

Export Data from DynamoDB to S3 as CSV File

Lab Steps

Task 1: Sign in to AWS Management Console

1. Click on the **Open Console** button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,
 - a. Leave the Account ID as default. Never edit/remove the 12 digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.
 - b. Now copy your **User Name** and **Password** in the Lab Console to the **IAM Username and Password** in AWS Console and click on the **Sign in** button.
3. Once Signed In to the AWS Management Console, Make the default AWS Region as **US East (N. Virginia) us-east-1**.

Task 2: Create an S3 Bucket

1. Make sure you are in the **N.Virginia** Region.
2. Navigate to the **Services** menu at the top. Click on **S3** in the **Storage** section.
3. On the S3 Page, click on the **Create Bucket** and fill in the bucket details.
 - a. Bucket type select : **General Purpose**
 - b. Bucket name: Enter **whizlabs-dynamodb-s3-<RANDOM NUMBER>**

(Note: S3 bucket names are globally unique. So, in the end, enter some random number so that the bucket name will be unique.)

The screenshot shows the 'Create bucket' page in the AWS Management Console. The 'General configuration' section is selected. Under 'AWS Region', 'US East (N. Virginia) us-east-1' is chosen. Under 'Bucket type', 'General purpose' is selected (indicated by a red box). The 'Bucket name' field contains 'whizlabs-dynamodb-s3-123' (also highlighted with a red box). Below the name field, a note states: 'Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)'.

- Object ownership: Select **ACLs disabled (recommended)** option
- Leave other settings as default. Click on the **Create Bucket**.

(Note: This newly-created bucket is the upload bucket and will be used to upload the CSV files.)

4. The AWS S3 bucket is now created.
5. Copy the S3 bucket in your local editor for later use.

Task 3: Create a DynamoDB Table

1. Make sure you are in the **US East (N. Virginia) us-east-1** Region.
2. Navigate to **the DynamoDB** page by clicking on the **Services** menu at the top.

DynamoDB is available under the **Database** section.

3. Click on **Create Table**.
 - a. Table Name: Enter **mydynamoDB**
 - b. Partition key: **Sid** and select **String**.
 - c. Leave other options as **Default**.
 - d. Click on **Create table** button
4. Your table will be created within 2-3 minutes.
5. Copy the dynamoDB name in your local editor for later use.

Task 4: Inserting Data into a DynamoDB Table

1. Next, we need to **create an Item** and then **insert data** into the table.
2. Select the DynamoDB table and click on the **Explore table items** button
3. Then click on **Create item** button
4. Create Item
 - a. **Attribute Name:** **Sid**
 - b. **Value:** Enter **1**
 - c. Click on **Add New Attribute**, and select **String**.
 - i. **Attribute name:** Enter **Name**
 - ii. **Value:** Enter **Nikhil**
 - d. Click on **Add New Attribute**, and select **String**.
 - i. **Attribute name:** Enter **Course**
 - ii. **Value:** Enter **AWS CSAA**

Attributes		
Attribute name	Value	Type
Sid - Partition key	1	String
Name	Nikhil	String
Course	AWS CSAA	String
Add new attribute ▾		

- Click on **Create Item**.
- Repeat the insertion for the **below values**.

Note: Attribute name of the table is case sensitive i.e., "Name" and "name" will be treated as two different attributes. Similarly for Course.

	Sid	Course	Name
<input type="checkbox"/>	3	AWS CSAA	Rahul
<input type="checkbox"/>	2	AWS Devops	Pavan
<input type="checkbox"/>	1	AWS CSAA	Nikhil

Task 5: Create a Lambda Function

1. Go to the **Services** menu and click on **Lambda**
2. Make sure you are in the **US East (N. Virginia)** region.
3. Click on the **Create Function** button.
 - a. Choose **Auto from Scratch**.
 - b. Function name : Enter ***myLambdaFunction***
 - c. Runtime: Select **Node.js 16.x**

- d. Permissions: Click on the **Change default execution role** and choose **Use an Existing role** under **Execution role**.
- e. **Existing role** : Select **Lambda-s3-dynamodb_RANDOM_NUMBER** from the dropdown list.

Permissions [Info](#)

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

Create a new role with basic Lambda permissions
 Use an existing role
 Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

Lambda-s3-dynamodb

[View the Lambda-s3-dynamodb role](#) on the IAM console.

- f. Click on **Create Function**.

4. **Configuration Page:** Here we need to configure our lambda function. If you scroll down, you can see the **Code Source** section. Here we need to write some **Node.js code** that will read the items in the DynamoDB table and store them in the S3 bucket as CSV format.
5. Download the zip file [lambda_code.zip](#) and extract the file.
6. Remove the existing code from AWS lambda **index.js**.
7. Replace the Lambda function code with the contents of the downloaded **index.js** file.

Note: Replace the bucket name to your S3 bucket in Line 4 and replace the DynamoDB table name with your table name in Line 10.

8. Save the function by clicking on **the Deploy** button.

Task 6: Configure Test Event

1. Click on the **Test** button.
2. On the **Configure test event** page,
 - a. Event Name: Enter **myTestFunction**
 - b. Leave other fields as **default**.
 - c. Click on **Save** button

Task 7: Exporting Data from DynamoDB to S3

1. Once the **myTestFunction** is configured, we can manually trigger the lambda using a test event.
2. Click on the **Test** button.
3. Lambda function will be executed, it will read all the items from the DynamoDB table **mydynamoDB** and save them to the S3 Bucket **whizlabs-dynamodb-s3**.
4. Once it's completed, you will see a **success message**.

```

▼ Execution results
Test Event Name
myTestFunction

Response
"200 : Success"

Function Logs
START RequestId: a7334fed-60a1-4754-beb0-14c96128de89 Ver
2021-06-10T14:19:28.668Z      a7334fed-60a1-4754-beb0-14c96
    { Sid: '2', Course: 'AWS Devops', Name: 'Pavan' },
    { Sid: '1', Course: 'AWS CSAA', Name: 'Nikhil' },
    { Sid: '3', Course: 'AWS CSAA', Name: 'Rahul' }
]
2021-06-10T14:19:28.688Z      a7334fed-60a1-4754-beb0-14c96
CSV File content
Sid,Name,Course
2,Pavan,AWS Devops
1,Nikhil,AWS CSAA
3,Rahul,AWS CSAA

```

Task 8: View the CSV file in the S3 bucket

1. Navigate to the **Services** menu at the top. Click on **S3** in the **Storage** section.
2. Click on the **whizlabs-dynamodb-s3-123** S3 Bucket.
3. Click on the object **myfile1.csv**
4. On the right top corner, click on the **Download** option.
- 5. It will ask you to save the file in your local system.**
6. Once it is downloaded to your local machine by selecting your favorite text editor, open the CSV file, see the contents.

Do you know?

Exporting data from Amazon DynamoDB to Amazon S3 as CSV files is a common task in data engineering and analytics workflows. DynamoDB is a fully managed NoSQL

database service provided by Amazon Web Services (AWS), while Amazon S3 (Simple Storage Service) is a scalable object storage service. This process allows you to efficiently store and analyze your DynamoDB data in a tabular format using CSV files.