Amazon DynamoDB

- Fully managed NoSQL database service
- Supports key-value and document data models.
- Serverless, schema less and event-driven programming
- Petabytes of data storage and tens of millions of read and write requests per second.
- Fine-grained access control
- Fast, flexible, and secure
- Single-digit millisecond performance at any scale
- Support 99.99% availability with replication across multi AZs
- Auto scaling capability and Scale horizontally
- Offers encryption at rest.
- Data is stored on SSDs.
- Cost effective.
- Working with an Online Transaction Processing (OLTP) workload
- Is made of tables.
- Each table has a primary key to supports GET/PUT operations.
- Each table can have infinite no. of items (=rows)
- Each item has attributes.
- Max. size of item is 400KB.
- Use cases: Ad Tech, Retail, Banking and Finance, Media and entertainment, Gaming, Software and internet.

Data types

1. Scalar Types – number, string, binary, Boolean, and null
2. Document Types – list and map.
3. Set Types – string set, number set, and binary set.

Table class

1. **Standard table** - recommended for most workloads.
2. **Standard-infrequent access (IA) table** - store infrequently accessed data, such as application logs, old social media posts, e-commerce order history, and past gaming achievements.

Read consistency model.

Eventually consistent reads (the default)	Strongly consistent reads
Propagation of latest update might take a few minutes longer. It is possible to miss the latest update.	You always read the latest update.
Fastest possible reads	Slower than eventually consistent reads
Cheapest possible reads. 2 Eventually consistent reads cost 1RCU	Twice as expensive as Eventually consistent reads. Each Strongly consistent reads cost 1RCU.
Low latency	High latency
Highly available	Weak availability

Read/write capacity mode.

Controls how you are charged for read and write throughput and how you manage capacity.

Provisioned mode (default, free-tier eligible)	On-demand mode
You need to specify the no. of reads/writes per second	Read/writes automatically scale up/down with your workloads
You need to plan capacity beforehand	No capacity planning needed
Pay for provisioned RCU & WCU	Pay for what you use, more expensive
Possibility to add auto scaling mode for RCU & WCU	Great for unpredictable workloads, steep sudden spikes

DynamoDB Global Tables

- fully managed, multi-Region, and multi-active database
- delivers fast, local, read & write performance for massively scaled global applications.
- Easy to set up and no application re-writes required.
- Support Eventual consistency.
- deliver low-latency data access to users.
- Must enable DynamoDB stream as a pre-requisite.

DynamoDB read and write operations.

- DynamoDB API or PartiQL
- Perform basic create, read, update, and delete (CRUD) functionality.

DynamoDB Time to Live (TTL)

- Automatically delete items after an expiry Timestamp
- Reduce extra cost by reducing stored data volumes by keeping only the current items

DynamoDB streaming models.

Supports streaming of item-level change data capture records in near-real time

Use cases:

- React to changes in real-time.
- Real-time usage analytics
- Insert into derivative tables.
- Implement cross-region replication.
- Invoke lambda on changes to your tables.

DynamoDB Streams	Kinesis Data Streams for DynamoDB
24 hours retention	1 year retention
Limited no. of consumers	High no. of consumers
Process stream records using AWS Lambda or DynamoDB Streams Kinesis adapter.	Process stream records using AWS Lambda, Kinesis Data Analytics, Kinesis data firehose, or AWS Glue streaming ETL.

DynamoDB Backup for DR

Continuous backups using Point-in-time recovery (PITR)

- protect your tables from accidental write or delete operations.
- Can restore a table to any point in time during the last 35 days.
- Recovery process always restores to a new table.

On-demand backups

- full backups for long-term retention and archiving for regulatory compliance needs.
- Doesn’t affect performance or latency.
- Can be configured and managed in AWS Backup
- Recovery process always restores to a new table.

DynamoDB Accelerator (DAX)

- DAX is a DynamoDB-compatible caching service.
- Fully managed, highly available, in-memory cache for Amazon DynamoDB
- Delivers millions of requests per Microsecond
- Applications remain fast and responsive.
- Start with a three-node DAX cluster and then scale on demand.
- Automates many common administrative tasks such as failure detection, failure recovery, and software patching.
- Secure, Easy to use and give you maximal flexibility.
- Supports server-side encryption.
- Only support eventually consistent reads
- Only available for the EC2-VPC platform
- Saves cost by reducing load (RCU) on DynamoDB
- By default, 5 minutes TTL for cache

Use cases for DAX:

- Applications that require the fastest possible response time for reads.
- Applications that read a small number of items more frequently than others.
- Applications that are read-intensive but are also cost-sensitive.
- Applications that require repeated reads against a large set of data.

