Getting Started with DynamoDB

Use the hands-on tutorials in this section to help you get started and learn more about Amazon DynamoDB.

Topics

Basic Concepts in DynamoDB

Before you begin, you should familiarize yourself with the basic concepts in Amazon DynamoDB. For more information, see DynamoDB Core Components.

Then continue on to Prerequisites to learn about setting up DynamoDB.

Prerequisites - Getting Started Tutorial

Before starting the Amazon DynamoDB tutorial, follow the steps in Setting Up DynamoDB. Then continue on to Step 1: Create a Table (p. 61).

Note

- If you plan to interact with DynamoDB only through the AWS Management Console, you don't need an AWS access key. Complete the steps in Signing Up for AWS, and then continue on to Step 1: Create a Table (p. 61).
- If you don't want to sign up for a free tier account, you can set up DynamoDB Local (Downloadable Version). Then continue on to Step 1: Create a Table (p. 61).

Step 1: Create a Table

In this step, you create a `Music` table in Amazon DynamoDB. The table has the following details:

- Partition key — `Artist`
- Sort key — `SongTitle`

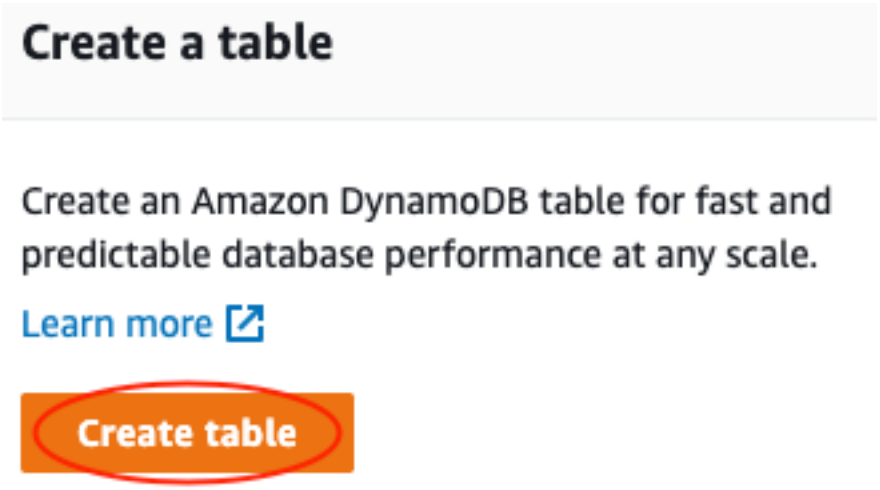
For more information about table operations, see Working with Tables and Data in DynamoDB (p. 332).

Note
Before you begin, make sure that you followed the steps in Prerequisites - Getting Started Tutorial (p. 61).

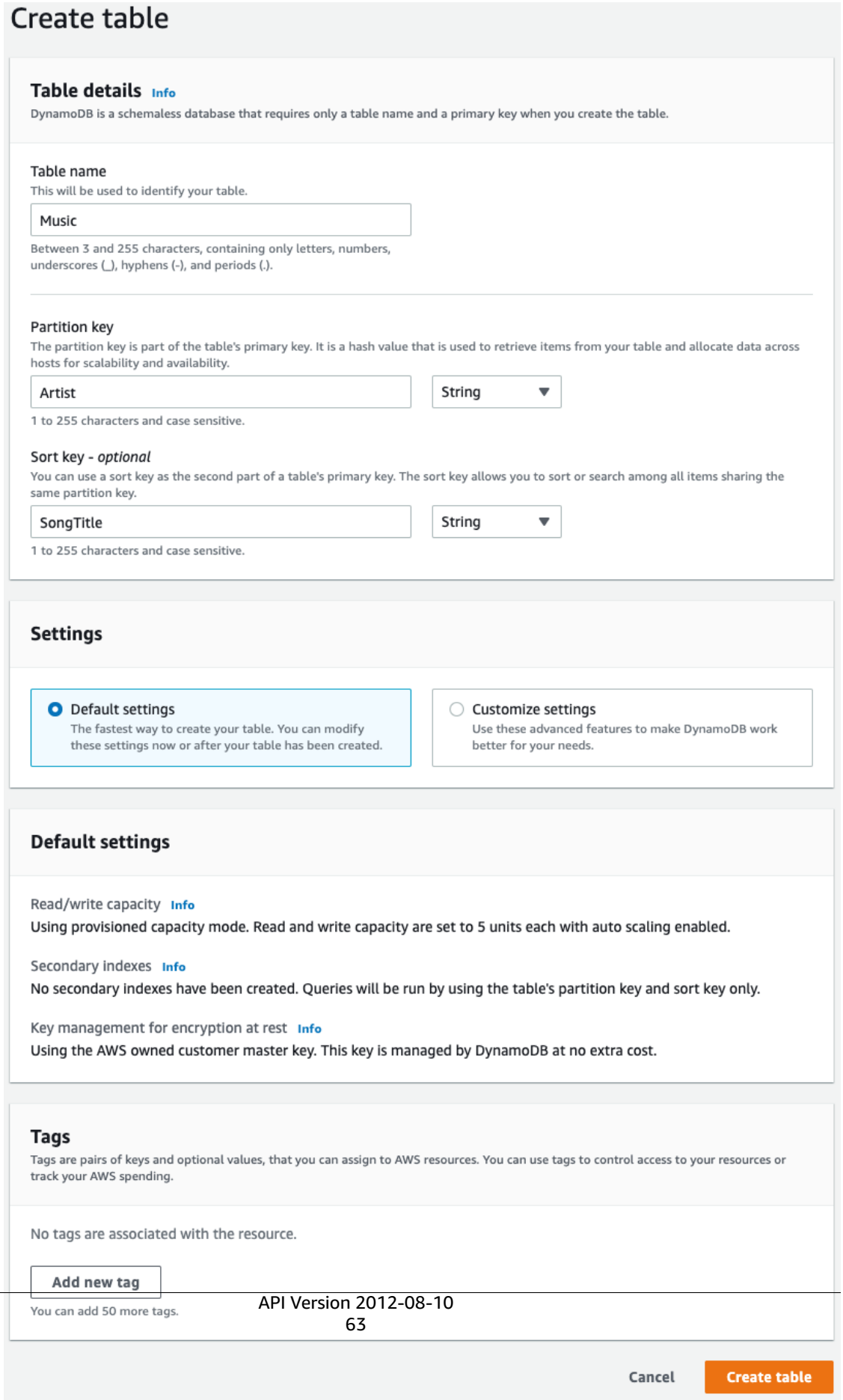
AWS Management Console

To create a new `Music` table using the DynamoDB console:

1. Sign in to the AWS Management Console and open the DynamoDB console at https://console.aws.amazon.com/dynamodb/.
2. In the navigation pane on the left side of the console, choose **Dashboard**.
3. On the right side of the console, choose **Create Table**.



