Cycle 08 AWS Homework

Task 1: Create and Configure a DynamoDB Table

Objective:

Gain understanding of table creation and configuration choices in DynamoDB.

Steps Explained:

1. **Log into AWS Management Console**

   Access the AWS Console using your credentials to start using AWS services.

2. **Navigate to DynamoDB Service**

   Locate and open the DynamoDB service for database management.

3. **Create a New Table**

   Use the DynamoDB interface to create a fresh table for storing data.

4. **Experiment with Keys**

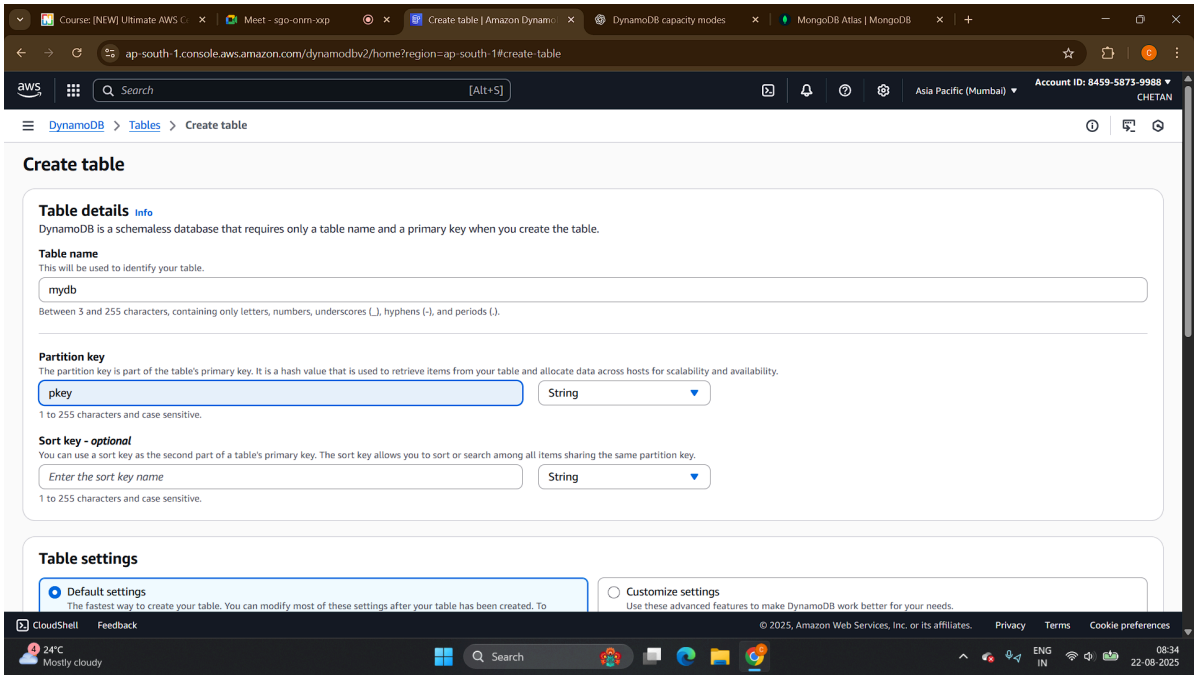
   - **Sub-task A:** Create a table with a simple primary key, e.g., `UserID`. This means the table's partition key is `UserID`, which uniquely identifies each item.
   - **Sub-task B:** Create another table with a composite primary key, e.g., `CustomerID` as the partition key and `OrderID` as the sort key. The composite key helps to group related data and enables sorting within each partition.

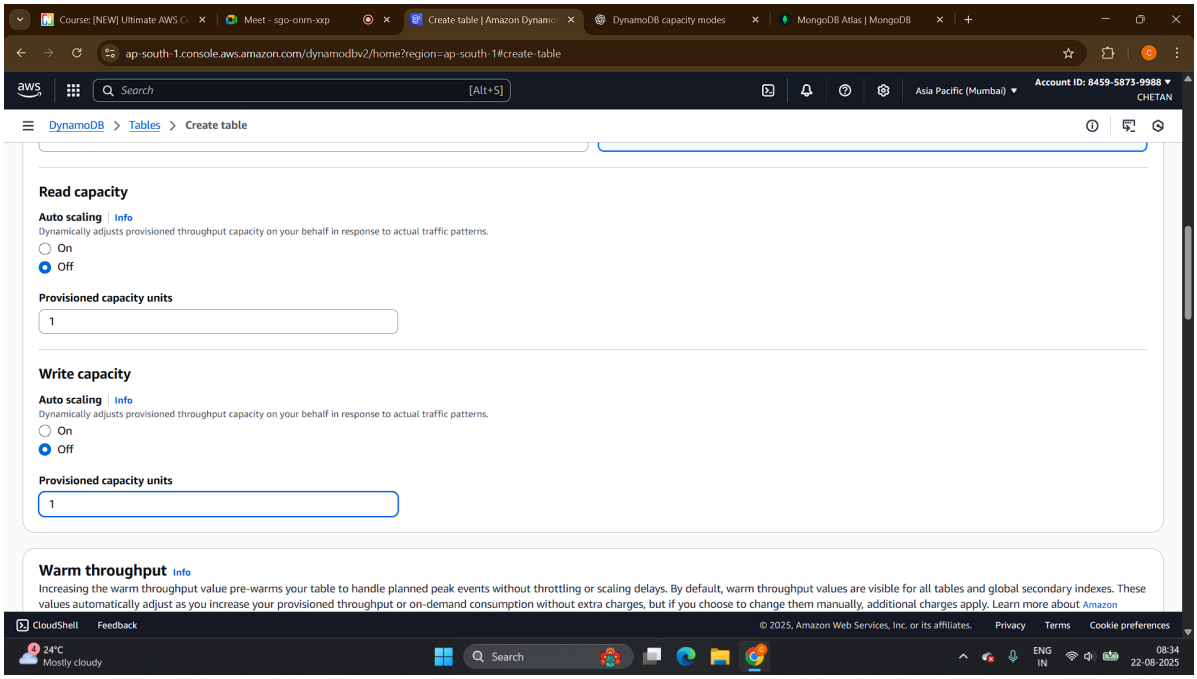
5. **Experiment with Capacity Modes**

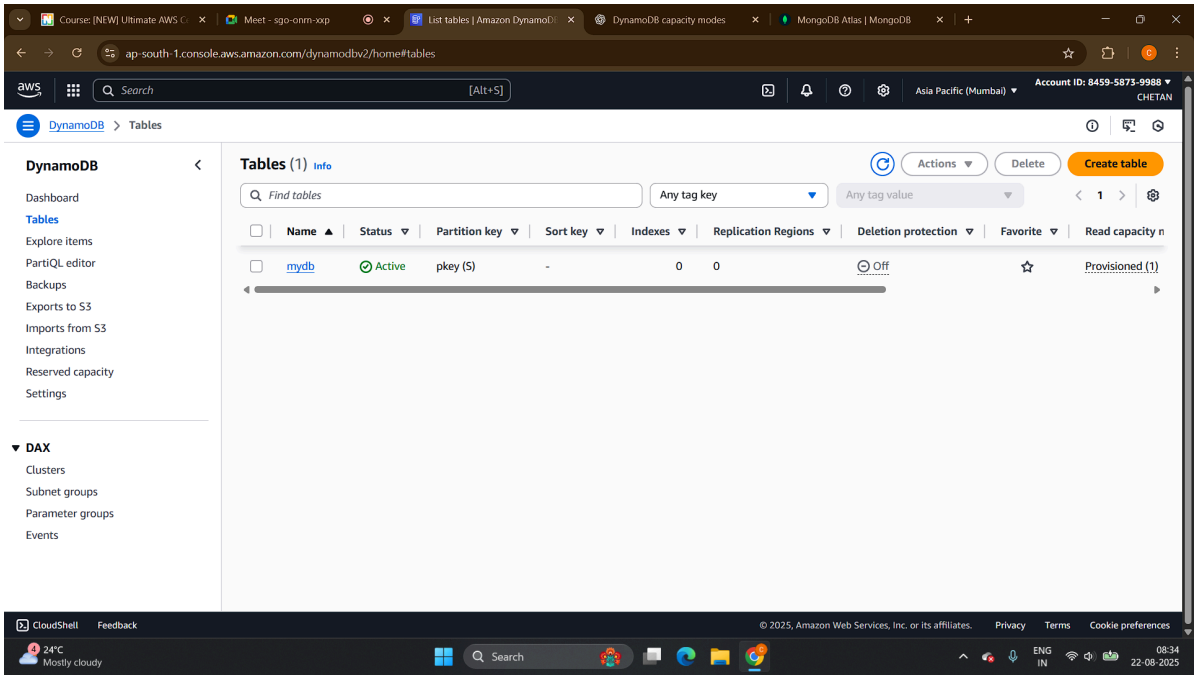
   - For one table, select **Provisioned Capacity** mode. Manually set Read Capacity Units (RCU) and Write Capacity Units (WCU) to a low count such as 5 each. This mode requires predicting usage and managing capacity.
   - For the other table, choose **On-Demand** capacity, where throughput is managed automatically based on requests, useful for unpredictable workloads.