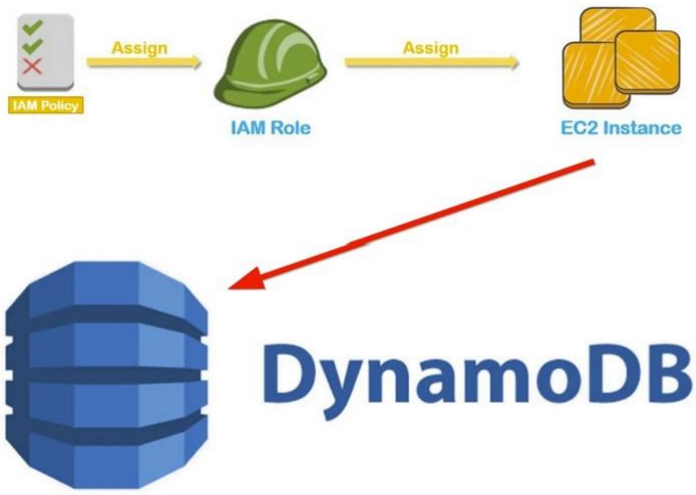
DynamoDB transactions

Provide Atomicity, Consistency, Isolation, and Durability (ACID) transactions.

Use cases:

- Processing financial transactions
- Fulfilling and managing orders
- Building multiplayer game engines
- Coordinating actions across distributed components and services

How to Create a DynamoDB Table with the AWS Console



Disha

- Create a DynamoDB table.
- Create a Ec2 instance.
- Using an IAM role and the principle of least privilege, grant the EC2 instance read access to DynamoDB.
- Use the AWS CLI in the EC2 instance to scan the DynamoDB table.
- Use the AWS CLI in the EC2 instance to validate you cannot write an item to the DynamoDB table.

Prerequisites:

AWS account and basic knowledge of DynamoDB

Step 1: Create a table.

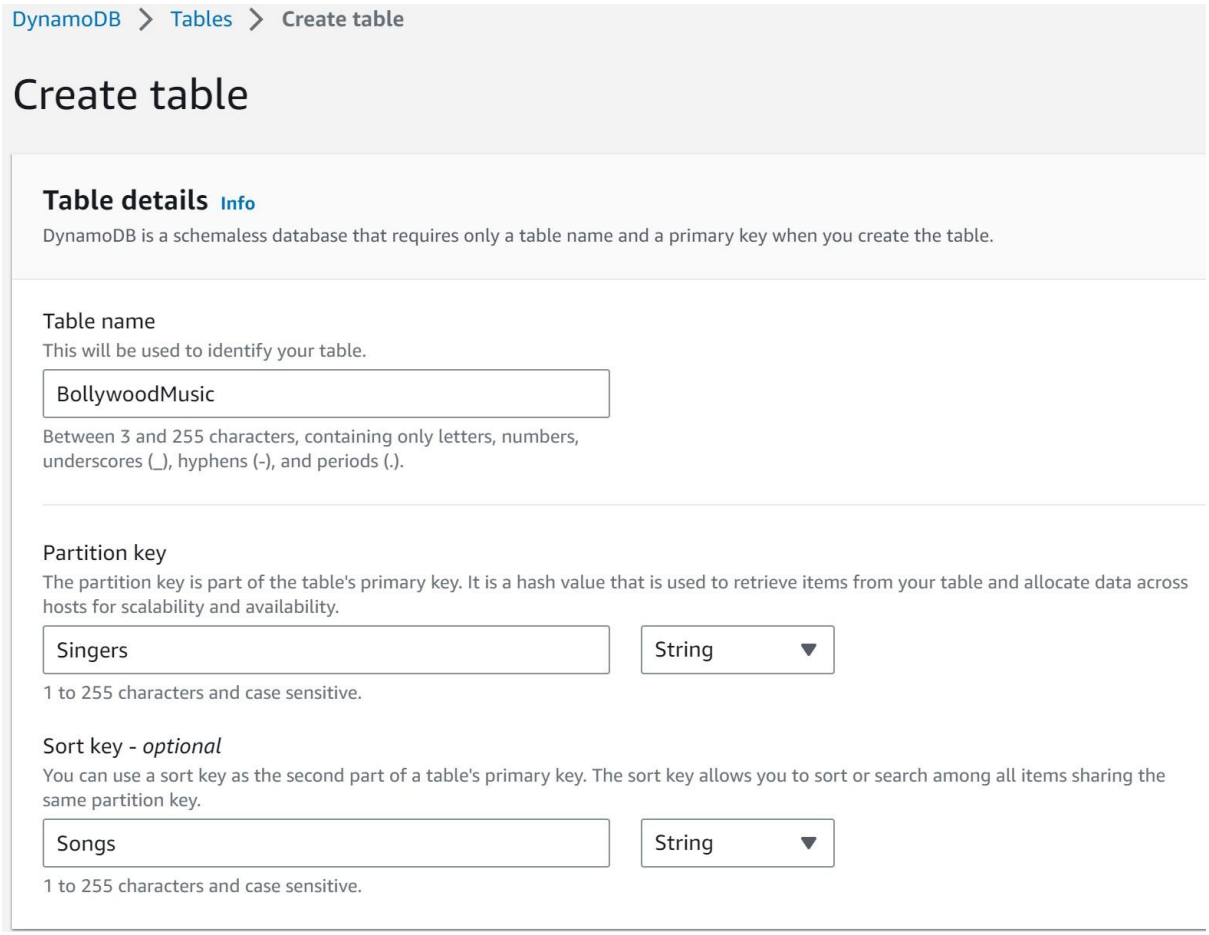
AWS console and navigate to Database and then DynamoDB. Click on Create Table.