4. Enter the table details as follows:
   a. For the table name, enter **`Music`**.
   b. For the partition key, enter **`Artist`**.
   c. Enter **`SongTitle`** as the sort key.
   d. Leave **Default settings** selected.
5. Choose **Create** to create the table.


Create table
Table details Info
DynamoDB is a schemaless database that requires only a table name and a primary key when you create the table.
Table name
This will be used to identify your table.
Music
Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.).
Partition key
The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability and availability.
Artist
String
1 to 255 characters and case sensitive.
Sort key - optional
You can use a sort key as the second part of a table's primary key. The sort key allows you to sort or search among all items sharing the same partition key.
SongTitle
String
1 to 255 characters and case sensitive.
Settings
Default settings
The fastest way to create your table. You can modify these settings now or after your table has been created.
Customize settings
Use these advanced features to make DynamoDB work better for your needs.
Default settings
Read/write capacity Info
Using provisioned capacity mode. Read and write capacity are set to 5 units each with auto scaling enabled.
Secondary indexes Info
No secondary indexes have been created. Queries will be run by using the table's partition key and sort key only.
Key management for encryption at rest Info
Using the AWS owned customer master key. This key is managed by DynamoDB at no extra cost.
Tags
Tags are pairs of keys and optional values, that you can assign to AWS resources. You can use tags to control access to your resources or track your AWS spending.
No tags are associated with the resource.
Add new tag
You can add 50 more tags.
Cancel
Create table

AWS CLI

The following AWS CLI example creates a new `Music` table using `create-table`.

```
aws dynamodb create-table \
    --table-name Music \
    --attribute-definitions \
        AttributeName=Artist,AttributeType=S \
        AttributeName=SongTitle,AttributeType=S \
    --key-schema \
        AttributeName=Artist,KeyType=HASH \
        AttributeName=SongTitle,KeyType=RANGE \
    --provisioned-throughput \
        ReadCapacityUnits=10,WriteCapacityUnits=5
```

Using `create-table` returns the following sample result.

```
{
    "TableDescription": {
        "TableArn": "arn:aws:dynamodb:us-west-2:522194210714:table/Music",
        "AttributeDefinitions": [
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "WriteCapacityUnits": 5,
            "ReadCapacityUnits": 10
        },
        "TableSizeBytes": 0,
        "TableName": "Music",
        "TableStatus": "CREATING",
        "TableId": "d04c7240-0e46-435d-b231-d54091fe1017",
        "KeySchema": [
            {
                "KeyType": "HASH",
                "AttributeName": "Artist"
            },
            {
                "KeyType": "RANGE",
                "AttributeName": "SongTitle"
            }
        ],
        "ItemCount": 0,
        "CreationDateTime": 1558028402.69
    }
}
```

Note that the value of the `TableStatus` field is set to `CREATING`.

To verify that DynamoDB has finished creating the `Music` table, use the `describe-table` command.

```
aws dynamodb describe-table --table-name Music | grep TableStatus
```

This command returns the following result. When DynamoDB finishes creating the table, the value of the `TableStatus` field is set to `ACTIVE`.

```
"TableStatus": "ACTIVE",
```

After creating the new table, proceed to Step 2: Write Data to a Table Using the Console or AWS CLI (p. 65).

Step 2: Write Data to a Table Using the Console or AWS CLI

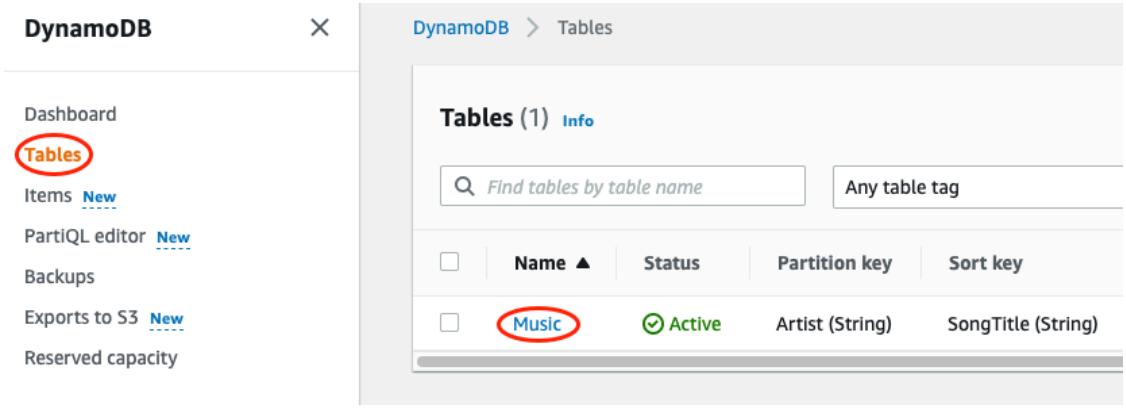
In this step, you insert two items into the `Music` table that you created in Step 1: Create a Table (p. 61).

For more information about write operations, see Writing an Item (p. 396).

