



Certified

Developer - Associate



lab



lab title

Using AWS DynamoDB with the CLI V1.02



Course title

AWS Certified Associate



Table of Contents

Contents

Table of Contents.....	1
About the Lab	1
Creating a DynamoDB Table using the Console	1
Importing Items into DynamoDB using batch-write-item	1
Querying DynamoDB Tables using the CLI.....	1

About the Lab

These lab notes are to support the instructional videos on Using Amazon DynamoDB using the CLI in the BackSpace AWS Certified Associate course.

We will first create a DynamoDB table using the console and then add items to the table.

We will then:

- Create a DynamoDB table.
- Upload a JSON file containing items using the batchWriteItem method.
- Query the data using the CLI.

Please refer to the AWS CLI documentation at:

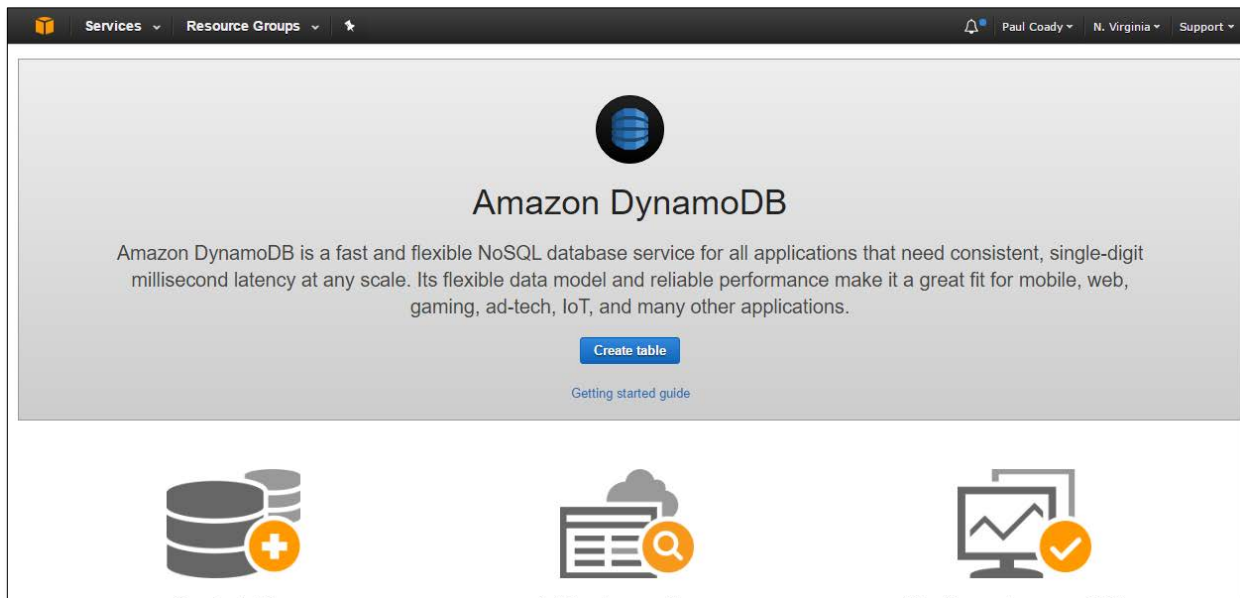
<http://docs.aws.amazon.com/cli/latest/reference/dynamodb/index.html#cli-aws-dynamodb>

Please note that AWS services change on a weekly basis and it is extremely important you check the version number on this document to ensure you have the latest version with any updates or corrections.

▶ Creating a DynamoDB Table using the Console

In this section we will use the **DynamoDB console** to create a table and then add items individually using the console.

Select the DynamoDB Console



Click "Create Table"

Enter the following details (enter exactly with correct case)

BE CAREFUL IF USING COPY/PASTE NOT TO INCLUDE ANY EXTRA SPACES ON THE END.