Create a “BollywoodMusic” table with following details:

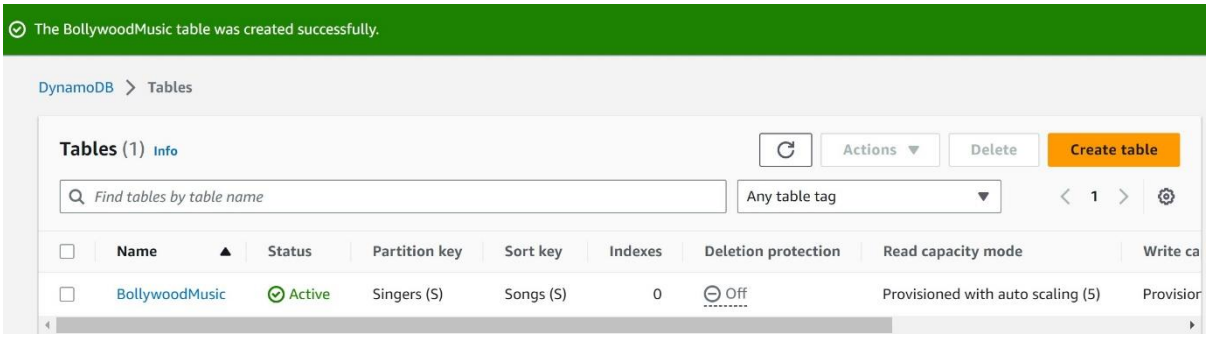
Partition key — Singers

Sort key — Songs

Table settings – Default settings

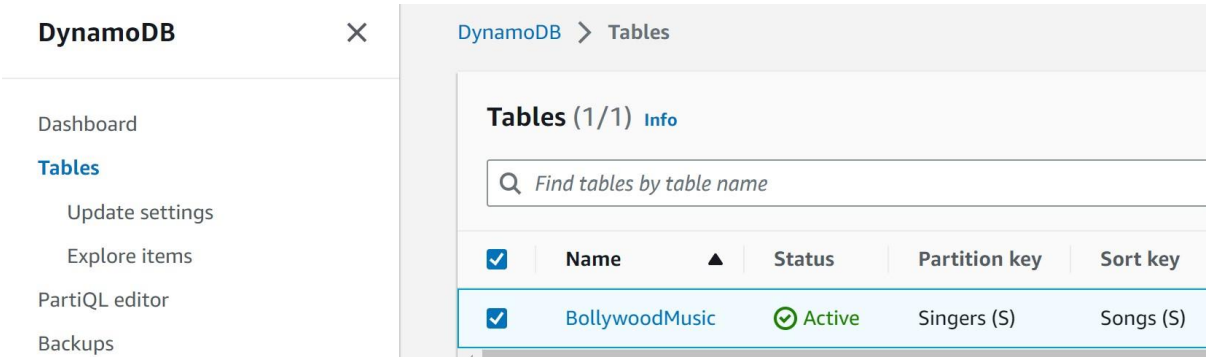


Click on “Create Table”.