6. **Review Additional Settings**

   Explore options like encryption (to secure data at rest), tags (for resource identification and billing), and other configurations available to customize the table according to needs.







Task 2: Perform Data Operations (CRUD)

Objective:

Hands-on practice with creating, reading, updating, and deleting items to understand DynamoDB's flexible schema.

Steps Explained:

1. **Navigate to "Explore Items"**

   This section in the DynamoDB console allows operating on data inside the table.

2. **Create Items**

   - Using **Form view**: Add items attribute by attribute interactively.
   - Using **JSON view**: Paste a full JSON document for each item, e.g.:

```
json{
 "UserID": "user_12345",
 "Name": "Jane Doe",
 "Email": "jane@example.com",
 "Address": {
   "Street": "123 Main St",
```

```
  "City": "Anytown"
 },
 "LastLogin": 1718214120
}
```

3. **Prove Schema Flexibility**

   Add an item with completely different attributes, for example only having `UserID` and `DeviceSerialNumber`. DynamoDB does not require a fixed schema other than key attributes.

4. **Read Items**

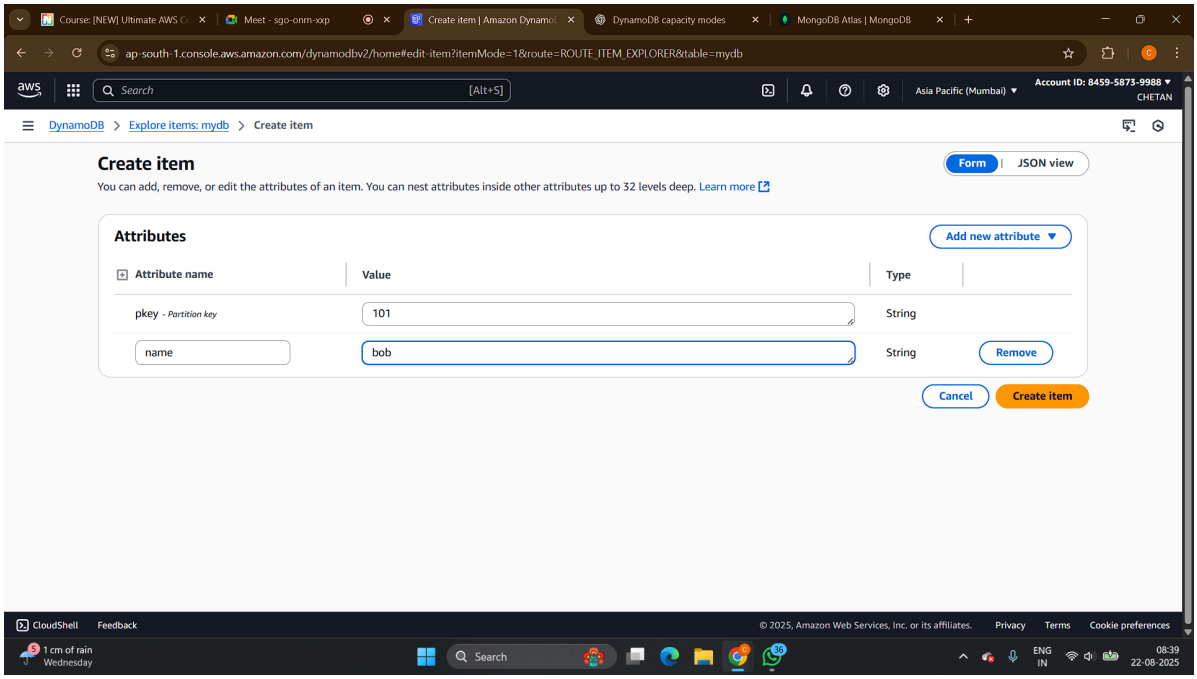
   - Use **Get item** to retrieve items by primary key.
   - Use **Scan** to read the entire table. Note that get-item is efficient for exact keys, while scan reads all data.

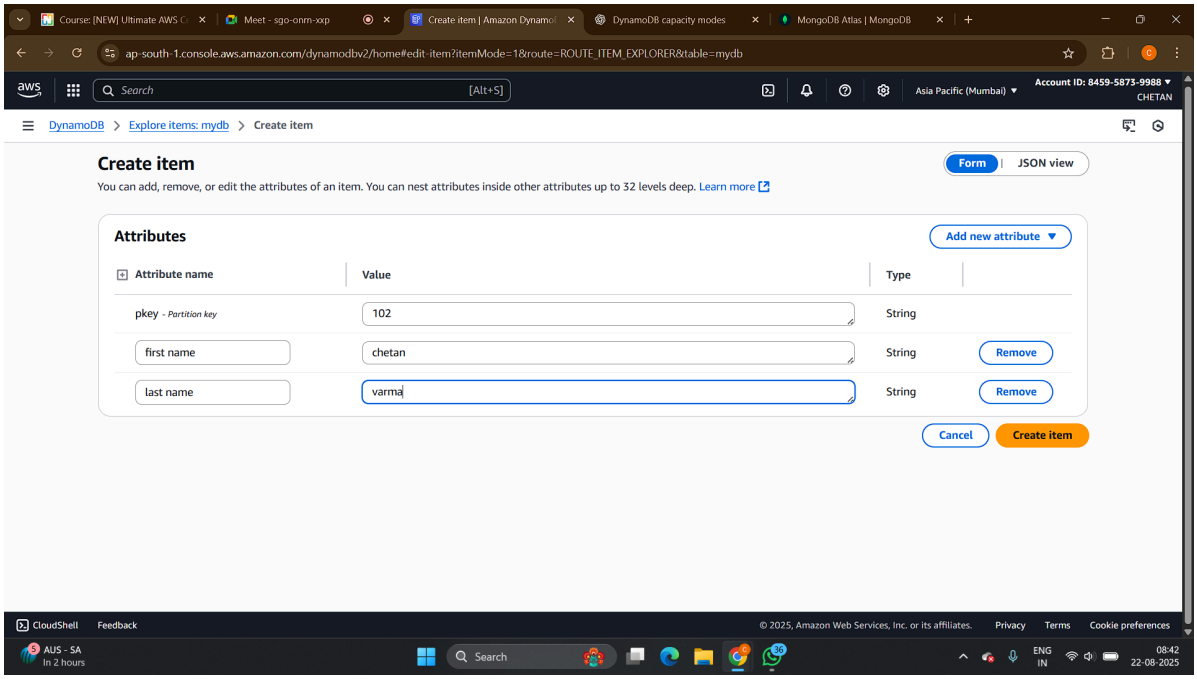
5. **Update an Item**

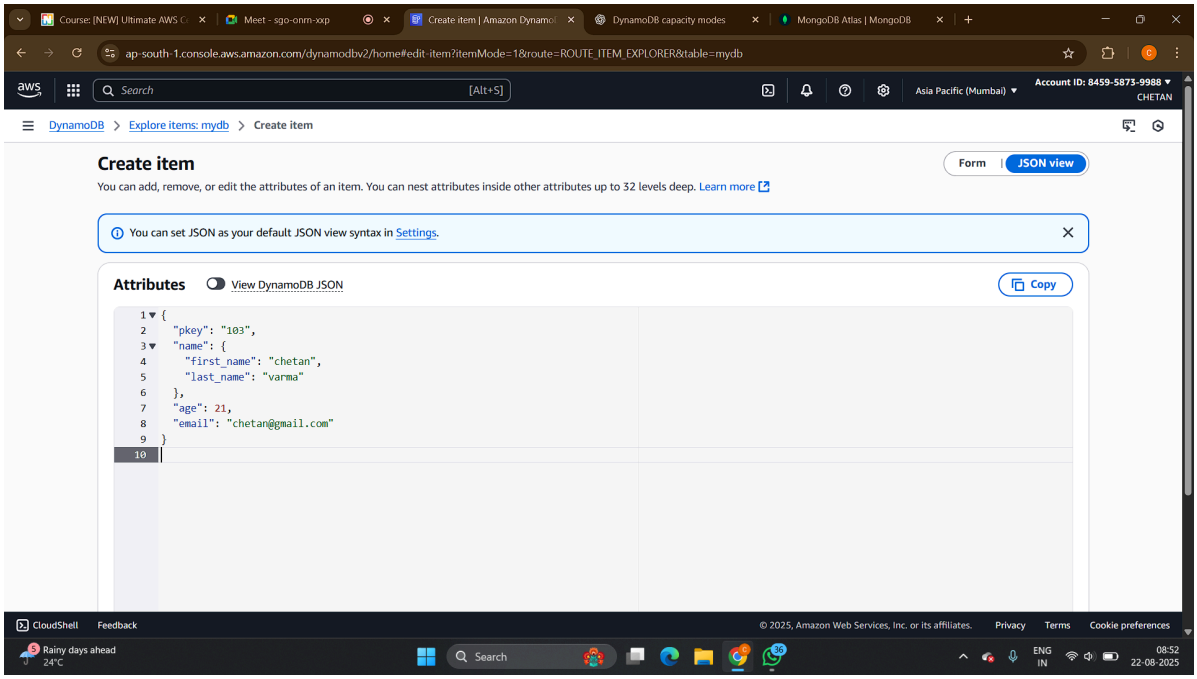
   Modify an existing item by adding a new attribute, e.g., `"AccountStatus": "Premium"`. This demonstrates that items can evolve flexibly over time.

