



# AWS DynamoDB Lab



⌚ 2021-01-29 | 📅 2022-02-15

| ⏺ Tech , AWS , AWS Certificate , Labs , AWS DAS , AWS , DB , DB , NoSQL

📄 15k | ⏰ 14 mins.

## AWS DynamoDB

AWS DynamoDB

## DynamoDB

### Introduction to AWS DynamoDB

[https://play.whizlabs.com/site/task\\_details?lab\\_type=1&task\\_id=13&quest\\_id=35](https://play.whizlabs.com/site/task_details?lab_type=1&task_id=13&quest_id=35)

#### Lab Details

1. This lab walks you through Amazon DynamoDB features. We will create a table in Amazon DynamoDB to store information and then query that information from the DynamoDB table.

#### Introduction

#### What is AWS DynamoDB?

- Definition
  - DynamoDB is a fast and flexible NoSQL database designed for applications that need consistent, single-digit millisecond latency at any scale. It is a fully managed database and it supports both document and key value data models.

- It has a very flexible data model. This means that you don't need to define your database schema upfront. It also has reliable performance.
- DynamoDB is a good fit for mobile gaming, ad-tech, IoT and many other applications.

## DynamoDB Tables

DynamoDB tables consist of

- **Items** (Think of a row of data in a table).
- **Attributes** (Think of a column of data in a table).
- Supports **key-value** and **document data structures**.
- Key = the name of the data. Value = the data itself.
- Document can be written in **JSON**, **HTML** or **XML**.

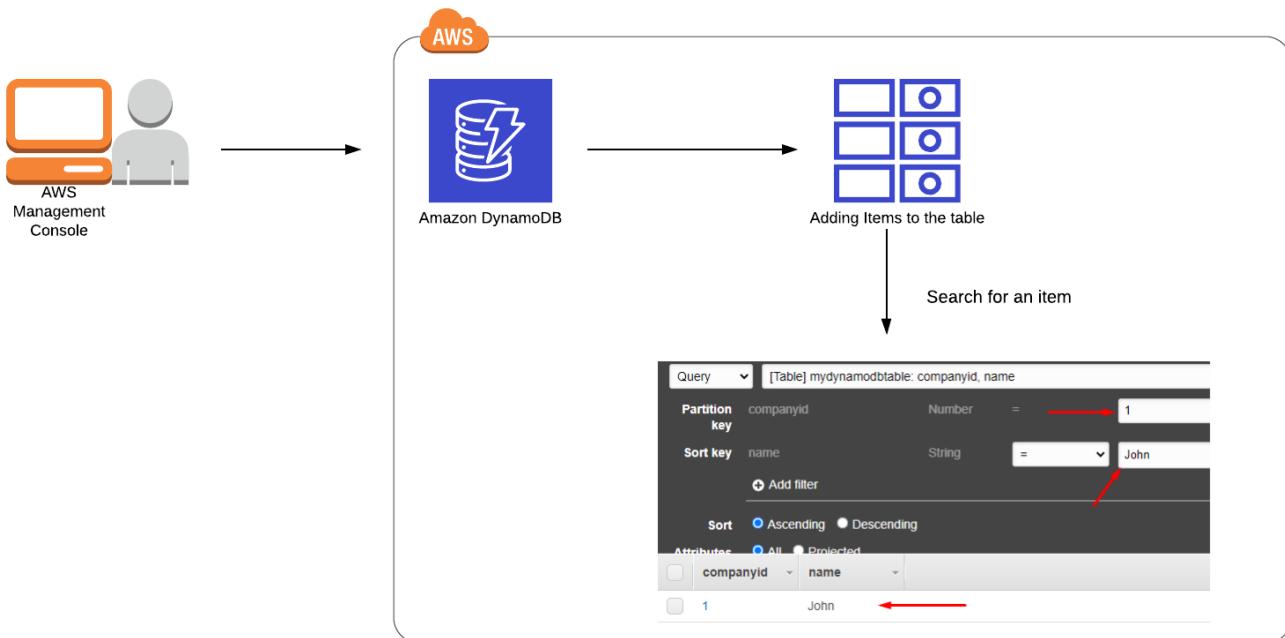
## DynamoDB- Primary Keys

- DynamoDB stores and retrieves data based on a **Primary key**.
- DynamoDB also uses **Partition keys** to determine the physical location data is stored.
- **If you are using a partition key as your Primary key, then no items will have the same Partition key.**
- **All items with the same partition key are stored together and then sorted according to the sort key value.**
- **Composite Keys (Partition Key + Sort Key) can be used in combination.**
- **Two items may have the same partition key, but must have a different sort key.**

## Lab Tasks

1. Log into AWS Management Console.
2. Create a DynamoDB table.
3. Insert data into that DynamoDB table.
4. Search for an item in the DynamoDB table.

## Architecture Diagram



## DynamoDB Configuration

### Services -> DynamoDB

Click on Create table .

The screenshot shows the 'Create table' page for Amazon DynamoDB. At the top, there's a large 'Amazon DynamoDB' logo and a brief description: 'Amazon DynamoDB is a fast and flexible NoSQL database service for all applications that need consistent, single-digit millisecond latency at any scale. Its flexible data model and reliable performance make it a great fit for mobile, web, gaming, ad-tech, IoT, and many other applications.' Below this is a 'Create table' button and a 'Getting started guide' link.

Below the main heading, there are three sections with icons and links:

- Create tables**: An icon of two databases with a plus sign. Link: [More about DynamoDB throughput](#)
- Add and query items**: An icon of a database with a magnifying glass. Link: [DynamoDB API reference](#)
- Monitor and manage tables**: An icon of a monitor with a graph and a checkmark. Link: [Monitoring tables](#)

At the bottom of the page, there's a 'DynamoDB documentation & support' section with links to 'Getting started guide', 'FAQ', 'Developer guide', 'Forums', and 'Report an issue'.

- Table Name: test
- Primary key: companyID and select Number
- Add sort key: Enter name in the respective field and select String .
- The combination of a Primary Key and a Sort Key uniquely identifies each item in a DynamoDB table.

Click on Create .

The screenshot shows the 'Create DynamoDB table' wizard on the AWS console. The 'Table name' field is set to 'test'. Under 'Primary key', 'companyID' is defined as a 'Number' type. A checkbox for 'Add sort key' is checked, and 'name' is defined as a 'String' type. In the 'Table settings' section, the 'Use default settings' checkbox is checked, and a note states: 'You do not have the required role to enable Auto Scaling by default. Please refer to documentation.' At the bottom right are 'Cancel' and 'Create' buttons.

Your table will be created within 2-3 minutes.

The screenshot shows the AWS DynamoDB console interface. A new table named "test" is being created. The "Overview" tab is active, showing the following table details:

Table name	test
Primary partition key	companyID (Number)
Primary sort key	name (String)
Point-in-time recovery	-
Encryption Type	DEFAULT
KMS Master Key ARN	Not Applicable
Encryption Status	- NEW
CloudWatch Contributor Insights	-
Time to live attribute	-
Table status	Creating
Creation date	November 6, 2020 at 11:34:15 PM UTC-8
Read/write capacity mode	Provisioned
Last change to on-demand mode	-
Provisioned read capacity units	5 (Auto Scaling Error)
Provisioned write capacity units	5 (Auto Scaling Error)
Last decrease time	-
Last increase time	-
Storage size (in bytes)	0 bytes
Item count	0
Region	US East (N. Virginia)

Other tabs visible include Items, Metrics, Alarms, Capacity, Indexes, Global Tables, Backups, Contributor Insights, Triggers, Access control, and Tags.

## Query DynamoDB

### Insert Items

Next, we are inserting data in the table we created.

Click on the Items tab.

The screenshot shows the AWS DynamoDB console with the 'test' table selected. The left sidebar shows navigation options like Dashboard, Tables, Backups, Reserved capacity, Preferences, DAX, and Events. The main area has tabs for Overview, Items, Metrics, Alarms, Capacity, Indexes, Global Tables, Backups, Contributor Insights, Triggers, Access control, and Tags. The 'Items' tab is active, showing a 'Scan' operation for the 'test: companyID, name' index. A tooltip at the bottom explains that an item consists of one or more attributes, each with a name, data type, and value. The status bar at the bottom indicates 'Viewing 0 to 0 items'.