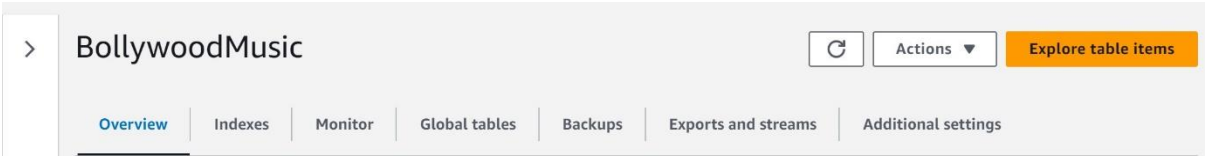


Step 2: Write data.

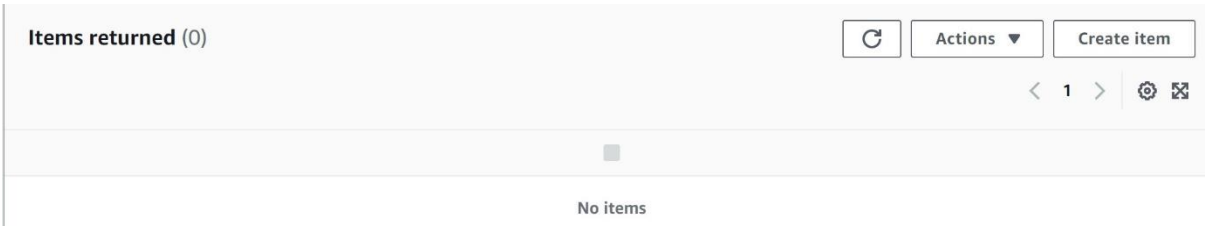
In the table list, choose the BlooywoodMusic table.



Select Explore Table Items.



In the Items view, choose Create item.



On this page it should show your partition key and sort key. Here you can add the data of your choosing to your table!

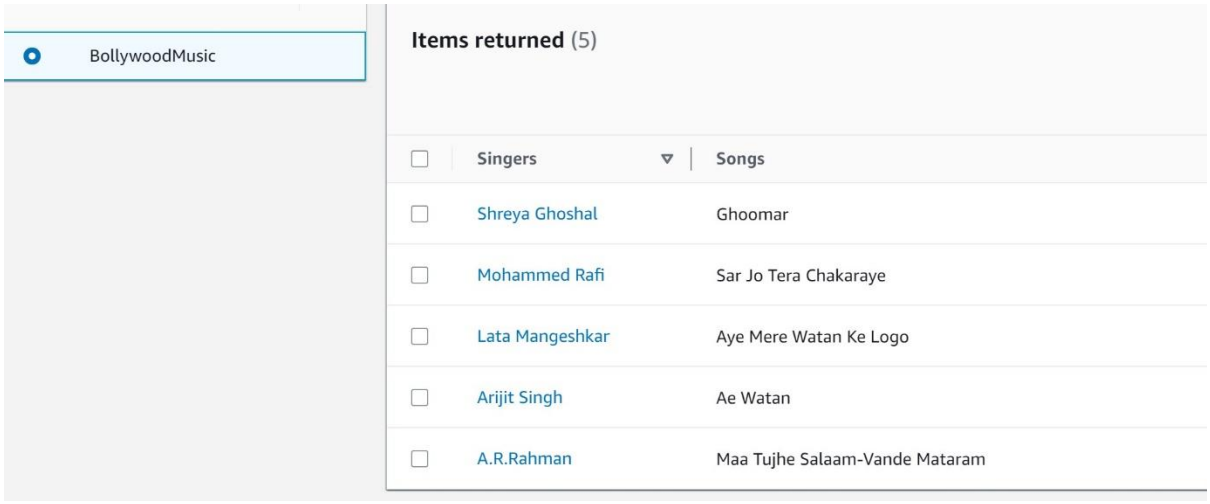
Enter the following values for your item:

For Singers – enter Singer's name as the value.

For Song – enter Singer's songs as the value.

Choose Create item.

Keep creating new items until you are satisfied with your table.



Step 3: Create a EC2 instance.

Click on the EC2 service.