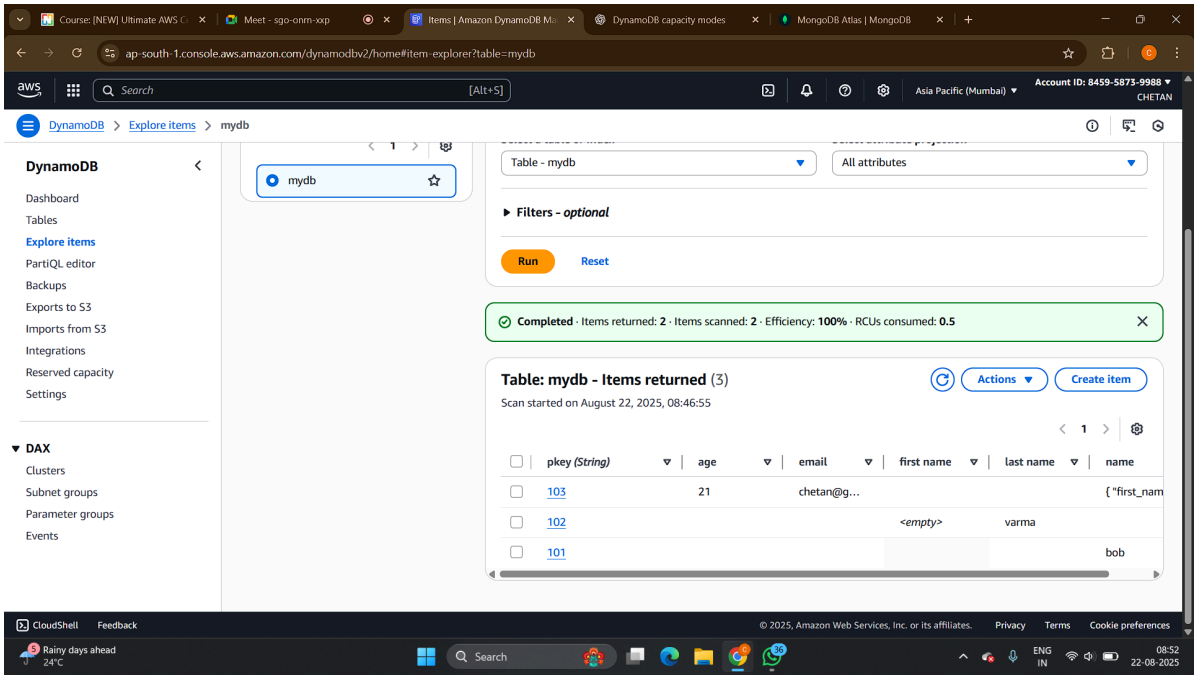
6. **Delete an Item**

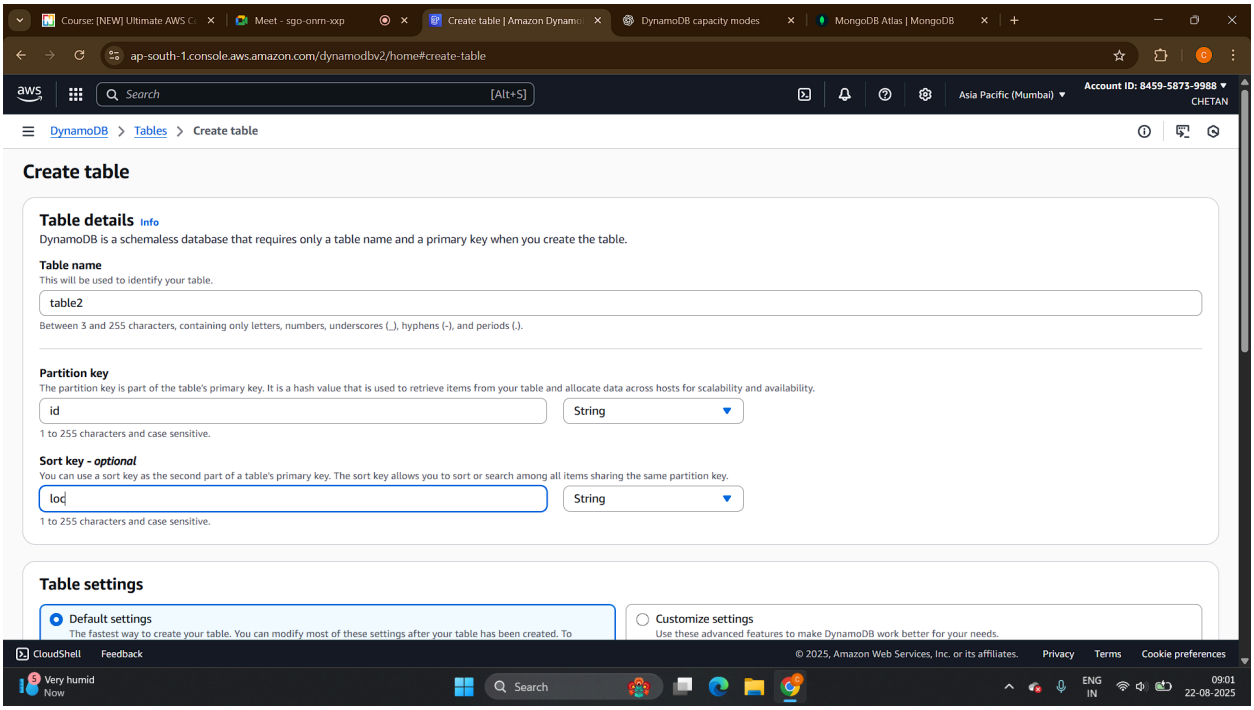
   Remove an item from the table to practice data deletion and cleanup.