Click on Create item.

Add new primary key and sort key values.

- companyid: 1
- name: ZacksAmber

Click on Save .

The screenshot shows the 'Create item' dialog in the AWS DynamoDB console. A single item is listed in the tree view:

- Item (2)
  - companyID Number : 1
  - name String : ZacksAmber

At the bottom right of the dialog are 'Cancel' and 'Save' buttons.

The screenshot shows the 'Items' tab for the 'test' table in the AWS DynamoDB console. The table contains one item:

companyID	name
1	ZacksAmber

At the bottom right of the table view are 'Viewing 1 to 1 items' and 'Actions' buttons.

For testing purposes, add 4-5 items as shown in the above step.

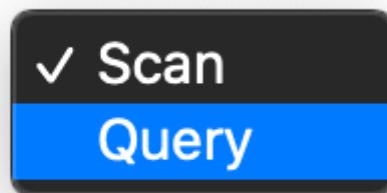
The screenshot shows the AWS DynamoDB console with a table named 'test'. The 'Items' tab is selected. The table structure is defined by partition key 'compayID' and sort key 'name'. The data in the table is:

compayID	name
1	ZacksAmber
2	Amazon
3	Apple
4	Google
5	Tesla

## Search for Items

We will query the items in our table by using scan.

Click the drop-down list showing the Scan in Items Tab (located below the Create item button) and change it to Query .



In the query window, enter the partition key and sort key which you want to search.

- Partition Key: 2
- Sort Key: Amazon

Click on Start search

The screenshot shows the AWS DynamoDB console interface. On the left, the navigation bar includes 'DynamoDB', 'Tables' (selected), 'Backups', 'Reserved capacity', 'Preferences', 'DAX', 'Dashboard', 'Clusters', 'Subnet groups', 'Parameter groups', and 'Events'. The main area is titled 'test' and shows the 'Items' tab selected. A search bar at the top says 'Scan: [Table] test: compayID, name'. Below it, the 'Query' section has 'Partition key' set to 'compayID' with value '2' and 'Sort key' set to 'name' with value 'Amazon'. The 'Sort' option is set to 'Ascending'. The 'Attributes' section has 'All' selected. The results table shows the following data:

compayID	name
1	ZacksAmber
2	Amazon
3	Apple
4	Google
5	Tesla

You will be able to see a result table with your filtered records. A sample screenshot is given below:

This screenshot is identical to the one above, showing the AWS DynamoDB console with the 'test' table. The 'Items' tab is selected, and the results table shows only one item: 2 (Amazon).

only search for partition key

The screenshot shows the AWS DynamoDB console interface. On the left, the navigation pane is visible with options like Dashboard, Tables, Backups, Reserved capacity, Preferences, DAX, and Events. The 'Tables' section is selected, and a table named 'test' is highlighted. The main area is titled 'test' and shows the 'Items' tab selected. A search bar at the top says 'Query by table name' and 'Choose a table ...'. Below it, there's a 'Create item' button and an 'Actions' dropdown. The 'Items' section has a heading 'Query: [Table] test: compayID, name ^'. It includes fields for 'Partition key' (compayID) and 'Sort key' (name), both set to '='. There are dropdowns for 'Number' (3) and 'String' (Enter value). Below these are 'Sort' options ('Ascending' is selected) and 'Attributes' ('All' is selected). A 'Start search' button is present. The results table shows one item: compayID 3 and name Apple.

## Completion and Conclusion

1. You have successfully created an Amazon DynamoDB table.
2. You have inserted multiple items into the table using a partition key and sort key.
3. You have successfully searched for items in the table using the partition and sort keys.

## DynamoDB & Global Secondary Index

[https://play.whizlabs.com/site/task\\_details?lab\\_type=1&task\\_id=40&quest\\_id=35](https://play.whizlabs.com/site/task_details?lab_type=1&task_id=40&quest_id=35)

## Lab Details

1. This lab walks you through the steps to create a DynamoDB database and use Global Secondary Indexes in a case study.

## Introduction

### What is AWS DynamoDB?

- Definition

- DynamoDB is a fast and flexible NoSQL database designed for applications that need consistent, single-digit millisecond latency at any scale. It is a fully managed database and it supports both document and key value data models.
- It has a very flexible data model. This means that you don't need to define your database schema upfront. It also has reliable performance.
- DynamoDB is a good fit for mobile gaming, ad-tech, IoT and many other applications.

## DynamoDB Tables

DynamoDB tables consist of

- **Items** (Think of a row of data in a table).
- **Attributes** (Think of a column of data in a table).
- Supports **key-value** and **document data structures**.
- Key = the name of the data. Value = the data itself.
- Document can be written in **JSON**, **HTML** or **XML**.

## DynamoDB - Primary Keys

- DynamoDB stores and retrieves data based on a **Primary key**.
- DynamoDB also uses **Partition keys** to determine the physical location data is stored.
- **If you are using a partition key as your Primary key, then no items will have the same Partition key.**
- **All items with the same partition key are stored together and then sorted according to the sort key value.**
- **Composite Keys (Partition Key + Sort Key) can be used in combination.**
- **Two items may have the same partition key, but must have a different sort key.**

## What is an Index in DynamoDB

- In SQL databases, an index is a data structure which allows you to perform queries on specific columns in a table.
- You select the column that is required to include in the index and run the searches on the index instead of on the entire dataset.

In DynamoDB, two types of indexes are supported to help speed-up your queries.

- **Local Secondary Index.**
- **Global Secondary Index.**

## Local Secondary Index

- **Can only be created when you are creating the table.**
- **Cannot be removed, added or modified later.**
- It has the same partition key as the original table.
- Has a different Sort key.
- Gives you a different view of your data, organized according to an alternate sort key.
- Any queries based on Sort key are much faster using the index than the main table.

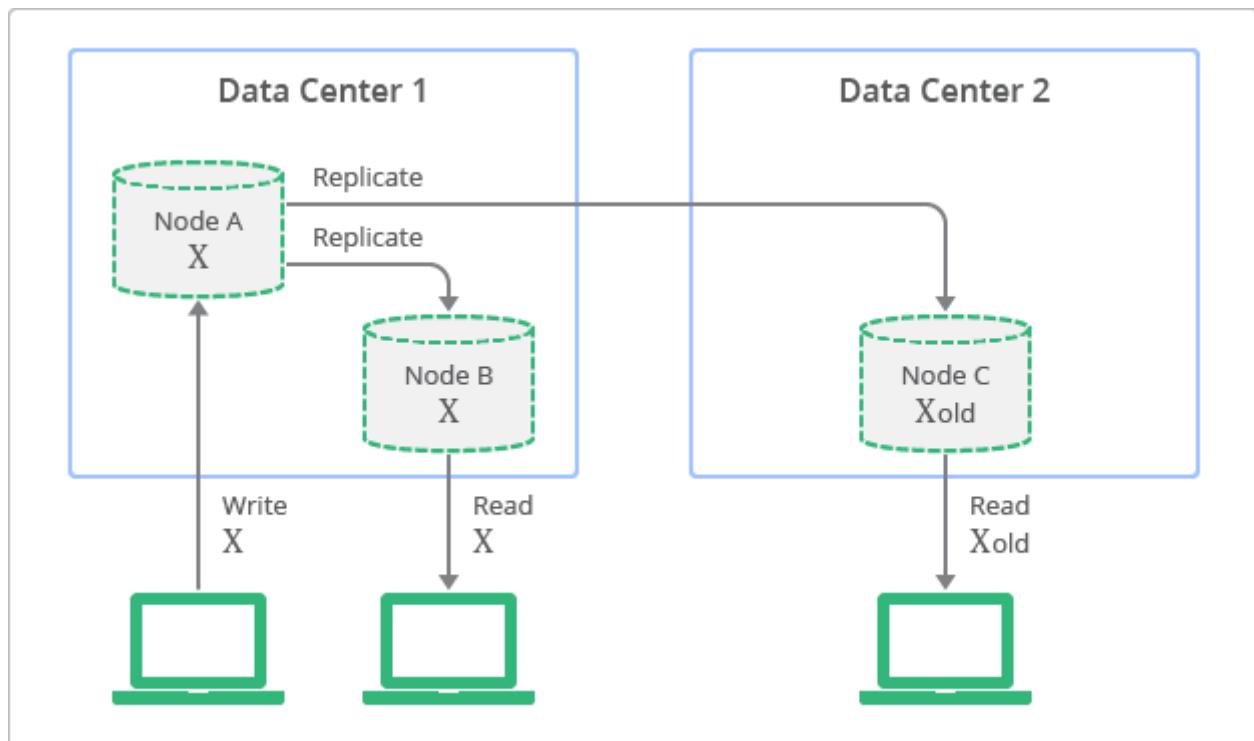
## Global Secondary Index

- You can create a GSI on creation or you can add it later.
- Different partition key as well as different sort key.
- It gives a completely different view of the data.
- Speeds up the queries relating to this alternative Partition or Sort Key.