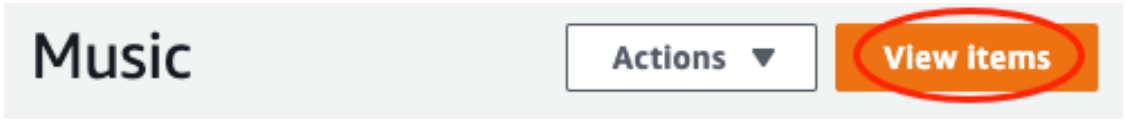
AWS Management Console

Follow these steps to write data to the `Music` table using the DynamoDB console.

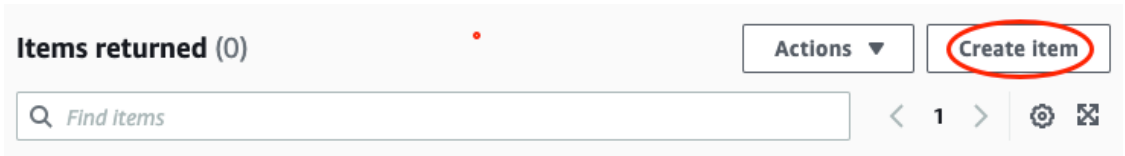
1. Open the DynamoDB console at https://console.aws.amazon.com/dynamodb/.
2. In the navigation pane on the left side of the console, choose **Tables**.
3. In the table list, choose the **Music** table.



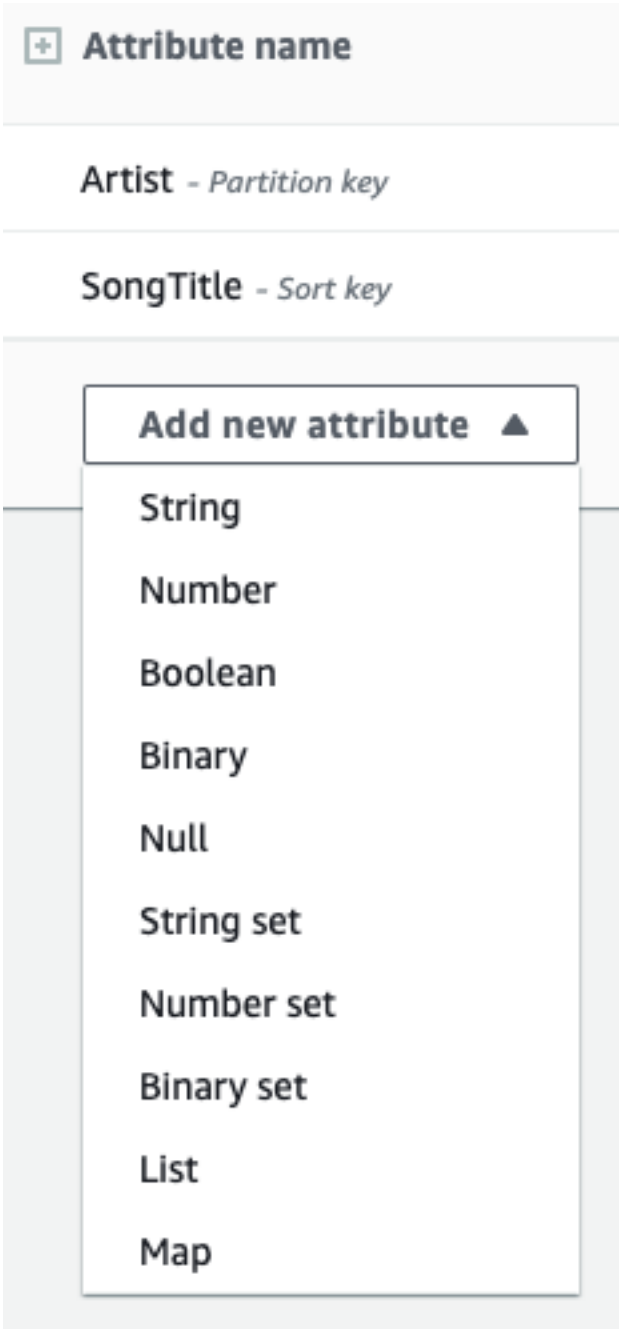
4. Select **View Items**.



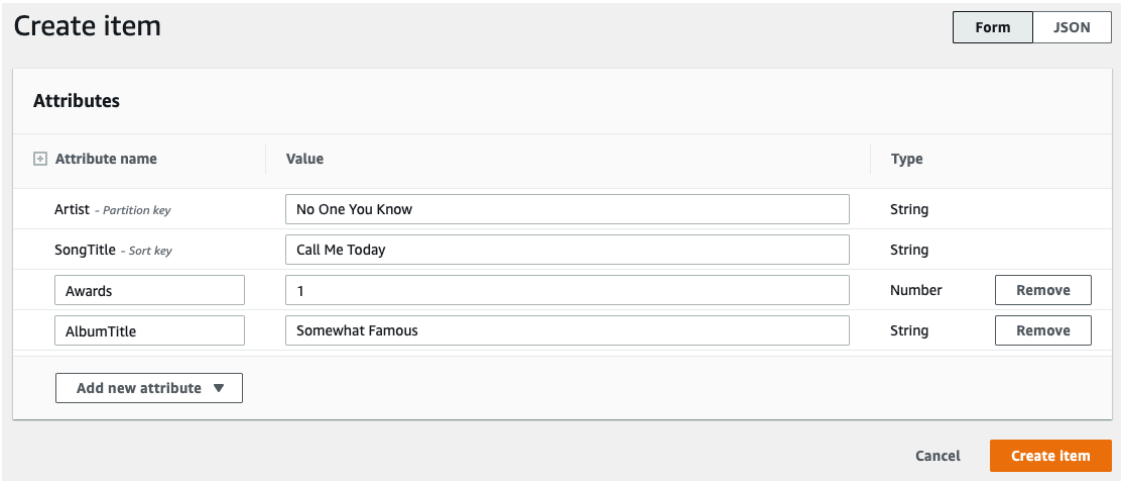
5. In the **Items** view, choose **Create item**.