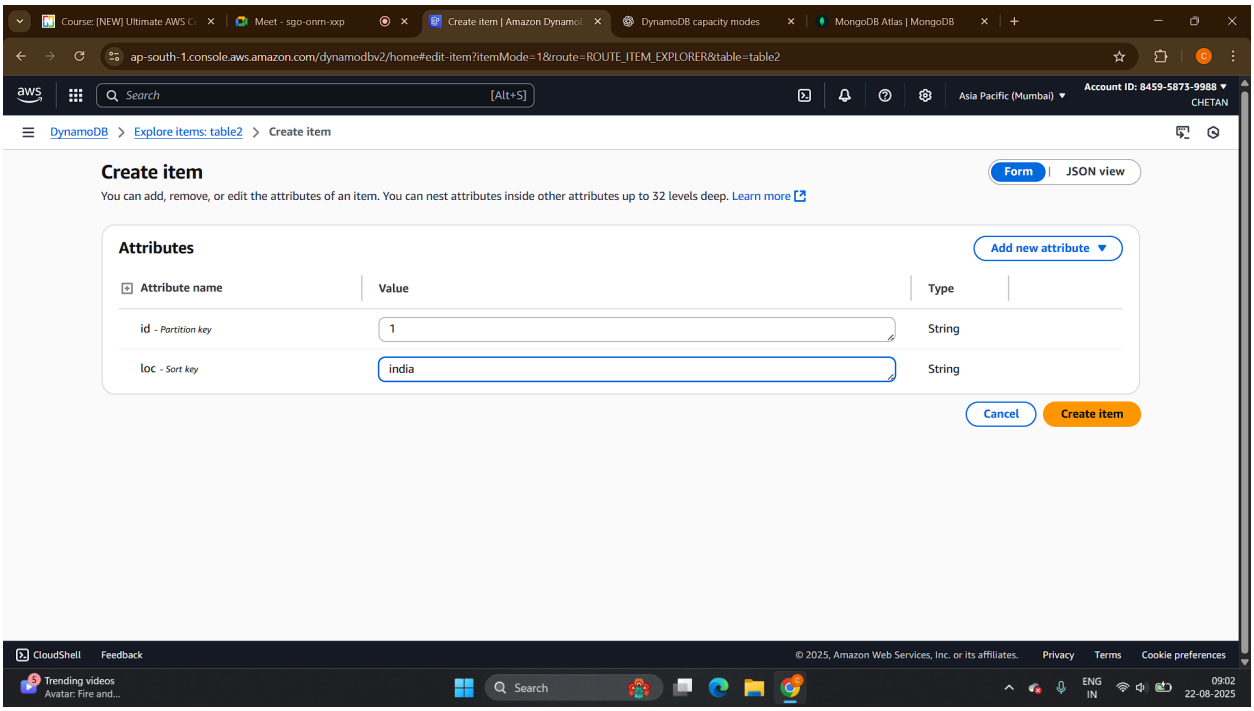


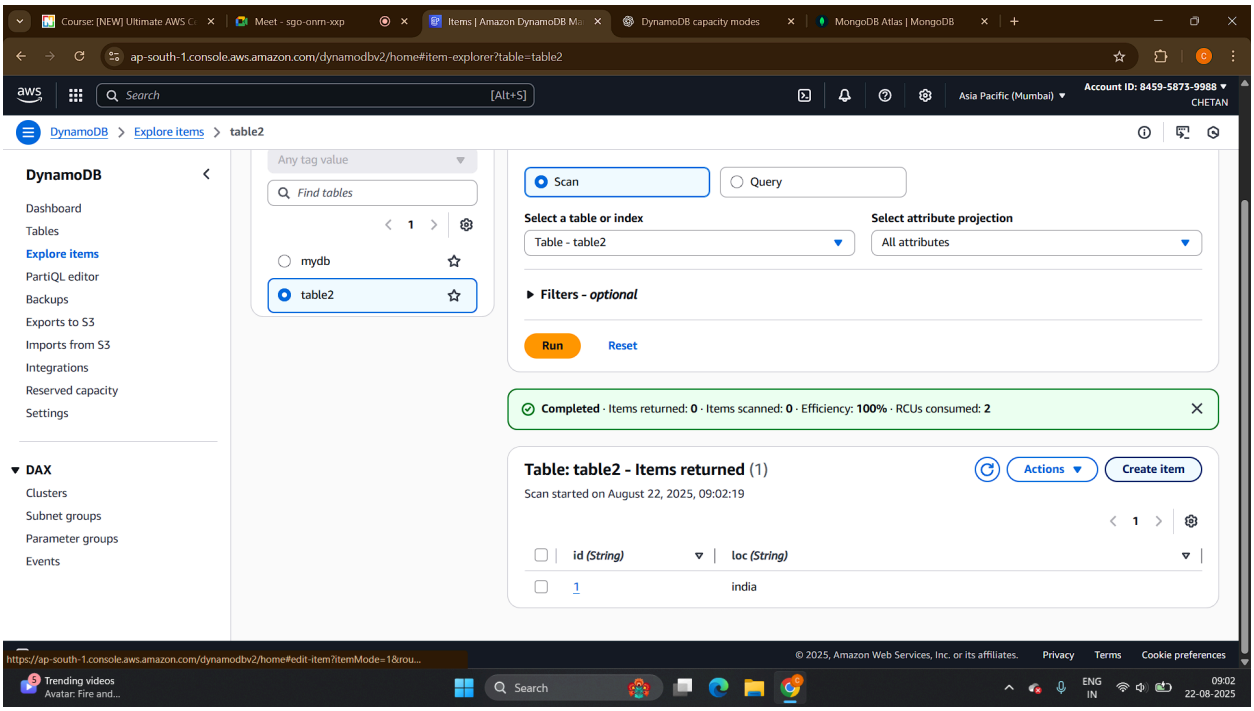

Create item | Amazon DynamoDB
ap-south-1.console.aws.amazon.com/dynamodbv2/home#edit-item?itemMode=1&route=ROUTE_ITEM_EXPLORER&table=mydb
Search [Alt+S]
Asia Pacific (Mumbai)
Account ID: 8459-5873-9988
CHETAN
DynamoDB > Explore items: mydb > Create item
Create item
Form | JSON view
You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. Learn more
Attributes
Add new attribute
Attribute name
Value
Type
pkey - Partition key
102
String
first name
chetan
String
Remove
last name
varma
String
Remove
Cancel
Create item
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
AUS - SA
In 2 hours
Search
ENG
IN
08:42
22-08-2025

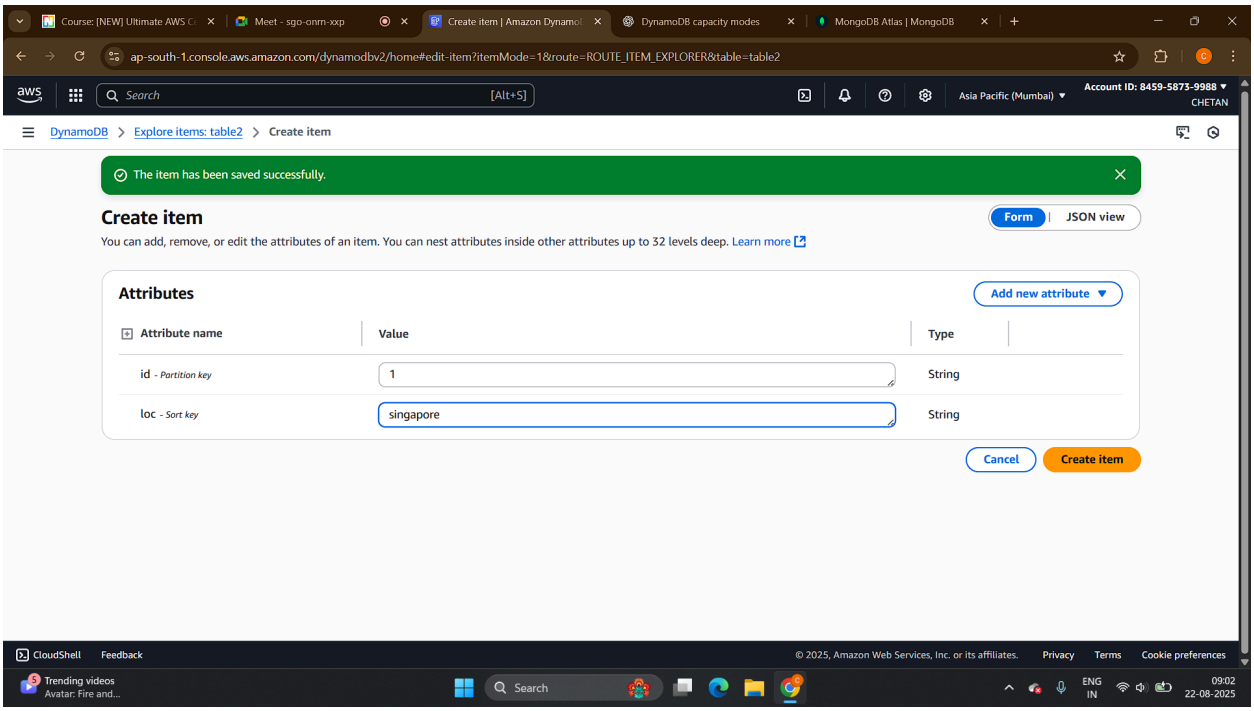

Create item | Amazon DynamoDB
ap-south-1.console.aws.amazon.com/dynamodbv2/home#edit-item?itemMode=1&route=ROUTE_ITEM_EXPLORER&table=mydb
Search [Alt+S]
Asia Pacific (Mumbai)
Account ID: 8459-5873-9988
CHETAN
DynamoDB > Explore items: mydb > Create item
Create item
Form | JSON view
You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. Learn more
You can set JSON as your default JSON view syntax in Settings.
Attributes
View DynamoDB JSON
Copy
{
"pkey": "103",
"name": {
"first_name": "chetan",
"last_name": "varma"
},
"age": 21,
"email": "chetan@gmail.com"
}
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Rainy days ahead
24°C
Search
ENG
IN
08:52
22-08-2025

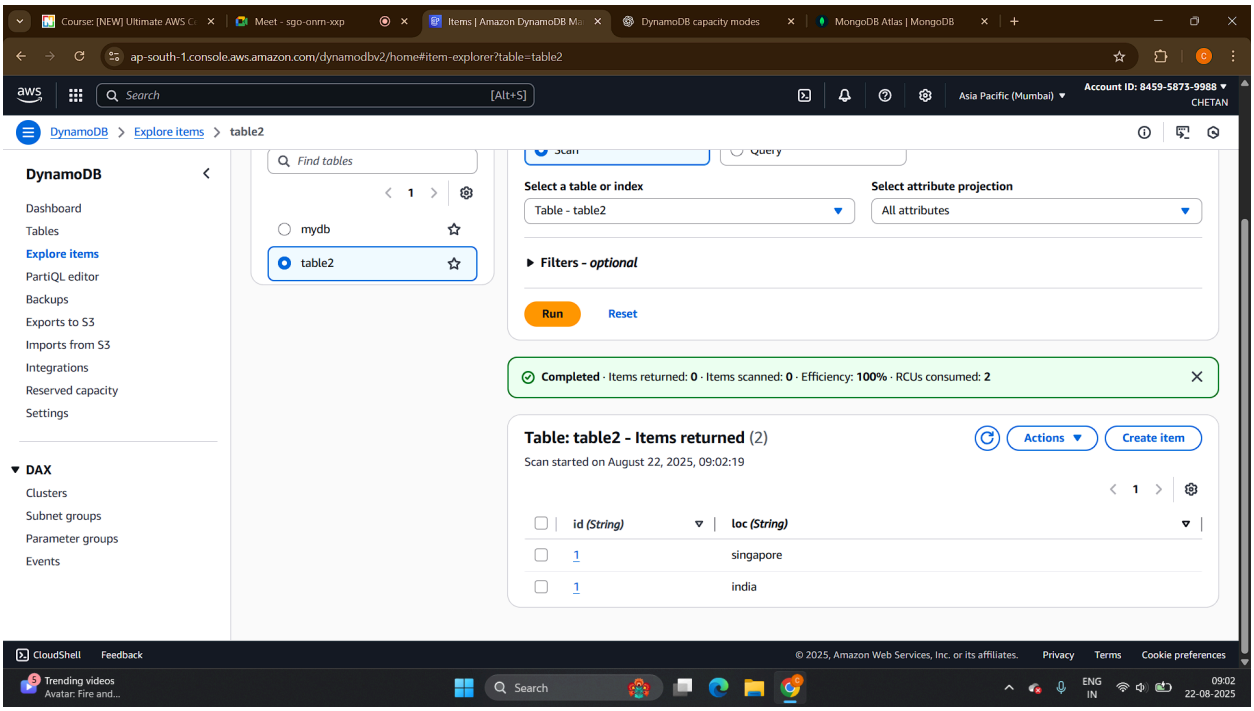

DynamoDB
Dashboard
Tables
Explore items
PartiQL editor
Backups
Exports to S3
Imports from S3
Integrations
Reserved capacity
Settings
DAX
Clusters
Subnet groups
Parameter groups
Events
mydb
Table - mydb
All attributes
Filters - optional
Run
Reset
Completed · Items returned: 2 · Items scanned: 2 · Efficiency: 100% · RCUs consumed: 0.5
Table: mydb - Items returned (3)
Scan started on August 22, 2025, 08:46:55
Actions
Create item
pkey (String)
age
email
first name
last name
name
103
21
chetan@g...
102
<empty>
varma
101
bob