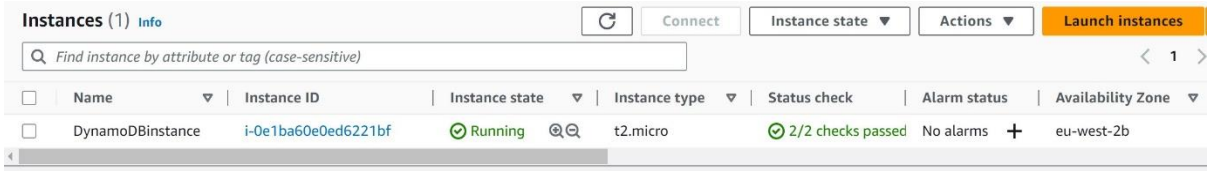
Click on the Launch Instance button to create a new instance – DynamoDBinstance.

Choose Amazon Linux AMI and T2. micro instance type.

Create new key pair – mydynamodbkey.pem.

Configure Default network setting and storage.

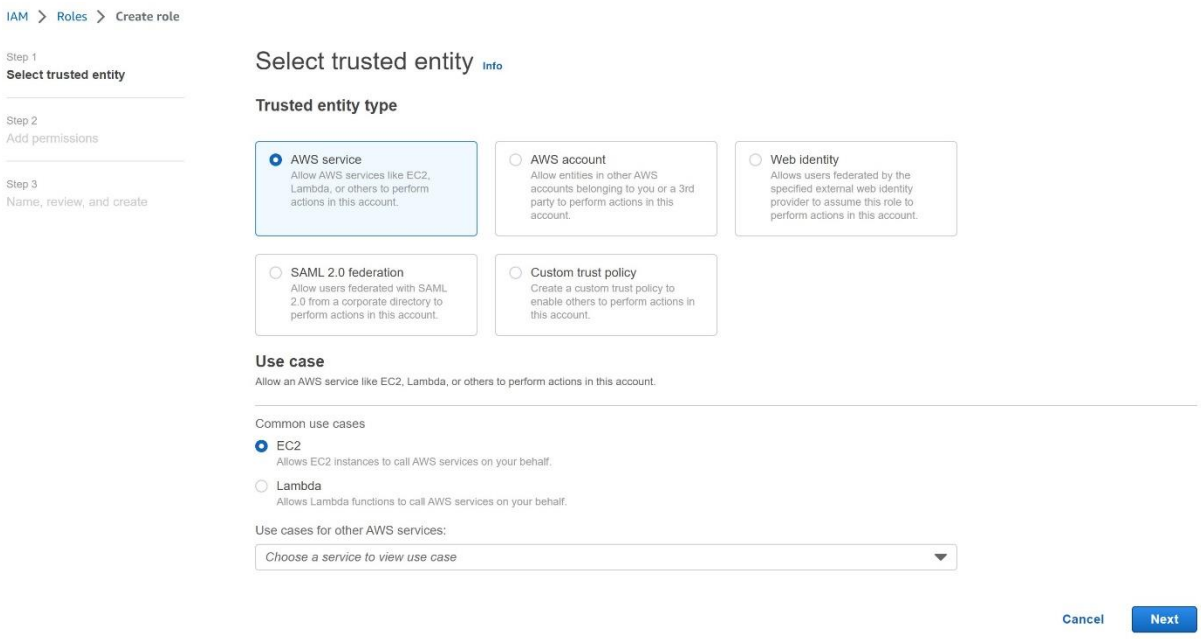
Click Create instance.



