

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

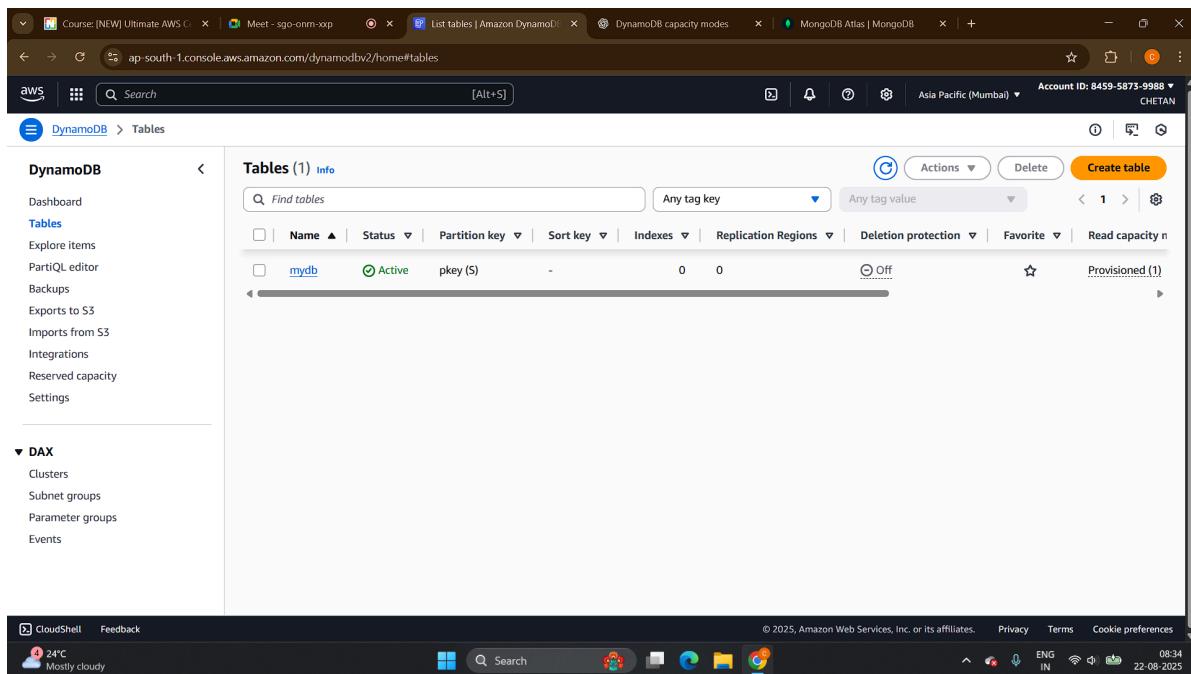
- For one table, select **Provisioned Capacity** mode. Manually set Read Capacity Units (RCU) and Write Capacity Units (WCU) to a low count such as 5 each. This mode requires predicting usage and managing capacity.
- For the other table, choose **On-Demand** capacity, where throughput is managed automatically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

The screenshot shows the 'Create table' wizard in the AWS Management Console. The 'Table details' section is active, displaying fields for 'Table name' (set to 'mydb'), 'Partition key' (set to 'pkey' of type 'String'), and 'Sort key - optional' (left empty). Below this, the 'Table settings' section is partially visible, showing 'Default settings' selected. The browser status bar indicates the URL is `ap-south-1.console.aws.amazon.com/dynamodbv2/home?region=ap-south-1#create-table`.

The screenshot shows the 'Create table' wizard continuing through the configuration steps. The 'Read capacity' section is shown, with 'Auto scaling' set to 'Off' and 'Provisioned capacity units' set to 1. The 'Write capacity' section follows, also with 'Auto scaling' set to 'Off' and 'Provisioned capacity units' set to 1. The 'Warm throughput' section is at the bottom, noting its purpose for handling peak events. The browser status bar indicates the URL is `ap-south-1.console.aws.amazon.com/dynamodbv2/home?region=ap-south-1#create-table`.



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"
  }
}
```

```
        "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.

The screenshot shows the 'Create item' interface in the AWS DynamoDB console. The URL is `ap-south-1.console.aws.amazon.com/dynamodbv2/home#edit-item?itemMode=1&route=ROUTE_ITEM_EXPLORER&table=mydb`. The page title is 'Create item | Amazon DynamoDB'. The main form has 'Form' selected. The 'Attributes' section contains two entries:

Attribute name	Value	Type
pkey - Partition key	101	String
name	bob	String

Buttons at the bottom include 'Cancel' and 'Create item'.