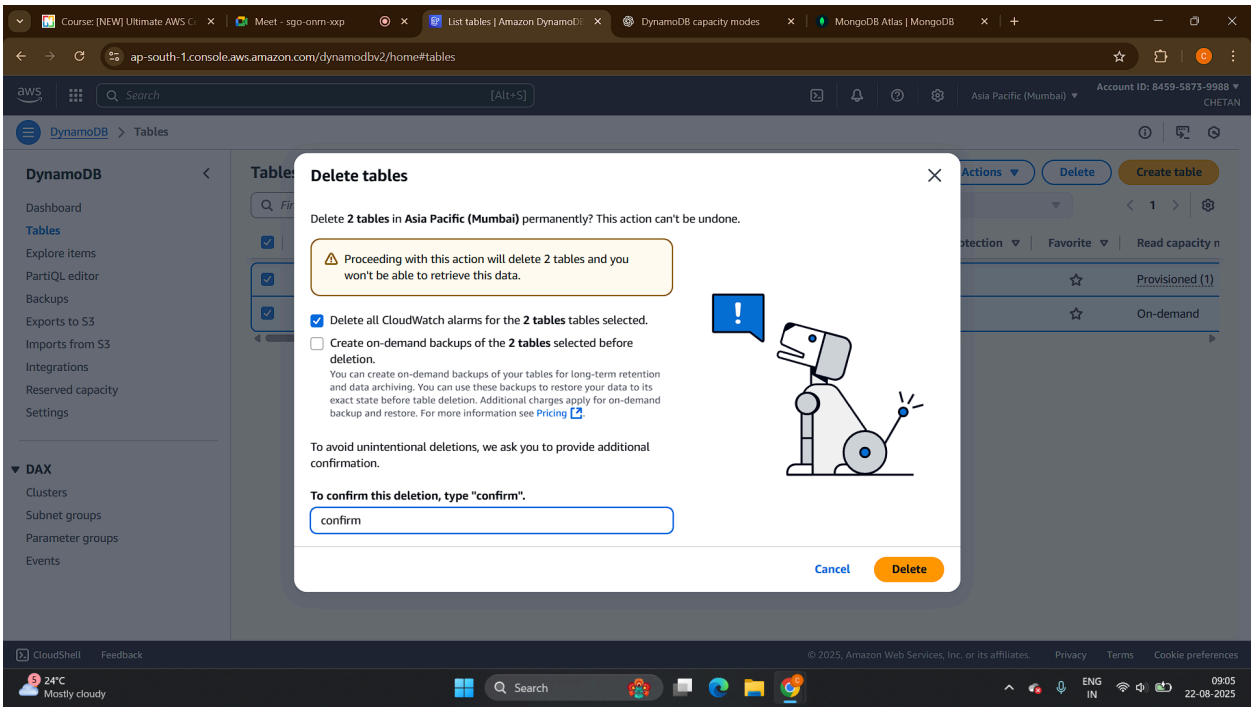

Create table
Table details Info
DynamoDB is a schemaless database that requires only a table name and a primary key when you create the table.
Table name
This will be used to identify your table.
table2
Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.).
Partition key
The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability and availability.
id
String
1 to 255 characters and case sensitive.
Sort key - optional
You can use a sort key as the second part of a table's primary key. The sort key allows you to sort or search among all items sharing the same partition key.
loc
String
1 to 255 characters and case sensitive.
Table settings
Default settings
Customize settings


Create item | Amazon DynamoDB
ap-south-1.console.aws.amazon.com/dynamodbv2/home#edit-item?itemMode=1&route=ROUTE_ITEM_EXPLORER&table=table2
Search [Alt+S]
Asia Pacific (Mumbai)
Account ID: 8459-5873-9988
CHETAN
DynamoDB > Explore items: table2 > Create item
Create item
Form | JSON view
You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. Learn more
Attributes
Add new attribute
Attribute name
Value
Type
id - Partition key
1
String
loc - Sort key
india
String
Cancel
Create item
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Items | Amazon DynamoDB
ap-south-1.console.aws.amazon.com/dynamodbv2/home#item-explorer?table=table2
DynamoDB > Explore items > table2
DynamoDB
Dashboard
Tables
Explore items
PartiQL editor
Backups
Exports to S3
Imports from S3
Integrations
Reserved capacity
Settings
DAX
Clusters
Subnet groups
Parameter groups
Events
Any tag value
Find tables
mydb
table2
Scan
Query
Select a table or index
Table - table2
Select attribute projection
All attributes
Filters - optional
Run
Reset
Completed · Items returned: 0 · Items scanned: 0 · Efficiency: 100% · RCUs consumed: 2
Table: table2 - Items returned (1)
Scan started on August 22, 2025, 09:02:19
Actions
Create item
id (String)
loc (String)
1
india
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Create item | Amazon DynamoDB
ap-south-1.console.aws.amazon.com/dynamodbv2/home#edit-item?itemMode=1&route=ROUTE_ITEM_EXPLORER&table=table2
Search [Alt+S]
Asia Pacific (Mumbai)
Account ID: 8459-5873-9988
CHETAN
DynamoDB > Explore items: table2 > Create item
The item has been saved successfully.
Create item
Form | JSON view
You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. Learn more
Attributes
Add new attribute
Attribute name
Value
Type
id - Partition key
1
String
loc - Sort key
singapore
String
Cancel
Create item
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Items | Amazon DynamoDB
ap-south-1.console.aws.amazon.com/dynamodbv2/home#item-explorer?table=table2
DynamoDB > Explore items > table2
DynamoDB
Dashboard
Tables
Explore items
PartiQL editor
Backups
Exports to S3
Imports from S3
Integrations
Reserved capacity
Settings
DAX
Clusters
Subnet groups
Parameter groups
Events
Find tables
mydb
table2
Select a table or index
Table - table2
Select attribute projection
All attributes
Filters - optional
Run
Reset
Completed · Items returned: 0 · Items scanned: 0 · Efficiency: 100% · RCUs consumed: 2
Table: table2 - Items returned (2)
Scan started on August 22, 2025, 09:02:19
Actions
Create item
id (String)
loc (String)
1
singapore
1
india



Task 3: Design a Data Model for a Real-World Scenario

Objective:

Apply primary key design strategies to model a forum application to efficiently store posts, comments, and user profiles.

Steps Explained:

1. **Design the Table(s)**

   Decide if all data will be in one table or multiple tables.

2. **Define the Primary Key**

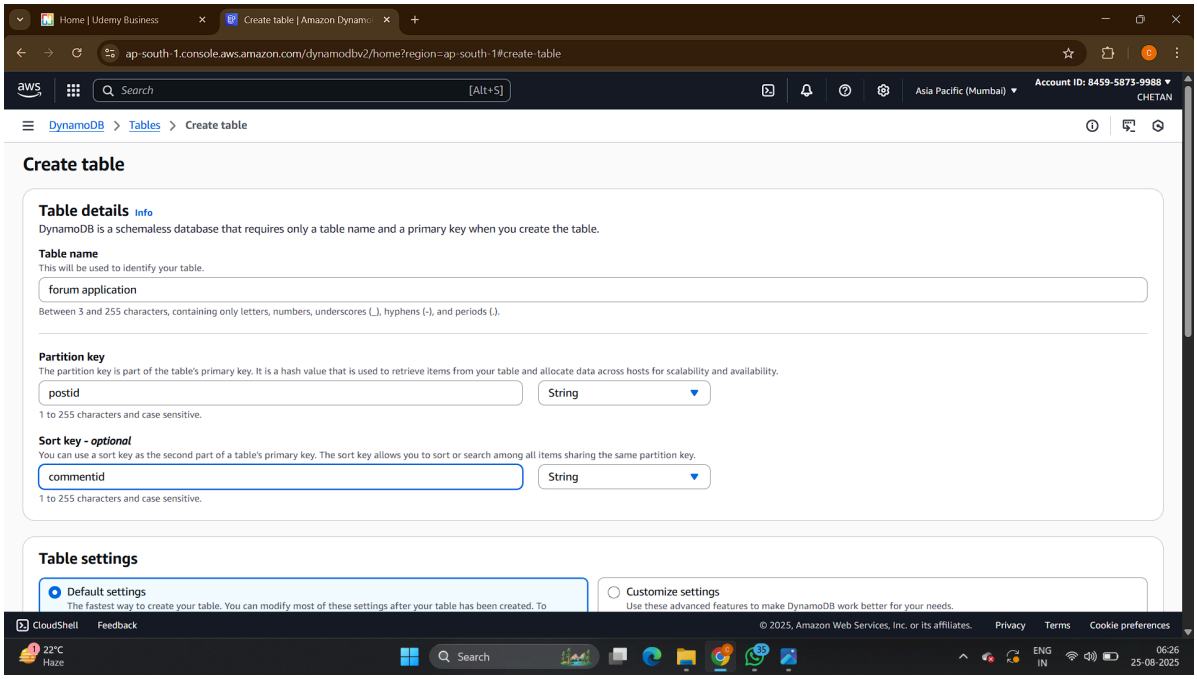
   - If using **one table** for posts and comments:
     - Partition Key: `POST#<PostID>` (to group all related items)
     - Sort Key: `#METADATA#<PostID>` for post data, and `COMMENT#<CommentID>` for comments (to enable sorting and querying by comment IDs).
   - If using **multiple tables**:

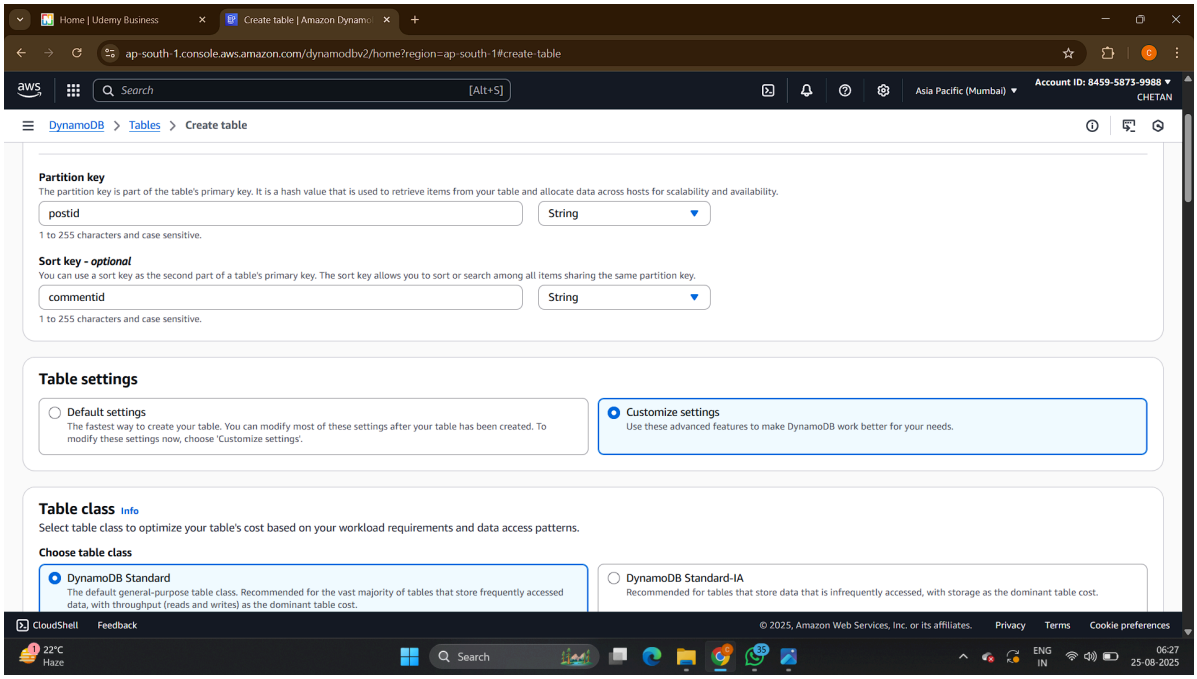
- Posts table with simple key: `PostID`
- Comments table with composite keys: partition key `PostID` (groups comments by post), sort key `CommentID` (identifies individual comments).

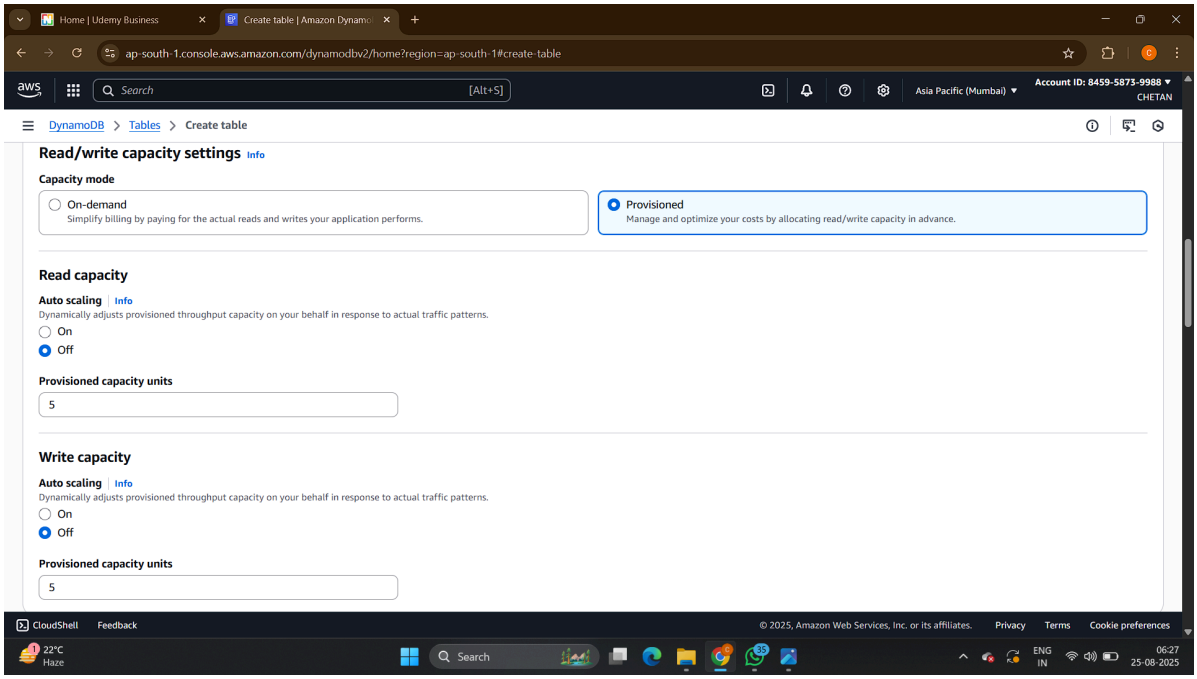
3. **Justify Design**

   Explain how the key design allows efficient queries, for example:

   - Querying all comments for a post by using the partition key `PostID`.
   - Sorting comments by comment ID using the sort key.







Task 4: Experiment with Consistency and Performance

Objective:

Understand and observe the differences between eventual and strong consistency in DynamoDB reads.

Steps Explained:

1. **Write a New Item**

   Insert a new data item into the DynamoDB table.

2. **Perform Get Item with Eventually Consistent Read**

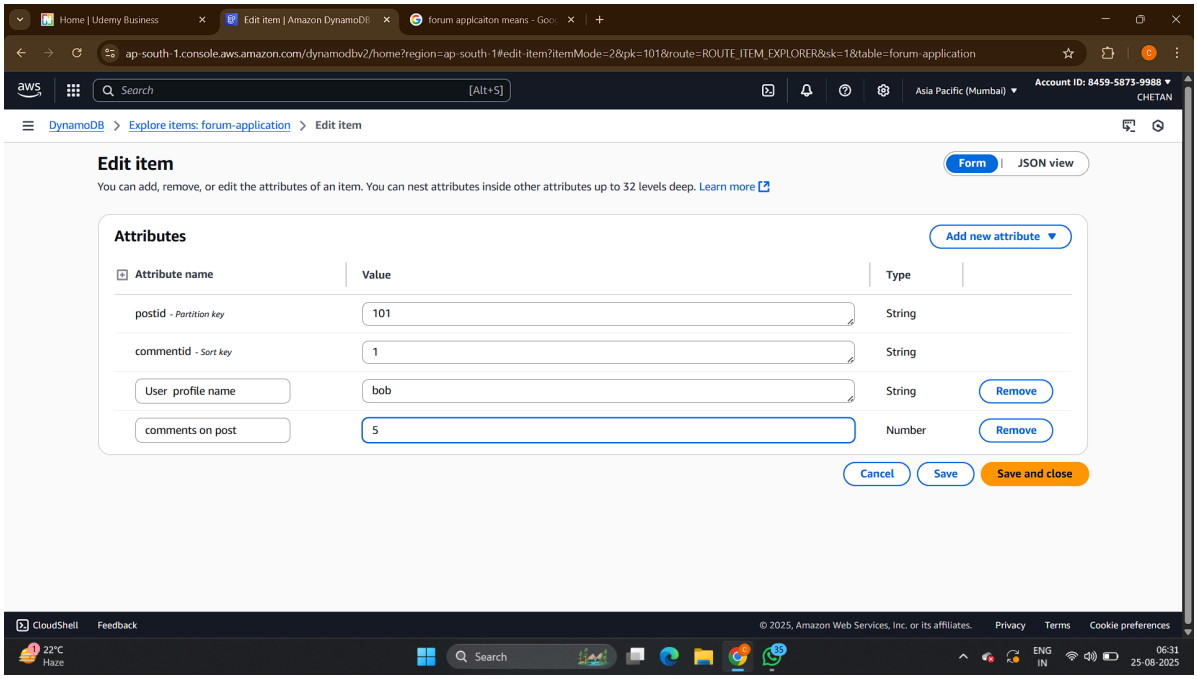
   By default, read operations in DynamoDB are eventually consistent. Immediately read the new item and note if the item is returned (it might not if the data is still propagating).