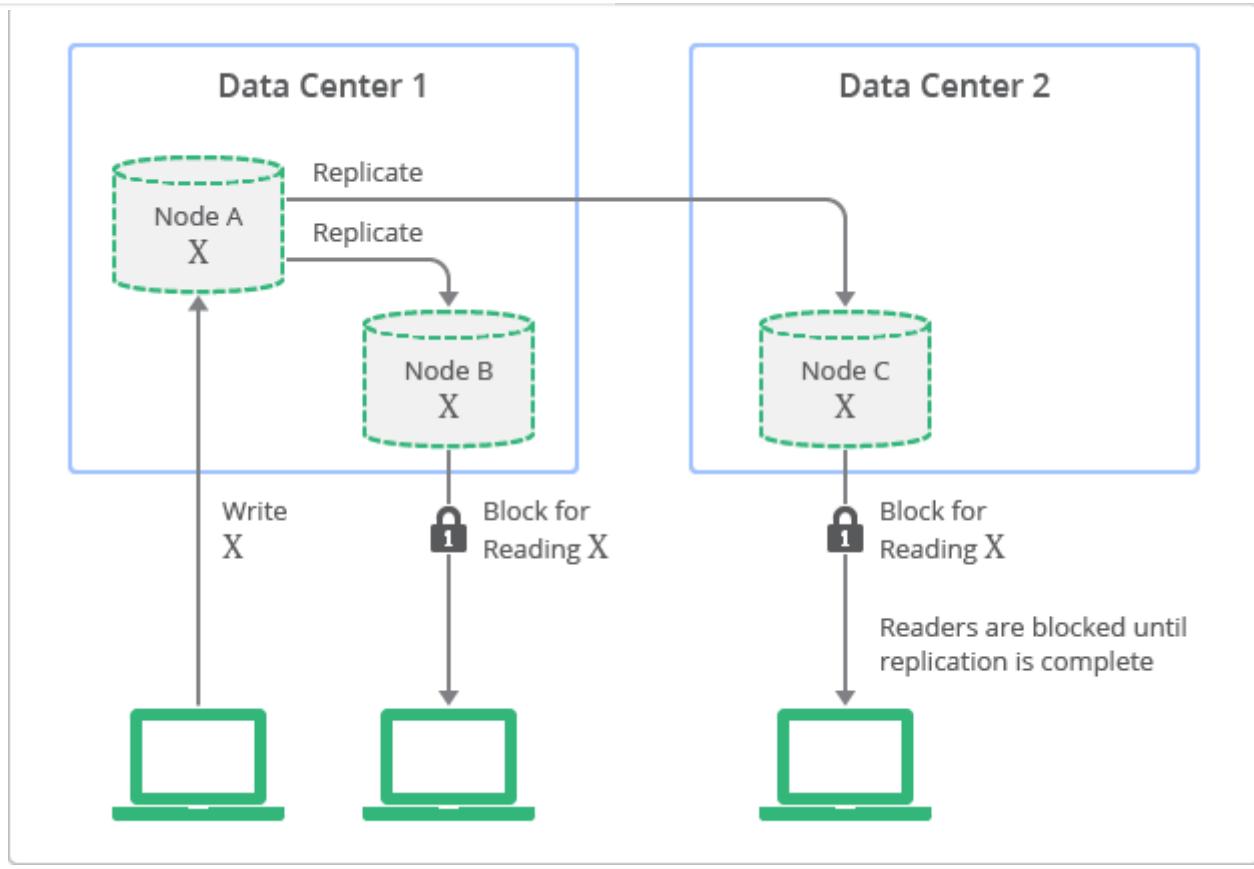
No.	LSI	GSI
1	Key = hash key and a range key	Key = hash or hash-and-range
2	Hash same attribute as that of the table. Range key can be any scalar table attribute	The index hash key and range key (if present) can be any scalar table attributes.
3	For each hash key, the total size of all indexed items must be 10GB or less.	No size restrictions for global secondary indexes.

No.	LSI	GSI
4	Query over a single partition, as specified by the hash key value in the query.	Query over the entire table, across all partitions.
5	Eventual consistency or strong consistency.	Eventual consistency only.
6	Read and write capacity units consumed from the table.	Every global secondary index has its own provisioned read and write capacity units.
7	Query will automatically fetch non-projected attributes from the table.	Query can only request projected attributes. It will not fetch attributes from the table.

## Eventual Consistency vs Strong Consistency



Eventual Consistency



Strong Consistency

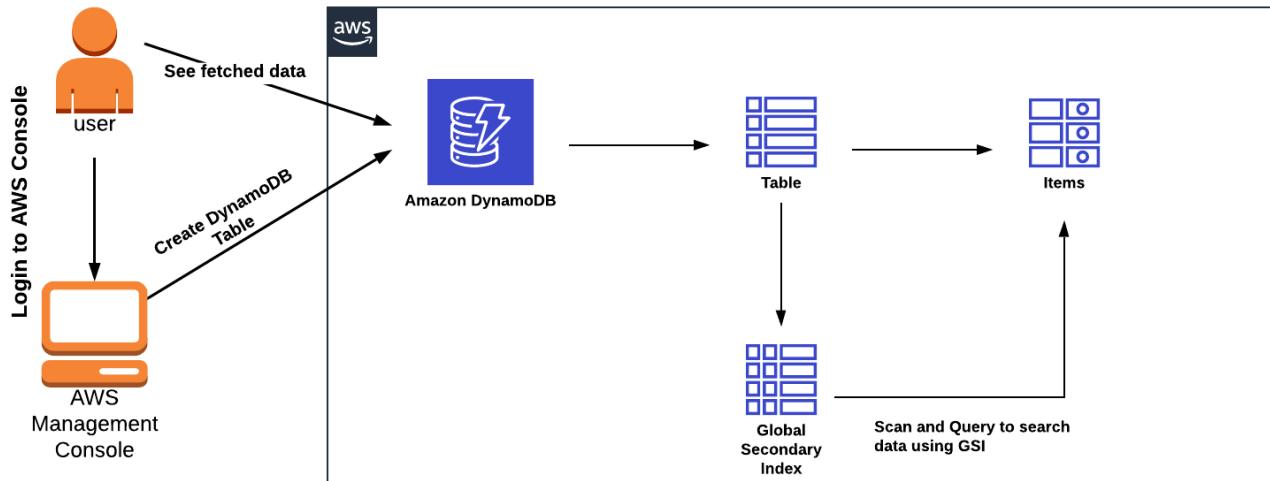
## Lab Details

### Case Study: Creating a Global Secondary Index

Let's get our hands dirty by creating a table and adding a GSI on the table.

In this example, imagine we want to keep track of orders that were returned by our users. We'll store the date of the return in a `ReturnDate` attribute. We'll also add a global secondary index with a composite key schema using `ReturnDate` as the HASH key and `UserAmount` as the RANGE key.

### Architecture Diagram



## DynamoDB Configuration

### Services -> DynamoDB

The screenshot shows the AWS Management Console interface for DynamoDB. At the top, there's a navigation bar with various services like CloudWatch, Lambda, and S3. Below it, the main content area has a large 'Amazon DynamoDB' logo and a brief description: 'Amazon DynamoDB is a fast and flexible NoSQL database service for all applications that need consistent, single-digit millisecond latency at any scale. Its flexible data model and reliable performance make it a great fit for mobile, web, gaming, ad-tech, IoT, and many other applications.' There are three main buttons: 'Create table', 'Getting started guide', and 'Create tables'. The 'Create tables' section includes a sub-section for 'More about DynamoDB throughput'. The 'Add and query items' section includes a sub-section for 'DynamoDB API reference'. The 'Monitor and manage tables' section includes a sub-section for 'Monitoring tables'. At the bottom, there's a footer with links to 'DynamoDB documentation & support', 'Getting started guide', 'FAQ', 'Developer guide', 'Forums', and 'Report an issue'.