Table Name: test-table

Primary Key Type: hash

Hash Attribute Type: Number

Hash Attribute Name: Id (case sensitive - make sure the first letter is capitalised)

Table name*	<input type="text" value="test-table"/>	?
Primary key*	Partition key	
	<input type="text" value="Id"/>	Number ?
<input type="checkbox"/> Add sort key		

Uncheck *Use Default Settings*

Table settings

Default settings provide the fastest way to get started with your table. You can modify these default settings now or after your table has been created.

☐ Use default settings

Now create a global secondary index with hash key string ProductCategory and sort key number Price.

Use index name ProductCategory-Price-index

Click Add index to table.

Secondary indexes

Name	Type	Partition key	Sort key	Projected Attributes
+ Add Index				

Enter index details

Add index

Primary key* Partition key

ProductCategory String

☒ Add sort key

Price Number

Index name* ProductCategory--Price-index

Projected attributes All

☐ Create as Local Secondary Index

[Cancel](#) [Add index](#)

Click *Add Index*

Continue using default settings.

Provisioned capacity

	Read capacity units	Write capacity units
Table	5	5
ProductCategory--Price-index	5	5

Estimated cost \$5.81 / month ([Capacity calculator](#))

Additional charges may apply if you exceed the AWS Free Tier levels for CloudWatch or Simple Notification Service. Advanced alarm settings are available in the CloudWatch management console.

[Cancel](#) [Create](#)

Click Create.

Press refresh until table status is listed as active.

Table details	
Table name	test-table
Primary partition key	Id (Number)
Primary sort key	-
Time to live attribute	DISABLED Manage TTL
Table status	Active
Creation date	May 12, 2017 at 2:19:58 AM UTC+10
Provisioned read capacity units	5
Provisioned write capacity units	5
Last decrease time	-
Last increase time	-
Storage size (in bytes)	0 bytes
Item count	0
Region	US East (N. Virginia)
Amazon Resource Name (ARN)	arn:aws:dynamodb:us-east-1:802694931986:table/test-table

Storage size and item count are not updated in real-time. They are updated periodically, roughly every six hours.

Click on Items tab

Click on Create Item

BE CAREFUL IF USING COPY/PASTE NOT TO INCLUDE ANY EXTRA SPACES ON THE END.

Enter the Id as 101

ProductCategory- String: Book

Price - Number:-2

and then click on the action menus box on the left of the entry.

Select Append then String

Enter field *Title* and value *Book 101 Title*

Enter the rest of the details for the item. Make sure you select the right data type of string or number or boolean:

InPublication - Boolean:true

PageCount - Number:500

Dimensions - String: 8.5 x 11.0 x 0.5

Authors - String: Author 1

ISBN- String: 111-1111111111

▼ Item {9}
⊕ Id Number : 101
⊕ ProductCategory String : Book
⊕ Price Number : 2
⊕ Title String : Book 101 Title
⊕ InPublication Boolean : true
⊕ PageCount Number : 500
⊕ Dimensions String : 8.5 x 11.0 x 0.5
⊕ Author String : Author 1
⊕ ISBN String : 111-1111111111

Click Save

test-table Close									
Overview Items Metrics Alarms Capacity Indexes Triggers Access control Tags									
Create item Actions ▼									
Scan: [Table] test-table: Id ↕									
Scan ▼ [Table] test-table: Id ^									
+ Add filter									
Start search									
<input type="checkbox"/>	Id	Author	Dimensions	ISBN	InPublication	PageCount	Price	ProductCategory	Title
<input type="checkbox"/>	101	Author 1	8.5 x 11.0 x 0.5	111-1111111111	true	500	2	Book	Book 101 Title

Importing Items into DynamoDB using batch-write-item

In this section we will use the DynamoDB CLI to import items from a JSON file into a DynamoDB table.