6. Choose **Add new attribute**, and then choose **Number**. Name the field **`Awards`**.



7. Repeat this process to create an **`AlbumTitle`** of type **String**.
8. Enter the following values for your item:
   a. For **Artist**, enter **`No One You Know`** as the value.
   b. For **SongTitle**, enter **`Call Me Today`**.
   c. For **AlbumTitle**, enter **`Somewhat Famous`**.
   d. For **Awards**, enter **`1`**.
9. Choose **Create item**.



10. Repeat this process and create another item with the following values:

    a. For **Artist**, enter **`Acme Band`**.

    b. For **SongTitle** enter **`Happy Day`**.

    c. For **AlbumTitle**, enter **`Songs About Life`**.

    d. For **Awards**, enter **`10`**.

AWS CLI

The following AWS CLI example creates two new items in the `Music` table using `put-item`.

```
aws dynamodb put-item \
    --table-name Music  \
    --item \
        '{"Artist": {"S": "No One You Know"}, "SongTitle": {"S": "Call Me Today"},
 "AlbumTitle": {"S": "Somewhat Famous"}, "Awards": {"N": "1"}}'

aws dynamodb put-item \
    --table-name Music \
    --item \
        '{"Artist": {"S": "Acme Band"}, "SongTitle": {"S": "Happy Day"}, "AlbumTitle":
 {"S": "Songs About Life"}, "Awards": {"N": "10"} }'
```

For more information about supported data types in DynamoDB, see Data Types.

For more information about how to represent DynamoDB data types in JSON, see Attribute Values.

After writing data to your table, proceed to Step 3: Read Data from a Table (p. 67).

Step 3: Read Data from a Table

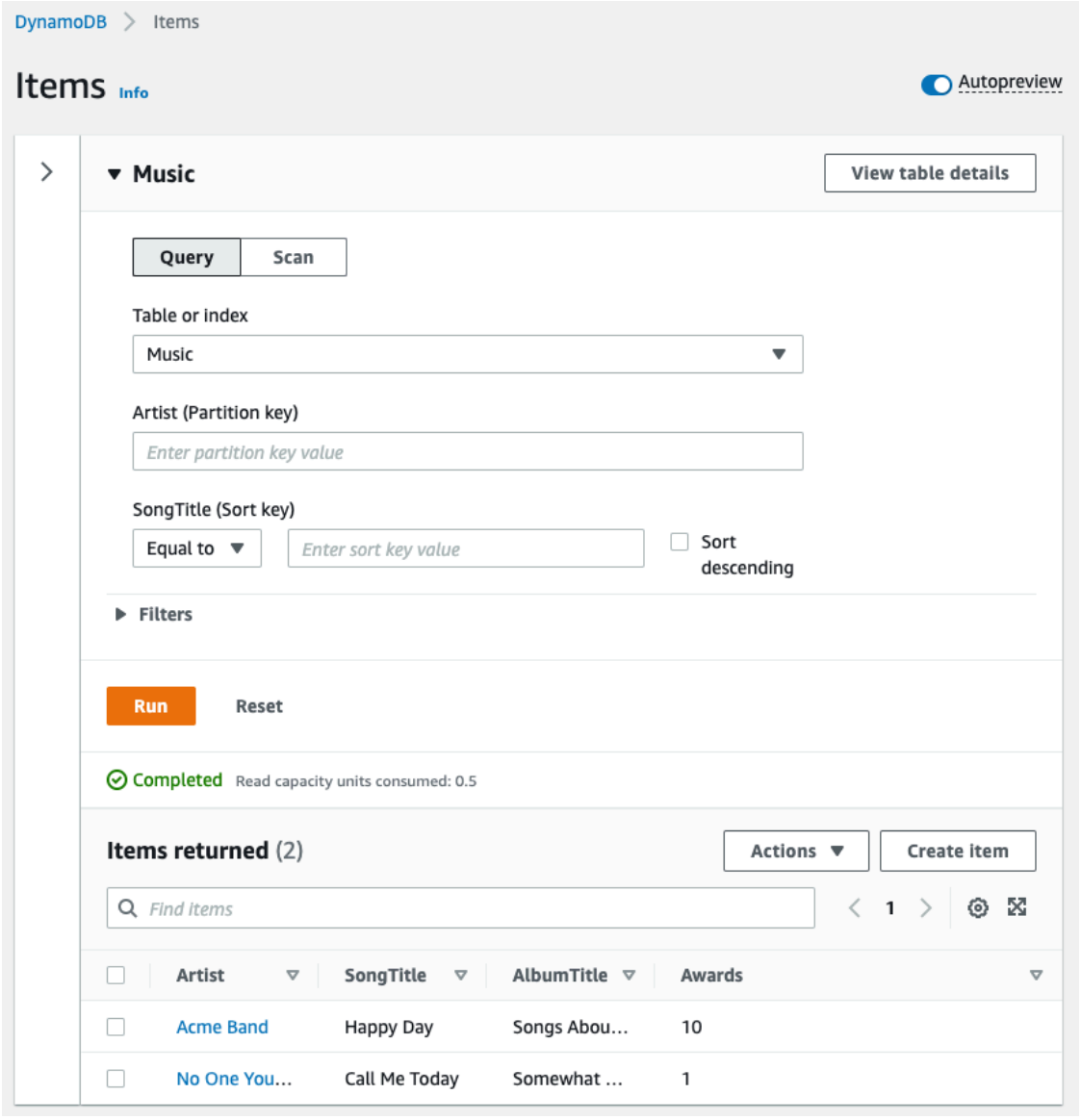
In this step, you will read back an item that was created in Step 2: Write Data to a Table Using the Console or AWS CLI (p. 65). You can use the DynamoDB console or the AWS CLI to read an item from the `Music` table by specifying `Artist` and `SongTitle`.

For more information about read operations in DynamoDB, see Reading an Item (p. 395).

AWS Management Console

Follow these steps to read data from the `Music` table using the DynamoDB console.

1. Open the DynamoDB console at https://console.aws.amazon.com/dynamodb/.
2. In the navigation pane on the left side of the console, choose **Tables**.
3. Choose the **Music** table from the table list.
4. Select the **View items**.
5. On the **Items** tab, view the list of items stored in the table, sorted by `Artist` and `SongTitle`. The first item in the list is the one with the `Artist` **Acme Band** and the `SongTitle` **Happy Day**.