- Table Name: test
- Primary key(Partition key): Username and select String
- Add sort key: Enter **OrderID** in the respective field and select String .
- The combination of a Primary Key and a Sort Key uniquely identifies each item in a DynamoDB table.
- Keep all other settings as default.

## Click on Create .

The preview of the new DynamoDB console is now available  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to [try it](#) and let us know what you think.

### Create DynamoDB table

DynamoDB is a schema-less database that only requires a table name and primary key. The table's primary key is made up of one or two attributes that uniquely identify items, partition the data, and sort data within each partition.

Table name*	<input type="text" value="test"/>
Primary key*	Partition key
	<input type="text" value="Username"/> String
<input checked="" type="checkbox"/> Add sort key	<input type="text" value="OrderID"/> String

**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

- No secondary indexes.
- Provisioned capacity set to 5 reads and 5 writes.
- Basic alarms with 80% upper threshold using SNS topic "dynamodb".
- Encryption at Rest with DEFAULT encryption type.

**Note:** You do not have the required role to enable Auto Scaling by default.  
Please refer to [documentation](#).

+ Add tags NEW!

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](#)

Feedback English (US) ▾ © 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

Your table will be created within 2-3 minutes.

The preview of the new DynamoDB console is now available  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to [try it](#) and let us know what you think.

[Create table](#) [Delete table](#)

[Choose a table ...](#) [Actions](#)

**test** Close

[Overview](#) [Items](#) [Metrics](#) [Alarms](#) [Capacity](#) [Indexes](#) [Global Tables](#) [Backups](#) [Contributor Insights](#) [Triggers](#) [Access control](#) [Tags](#)

Table is being created

Recent alerts

No CloudWatch alarms have been triggered for this table.

Stream details

Stream enabled	No
View type	-
Latest stream ARN	-

[Manage Stream](#)

Table details

Table name	test
Primary partition key	Username (String)
Primary sort key	OrderID (String)
Point-in-time recovery	-
Encryption Type	DEFAULT
KMS Master Key ARN	Not Applicable
Encryption Status	-
CloudWatch Contributor Insights	- <small>NEW!</small>
Time to live attribute	-
Table status	Creating
Creation date	November 20, 2020 at 2:49:22 PM UTC-8
Read/write capacity mode	Provisioned
Last change to on-demand mode	-
Provisioned read capacity units	5 (Auto Scaling Disabled)
Provisioned write capacity units	5 (Auto Scaling Disabled)
Last decrease time	-
Last increase time	-

Feedback English (US) ▾ © 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

Scan & Query DynamoDB

## Create Item

Select the Items tab next to the overview tab and click on Create Item .

Once you select the create item, you'll see Username and OrderID , but we need 2 more attributes in our table. Click on + on OrderID then select Append . Choose String in the dropdown list.

Add two attributes: ReturnDate and UserAmount . Enter the string value in the field and click on Save .

- UserName : HarryPotter
- OrderID : 20160630-12928
- ReturnDate : 20190705
- UserAmount : 142.23

The screenshot shows the 'Create item' interface in the AWS DynamoDB console. A tree view displays four items under 'Item (4)'. The first item has attributes: Username (String: HarryPotter), OrderID (String: 20160630-12928), ReturnDate (String: 20190705), and UserAmount (String: 142.23). A validation error message at the bottom states: 'One or more parameter values are not valid. The AttributeValue for a key attribute cannot contain an empty string value. Key: Username (Service: AmazonDynamoDBv2; Status Code: 400; Error Code: ValidationException; Request ID: P8VT1R90SLJVRBOSDIEVAKWV3FV4KQNS05AEMVJF66Q9ASUAAJG; Proxy: null)'.

Navigate to the Indexes tab section (next to the capacity tab on the header) and click on Create index .

The screenshot shows the table configuration interface in the AWS DynamoDB console. The 'Indexes' tab is selected. A sub-section titled 'Global Secondary Indexes (GSIs) allow you to query efficiently over any field (attribute) in your DynamoDB table. GSIs can treat any table attribute as a key, even attributes not present in all items.' is visible. The 'Create index' button is highlighted.

↓ the parameters ReturnDate and UserAmount . Click on Create index and wait until the index state changes to **Active**. The nomenclature for Index name is a combination of the

columns we selected, followed by Index. You can modify the index name as per the requirement.

The preview of the new DynamoDB console is now available.  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

**Create index**

**Primary key\*** Partition key  
 String  Add sort key  
 String

**Index name\*** ReturnDate-UserAmount-index

**Projected attributes** All

Read capacity units Write capacity units

Estimated cost \$2.91 / month (Capacity calculator)

Approximate creation time is 5 minutes. Additional write capacity may decrease creation time. A notification will be sent to the SNS topic dynamodb once the index creation is completed. Basic Alarms with 80% upper threshold using SNS topic 'dynamodb' will be automatically created. Additional charges may apply if you exceed the AWS Free Tier levels for CloudWatch or Simple Notification Service. Advanced configuration for alarms can be done in the alarms tab.

**Create index**

Once the GSI is active, you can check the Indexes tab to confirm. Check the index type.

- Note: It will take 5-10 minutes to be active.

The preview of the new DynamoDB console is now available.  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

**Indexes**

Name	Status	Type	Partition key	Sort key	Attributes	Read capacity	Write capacity	Size	Item count	Auto
ReturnDate-UserAmount	Active	GSI	ReturnDate (String)	UserAmount (String)	ALL	5	5	0 bytes	0	DISA

Move to the Items tab and click on Create item. Insert data into the table and click on Save .

- Note: Refresh the console a few times if the newly-added attributes are not displayed in the field.

Add remaining values to the table.

- UserName : HarryPotter
- OrderID : 20160630-28176
- ReturnDate : 20190513
- UserAmount : 88.30
- UserName : Ron
- OrderID : 20170609-25875
- ReturnDate : 20190628
- UserAmount : 116.86
- UserName : Ron
- OrderID : 20170609-4177
- ReturnDate : 20190731
- UserAmount : 27.89
- UserName : Voldemort
- OrderID : 20170609-17146
- ReturnDate : 20190511

- UserAmount : 114.00
- UserName : Voldemort
- OrderID : 20170609-18618
- ReturnDate : 20190615
- UserAmount : 122.45

In case you added a wrong value, you can edit with the `edit` option on the column.

Username	OrderID	ReturnDate	UserAmount
HarryPotter	20160630-28176	20190513	88.30
HarryPotter	20160630-12928	20190705	142.23
Ron	20170609-4177	20190731	27.89
Ron	20170609-25875	20190628	116.86
Voldemort	20170609-18618	20190615	122.45
Voldemort	20170609-17146	20190511	114.00

Once you have added all the data in the table, please review it.

The preview of the new DynamoDB console is now available  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Username	OrderID	ReturnDate	UserAmount
HarryPotter	20160630-28176	20190513	88.30
HarryPotter	20160630-12928	20190705	142.23
Ron	20170609-4177	20190731	27.89
Ron	20170609-25875	20190628	116.86
Voldemort	20170609-18618	20190615	122.45
Voldemort	20170609-17146	20190511	114.00

## Use Global Secondary Index to Fetch Data

1. Now with the help of **GSI** we will try to fetch the data from the table, avoiding a full scan. This will lead to better performance and saving resources. We'll be fetching data and adding filter conditions on the return date.

2. Lets try with the **Scan** option to search for data. We need a **ReturnDate** (PartitionKey) and we need to check which users returned an item on that date. To do so, we can use the sort key to qualify the amount as per the requirement.