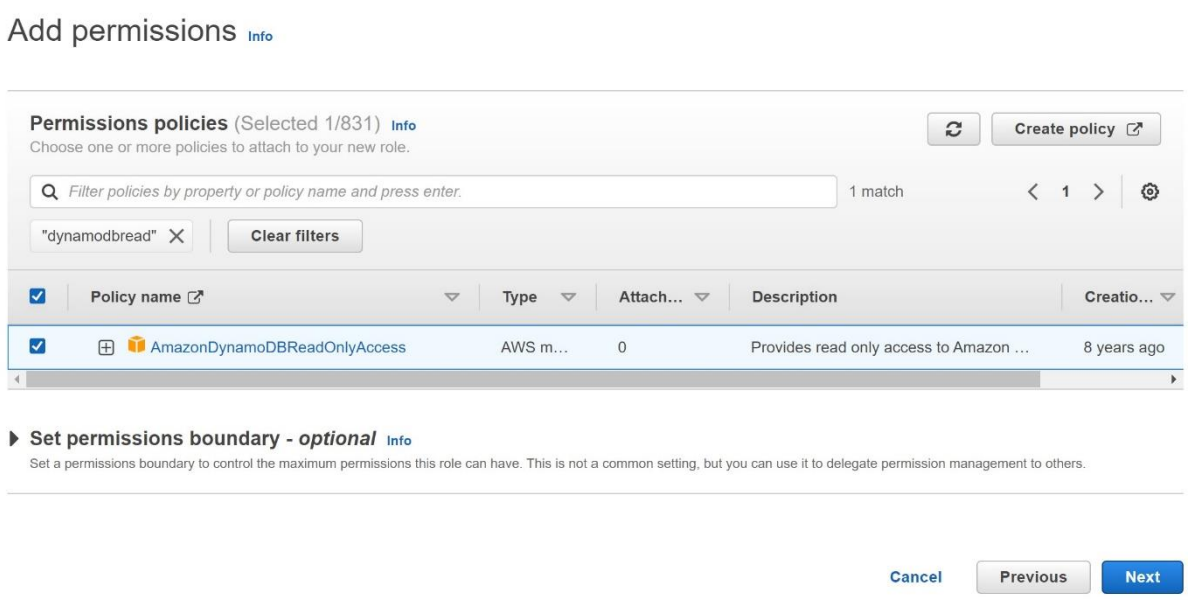
Step 4: Create IAM Role.

Using an IAM role and the principle of least privilege, grant the EC2 instance read access to DynamoDB.

Now, go to the Identity and Access Management dashboard to create a new IAM role. Click on Roles and Create a role. For trusted entity select AWS and for use case select EC2.



Click Next and Choose DynamoDBReadOnlyAccess on the Add Permissions screen and select that permission.

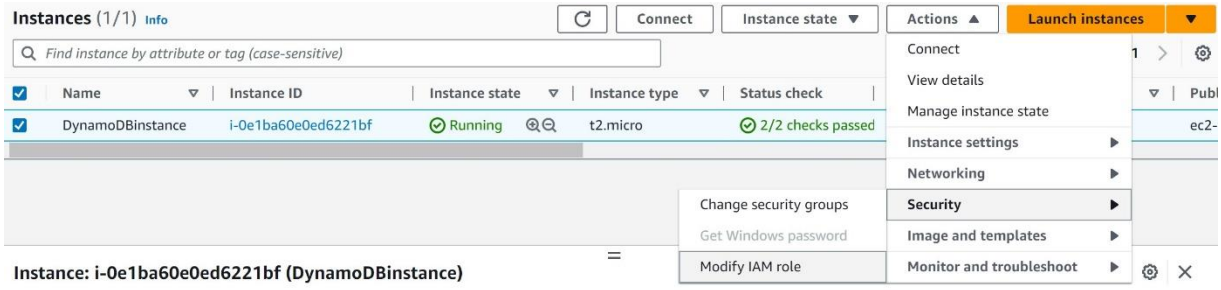


Click Next and Give the role a name – EC2-DynamoDB-Role and click Create role.

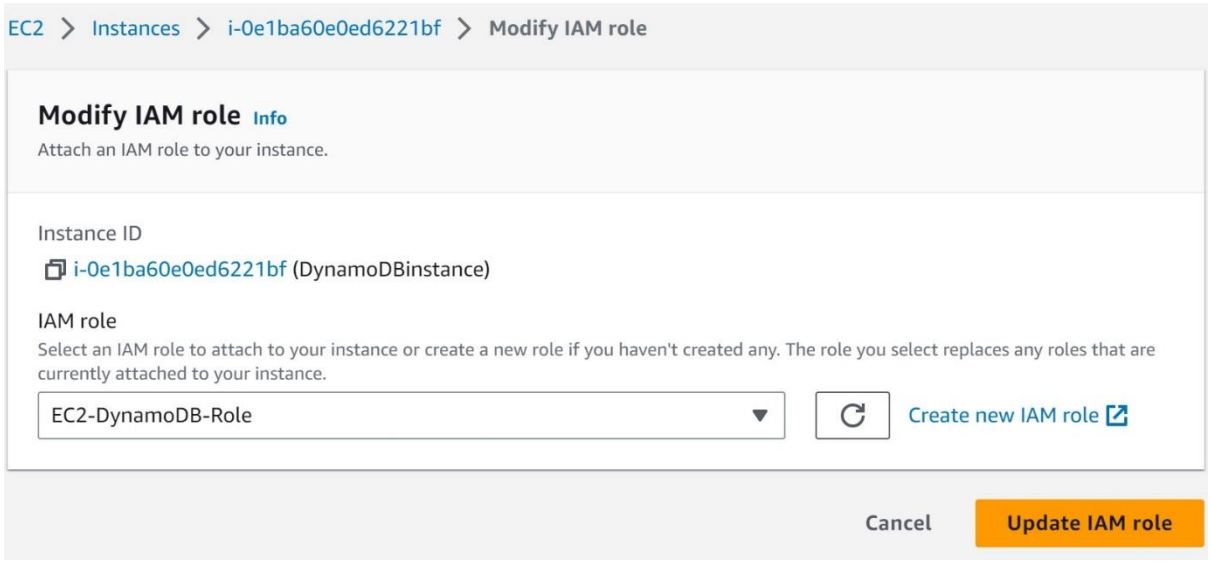
Role name	Trusted entities
AWSServiceRoleForAmazonElasticFileSystem	AWS Service: elasticfilesystem (Service-Linked Role)
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable	AWS Service: dynamodb.application-autoscaling (Service-Linked Role
AWSServiceRoleForBackup	AWS Service: backup (Service-Linked Role)
AWSServiceRoleForElasticLoadBalancing	AWS Service: elasticloadbalancing (Service-Linked Role)
AWSServiceRoleForGlobalAccelerator	AWS Service: globalaccelerator (Service-Linked Role)
AWSServiceRoleForRDS	AWS Service: rds (Service-Linked Role)
AWSServiceRoleForSupport	AWS Service: support (Service-Linked Role)
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linked Role)
EC2-DynamoDB-Role	AWS Service: ec2