AWS CLI

The following AWS CLI example reads an item from the `Music` table using `get-item`.

Note
The default behavior for DynamoDB is eventually consistent reads. The `consistent-read` parameter is used below to demonstrate strongly consistent reads.

```
aws dynamodb get-item --consistent-read \
    --table-name Music \
    --key '{ "Artist": {"S": "Acme Band"}, "SongTitle": {"S": "Happy Day"}}'
```

Using `get-item` returns the following sample result.

```
{
    "Item": {
        "AlbumTitle": {
            "S": "Songs About Life"
        },
        "Awards": {
            "N": "10"
        },
        "SongTitle": {
            "S": "Happy Day"
        },
        "Artist": {
            "S": "Acme Band"
        }
    }
}
```

To update the data in your table, proceed to Step 4: Update Data in a Table (p. 69).

Step 4: Update Data in a Table

In this step, you update an item that you created in Step 2: Write Data to a Table Using the Console or AWS CLI (p. 65). You can use the DynamoDB console or the AWS CLI to update the `AlbumTitle` of an item in the `Music` table by specifying `Artist`, `SongTitle`, and the updated `AlbumTitle`.

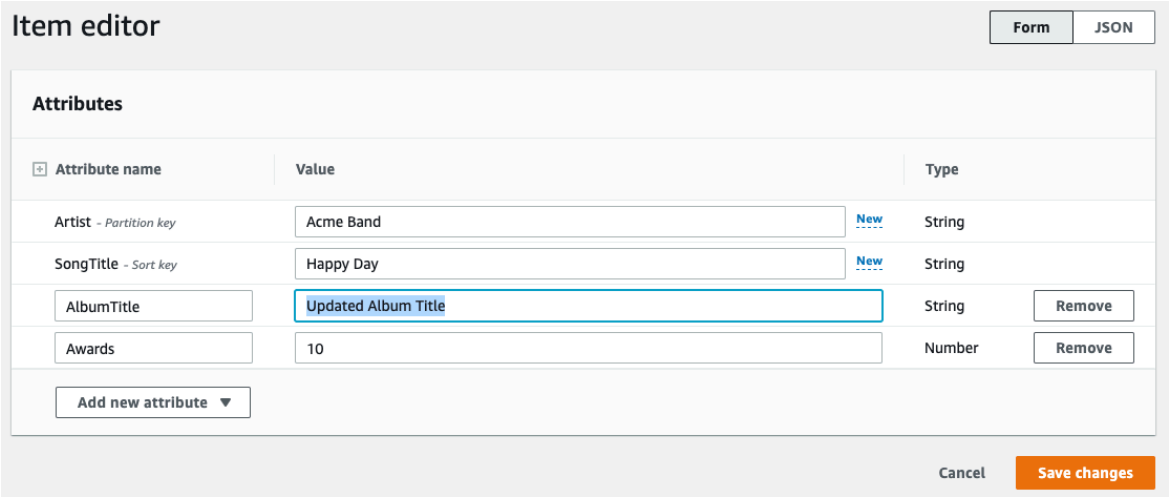
For more information about write operations, see Writing an Item (p. 396).

AWS Management Console

You can use the DynamoDB console to update data in the `Music` table.

1. Open the DynamoDB console at https://console.aws.amazon.com/dynamodb/.
2. In the navigation pane on the left side of the console, choose **Tables**.
3. Choose the **Music** table from the table list.
4. Choose **View items**.
5. Choose the item whose `Artist` value is **Acme Band** and `SongTitle` value is **Happy Day**.
6. Update the **AlbumTitle** value to **`Updated Album Title`**, and then choose **Save**.

The following image shows the updated item on the console.



AWS CLI

The following AWS CLI example updates an item in the `Music` table using `update-item`.

```
aws dynamodb update-item \
    --table-name Music \
    --key '{ "Artist": {"S": "Acme Band"}, "SongTitle": {"S": "Happy Day"}}' \
    --update-expression "SET AlbumTitle = :newval" \
    --expression-attribute-values '{":newval":{"S":"Updated Album Title"}}' \
    --return-values ALL_NEW
```

Using `update-item` returns the following sample result.

```
{
    "Attributes": {
        "AlbumTitle": {
            "S": "Updated Album Title"
        },
        "Awards": {
            "N": "10"
        },
        "SongTitle": {
            "S": "Happy Day"
        },
        "Artist": {
            "S": "Acme Band"
        }
    }
}
```

To query the data in the `Music` table, proceed to Step 5: Query Data in a Table (p. 70).