- Select the “**Scan**” option (which is in the left upper corner).
- In this example, we are trying to fetch the users who returned their orders in the month of May.
- Select the “**Index**” option which we created and add a filter condition on “**ReturnDate**”. Select data type “**String**” because our attributes are a “String” data type, and then select the clause “**Between**”– 20190501 and 20190531

The preview of the new DynamoDB console is now available  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

test Close

Overview Items Metrics Alarms Capacity Indexes Global Tables Backups Contributor Insights Triggers Access control Tags

Create item Actions

Scan: [Table] test: Username, OrderID ^

Scan [Index] ReturnDate-UserAmount-index: ReturnDate, UserAmount

Filter ReturnData String Between 20190501 And 20190531

Add filter

Start search Cancel changes

Username	OrderID	ReturnDate	UserAmount
HarryPotter	20160630-28176	20190513	88.30
HarryPotter	20160630-12928	20190705	142.23
Ron	20170609-4177	20190731	27.89
Ron	20170609-25875	20190628	116.86
Voldemort	20170609-18618	20190615	122.45
Voldemort	20170609-17146	20190511	114.00

Note: If the Global secondary index is not showing in the Scan list, Please wait another 5 to 10 minutes and reload the entire page. This is an AWS side delay.

Click on Start search

The preview of the new DynamoDB console is now available  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

test Close

Overview Items Metrics Alarms Capacity Indexes Global Tables Backups Contributor Insights Triggers Access control Tags

Create item Actions

Scan: [Index] ReturnDate-UserAmount-index: ReturnDate...

Scan [Index] ReturnDate-UserAmount-index: ReturnDate, UserAmount

Filter ReturnDate String Between 20190501 And 20190531

Add filter

Start search

Username	OrderID	ReturnDate	UserAmount
HarryPotter	20160630-28176	20190513	88.30
Voldemort	20170609-17146	20190511	114.00

3. Lets try with the **Query** option to search for some data. We need a **ReturnDate** (PartitionKey) and we need to check which users returned an item on that day. To do so, we'll use the sort key to qualify the amount, as per the requirement.

The screenshot shows the AWS DynamoDB console interface. On the left, there's a sidebar with options like Dashboard, Tables, Backups, Reserved capacity, Preferences, DAX, and Events. Under Tables, a table named 'test' is selected. The main area shows a preview of the new console, stating 'The preview of the new DynamoDB console is now available'. Below this, the 'Items' tab is selected in the navigation bar. A query is being run: 'Scan: [Index] ReturnDate-UserAmount-index: ReturnDate, UserAmount'. The 'Partition key' is set to 'ReturnDate' with the value '20190705'. The 'Sort key' is set to 'UserAmount' with the condition ' $\geq$ ' and the value '100'. The results table shows two items:

Username	OrderID	ReturnDate	UserAmount
HarryPotter	20160630-28176	20190513	88.30
Voldemort	20170609-17146	20190511	114.00

Click on Start search

This screenshot is identical to the previous one, showing the same query setup and results. The results table now shows only one item:

Username	OrderID	ReturnDate	UserAmount
HarryPotter	20160630-12928	20190705	142.23

This global secondary index allows us to quickly find all the returns entered on various dates, which would normally require full table scans.

## Completion and Conclusion

1. You have created a DynamoDB table with Global Secondary Indexes.
2. You have successfully fetched data using Global Secondary Indexes.

## One more thing

### Best Practices for Querying and Scanning Data

#### Performance Considerations for Scans

In general, Scan operations are less efficient than other operations in DynamoDB. A Scan operation always scans the entire table or secondary index. It then filters out values to provide the result you want, essentially adding the extra step of removing data from the result set.

If possible, you should avoid using a Scan operation on a large table or index with a filter that removes many results. Also, as a table or index grows, the Scan operation slows. The Scan operation examines every item for the requested values and can use up the provisioned throughput for a large table or index in a single operation. For faster response times, design your tables and indexes so that your applications can use Query instead of Scan. (For tables, you can also consider using the GetItem and BatchGetItem APIs.)

Alternatively, design your application to use Scan operations in a way that minimizes the impact on your request rate.

## DynamoDB Global Table & Load Data from S3

### DynamoDB

### Lab Details

- DynamoDB Tables
- DynamoDB Two Read Models
- Cost vs. Performance - Two Capacity Modes
- DynamoDB Global Tables

## Task Details

- Create a DynamoDB Table with a Local Secondary Index
- Make a DynamoDB replica in another region, which is high availability
- Load data from S3 with COPY command

## DynamoDB Configuration

Services -> DynamoDB

### Create Table

Click on Tables navigation

Click on Create table

Name	Status	Partition key	Sort key	Indexes	Total read capacity	Total write capacity	Auto Scaling	Encrypt
Medical_Company	Active	Medical_Id (String)	-		0	5	5	DEFAULT

## Create DynamoDB table

- Table name: dynamodb\_lab
- Primary key
  - Partition key: String
    - customer\_id
  - Check Add sort key: String
    - order\_timestamp

The screenshot shows the 'Create DynamoDB table' wizard. The 'Table name' is set to 'dynamodb\_lab'. Under 'Primary key', 'customer\_id' is defined as a String partition key. An 'Add sort key' checkbox is checked, and 'order\_timestamp' is defined as a String sort key. In the 'Table settings' section, the 'Use default settings' checkbox is unchecked. The 'Secondary indexes' section shows a table with columns: Name, Type, Partition key, Sort key, and Projected Attributes. A '+ Add index' button is present. In the 'Read/write capacity mode' section, 'Provisioned (free-tier eligible)' is selected. The 'Provisioned capacity' section is collapsed. At the bottom, there are links for Feedback, English (US), Privacy Policy, and Terms of Use.

## Table settings

Uncheck Use default settings

## Secondary indexes

Click on + Add index

- Partition key: string
  - customer\_id

- Add sort key: string
  - order\_timestamp
- Projected attributes: All
- Check Create as Local Secondary Index

The preview of the new DynamoDB console is now available  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available  
You now can capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**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.

**Secondary indexes**

Name	Type	Partition key	Sort key	Projected Attributes
customer_id-order_timestamp-index	Local Secondary Index	customer_id	order_timestamp	All

**Read/write capacity mode**

Select on-demand if you want to pay only for the reads and writes you perform, with no capacity planning required. Select provisioned to save on throughput costs if you can reliably estimate your application's throughput requirements. See the [DynamoDB pricing page](#) and [DynamoDB Developer Guide](#) to learn more.

**Provisioned capacity**

Table	Read capacity units	Write capacity units
Table	5	5

Estimated cost \$2.91 / month ([Capacity calculator](#))

**Auto Scaling**

The preview of the new DynamoDB console is now available  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available  
You now can capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**Create DynamoDB table**

DynamoDB is a schema-less database that only requires a table name and primary key. The table's primary key is made up of one or two attributes that uniquely identify items, partition the data, and sort data within each partition.

**Table name\*** dynamodb\_lab

**Primary key\*** Partition key

Attribute	Type
customer_id	String
order_timestamp	String

**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

**Secondary indexes**

Name	Type	Partition key	Sort key	Projected Attributes
customer	Local Secondary Index	customer_id	order_timestamp	ALL

**Read/write capacity mode**

Select on-demand if you want to pay only for the reads and writes you perform, with no capacity planning required. Select provisioned to save on throughput costs if you can reliably estimate your application's throughput requirements. See the [DynamoDB pricing page](#) and [DynamoDB Developer Guide](#) to learn more.

Read/write capacity mode can be changed later.

Provisioned (free-tier eligible)

On-demand

Feedback English (US) ▾ © 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

**Projected attributes:** Projected attributes are attributes stored in the index, and can be returned by queries and scans performed on the index. Local secondary index queries can also return attributes that are not projected by fetching them from the table. Global secondary index queries can only return projected attributes. Note, that projected attributes incur write and storage costs. For more information and performance tuning tips, see [Projections](#).

## Read/write capacity mode

Check Provisioned (free-tier eligible)

## Provisioned capacity

For modifying **Read capacity units** or **Write capacity units**, uncheck **Auto Scaling** first.