The screenshot shows two views of the same item creation process in the AWS DynamoDB console:

- Form View:** On the left, the 'Create item' form is displayed. It has three attributes: 'pkey' (String type) with value '102', 'first name' (String type) with value 'chetan', and 'last name' (String type) with value 'varma'. There are 'Remove' buttons next to each attribute entry.
- JSON View:** On the right, the JSON representation of the item is shown. The JSON object has three properties: 'pkey' (value '103'), 'name' (Object type), and 'age' (Number type). The 'name' object contains 'first_name' ('chetan') and 'last_name' ('varma').

Both views have 'Cancel' and 'Create item' buttons at the bottom.

The screenshot shows the AWS DynamoDB console with the 'Explore items' page for the 'mydb' table. The left sidebar includes links for Dashboard, Tables, Explore items (which is selected), PartiQL editor, Backups, Exports to S3, Imports from S3, Integrations, Reserved capacity, Settings, and DAX. The main area displays the 'Table - mydb' section with a search bar containing 'mydb'. Below it are 'Filters - optional' buttons for 'Run' and 'Reset'. A green status bar indicates 'Completed · Items returned: 2 · Items scanned: 2 · Efficiency: 100% · RCU consumed: 0.5'. The table data view shows three items:

pkey	age	email	first name	last name	name
103	21	chetan@g...	{"first_name": "Chetan", "last_name": "Varma", "name": "Chetan Varma"}		
102	<empty>		{"first_name": null, "last_name": "Varma", "name": "Varma"}		
101			{"first_name": null, "last_name": "Bob", "name": "Bob"}		

Course [NEW] Ultimate AWS Course | Meet - sgo-onm-xxp | Create table | Amazon DynamoDB | DynamoDB capacity modes | MongoDB Atlas | MongoDB | +

ap-south-1.console.aws.amazon.com/dynamodbv2/home#create-table

aws Search [Alt+S]

Asia Pacific (Mumbai) Account ID: 8459-5873-9988 CHETAN

DynamoDB > Tables > Create table

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
The fastest way to create your table. You can modify most of these settings after your table has been created. To

Customize settings
Use these advanced features to make DynamoDB work better for your needs.

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Very humid Now ENG IN 09:01 22-08-2025

The screenshot shows the 'Create item' page in the AWS DynamoDB console. The 'Form' tab is selected. There are two attributes defined:

- id** - Partition key: Value 1, Type String
- loc** - Sort key: Value india, Type String

Buttons at the bottom include 'Cancel' and 'Create item'.

The screenshot shows the 'Items' page in the AWS DynamoDB console, displaying the contents of the 'table2' table.

Scan Configuration:

- Mode: Scan
- Select a table or index: Table - table2
- Select attribute projection: All attributes
- Filters - optional: None

Completed Scan Summary:

- Items returned: 0
- Items scanned: 0
- Efficiency: 100%
- RCUs consumed: 2

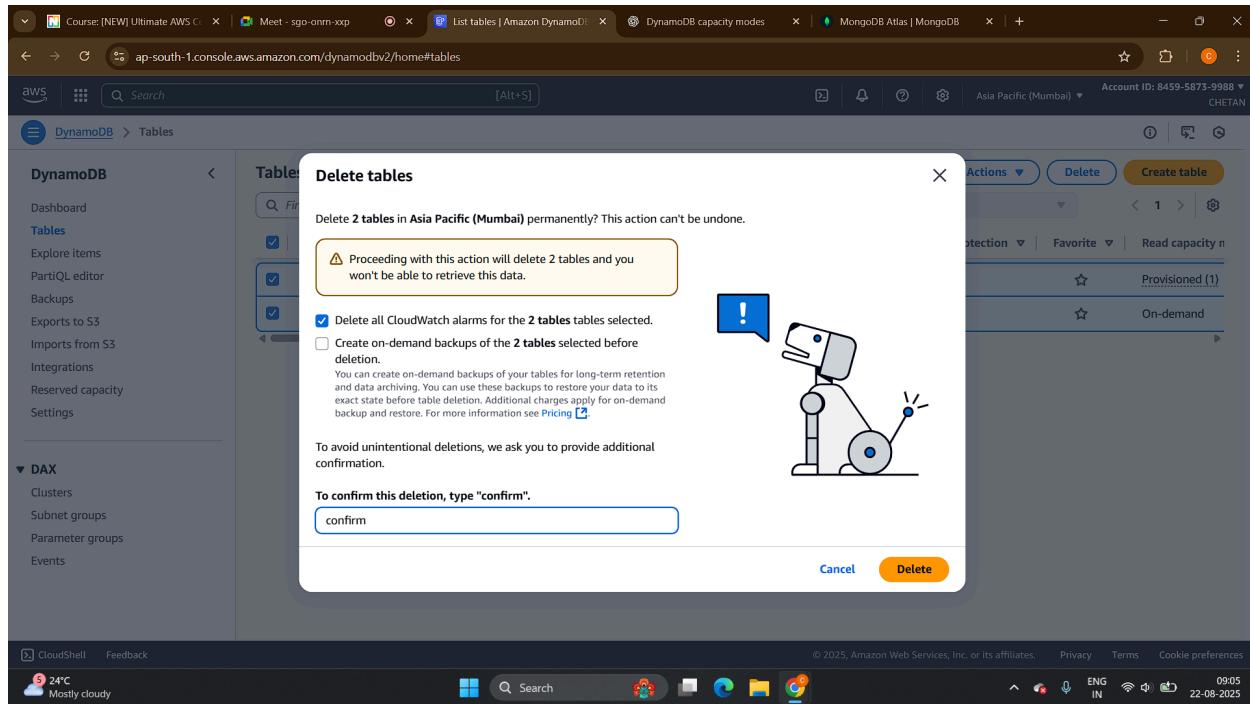
Table: table2 - Items returned (1)

	id (String)	loc (String)
1	1	india

The screenshot shows the AWS DynamoDB 'Create item' interface. A green success message at the top states 'The item has been saved successfully.' Below it, the 'Create item' form is displayed with two attributes: 'id - Partition key' set to '1' and 'String type', and 'loc - Sort key' set to 'singapore' and also 'String type'. At the bottom right are 'Cancel' and 'Create item' buttons.

The screenshot shows the AWS DynamoDB 'Items Explorer' for the 'table2' table. The left sidebar shows 'Explore items' selected. The main area displays the table schema with 'id (String)' and 'loc (String)'. A completed scan operation is shown with 2 items returned. The table data is as follows:

	id (String)	loc (String)
1	1	singapore
2	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

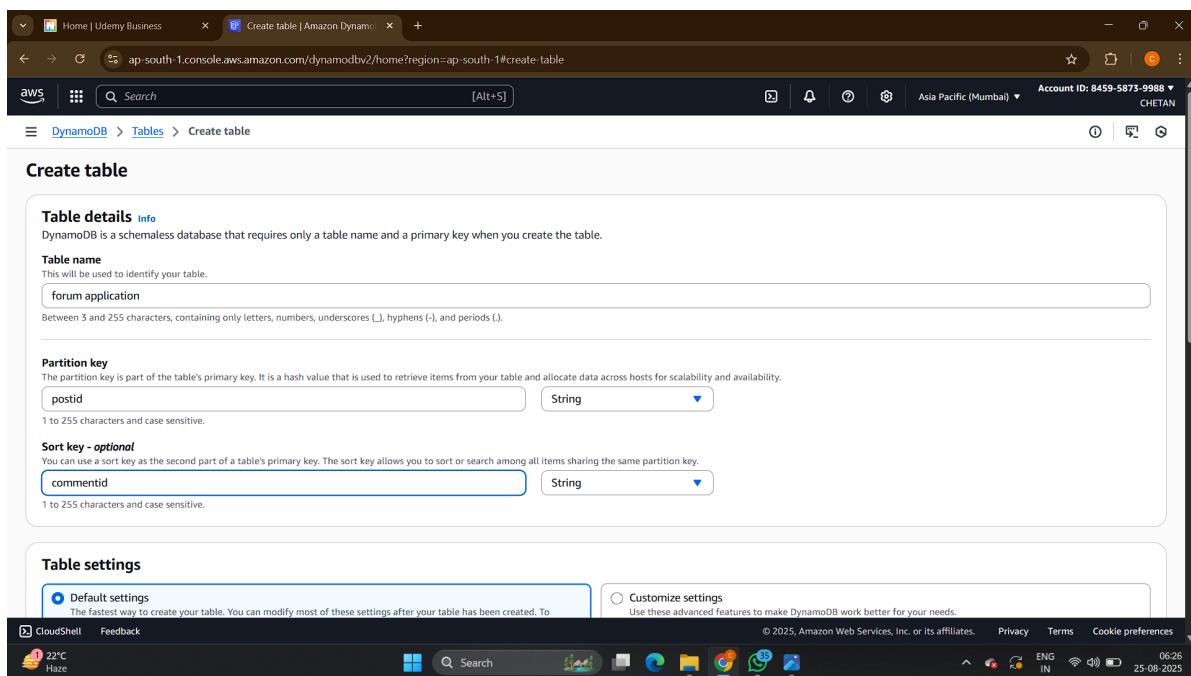
- If using **one table** for posts and comments:
 - Partition Key: `POST#<PostID>` (to group all related items)
 - Sort Key: `#METADATA#<PostID>` for post data, and `COMMENT#<CommentID>` for comments (to enable sorting and querying by comment IDs).
- If using **multiple tables**:

- o Posts table with simple key: `PostID`
- o 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.



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.

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.

Table settings

Default settings
The fastest way to create your table. You can modify most of these settings after your table has been created. To modify these settings now, choose 'Customize settings'.

Customize settings
Use these advanced features to make DynamoDB work better for your needs.

Table class [Info](#)

Select table class to optimize your table's cost based on your workload requirements and data access patterns.

Choose table class

DynamoDB Standard
The default general-purpose table class. Recommended for the vast majority of tables that store frequently accessed data, with throughput (reads and writes) as the dominant table cost.

DynamoDB Standard-IA
Recommended for tables that store data that is infrequently accessed, with storage as the dominant table cost.

CloudShell Feedback 22°C Haze © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 06:27 IN 25-08-2025

Read/write capacity settings [Info](#)

Capacity mode

On-demand
Simplify billing by paying for the actual reads and writes your application performs.

Provisioned
Manage and optimize your costs by allocating read/write capacity in advance.

Read capacity

Auto scaling [Info](#)
Dynamically adjusts provisioned throughput capacity on your behalf in response to actual traffic patterns.

On

Off

Provisioned capacity units
5

Write capacity

Auto scaling [Info](#)
Dynamically adjusts provisioned throughput capacity on your behalf in response to actual traffic patterns.

On

Off

Provisioned capacity units
5

CloudShell Feedback 22°C Haze © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 06:27 IN 25-08-2025

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.

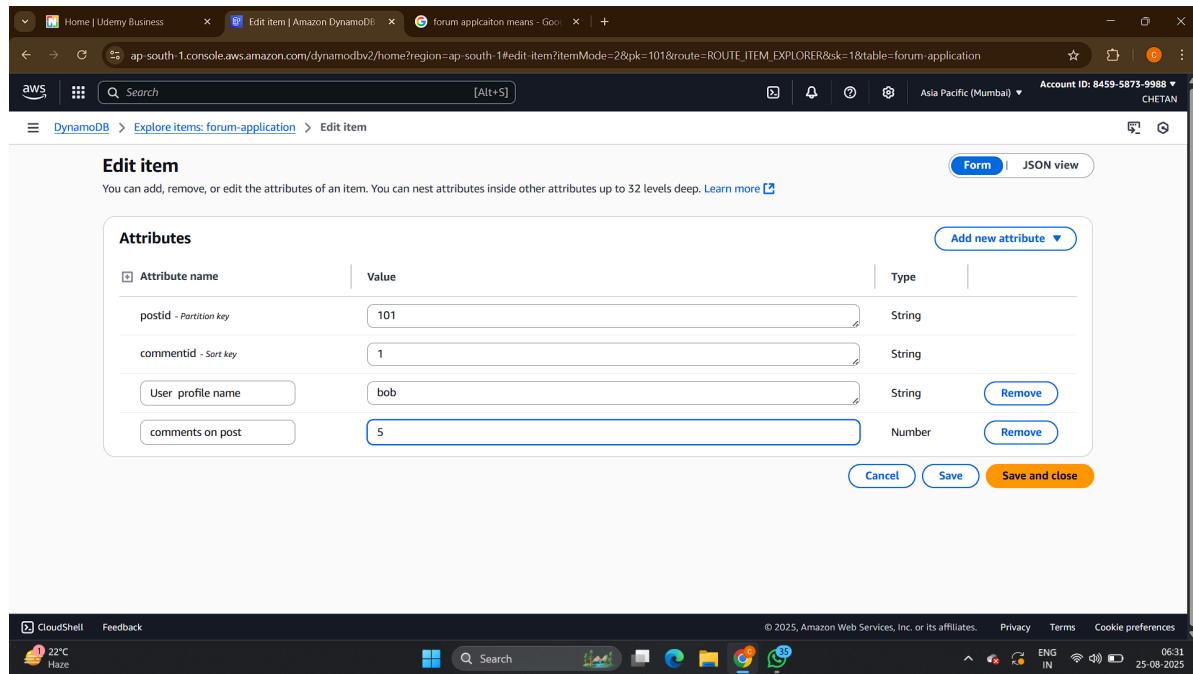
Immediately read the new item and note if the item is returned (it might not if the data is still propagating).

