence number.
Type: String
Required: No

StartingSequenceNumber
The first sequence number.
Type: String
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
**Shard**

Service: Amazon DynamoDB Streams

A uniquely identified group of stream records within a stream.

**Contents**

**Note**

In the following list, the required parameters are described first.

**ParentShardId**

  The shard ID of the current shard's parent.

  Type: String


  Required: No

**SequenceNumberRange**

  The range of possible sequence numbers for the shard.

  Type: SequenceNumberRange (p. 333) object

  Required: No

**ShardId**

  The system-generated identifier for this shard.

  Type: String


  Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Stream
Service: Amazon DynamoDB Streams

Represents all of the data describing a particular stream.

Contents

Note
In the following list, the required parameters are described first.

StreamArn

The Amazon Resource Name (ARN) for the stream.

Type: String


Required: No

StreamLabel

A timestamp, in ISO 8601 format, for this stream.

Note that LatestStreamLabel is not a unique identifier for the stream, because it is possible that a stream from another table might have the same timestamp. However, the combination of the following three elements is guaranteed to be unique:
- the AWS customer ID.
- the table name
- the StreamLabel

Type: String

Required: No

TableName

The DynamoDB table with which the stream is associated.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
StreamDescription

Service: Amazon DynamoDB Streams

Represents all of the data describing a particular stream.

Contents

**Note**

In the following list, the required parameters are described first.

**CreationRequestDateTime**

The date and time when the request to create this stream was issued.

Type: Timestamp

Required: No

**KeySchema**

The key attribute(s) of the stream's DynamoDB table.

Type: Array of KeySchemaElement (p. 330) objects

Array Members: Minimum number of 1 item. Maximum number of 2 items.

Required: No

**LastEvaluatedShardId**

The shard ID of the item where the operation stopped, inclusive of the previous result set. Use this value to start a new operation, excluding this value in the new request.

If LastEvaluatedShardId is empty, then the "last page" of results has been processed and there is currently no more data to be retrieved.

If LastEvaluatedShardId is not empty, it does not necessarily mean that there is more data in the result set. The only way to know when you have reached the end of the result set is when LastEvaluatedShardId is empty.

Type: String


Required: No

**Shards**

The shards that comprise the stream.

Type: Array of Shard (p. 334) objects

Required: No

**StreamArn**

The Amazon Resource Name (ARN) for the stream.

Type: String


Required: No
StreamLabel

A timestamp, in ISO 8601 format, for this stream.

Note that LatestStreamLabel is not a unique identifier for the stream, because it is possible that a stream from another table might have the same timestamp. However, the combination of the following three elements is guaranteed to be unique:

- the AWS customer ID.
- the table name
- the StreamLabel

Type: String

Required: No

StreamStatus

Indicates the current status of the stream:

- **ENABLING** - Streams is currently being enabled on the DynamoDB table.
- **ENABLED** - the stream is enabled.
- **DISABLING** - Streams is currently being disabled on the DynamoDB table.
- **DISABLED** - the stream is disabled.

Type: String

Valid Values: **ENABLING** | **ENABLED** | **DISABLING** | **DISABLED**

Required: No

StreamViewType

Indicates the format of the records within this stream:

- **KEYS_ONLY** - only the key attributes of items that were modified in the DynamoDB table.
- **NEW_IMAGE** - entire items from the table, as they appeared after they were modified.
- **OLD_IMAGE** - entire items from the table, as they appeared before they were modified.
- **NEW_AND_OLD_IMAGES** - both the new and the old images of the items from the table.

Type: String

Valid Values: **NEW_IMAGE** | **OLD_IMAGE** | **NEW_AND_OLD_IMAGES** | **KEYS_ONLY**

Required: No

TableName

The DynamoDB table with which the stream is associated.

Type: String


Pattern: [a-zA-Z0-9_.-]+

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for Ruby V2
StreamRecord
Service: Amazon DynamoDB Streams

A description of a single data modification that was performed on an item in a DynamoDB table.

Contents

Note
In the following list, the required parameters are described first.

ApproximateCreationDateTime
The approximate date and time when the stream record was created, in UNIX epoch time format.
Type: Timestamp
Required: No

Keys
The primary key attribute(s) for the DynamoDB item that was modified.
Type: String to AttributeValue (p. 327) object map
Key Length Constraints: Maximum length of 65535.
Required: No

NewImage
The item in the DynamoDB table as it appeared after it was modified.
Type: String to AttributeValue (p. 327) object map
Key Length Constraints: Maximum length of 65535.
Required: No

OldImage
The item in the DynamoDB table as it appeared before it was modified.
Type: String to AttributeValue (p. 327) object map
Key Length Constraints: Maximum length of 65535.
Required: No

SequenceNumber
The sequence number of the stream record.
Type: String
Required: No

SizeBytes
The size of the stream record, in bytes.
Type: Long
Valid Range: Minimum value of 1.
StreamViewType

The type of data from the modified DynamoDB item that was captured in this stream record:
- **KEYS_ONLY** - only the key attributes of the modified item.
- **NEW_IMAGE** - the entire item, as it appeared after it was modified.
- **OLD_IMAGE** - the entire item, as it appeared before it was modified.
- **NEW_AND_OLD_IMAGES** - both the new and the old item images of the item.

Type: String

Valid Values: NEW_IMAGE | OLD_IMAGE | NEW_AND_OLD_IMAGES | KEYS_ONLY

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Common Errors

This section lists the errors common to the API actions of all AWS services. For errors specific to an API action for this service, see the topic for that API action.

**AccessDeniedException**

You do not have sufficient access to perform this action.

HTTP Status Code: 400

**IncompleteSignature**

The request signature does not conform to AWS standards.

HTTP Status Code: 400

**InternalFailure**

The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

**InvalidAction**

The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

**InvalidClientTokenId**

The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

**InvalidParameterCombination**

Parameters that must not be used together were used together.

HTTP Status Code: 400

**InvalidParameterValue**

An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400

**InvalidQueryParameter**

The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400

**MalformedQueryString**

The query string contains a syntax error.

HTTP Status Code: 404

**MissingAction**

The request is missing an action or a required parameter.

HTTP Status Code: 400
**MissingAuthenticationToken**

The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403

**MissingParameter**

A required parameter for the specified action is not supplied.

HTTP Status Code: 400

**OptInRequired**

The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

**RequestExpired**

The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

**ServiceUnavailable**

The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

**ThrottlingException**

The request was denied due to request throttling.

HTTP Status Code: 400

**ValidationError**

The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400