You can also click on Capacity calculator

The screenshot shows the AWS DynamoDB console interface. At the top, there are two informational banners: one about the new console preview and another about Kinesis Data Streams for DynamoDB. Below these, the main configuration area is shown. Under 'Provisioned capacity', the 'Table' section has 'Read capacity units' set to 5 and 'Write capacity units' set to 5. The 'Estimated cost' is \$2.91 / month. A 'Capacity calculator' dialog is open, showing detailed settings: 'Avg. item size' is 1 KB, 'Item read/sec' is 1 (with 'Eventually consistent' checked), and 'Item write/sec' is 1 (with 'Strongly consistent' and 'Transactional' options available). The 'Capacity calculator' also displays 'Read capacity' as 1, 'Write capacity' as 1, and an 'Estimated cost' of \$0.59 / month per table/index. The 'Auto Scaling' section is visible below, with 'Read capacity' checked. Other settings like 'Target utilization' (70%), 'Minimum provisioned capacity' (5 units), and 'Maximum provisioned capacity' (40000 units) are also present. A note at the bottom of the main page says to check IAM permissions for Auto Scaling. The bottom of the screen includes standard AWS navigation links like Feedback, English (US), Privacy Policy, and Terms of Use.

See how the estimated cost change

The screenshot shows the AWS DynamoDB console with the Capacity calculator dialog box open. The dialog box displays the following settings:

- Avg. item size: 4 KB
- Item read/sec: 2 (Eventually consistent)
- Item write/sec: 1 (Standard)
- Read capacity: 1
- Write capacity: 4
- Estimated cost: \$2.04 / month per table/index

The main interface shows a table with 5 Read capacity units and 5 Write capacity units. The Auto Scaling section has a checkbox for "Read capacity" checked. A note says "Please check your IAM permissions to create new service linked role for enabling Auto Scaling. See permissions."

The screenshot shows the AWS DynamoDB console with the Capacity calculator dialog box open. The dialog box displays the following settings:

- Avg. item size: 4 KB
- Item read/sec: 2 (Strongly consistent)
- Item write/sec: 1 (Standard)
- Read capacity: 2
- Write capacity: 4
- Estimated cost: \$2.13 / month per table/index

The main interface shows a table with 5 Read capacity units and 5 Write capacity units. The Auto Scaling section has a checkbox for "Read capacity" checked. A note says "Please check your IAM permissions to create new service linked role for enabling Auto Scaling. See permissions."

The screenshot shows the AWS DynamoDB console with the 'Capacity calculator' dialog box open. The dialog box contains fields for 'Avg. item size' (4 KB), 'Item read/sec' (2 Transactional), and 'Item write/sec' (1 Standard). It also displays 'Read capacity' (4) and 'Write capacity' (4), with an estimated cost of '\$2.33 / month per table/index'. Below the dialog, there's a note about IAM permissions for Auto Scaling.

- Provisioned capacity
  - Specify read capacity units (RCUs) and write capacity units (WCUs) per second for your application
  - For Avg. item size up to 4 KB, every 4 KB of Avg. item will be counted as a coefficient
    - 1 RCU = 1 read/sec strongly consistent
    - 1 RCU = 2 read/rec eventually consistent
    - 2 RCU = 1 read/rec transactional
  - For Avg. item size up to 1 KB, every 1 KB of Avg. item will be counted as a coefficient
    - 1 WCU = 1 write/sec
  - Can use auto scaling to automatically calibrate your table's capacity
    - Helps manage cost

## Auto Scaling

Set as the screenshot.

The screenshot shows the AWS DynamoDB console with the 'Auto Scaling' configuration page. It includes sections for Read and Write capacity, target utilization, and minimum/maximum provisioned capacity. A note about IAM permissions is displayed in a box. The IAM Role section shows 'DynamoDB AutoScaling Service Linked Role' selected. The Encryption At Rest section shows 'DEFAULT' selected. The bottom of the page includes standard AWS navigation links.

## Encryption At Rest

Encryption service is free for DynamoDB.

KMS - AWS managed CMK is a good choice. But for this lab let we select Default .

Click on Create

The preview of the new DynamoDB console is now available  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available  
You now can capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

IAM Role I authorize DynamoDB to scale capacity using the following role:

- DynamoDB AutoScaling Service Linked Role
- Existing role with pre-defined policies [\[Instructions\]](#)

**Note:** If you have the required IAM permissions, a Service Linked Role will automatically be created on your behalf. [Learn more](#)

Role Name\*

**Encryption At Rest**

Select Server-side encryption settings for your DynamoDB table to help protect data at rest. [Learn more](#)

**DEFAULT**  
The key is owned by Amazon DynamoDB. You are not charged any fee for using these CMKs.

**KMS - Customer managed CMK**  
The key is stored in your account that you create, own, and manage. AWS Key Management Service (KMS) charges apply. [Learn more](#)

**KMS - AWS managed CMK**  
The key is stored in your account and is managed by AWS Key Management Service (KMS). AWS KMS charges apply.

aws/dynamodb

+ Add tags [\[NEW\]](#)

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**

## Global Tables Configuration

Click on Items tab.

The preview of the new DynamoDB console is now available  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available  
You now can capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**dynamodb\_lab** Close

**Overview** **Items** Metrics Alarms Capacity Indexes Global Tables Backups Contributor Insights Triggers Access control Tags

**Recent alerts**  
No CloudWatch alarms have been triggered for this table.

**Kinesis data stream details**  
Use Amazon Kinesis Data Streams for DynamoDB to capture item-level changes in your table as a Kinesis data stream. [Learn more](#)

Stream enabled	No
View type	-
Latest stream ARN	-

**Manage DynamoDB stream**

**Table details**

Table name	dynamodb_lab
Primary partition key	customer_id (String)
Primary sort key	order_timestamp (String)
Point-in-time recovery	DISABLED <a href="#">Enable</a>
Encryption Type	KMS <a href="#">Manage Encryption</a>
KMS Master Key ARN	arn:aws:kms:us-east-1:542892269888:key/1ac1968c-0ffa-4685-b750-191d5723db5d
Encryption Status	Enabled
CloudWatch Contributor Insights	DISABLED <a href="#">Manage Contributor Insights</a> <a href="#">[NEW]</a>
Time to live attribute	DISABLED <a href="#">Manage TTL</a>
Table status	Active
Creation date	January 19, 2021 at 11:38:58 PM UTC-8
Read/write capacity mode	Provisioned
Last change to on-demand mode	-
Provisioned read capacity units	5 (Auto Scaling Enabled)
Provisioned write capacity units	5 (Auto Scaling Enabled)
Last decrease time	-

Now we don't have any item.

## Click on Global Tables

The preview of the new DynamoDB console is now available. We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available. You now can capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**dynamodb\_lab** Close

- Overview
- Items**
- Metrics
- Alarms
- Capacity
- Indexes
- Global Tables
- Backups
- Contributor Insights
- Triggers
- Access control
- Tags

**Create item** Actions ▾

Scan: [Table] dynamodb\_lab: customer\_id, order\_timestamp

Start search

customer\_id order\_timestamp

An item consists of one or more attributes. Each attribute consists of a name, a data type, and a value. When you read or write an item, the only attributes that are required are those that make up the primary key. [More info](#)

## Click on Enable streams

The preview of the new DynamoDB console is now available. We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available. You now can capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**dynamodb\_lab** Close

- Overview
- Items
- Metrics
- Alarms
- Capacity
- Indexes
- Global Tables**
- Backups
- Contributor Insights
- Triggers
- Access control
- Tags

Global Tables enable you to use DynamoDB as a fully-managed, multi-region, multi-master database. [Learn more](#)

**Streams:** Disabled

**Stream type:** -

**Enable streams**

**IAM role** AWSServiceRoleForDynamoDBReplication Automatically created on your behalf.

**Global Table regions**

**Global table version:** 2019.11.21

Create a replica table in another region. [Learn more](#)

**Add region** Delete region

Region Name	Status	Read capacity units	Write capacity units	Auto Scaling	Endpoint

You do not have a global table with this name. Click "Add region" to create one.

## Click on Enable

The preview of the new DynamoDB console is now available  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available  
You can now capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**dynamodb\_lab** Close

Overview Items Metrics Alarms Capacity Indexes **Global Tables** Backups Contributor Insights Triggers Access control Tags

Global Tables enable you to use DynamoDB as a fully-managed, multi-region, multi-master database. [Learn more](#)

**Streams:** **Disabled**

**Manage Stream**

View type  New and old images - both the new and the old images of the item

**Enable**

IAM role

Global Table regions

Global table version: 2019.11.21

Create a replica table in another region. [Learn more](#)

Add region Delete region

Region Name	Status	Read capacity units	Write capacity units	Auto Scaling	Endpoint

You do not have a global table with this name. Click "Add region" to create one.