Step 5: Query Data in a Table

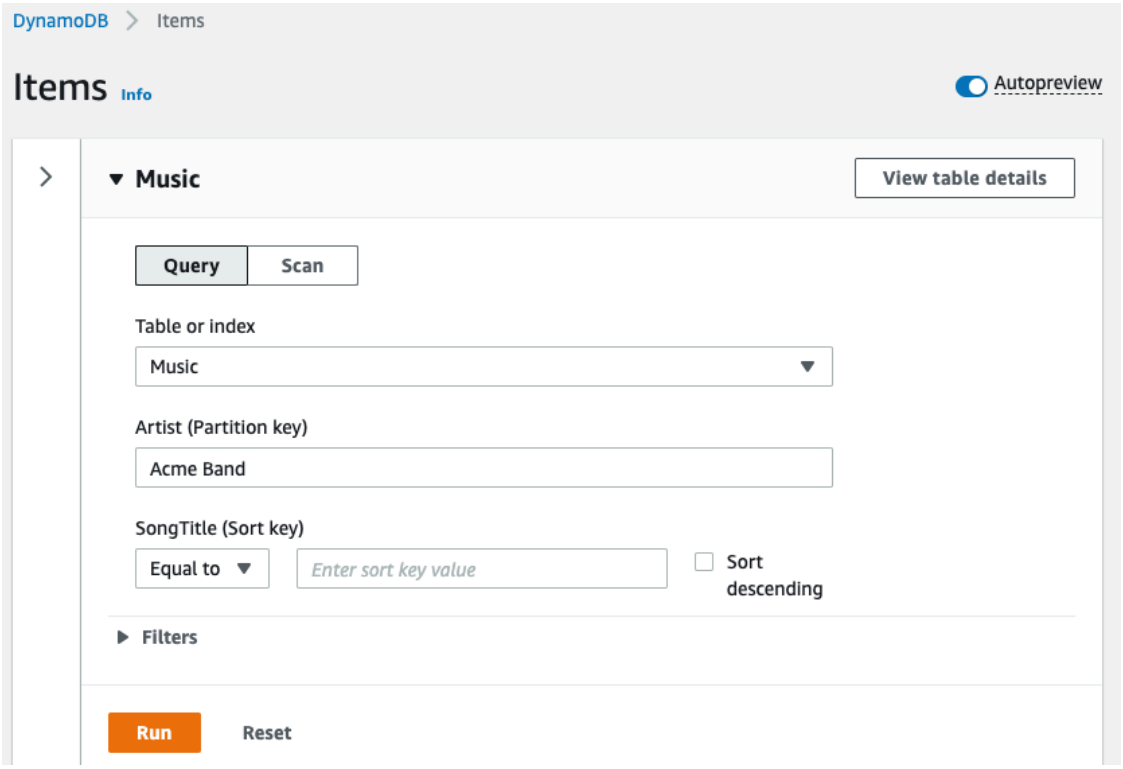
In this step, you query the data that you wrote to the `Music` table in the section called “Step 2: Write Data” (p. 65) by specifying `Artist`.

For more information about query operations, see Working with Queries in DynamoDB (p. 477).

AWS Management Console

Follow these steps to use the DynamoDB console to query data in the `Music` table.

1. Open the DynamoDB console at https://console.aws.amazon.com/dynamodb/.
2. In the navigation pane on the left side of the console, choose **Tables**.
3. Choose the **Music** table from the table list.
4. Choose **View items**.
5. Choose **Query**.
6. For **Partition key**, enter **`Acme Band`**, and then choose **Run**.



AWS CLI

The following AWS CLI example queries an item in the `Music` table using `query`.

```
aws dynamodb query \
    --table-name Music \
    --key-condition-expression "Artist = :name" \
    --expression-attribute-values  '{":name":{"S":"Acme Band"}}'
```

Using `query` returns the following sample result.

```
{
    "Count": 1,
    "Items": [
        {
            "AlbumTitle": {
```

```
                "S": "Updated Album Title"
            },
            "Awards": {
                "N": "10"
            },
            "SongTitle": {
                "S": "Happy Day"
            },
            "Artist": {
                "S": "Acme Band"
            }
        }
    ],
    "ScannedCount": 1,
    "ConsumedCapacity": null
}
```

To create a global secondary index for your table, proceed to Step 6: Create a Global Secondary Index (p. 72).

Step 6: Create a Global Secondary Index

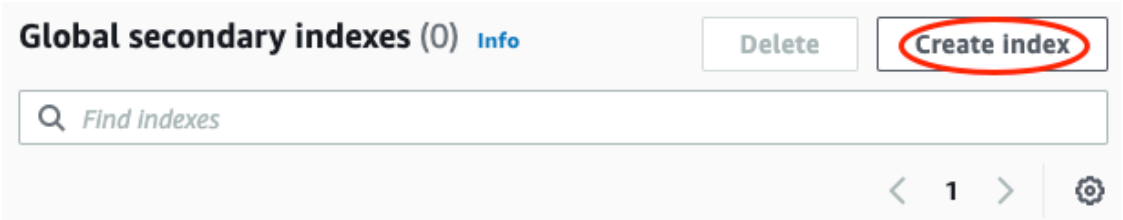
In this step, you create a global secondary index for the `Music` table that you created in Step 1: Create a Table (p. 61).

For more information about global secondary indexes, see Using Global Secondary Indexes in DynamoDB (p. 547).

