

Create a DynamoDB table named "UserActivity" with a primary key "UserID" (partition key) and "ActivityTime" (sort key).

The screenshot shows the AWS Management Console interface for the 'UserActivity' table. The left sidebar contains navigation links for DynamoDB, DAX, Clusters, Subnet groups, Parameter groups, and Events. The main content area displays the 'UserActivity' table settings under the 'Settings' tab. The 'General information' section shows the Partition key as 'UserID (String)' and the Sort key as 'ActivityTime (String)'. The Capacity mode is set to 'On-demand'. The Table status is 'Active', and the Table size is '192 bytes'. The Item count is '9'. The Average item size is '21.33 bytes'. The Amazon Resource Name (ARN) is 'arn:aws:dynamodb:us-east-2:842676015714:table/UserActivity'. The 'Additional info' section is expanded, showing 'Alarms' as 'No active alarms', 'Point-in-time recovery (PITR)' as 'Off', and 'Resource-based policy' as 'Not active'.

Added 5 items in the database

The screenshot shows the AWS Management Console interface for the 'UserActivity' table, displaying the 'Explore items' view. The 'Scan or query items' section shows the 'Scan' button selected. The 'Select a table or index' dropdown is set to 'Table - UserActivity', and the 'Select attribute projection' dropdown is set to 'All attributes'. The 'Filters - optional' section is empty. The 'Run' button is highlighted. A green banner indicates 'Completed - Items returned: 9 - Items scanned: 9 - Efficiency: 100% - RCUs consumed: 2'. Below this, the 'Table: UserActivity - Items returned (9)' section shows a list of items. The items are displayed in a table with columns for 'UserID (String)', 'ActivityTime (String)', and 'duration'. The items are:

UserID (String)	ActivityTime (String)	duration
2	2	
8	8	
1	2	
6	6	
5	5	
4	4	
7	7	
3	3	
10	14	55

Query to get all the activities of specific user id.

The screenshot shows the AWS DynamoDB console interface. On the left, the 'DynamoDB' sidebar is visible with options like Dashboard, Tables, Explore items, PartiQL editor, Backups, Exports to S3, Imports from S3, Integrations, Reserved capacity, and Settings. The main area is titled 'UserActivity' and shows a 'Scan or query items' section. The 'Scan' option is selected. Below this, 'Table - UserActivity' is chosen, and 'Select attribute projection' is set to 'All attributes'. A filter is added for 'Attribute name' 'UserID' with the condition 'Equal to' and value '1'. The 'Run' button is clicked. A green status bar indicates 'Completed - Items returned: 1 - Items scanned: 9 - Efficiency: 11.11% - RCUs consumed: 2'. Below this, a table titled 'Table: UserActivity - Items returned (1)' shows the scan results. The table has two columns: 'UserID (String)' and 'ActivityTime (String)'. The first row shows '1' for UserID and '2' for ActivityTime.

UserID (String)	ActivityTime (String)
1	2

Code to insert and get data from dynamodb

The screenshot shows the AWS Lambda console 'Code source' page for a function named 'lambda\_function.py'. The code is written in Python and uses the boto3 library to interact with DynamoDB. It defines a lambda\_handler function that takes an event and context as input. The handler checks the 'action' in the event. If the action is 'insert', it extracts 'user\_id', 'activity\_time', 'activity\_type', and 'duration' from the event and inserts a new item into the 'UserActivity' table. If the action is 'query', it extracts 'user\_id' and queries the table for items with that user\_id. The code is as follows:

```
1 import json
2 import boto3
3 from boto3.dynamodb.conditions import Key
4 from datetime import datetime
5
6 dynamodb = boto3.resource('dynamodb')
7 table = dynamodb.Table('UserActivity')
8
9 def lambda_handler(event, context):
10     action = event.get('action')
11
12     if action == 'insert':
13         user_id = event['UserID']
14         activity_time = event.get('ActivityTime', datetime.utcnow().isoformat())
15         activity_type = event.get('ActivityType', 'unknown')
16         duration = event.get('Duration', 0)
17
18         response = table.put_item(
19             Item={
20                 'UserID': user_id,
21                 'ActivityTime': activity_time,
22                 'ActivityType': activity_type,
23                 'Duration': duration
24             }
25         )
26         return {
27             'statusCode': 200,
28             'body': json.dumps('Activity inserted successfully')
29         }
30
31     elif action == 'query':
32         user_id = event['UserID']
33         response = table.query(
34             KeyConditionExpression=Key('UserID').eq(user_id)
```

