

1) Assemble a small data file, that contains a list of JSON objects, **at least 20 to 30 objects**. Use data that comes from fakestore API or some other site.

2) Place the data file on an S3 bucket. You can use the same bucket you used for your web page assignment.

The file should be accessible through a URL. Give it appropriate ACL public permissions and enable CORS.

#### NOTES:

Please check Enable ACL (access control lists) when you create the bucket. That way you'd be able to make objects in the bucket public, by ACL.

Please make sure that the .json datafile that shows on the bucket link is well formed. That is, a list of json objects, each object is a complete object. If the data is not well formed, it will be rejected by DynamoDB.

[{ }, { }, { }, { }, { }]

3) Create a new DynamoDB table, make sure the Partition key agrees with what you'd use as index in the data file. My sample data file uses id (Number)

The screenshot shows the AWS DynamoDB 'fakeproductstable' table details. The table has a single partition key 'id (Number)'. It is set to 'On-demand' capacity mode and is currently 'Active'. There are no active alarms. Point-in-time recovery (PITR) is disabled. An 'Additional info' button is visible at the bottom left.

Partition key	Sort key	Capacity mode	Table status
id (Number)	-	On-demand	Active

**Good practice:** Actions, Edit Capacity, **on-demand** so that you won't get charged just for having the table.

4) Create an execution role for the Lambda function with a policy to access DynamoDB.

IAM > Roles > DynamoDB\_FullAccess

## DynamoDB\_FullAccess

[Delete](#)

Allows Lambda functions to call AWS services on your behalf.

### Summary

[Edit](#)

Creation date  
September 02, 2022, 11:45 (UTC-05:00)

ARN  
[arn:aws:iam::117474402122:role/DynamoDB\\_FullAccess](#)

Last activity  
 19 minutes ago

Maximum session duration  
1 hour

[Permissions](#) [Trust relationships](#) [Tags](#) [Access Advisor](#) [Revoke sessions](#)

#### Permissions policies (1) Info

You can attach up to 10 managed policies.



[Simulate](#)

[Remove](#)

[Add permissions](#) ▾

<input type="checkbox"/>	Policy name	Type	Description
<input type="checkbox"/>	<a href="#">AmazonDynamoDBFullAccess</a>	AWS managed	Provides full access to Amazon

- 5) Write a Lambda function (**Runtime Python 3.13**) that reads the data from the URL of step 2) and batch writes it to the DynamoDB table. The function uses **Environment variables**.

// Code given in class

You may need to increase the **timeout** of the function.

- 6) If all OK, Deploy and Test the function to see that the function has added the Items to the table! It may take some time to get the table updated with the new Items!

fakeproductstable

[Run](#) [Reset](#)

Completed · Items returned: 20 · Items scanned: 20 · Efficiency: 100% · RCUs consumed: 2.5

**Table: fakeproductstable - Items returned (20)**

Scan started on October 04, 2025, 22:09:38

<input type="checkbox"/>	<a href="#">id (Number)</a>	<a href="#">category</a>	<a href="#">description</a>	<a href="#">image</a>	<a href="#">price</a>	<a href="#">rating</a>
<input type="checkbox"/>	<a href="#">7</a>	jewelery	Classic Create...	<a href="https://fak...">https://fak...</a>	9.99	{ "count":
<input type="checkbox"/>	<a href="#">8</a>	jewelery	Rose Gold Pla...	<a href="https://fak...">https://fak...</a>	10.99	{ "count":
<input type="checkbox"/>	<a href="#">10</a>	electronics	Easy upgrade ...	<a href="https://fak...">https://fak...</a>	109	{ "count":