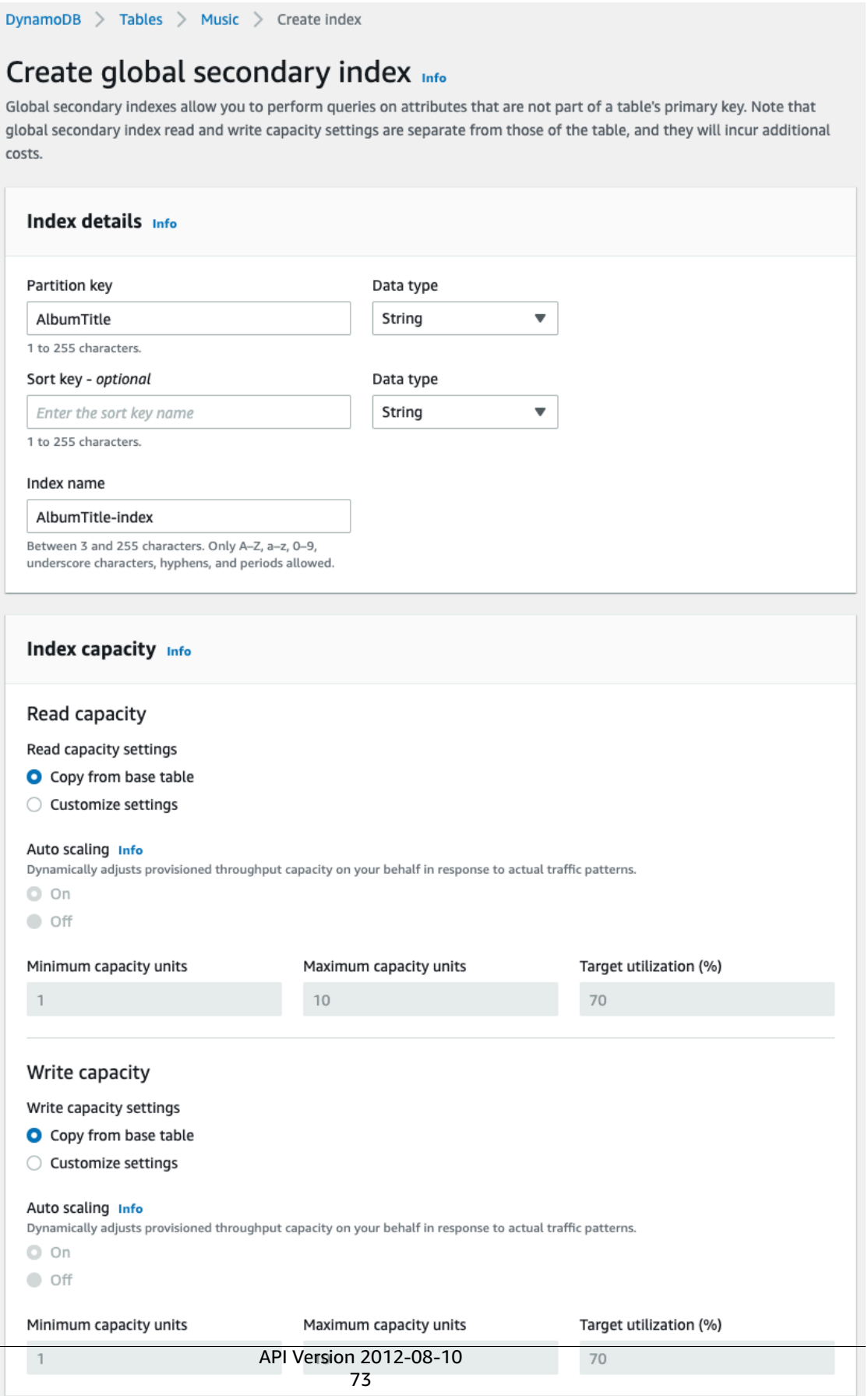
AWS Management Console

To use the Amazon DynamoDB console to create a global secondary index `AlbumTitle-index` for the `Music` table:

1. Open the DynamoDB console at https://console.aws.amazon.com/dynamodb/.
2. In the navigation pane on the left side of the console, choose **Tables**.
3. Choose the **Music** table from the table list.
4. Choose the **Indexes** tab for the Music table.
5. Choose **Create index**.



6. For the **Partition key**, enter **`AlbumTitle`**.
7. For **Index name**, enter **`AlbumTitle-index`**.
8. Leave the other settings on their default values and choose **Create index**.


DynamoDB > Tables > Music > Create index
Create global secondary index Info
Global secondary indexes allow you to perform queries on attributes that are not part of a table's primary key. Note that global secondary index read and write capacity settings are separate from those of the table, and they will incur additional costs.
Index details Info
Partition key
AlbumTitle
1 to 255 characters.
Data type
String
Sort key - optional
Enter the sort key name
1 to 255 characters.
Data type
String
Index name
AlbumTitle-index
Between 3 and 255 characters. Only A–Z, a–z, 0–9, underscore characters, hyphens, and periods allowed.
Index capacity Info
Read capacity
Read capacity settings
Copy from base table
Customize settings
Auto scaling Info
Dynamically adjusts provisioned throughput capacity on your behalf in response to actual traffic patterns.
On
Off
Minimum capacity units
1
Maximum capacity units
10
Target utilization (%)
70
Write capacity
Write capacity settings
Copy from base table
Customize settings
Auto scaling Info
Dynamically adjusts provisioned throughput capacity on your behalf in response to actual traffic patterns.
On
Off
Minimum capacity units
1
Maximum capacity units
Target utilization (%)
70
Attribute projections Info

AWS CLI

The following AWS CLI example creates a global secondary index `AlbumTitle-index` for the `Music` table using `update-table`.

```
aws dynamodb update-table \
    --table-name Music \
    --attribute-definitions AttributeName=AlbumTitle,AttributeType=S \
    --global-secondary-index-updates \
        "[{\"Create\":{\"IndexName\": \"AlbumTitle-index\",\"KeySchema\":[{\"AttributeName
\":\"AlbumTitle\",\"KeyType\":\"HASH\"}], \
        \"ProvisionedThroughput\": {\"ReadCapacityUnits\": 10, \"WriteCapacityUnits\": 5
   },\"Projection\":{\"ProjectionType\":\"ALL\"}}}]"
```

Using `update-table` returns the following sample result.

```
{
    "TableDescription": {
        "TableArn": "arn:aws:dynamodb:us-west-2:522194210714:table/Music",
        "AttributeDefinitions": [
            {
                "AttributeName": "AlbumTitle",
                "AttributeType": "S"
            },
            {
                "AttributeName": "Artist",
                "AttributeType": "S"
            },
            {
                "AttributeName": "SongTitle",
                "AttributeType": "S"
            }
        ],
        "GlobalSecondaryIndexes": [
            {
                "IndexSizeBytes": 0,
                "IndexName": "AlbumTitle-index",
                "Projection": {
                    "ProjectionType": "ALL"
                },
                "ProvisionedThroughput": {
                    "NumberOfDecreasesToday": 0,
                    "WriteCapacityUnits": 5,
                    "ReadCapacityUnits": 10
                },
                "IndexStatus": "CREATING",
                "Backfilling": false,
                "KeySchema": [
                    {
                        "KeyType": "HASH",
                        "AttributeName": "AlbumTitle"
                    }
                ],
                "IndexArn": "arn:aws:dynamodb:us-west-2:522194210714:table/Music/index/
AlbumTitle-index",
                "ItemCount": 0
            }
        ],
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "WriteCapacityUnits": 5,
            "ReadCapacityUnits": 10
        },
```

```
        "TableSizeBytes": 0,
        "TableName": "Music",
        "TableStatus": "UPDATING",
        "TableId": "d04c7240-0e46-435d-b231-d54091fe1017",
        "KeySchema": [
            {
                "KeyType": "HASH",
                "AttributeName": "Artist"
            },
            {
                "KeyType": "RANGE",
                "AttributeName": "SongTitle"
            }
        ],
        "ItemCount": 0,
        "CreationDateTime": 1558028402.69
    }
}
```

Note that the value of the `IndexStatus` field is set to `CREATING`.