Step 5: Grant the EC2 instance read access to DynamoDB.

Now, go to EC2 instance and select it. Then click on Actions > Security > Modify IAM Role.



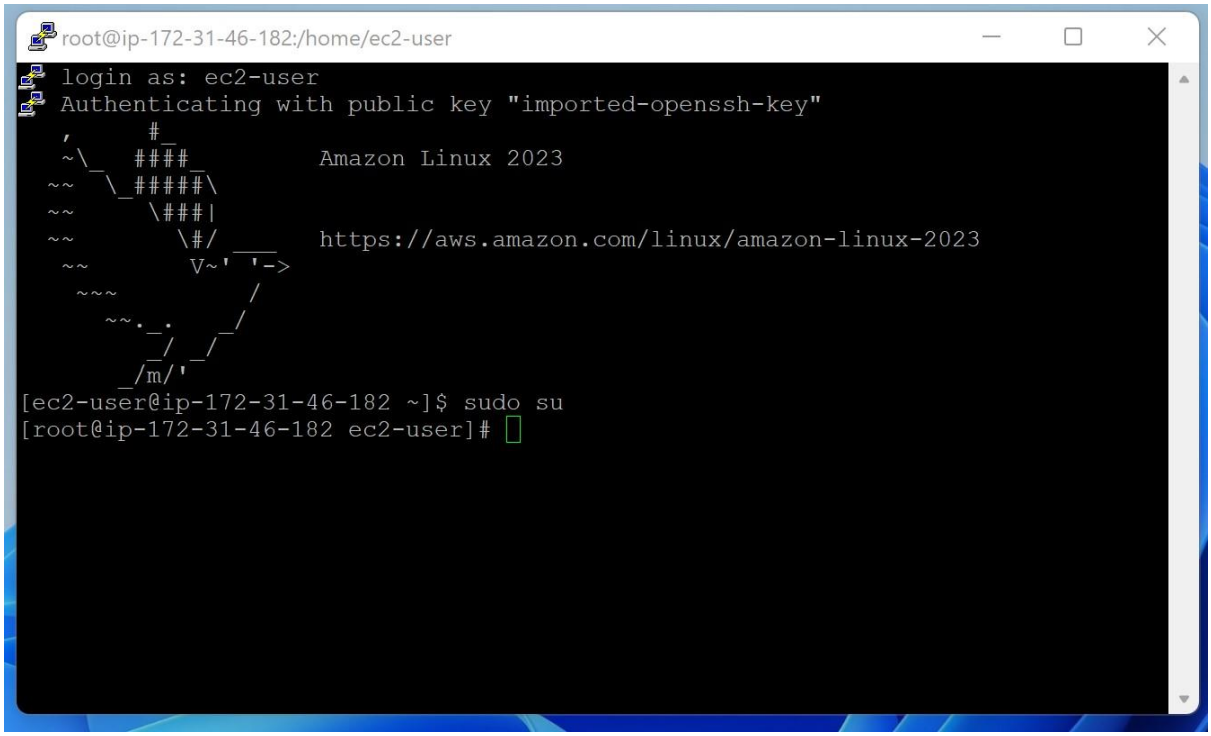
Select the DynamoDB read access role and click Update IAM role.