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.



The screenshot shows the AWS Lambda function configuration page. At the top, there are tabs for 'Function' and 'Configuration'. Below these, the 'Handler' is set to 'lambda_function.lambda_handler'. The 'Runtime' is 'Python 3.8'. Under the 'Memory' section, 'Memory size' is set to 128 MB and 'Maximum memory' is set to 256 MB. The 'Timeout' is 3 seconds. The 'Code' section shows the ARN: arn:aws:lambda:ap-south-1:845958739988:function:forum-application. The 'Environment' section shows the environment variables: AWS_LAMBDA_FUNCTION_NAME=forum-application, AWS_LAMBDA_FUNCTION_MEMORY_SIZE=128, AWS_LAMBDA_FUNCTION_TIMEOUT=3, and AWS_LAMBDA_HANDLER=lambda_function.lambda_handler. The 'Logs' section shows the log group: /aws/lambda/forum-application. The 'Logs' tab is selected. The log output shows several entries, including the deployment of the function and its execution. The log stream is forum-application/1d4f3e3c-1330-4330-8330-133013301330. The log level is INFO.

The screenshot shows the AWS DynamoDB console with the 'Explore items' page for the 'forum-application' table. The left sidebar includes links for Dashboard, Tables, Explore items (selected), PartiQL editor, Backups, Exports to S3, Imports from S3, Integrations, Reserved capacity, and Settings. Under 'DAX', there are links for Clusters, Subnet groups, Parameter groups, and Events. The main area has a search bar with 'Any tag value' and a dropdown for 'Find tables'. A table navigation bar shows '1' item. The table configuration section includes 'Scan' and 'Query' buttons, 'Select a table or index' set to 'Table - forum-application', 'Select attribute projection' set to 'All attributes', and a 'Filters - optional' section. Below is a green status bar: 'Completed - Items returned: 0 - Items scanned: 0 - Efficiency: 100% - RCU's consumed: 0.5'. The table view shows the following data:

	postid (String)	commentid (String)	comments on post	User profile name
1	101	1	5	bob

The screenshot shows the 'Create item' interface for the 'forum-application' table. The top navigation bar includes 'CloudShell', 'Feedback', 'Search', and various browser icons. The main area has tabs for 'Form' (selected) and 'JSON view'. The 'Attributes' section lists the following fields:

Attribute name	Value	Type
postid - Partition key	102	String
commentid - Sort key	2	String
User profile name	bob	String
comments on post	10	Number

Buttons at the bottom include 'Cancel' and 'Create item'.

DynamoDB > Explore items > forum-application

Select a table or index: Table - forum-application | Select attribute projection: All attributes

Completed - Items returned: 0 - Items scanned: 0 - Efficiency: 100% - RLCUs consumed: 0.5

	postid (String)	commentid (String)	comments on post	comments on post	User
102	2	10			bob
101	1	5			bob

```

import boto3.py

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.python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '61554' '--' '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> [

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The main editor area displays a Python script named `boto3.py`. The code uses the `boto3` library to interact with an AWS DynamoDB table named `forum-application`. It attempts to retrieve an item with `postid=102` and `commentid=2`, printing the result if found or an error message if not.

```
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.")
```

The terminal tab below shows the execution of the script. The output indicates that no item was found, matching the expected behavior of the code.

```
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.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>
```