To verify that DynamoDB has finished creating the `AlbumTitle-index` global secondary index, use the `describe-table` command.

```
aws dynamodb describe-table --table-name Music | grep IndexStatus
```

This command returns the following result. The index is ready for use when the value of the `IndexStatus` field returned is set to `ACTIVE`.

```
"IndexStatus": "ACTIVE",
```

Next, you can query the global secondary index. For details, see Step 7: Query the Global Secondary Index (p. 75).

Step 7: Query the Global Secondary Index

In this step, you query a global secondary index on the `Music` table using the Amazon DynamoDB console or the AWS CLI.

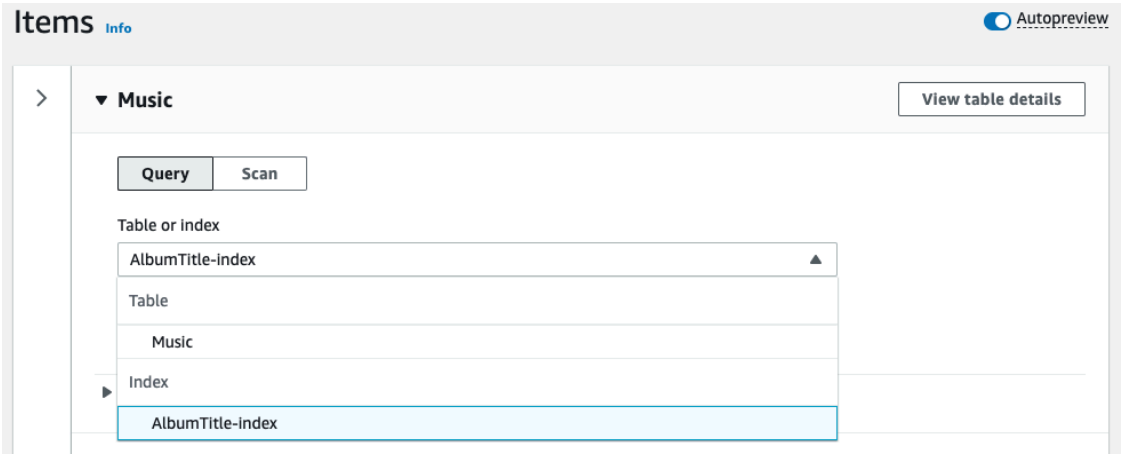
For more information about global secondary indexes, see Using Global Secondary Indexes in DynamoDB (p. 547).

AWS Management Console

Follow these steps to use the DynamoDB console to query data through the `AlbumTitle-index` global secondary index.

1. Open the DynamoDB console at https://console.aws.amazon.com/dynamodb/.
2. In the navigation pane on the left side of the console, choose **Tables**.
3. Choose the **Music** table from the table list.
4. Select the **View items**.
5. Choose **Query**.
6. In the drop-down list under **Query**, choose **AlbumTitle-index**.

   For **AlbumTitle**, enter **`Somewhat Famous`**, and then choose **Run**.



AWS CLI

The following AWS CLI example queries a global secondary index `AlbumTitle-index` on the `Music` table.

```
aws dynamodb query \
    --table-name Music \
    --index-name AlbumTitle-index \
    --key-condition-expression "AlbumTitle = :name" \
    --expression-attribute-values  '{":name":{"S":"Somewhat Famous"}}'
```

Using `query` returns the following sample result.

```
{
    "Count": 1,
    "Items": [
        {
            "AlbumTitle": {
                "S": "Somewhat Famous"
            },
            "Awards": {
                "N": "1"
            },
            "SongTitle": {
                "S": "Call Me Today"
            },
            "Artist": {
                "S": "No One You Know"
            }
        }
    ],
    "ScannedCount": 1,
    "ConsumedCapacity": null
}
```

Step 8: (Optional) Clean Up Resources

If you no longer need the Amazon DynamoDB table that you created for the tutorial, you can delete it. This step helps ensure that you aren't charged for resources that you aren't using. You can use the

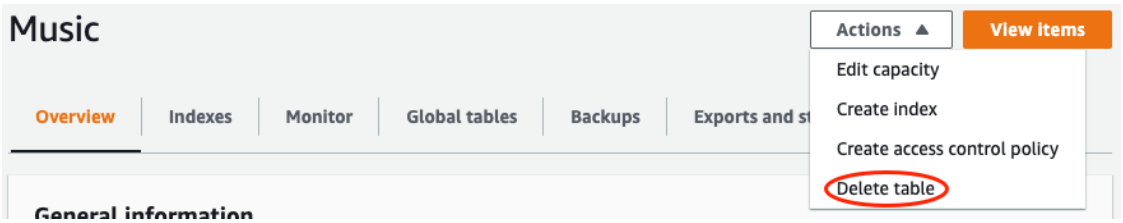
DynamoDB console or the AWS CLI to delete the `Music` table that you created in Step 1: Create a Table (p. 61).

For more information about table operations in DynamoDB, see Working with Tables and Data in DynamoDB (p. 332).

AWS Management Console

To delete the `Music` table using the console:

1. Open the DynamoDB console at https://console.aws.amazon.com/dynamodb/.
2. In the navigation pane on the left side of the console, choose **Tables**.
3. Choose the **Music** table from the table list.
4. Choose the **Indexes** tab for the `Music` table.
5. Choose **Delete table**.



AWS CLI

The following AWS CLI example deletes the `Music` table using `delete-table`.

```
aws dynamodb delete-table --table-name Music
```

Getting Started with DynamoDB: Next Steps

For more information about using Amazon DynamoDB, see the following topics:

- Working with Tables and Data in DynamoDB (p. 332)
- Working with Items and Attributes (p. 394)
- Working with Queries in DynamoDB (p. 477)
- Using Global Secondary Indexes in DynamoDB (p. 547)
- Working with Transactions (p. 662)
- In-Memory Acceleration with DynamoDB Accelerator (DAX) (p. 692)
- Getting Started with DynamoDB and AWS SDKs (p. 78)
- Programming with DynamoDB and the AWS SDKs (p. 207)