## Query Executed Successfully

aws

Search

[Option+S]

United States (Ohio)

gaurav @ buddha-aws-v3

Lambda > Functions > query-user-activity-dynamodb

Executing function: succeeded (logs)

Details

```
{
  "statusCode": 200,
  "body": "\"Activity inserted successfully\""
}
```

Summary

Code SHA-256

r1CtcbuQxHMKY7j6PE3sHESiyc8OSDzuJkz/JV9HI=

Function version

\$LATEST

Duration

318.99 ms

Resources configured

128 MB

Log output

The area below shows the last 4 KB of the execution log. [Click here](#) to view the corresponding CloudWatch log group.

Execution time

58 seconds ago

Request ID

6c2d5ed5-f2e4-44ff-8479-a1daffbf250e

Billed duration

319 ms

Max memory used

81 MB

START RequestId: 6c2d5ed5-f2e4-44ff-8479-a1daffbf250e Version: \$LATEST

END RequestId: 6c2d5ed5-f2e4-44ff-8479-a1daffbf250e

REPORT RequestId: 6c2d5ed5-f2e4-44ff-8479-a1daffbf250e Duration: 318.99 ms Billed Duration: 319 ms Memory Size: 128 MB Max Memory Used: 81 MB

## IAM Role for S3 to access dynamodb

aws

Search

[Option+S]

Global

gaurav @ buddha-aws-v3

IAM > Roles > query-user-activity-dynamodb-role-bd1ri8s2

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management

Users

Roles

Policies

Identity providers

Account settings

Root access management

Access reports

Access Analyzer

Archive rules

Analizers

Settings

Credential report

Organization activity

Service control policies

Resource control policies

Policy DynamoDBPutAccess created.

query-user-activity-dynamodb-role-bd1ri8s2

Summary

Creation date

April 14, 2025, 07:40 (UTC-05:00)

Last activity

-

ARN

arn:aws:iam::842676015714:role/service-role/query-user-activity-dynamodb-role-bd1ri8s2

Maximum session duration

1 hour

Permissions

Trust relationships

Tags

Last Accessed

Revoke sessions

Permissions policies (3)

You can attach up to 10 managed policies.

Filter by Type

Search

All types

Policy name	Type	Attached entities
AmazonDynamoDBFullAccess	AWS managed	1
AWSLambdaBasicExecutionRole-e8f5bb81-db15-4...	Customer managed	1
DynamoDBPutAccess	Customer inline	0

## Task 2

### EC2 Instance to connect to the Document DB

The screenshot displays the AWS Management Console interface for an EC2 instance. The left sidebar shows the navigation menu with categories like EC2, Images, Elastic Block Store, and Network & Security. The main content area shows the 'Instance summary for i-030111fb8a9a1790c (documentdbqueryec2)'. The instance is in a 'Running' state. Key details include: Public IPv4 address (3.22.250.196), Private IPv4 address (172.31.6.160), Public IPv4 DNS (ec2-3-22-250-196.us-east-2.compute.amazonaws.com), Private IP DNS name (ip-172-31-6-160.us-east-2.compute.internal), Instance type (t2.micro), VPC ID (vpc-04a8ca94e4da210c5), Subnet ID (subnet-0a3c5906cd2beab69), and Instance ARN (arn:aws:ec2:us-east-2:842676015714:instance/i-030111fb8a9a1790c). The instance is associated with the IAM Role 'documentdbqueryec2' and the IMDSv2 'Required' setting. The 'Managed' flag is set to 'false'. The bottom of the console shows tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags.