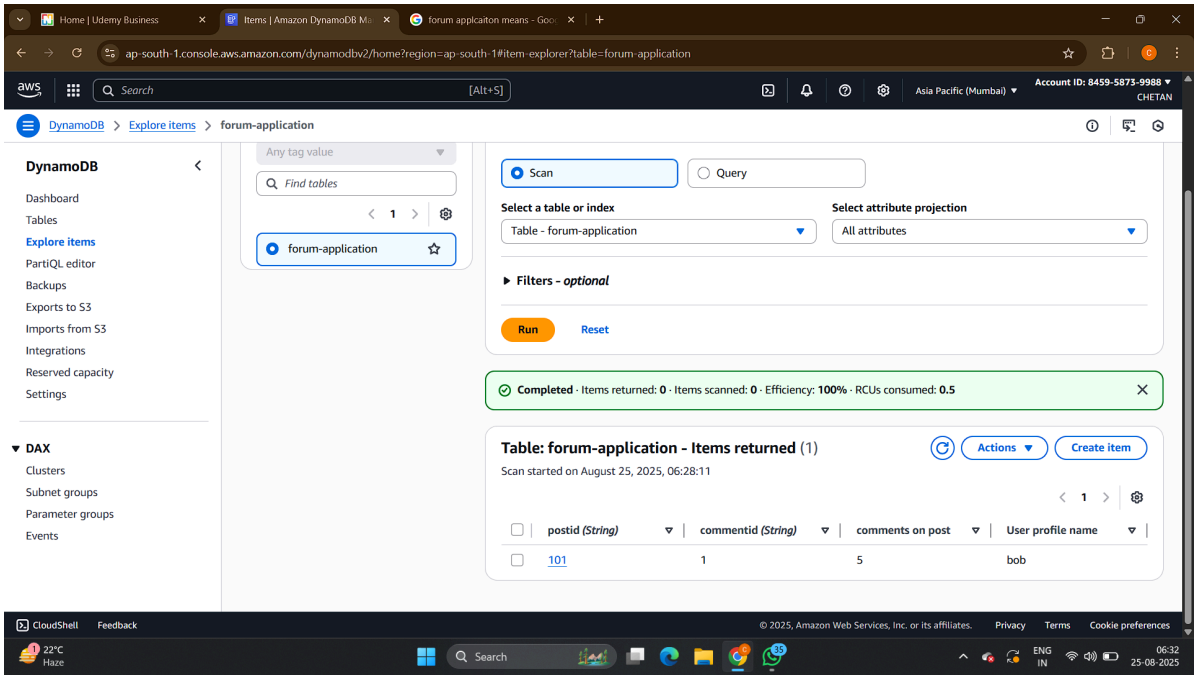
3. **Perform Get Item with Strongly Consistent Read**

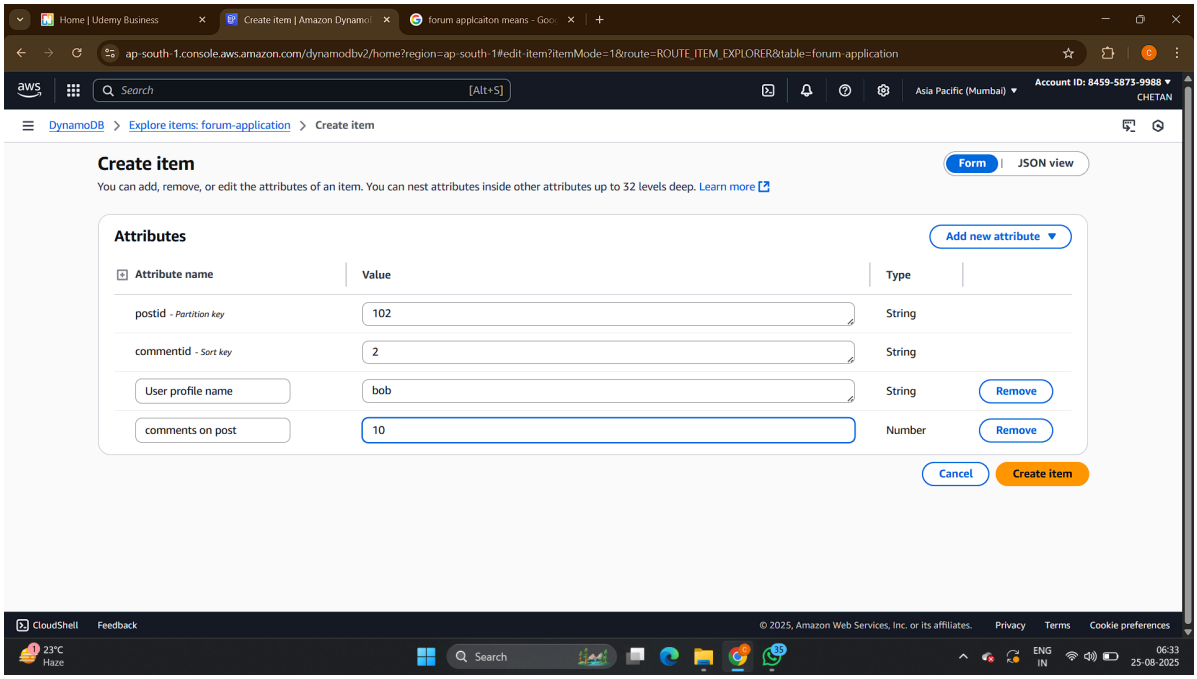
   Retry reading with the `ConsistentRead=True` parameter to guarantee receiving the latest data.

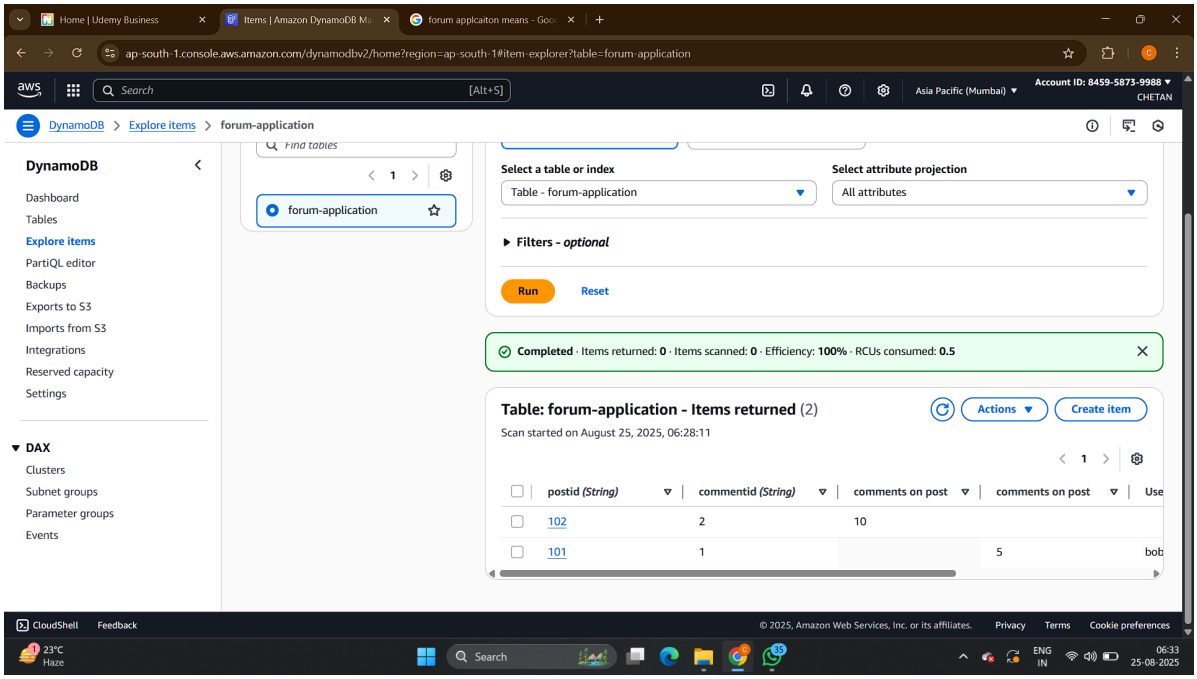
4. **Advanced Experiment with Automation**