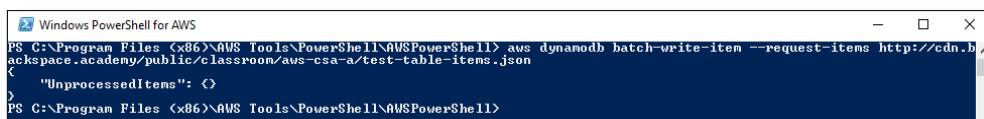
The following JSON file contains our list of items to be imported:

<http://cdn.backspace.academy/public/classroom/aws-csa-a/test-table-items.json>

We will use the batch-write-item from the CLI to download the file and write the items to DynamoDB:

```
aws dynamodb batch-write-item --request-items
http://cdn.backspace.academy/public/classroom/aws-csa-a/test-table-items.json
```

After running the command you will receive a message: "UnprocessedItems": {}



```
Windows PowerShell for AWS
PS C:\Program Files (x86)\AWS Tools\PowerShell\AWSPowerShell> aws dynamodb batch-write-item --request-items http://cdn.backspace.academy/public/classroom/aws-csa-a/test-table-items.json
{
  "UnprocessedItems": {}
}
PS C:\Program Files (x86)\AWS Tools\PowerShell\AWSPowerShell>
```

Now go to the DynamoDB console and view the added items:

Id	Author	Dimensions	ISBN	InPublication	PageCount	Price	ProductCategory	Title	BicycleType	Brand	Color	Description
205						500	Bike	20-Bicycle 205	Hybrid	Brand-Compa...	{ "Black", "Re...	205 descripti
203						300	Bike	19-Bicycle 203	Road	Brand-Compa...	{ "Black", "Gre...	203 descripti
202						200	Bike	21-Bicycle 202	Road	Brand-Compa...	{ "Black", "Re...	202 descripti
201						100	Bike	18-Bicycle 201	Road	Brand-Compa...	{ "Black", "Re...	201 descripti
204						400	Bike	18-Bicycle 204	Mountain	Brand-Compa...	Red	204 descripti
102		8.5 x 11.0 x 0.8	222-2222222...	true	600	20	Book	Book 102 Title				
103		8.5 x 11.0 x 1.5	333-3333333...	false	700	200	Book	Book 103 Title				
101	Author 1	8.5 x 11.0 x 0.5	111-1111111111	true	500	2	Book	Book 101 Title				

▶ Querying DynamoDB Tables using the CLI

In this section we will use CLI to query items in a DynamoDB table.

The details of our query are located in a json file at:

<http://cdn.backspace.academy/public/classroom/aws-csa-a/test-table-query.json>

This query will be based upon:

ProductCategory: Bike

Price: Less than or equal to 300

We can use the query command to query our table:

```
aws dynamodb query --table-name test-table --index-name ProductCategory-Price-index --key-conditions http://cdn.backspace.academy/public/classroom/aws-csa-a/test-table-query.json
```

This produces the Bike items less than or equal to \$300 in JSON format:

```

{
  "BicycleType": {
    "S": "Road"
  },
  "Description": {
    "S": "202 description"
  },
  "Title": {
    "S": "21-Bicycle 202"
  },
  "Color": {
    "SS": [
      "Black",
      "Red"
    ]
  },
  "Gender": {
    "S": "M"
  },
  "Price": {
    "N": "200"
  },
  "ProductCategory": {
    "S": "Bike"
  },
  "Brand": {
    "S": "Brand-Company A"
  },
  "Id": {
    "N": "202"
  }
},
{
  "BicycleType": {
    "S": "Road"
  },
  "Description": {
    "S": "203 description"
  },
  "Title": {
    "S": "19-Bicycle 203"
  },
  "Color": {
    "SS": [
      "Black",
      "Green",
      "Red"
    ]
  },
  "Gender": {
    "S": "M"
  },
  "Price": {
    "N": "300"
  },
  "ProductCategory": {
    "S": "Bike"
  },
  "Brand": {
    "S": "Brand-Company B"
  },
  "Id": {
    "N": "203"
  }
}
]
"ScannedCount": 3,
"ConsumedCapacity": null

```