## Click on Add region

Select a region that is different from your **Current region**

Click on Create replica

The preview of the new DynamoDB console is now available  
We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available  
You can now capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**dynamodb\_lab** Close

Overview Items Metrics Alarms Capacity Indexes **Global Tables** Backups Contributor Insights Triggers Access control Tags

Global Tables enable you to use DynamoDB as a fully-managed, multi-region, multi-master database. [Learn more](#)

**Streams:** **Disabled**

**Add replica to global table**

Current region: US East (N. Virginia)

Global table version: 2019.11.21

Region **US West (Oregon)**

Region is ready. Click create replica to proceed.

You are creating the 2019.11.21 version global table. If you need to create a version 2017.11.29 global table, please follow the steps outlined here: [Learn more](#)

My application requires version 2017.11.29 global tables. [Global table 2017.11.29 mode](#)

**Create replica**

IAM role

Global Table regions

Global table version: 2019.11.21

Create a replica table in another region. [Learn more](#)

Add region Delete region

Region Name	Status	Read capacity units	Write capacity units	Auto Scaling	Endpoint

You do not have a global table with this name. Click "Add region" to create one.

Review the settings and click on Close

The preview of the new DynamoDB console is now available. We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available. You can now capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**Add replica to global table**

Current region: US East (N. Virginia)

Global table version: 2019.11.21

Replica in the intended region has been initiated successfully. It may take a few minutes for the replica to be created.

**dynamodb\_lab**

Table name: dynamodb\_lab  
Region: US West (Oregon)  
Primary partition key: customer\_id (String)  
Primary sort key: order\_timestamp (String)  
Read/write capacity mode: Provisioned  
Provisioned read capacity units: 5  
Provisioned write capacity units: 5  
Auto Scaling: READ\_AND\_WRITE  
Stream enabled: Yes  
Encryption Type: KMS

**Close**

You do not have a global table with this name. Click "Add region" to create one.

We are now in US EAST (N. Virginia). So we click on US WEST (Oregon) to see the replica.

The preview of the new DynamoDB console is now available. We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available. You can now capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**dynamodb\_lab**

**Overview** **Items** **Metrics** **Alarms** **Capacity** **Indexes** **Global Tables** **Backups** **Contributor Insights** **Triggers** **Access control** **Tags**

Global Tables enable you to use DynamoDB as a fully-managed, multi-region, multi-master database. [Learn more](#)

IAM role: **AWSServiceRoleForDynamoDBReplication** (Automatically created on your behalf.)

Global table version: 2019.11.21

Create a replica table in another region. [Learn more](#)

**Add region** **Delete region**

Region Name	Status	Read capacity units	Write capacity units	Auto Scaling	Endpoint
US East (N. Virginia)	Active	5	5	READ_AND_WRITE	dynamodb.us-east-1.amazonaws.com
US West (Oregon)	Active	5	5	READ_AND_WRITE	dynamodb.us-west-2.amazonaws.com

[View all CloudWatch metrics](#) (Last updated 1 hour ago)

Time Range: Last Hour

Global table metrics

Replication latency (Milliseconds)

1

The preview of the new DynamoDB console is now available. We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available. You now can capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**dynamodb\_lab**

**Overview** **Items** **Metrics** **Alarms** **Capacity** **Indexes** **Global Tables** **Backups** **Contributor Insights** **Triggers** **Access control** **Tags**

**Recent alerts**  
No CloudWatch alarms have been triggered for this table.

**Kinesis data stream details**  
Use Amazon Kinesis Data Streams for DynamoDB to capture item-level changes in your table as a Kinesis data stream. [Learn more](#)

<b>Stream enabled</b>	Yes
<b>View type</b>	New and old images
<b>Latest stream ARN</b>	arn:aws:dynamodb:us-west-2:542892269888:table/dynamodb_lab/stream/2021-01-20T07:48:58.154

**Table details**

Table name	dynamodb_lab
Primary partition key	customer_id (String)
Primary sort key	order_timestamp (String)
Point-in-time recovery	DISABLED <a href="#">Enable</a>
Encryption Type	KMS <a href="#">Manage Encryption</a>
KMS Master Key ARN	arn:aws:kms:us-west-2:542892269888:key/af483552-c666-4faf-b77a-4636e77f863b
Encryption Status	Enabled
CloudWatch Contributor Insights	DISABLED <a href="#">Manage Contributor Insights</a> NEW
Time to live attribute	DISABLED <a href="#">Manage TTL</a>
Table status	Active
Creation date	January 19, 2021 at 11:48:58 PM UTC-8
Read/write capacity mode	Provisioned
Last change to on-demand mode	-
Provisioned read capacity units	5 (Auto Scaling Enabled)
Provisioned write capacity units	5 (Auto Scaling Enabled)
Last decrease time	-

## Add items to the primary region by Python

```

1 import boto3
2
3 # Get the DynamoDB service resource.
4 dynamodb = boto3.resource('dynamodb', region_name='us-east-1')
5
6 # Instantiate a table resource object without actually
7 # creating a DynamoDB table in our code. Know that the attributes of
8 # the table are lazy-loaded: the attribute
9 # values are not populated until the attributes
10 # on the table resource are accessed or its load() method is called.
11 table = dynamodb.Table('dynamodb_lab')
12
13 # Print out some data about the table.
14 # This will cause a request to DynamoDB and the table attribute
15 # values will be set via the response.
16 print("\n\ttable creatin date-time {}\n\ttable key schema {}".format(
17         table.creation_date_time, table.key_schema))
18
19 # Put an item into our table
20 table.put_item(

```

```

21     Item={
22         'username': 'Zacks',
23         'first_name': 'Zacks',
24         'last_name': 'Shen',
25         'age': 26,
26         'customer_id': '00000001',
27         'order_timestamp': '05-29-2018'
28     }
29 )

```

Save the code to your local machine then change directory to the program and run it.

If an permission error pops up, go check AWS CLI.

## Python

1 python3 DynamoDBLoader.py

```

aws-certification.md  KinesisProducer-C&F.py  aws-kinesis-lab.md  aws-certified-data-analytics.md  aws-database-lab.md  DynamoDBLoader.py  aws-glue-lab.md
Code > AWS > AWS Data Analytics > DynamoDBLoader.py > ...
19
20 import boto3
21
22 # Get the DynamoDB service resource.
23 dynamodb = boto3.resource('dynamodb', region_name='us-east-1')
24
25 # Instantiate a table resource object without actually
26 # creating a DynamoDB table in our code. Note that the attributes of
27 # the table are lazy-loaded: the attribute
28 # values are not populated until the attributes
29 # on the table resource are accessed or its load() method is called.
30 table = dynamodb.Table('dynamodb_lab')
31
32 # Print out some data about the table.
33 # This will cause a request to DynamoDB and the table attribute
34 # values will be set via the response.
35 print(table.creation_date_time)
36
37
38 # Put an item into our table
39 table.put_item(
40     Item={
41         'username': 'Zacks',
42         'first_name': 'Zacks',
43         'last_name': 'Shen',
44         'age': 26,
45         'customer_id': '00000001',
46         'order_timestamp': '05-29-2018'
47     }
48 )

```

OUTPUT TERMINAL SQL CONSOLE MESSAGES DEBUG CONSOLE PROBLEMS 157

```

(base) ➜ AWS Data Analytics git:(master) ✘ 11
total 40
-rw-r--r-- 1 zacks staff 2.2K Jan 29 00:16 DynamoDBLoader.py
-rw-r--r-- 1 zacks staff 2.7K Jan 29 00:16 KinesisClient.py
-rw-r--r-- 1 zacks staff 3.0K Jan 18 18:21 KinesisProducer-C&F.py
-rw-r--r-- 1 zacks staff 2.4K Jan 16 23:51 KinesisProducer.py
-rw-r--r-- 1 zacks staff 306B Jan 18 22:41 Surveillance.csv
(base) ➜ AWS Data Analytics git:(master) ✘ 12
(base) ➜ AWS Data Analytics git:(master) ✘ 13
(base) ➜ AWS Data Analytics git:(master) ✘ 14
table creation date-time 2021-01-19 23:38:58.074000+08:00
table key schema [{AttributeName: 'customer_id', KeyType: 'HASH'}, {AttributeName: 'order_timestamp', KeyType: 'RANGE'}]
(base) ➜ AWS Data Analytics git:(master) ✘ 15

```

Python 3.8.3 64-bit ('anaconda3': virtualenv) ② 26 ▲ 4 ① 127 Connect HelloAspNetCore ✓ python | ✓ DynamoDBLoader.py

See the information from terminal: table key schema [`{'AttributeName': 'customer_id', 'KeyType': 'HASH'}`, `{'AttributeName': 'order_timestamp', 'KeyType': 'RANGE'}`]

## Validation Test

Go to your regions to check the DynamoDB database.