Step 6: Connect to a DynamoDB instance.

Now, connect to a DynamoDB instance using PuTTY. Download Puttygen and Putty.

Open Puttygen, select Load, Go to Download, Select All Files with mydynamodbkey.pem and then open and Save private Key on Desktop as mydynamodbkey.ppk. Open Putty, paste your Instance Public IP, click on +SSH, select AUTH, Through Browse open mynewdynamodbkey.ppk with login as: ec2-user and $ sudo su.



Once connected to the instance, enter the following command to scan the table created in DynamoDB:

```
aws dynamodb scan --table-name BollywoodMusic --region eu-west-2
```

The following was the output.

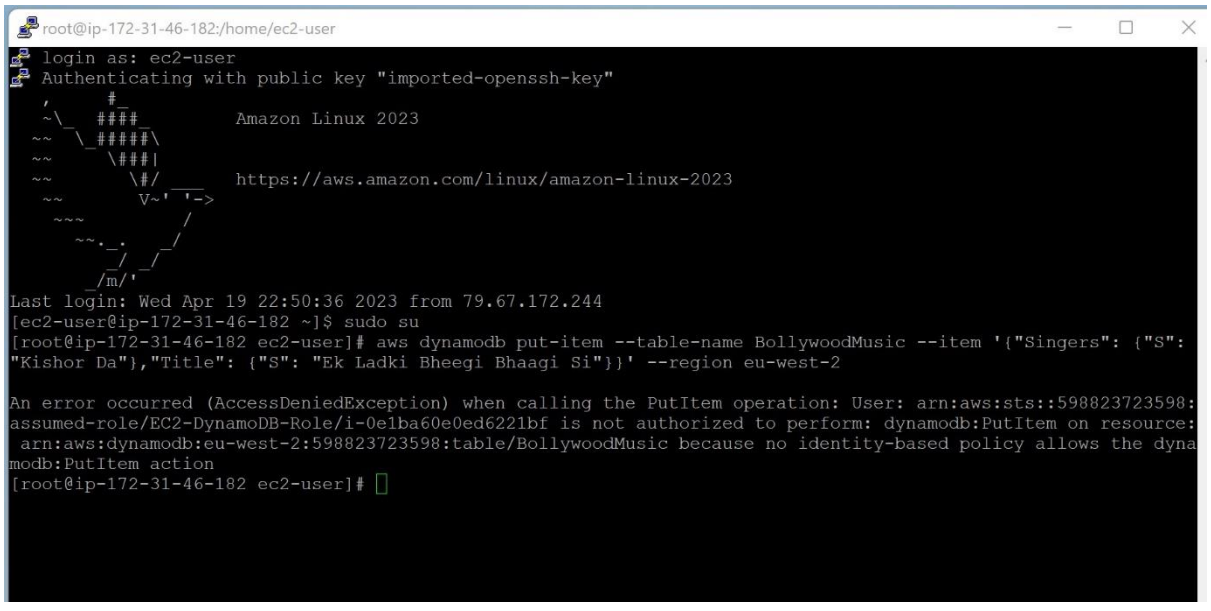
```
[ec2-user@ip-172-31-46-182 ~]$ sudo su
[root@ip-172-31-46-182 ec2-user]# aws dynamodb scan --table-name BollywoodMusic
--region eu-west-2
{
    "Items": [
        {
            "Singers": {
                "S": "Lata Mangeshkar"
            },
            "Songs": {
                "S": "Aye Mere Watan Ke Logo"
            }
        },
        {
            "Singers": {
                "S": "Arijit Singh"
            },
            "Songs": {
                "S": "Ae Watan"
            }
        },
```

Step 7: Try to validate read-only access to the DynamoDB table.

To test the read-only role set for EC2, use the following code to try to write a new item to the table:

```
aws dynamodb put-item --table-name BollywoodMusic --item '{"Singers": {"S": "Kishor Da"},"Title": {"S": "Ek
Ladki Bheegi Bhaagi Si"}}' --region eu-west-2
```

The error below shows that the IAM role was set up correctly.



Step 8: Clean up.

Delete the EC2 instance, IAM Role and DynamoDB table without on-demand backup.