**Instance summary for i-030111fb8a9a1790c (documentdbqueryec2)**

Updated less than a minute ago

**Instance ID**  
i-030111fb8a9a1790c

**Public IPv4 address**  
3.22.250.196 | [open address](#)

**Private IPv4 addresses**  
172.31.6.160

**Instance state**  
Running

**Public IPv4 DNS**  
ec2-3-22-250-196.us-east-2.compute.amazonaws.com | [open address](#)

**Private IP DNS name (IPv4 only)**  
ip-172-31-6-160.us-east-2.compute.internal

**Instance type**  
t2.micro

**Hostname type**  
IP name: ip-172-31-6-160.us-east-2.compute.internal

**Answer private resource DNS name**  
IPv4 (A)

**VPC ID**  
vpc-04a8ca94e4da210c5

**Auto-assigned IP address**  
3.22.250.196 [Public IP]

**Subnet ID**  
subnet-0a3c5906cd2beab69

**IAM Role**  
-

**Instance ARN**  
arn:aws:ec2:us-east-2:842676015714:instance/i-030111fb8a9a1790c

**IMDSv2**  
Required

**Elastic IP addresses**  
-

**AWS Compute Optimizer finding**  
Opt-in to AWS Compute Optimizer for recommendation s. | [Learn more](#)

**Auto Scaling Group name**  
-

**Managed**  
false

**Operator**  
-

**Details** | Status and alarms | Monitoring | Security | Networking | Storage | Tags

### Document DB cluster

The screenshot displays the AWS Management Console interface for Amazon DocumentDB. The left sidebar shows the navigation menu with categories like Amazon DocumentDB, Clusters, Performance Insights, Snapshots, zero-ETL integrations, and Subnet groups. The main content area shows the 'Clusters (1)' section. The cluster is named 'docdb-2025-04-14-13-01-39' and is in an 'Available' state. Key details include: Role (Regional cluster), Engine version (5.0.0), Region & AZ (us-east-2), and Instance health (-). The cluster is associated with the IAM Role 'documentdbqueryec2' and the IMDSv2 'Required' setting. The 'Managed' flag is set to 'false'. The bottom of the console shows tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags.

**Amazon DocumentDB**

**Clusters (1)**

Filter Resources

Cluster identifier	Status	Role	Engine version	Region & AZ	Instance health
docdb-2025-04-14-13-01-39	Available	Regional cluster	5.0.0	us-east-2	-
docdb-2025-04-14-13-01-39	Creating	Primary instance	5.0.0	us-east-2b	-

Connected to the Document DB from EC2 instance and ran the command.

The screenshot shows the AWS IAM console with the **ec2documentDBRole** selected. A green notification bar at the top states "Policy was successfully attached to role." The role's summary shows it was created on April 14, 2025, and allows EC2 instances to call AWS services on their behalf. The **Permissions** tab is active, showing the **AmazonDocDBFullAccess** policy attached. An inset terminal window shows the following commands and output:

```
ec2-user@ip-172-31-6-160:~/aws-basic-apps/node-docdb$ node app.js
(node:27679) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version
(node:27679) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version
DB connected
ec2-user@ip-172-31-6-160:~/aws-basic-apps/node-docdb$ vim app.js
ec2-user@ip-172-31-6-160:~/aws-basic-apps/node-docdb$ node app.js
(node:27818) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version
(node:27818) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version
DB connected
```

### Task 3:

### Created Elastic cache cluster

The screenshot shows the AWS ElastiCache console. A notification bar at the top mentions "Amazon ElastiCache announces Valkey starting as low as \$6/month". The **Redis OSS caches (1)** section is active, showing a table with one cache cluster:

Cache name	Status	Description	Engine version	Configuration	Created
elasticache-cloud-course	Available	-	7.1.0	cache.t2.micro	April 14,

### EC2 Instance to connect to Elastic cache

The screenshot shows the AWS EC2 console with the **Instances (1)** section active. A table lists the instance:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
elasticachecl...	i-030111fb8a9a1790c	Running	t2.micro	2/2 checks passed	View alarms +	us-east-2a	ec2-3-22-

Connected to the the cluster and did some insert and query

```
elasticache-cloud-course.psc8m8.ng.0001.use2.cache.amazonaws.com:6379> set user 123
OK
elasticache-cloud-course.psc8m8.ng.0001.use2.cache.amazonaws.com:6379> get user
"123"
elasticache-cloud-course.psc8m8.ng.0001.use2.cache.amazonaws.com:6379> █
```