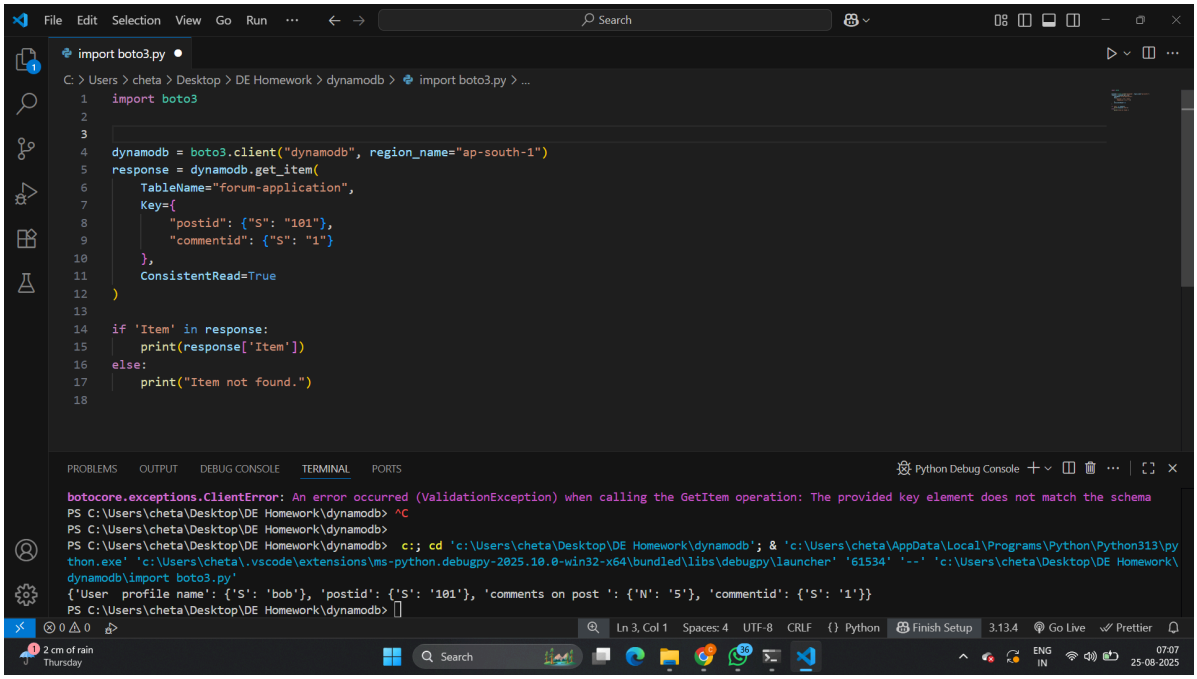
   Use AWS CLI or SDK script to write an item and repeatedly read it 100 times with eventual consistency. Measure how many times the new item was missing to showcase eventual consistency delay windows.

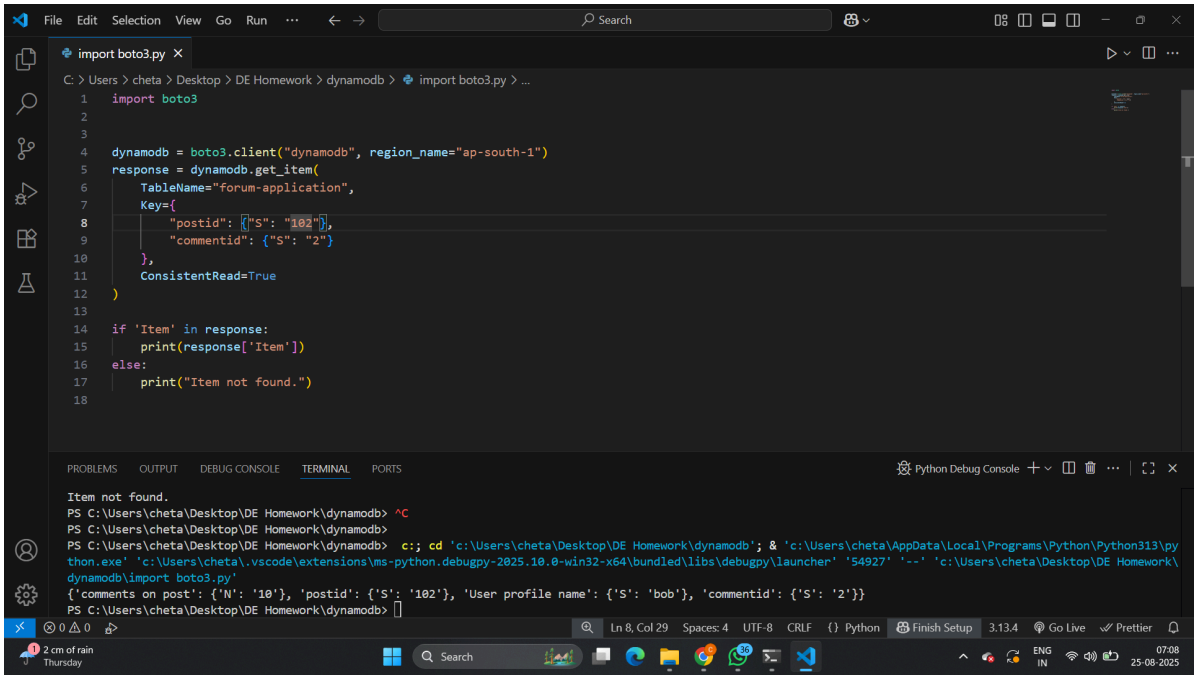



DynamoDB
Explore items
forum-application
Scan
Query
Select a table or index
Table - forum-application
Select attribute projection
All attributes
Filters - optional
Run
Reset
Completed · Items returned: 0 · Items scanned: 0 · Efficiency: 100% · RCUs consumed: 0.5
Table: forum-application - Items returned (1)
Scan started on August 25, 2025, 06:28:11
Actions
Create item
postid (String)
commentid (String)
comments on post
User profile name
101
1
5
bob


Create item
Form
JSON view
You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. Learn more
Attributes
Add new attribute
Attribute name
Value
Type
postid - Partition key
102
String
commentid - Sort key
2
String
User profile name
bob
String
Remove
comments on post
10
Number
Remove
Cancel
Create item


DynamoDB
Dashboard
Tables
Explore items
PartiQL editor
Backups
Exports to S3
Imports from S3
Integrations
Reserved capacity
Settings
DAX
Clusters
Subnet groups
Parameter groups
Events
forum-application
Select a table or index
Table - forum-application
Select attribute projection
All attributes
Filters - optional
Run
Reset
Completed · Items returned: 0 · Items scanned: 0 · Efficiency: 100% · RCUs consumed: 0.5
Table: forum-application - Items returned (2)
Scan started on August 25, 2025, 06:28:11
Actions
Create item
postid (String)
commentid (String)
comments on post
comments on post
102
2
10
101
1
5
bob
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


import boto3.py
import boto3
dynamodb = boto3.client("dynamodb", region_name="ap-south-1")
response = dynamodb.get_item(
TableName="forum-application",
Key={
"postid": {"S": "101"},
"commentid": {"S": "1"}
},
ConsistentRead=True
)
if 'Item' in response:
print(response['Item'])
else:
print("Item not found.")
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
botocore.exceptions.ClientError: An error occurred (ValidationException) when calling the GetItem operation: The provided key element does not match the schema
PS C:\Users\cheta\Desktop\DE Homework\dynamodb> ^C
PS C:\Users\cheta\Desktop\DE Homework\dynamodb>
PS C:\Users\cheta\Desktop\DE Homework\dynamodb> c:; cd 'c:\Users\cheta\Desktop\DE Homework\dynamodb'; & 'c:\Users\cheta\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\cheta\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '61534' '--' 'c:\Users\cheta\Desktop\DE Homework\dynamodb\import boto3.py'
{'User profile name': {'S': 'bob'}, 'postid': {'S': '101'}, 'comments on post ': {'N': '5'}, 'commentid': {'S': '1'}}
PS C:\Users\cheta\Desktop\DE Homework\dynamodb>


File Edit Selection View Go Run
Search
import boto3.py
C: > Users > cheta > Desktop > DE Homework > dynamodb > import boto3.py > ...
import boto3
dynamodb = boto3.client("dynamodb", region_name="ap-south-1")
response = dynamodb.get_item(
TableName="forum-application",
Key={
"postid": {"S": "102"},
"commentid": {"S": "2"}
},
ConsistentRead=True
)
if 'Item' in response:
print(response['Item'])
else:
print("Item not found.")
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python Debug Console
Item not found.
PS C:\Users\cheta\Desktop\DE Homework\dynamodb> ^C
PS C:\Users\cheta\Desktop\DE Homework\dynamodb>
PS C:\Users\cheta\Desktop\DE Homework\dynamodb> c:; cd 'c:\Users\cheta\Desktop\DE Homework\dynamodb'; & 'c:\Users\cheta\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\cheta\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '54927' '--' 'c:\Users\cheta\Desktop\DE Homework\dynamodb\import boto3.py'
{'comments on post': {'N': '10'}, 'postid': {'S': '102'}, 'User profile name': {'S': 'bob'}, 'commentid': {'S': '2'}}
PS C:\Users\cheta\Desktop\DE Homework\dynamodb>
Ln 8, Col 29 Spaces: 4 UTF-8 CRLF {} Python Finish Setup 3.13.4 Go Live Prettier
2 cm of rain
Thursday
Search
ENG
IN
07:08
25-08-2025