### US EAST (N. Virginia)

Click on Items tab then click on refresh button.

The screenshot shows the AWS DynamoDB console for the US EAST (N. Virginia) region. On the left, the navigation menu is visible with options like Dashboard, Tables, Backups, Reserved capacity, Preferences, DAX, and Events. A message at the top says "The preview of the new DynamoDB console is now available" and "We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think." Another message below it says "Kinesis Data Streams for DynamoDB is now available" and "You now can capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications." The main area shows the "dynamodb\_lab" table. The "Items" tab is selected, showing a single item with the following details:

customer_id	order_timestamp	age	first_name	last_name	username
00000001	05-29-2018	26	Zacks	Shen	Zacks

### US WEST (Oregon)

Click on Items tab then click on refresh button.

The preview of the new DynamoDB console is now available. We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available. You now can capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**dynamodb\_lab**

**Items**

Scan: [Table] dynamodb\_lab: customer\_id, order\_timestamp

customer_id	order_timestamp	age	first_name	last_name	username
00000001	05-29-2018	26	Zacks	Shen	Zacks

## Add items to the replica region by Python

Modify the program, `region_name` and `Item`.

```
Python3 DynamoDBLoader.py

1 import boto3
2
3 # Get the DynamoDB service resource.
4 dynamodb = boto3.resource('dynamodb', region_name='us-west-2')
5
6 # Instantiate a table resource object without actually
7 # creating a DynamoDB table in our code. Know that the attributes of
8 # the table are lazy-loaded: the attribute
9 # values are not populated until the attributes
10 # on the table resource are accessed or its load() method is called.
11 table = dynamodb.Table('dynamodb_lab')
12
13 # Print out some data about the table.
14 # This will cause a request to DynamoDB and the table attribute
15 # values will be set via the response.
16 print("\n\ttable creatin date-time {}\n\ttable key schema {}".format(
17         table.creation_date_time, table.key_schema))
18
```

```

19 # Put an item into our table
20 table.put_item(
21     Item={
22         'username': 'Amber',
23         'first_name': 'Amber',
24         'last_name': 'Lyu',
25         'age': 20,
26         'customer_id': '00000002',
27         'order_timestamp': '10-15-2020'
28     }
29 )

```

Run the program.

If an permission error pops up, go check AWS CLI.



## Python

```
1 python3 DynamoDBLoader.py
```

```

aws-certification.md  KinesisProducer-C&F.py  aws-kinesis-lab.md  aws-certified-data-analytics.md  aws-database-lab.md  DynamoDBLoader.py  aws-glue-lab.md
Code > AWS Data Analytics > DynamoDBLoader.py >
1
2 import boto3
3
4 # Get the DynamoDB service resource.
5 dynamodb = boto3.resource('dynamodb', region_name='us-west-2')
6
7 # Instantiate a table resource object without actually
8 # creating a DynamoDB table in our code. Note that the attributes of
9 # the table are lazy-loaded: the attribute
10 # values are not populated until the attributes
11 # on the table resource are accessed or its load() method is called.
12 table = dynamodb.Table('dynamodb_lab')
13
14 # Print out some data about the table.
15 # This will cause a request to DynamoDB and the table attribute
16 # values will be set via the response.
17 print("\ntable creation date-time {} \ntable key schema {}".format(
18     table.creation_date_time, table.key_schema))
19
20 # Put an item into our table
21 table.put_item(
22     Item={
23         'username': 'Amber',
24         'first_name': 'Amber',
25         'last_name': 'Lyu',
26         'age': 20,
27         'customer_id': '00000002',
28         'order_timestamp': '10-15-2020'
29     }
30 )

```

OUTPUT TERMINAL SQL CONSOLE: MESSAGES DEBUG CONSOLE PROBLEMS 160

```

(base) AWS Data Analytics git:(master) x ll
total 40
-rw-r--r-- 1 zacks staff 2.2K Jan 20 00:16 DynamoDBLoader.py
-rw-r--r-- 1 zacks staff 2.7K Jan 18 18:03 KinesisClient.py
-rw-r--r-- 1 zacks staff 3.8K Jan 18 18:23 KinesisProducer-C&F.py
-rw-r--r-- 1 zacks staff 2.7K Jan 18 22:51 KinesisProducer.py
-rw-r--r-- 1 zacks staff 306B Jan 18 22:41 Surveillance.csv
(base) AWS Data Analytics git:(master) x py DynamoDBLoader.py
(base) AWS Data Analytics git:(master) x py DynamoDBLoader.py
table create-date-time 2021-01-19 23:38:58.074000-08:00
table key schema [ { 'AttributeName': 'customer_id', 'KeyType': 'HASH' }, { 'AttributeName': 'order_timestamp', 'KeyType': 'RANGE' } ]
(base) AWS Data Analytics git:(master) x py DynamoDBLoader.py
table create-date-time 2021-01-19 23:38:58.074000-08:00
table key schema [ { 'AttributeName': 'customer_id', 'KeyType': 'HASH' }, { 'AttributeName': 'order_timestamp', 'KeyType': 'RANGE' } ]
(base) AWS Data Analytics git:(master) x

```

Ln 48, Col 2 Spaces: 4 UTF-8 LF Python

See the information from terminal: table key schema [ { 'AttributeName': 'customer\_id', 'KeyType': 'HASH' }, { 'AttributeName': 'order\_timestamp', 'KeyType': 'RANGE' } ]

## Validation Test

Go to your regions to check the DynamoDB database.

### US WEST (Oregon)

Click on Items tab then click on refresh button.

The screenshot shows the AWS DynamoDB console interface. On the left, the navigation menu is visible with 'Tables' selected. In the center, the 'dynamodb\_lab' table is selected. The top navigation bar has tabs for Overview, Items, Metrics, Alarms, Capacity, Indexes, Global Tables, Backups, Contributor Insights, Triggers, Access control, and Tags. The 'Items' tab is currently active. Below the tabs, there is a search bar and a filter section. The main area displays a table with two items:

customer_id	order_timestamp	age	first_name	last_name	username
00000001	05-29-2018	26	Zacks	Shen	Zacks
00000002	10-15-2020	20	Amber	Lyu	Amber

### US EAST (N. Virginia)

Click on Items tab then click on refresh button.

SHARES

The preview of the new DynamoDB console is now available. We are redesigning the DynamoDB console. The preview of the new console is a work in progress, but we encourage you to try it and let us know what you think.

Kinesis Data Streams for DynamoDB is now available. You now can capture item-level changes in your DynamoDB tables as a Kinesis data stream and start taking advantage of Kinesis services to build advanced streaming applications.

**dynamodb\_lab**

**Items**

Scan: [Table] dynamodb\_lab: customer\_id, order\_timestamp

customer_id	order_timestamp	age	first_name	last_name	username
00000001	05-29-2018	26	Zacks	Shen	Zacks
00000002	10-15-2020	20	Amber	Lyu	Amber

Feedback English (US) ▾

© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

[Privacy Policy](#)[Terms of Use](#) [Tech](#) [AWS](#) [Labs](#) [AWS DAS](#) [DB](#) [Analytics](#) [NoSQL](#) [AWS Redshift Lab](#) [AWS Elasticsearch Service](#)

© 2023 Zacks Shen

3.8m | 57:57

Powered by [Hexo](#) & [NexT.Mist](#)

SHARES