



Query Data with DynamoDB



Prajit Venkatachalam

<https://www.linkedin.com/in/prajit-venkatachalam/>

```
}  
}  
[cloudshell-user@ip-10-132-71-32 ~]$ aws dynamodb get-item \  
> --table-name ContentCatalog \  
> --key '{"Id":{"N":"101"}}' \  
> --consistent-read \  
> --projection-expression "Title, ContentType, Services" \  
> --return-consumed-capacity TOTAL  
{  
  "ConsumedCapacity": {  
    "TableName": "ContentCatalog",  
    "CapacityUnits": 1.0  
  }  
}  
[cloudshell-user@ip-10-132-71-32 ~]$  
[cloudshell-user@ip-10-132-71-32 ~]$
```

CloudShell [Feedback](#)



Introducing Today's Project!

What is Amazon DynamoDB?

Amazon DynamoDB is a NoSQL database service. It organizes data into tables, with information structured in the form of items and attributes.

How I used Amazon DynamoDB in this project

In today's project, I used DynamoDB to create and load tables, and I executed queries using the AWS CLI. I also processed transactions, which allowed me to perform multiple operations as a single unit to manage related data.

One thing I didn't expect in this project was...

One thing I did not expect in this project was that DynamoDB queries must use the partition key. I encountered an error in scenarios where I tried to find data using only other attributes.

This project took me...

This project took me 3 hours to complete include documentation.



Querying DynamoDB Tables

A partition key is the main tag/ID that DynamoDB uses to partition its table items. Every item must have a partition key, and the partition key's value must be unique unless the table also has a sort key.

A sort key is a secondary filter that DynamoDB uses to search for specific items. Each item's primary key is made up of its partition key and sort key in a table. This means partition key values can be the same as long as the sort keys are different.

▼ Scan or query items

☐ Scan

☒ Query

Select a table or index

Table - Comment ▼

Select attribute projection

All attributes ▼

Id (Partition key)

I have a question/Location of Documentation/ IAM User Setup for the projects.

CommentDateTime (Sort key)

Greater than ▼

2024-09-20

☐ Sort descending

► Filters

Run

Reset



Limits of Using DynamoDB

I encountered an error when I queried for all comments made by a specific user. This happened because queries must use the table's partition key, and the 'PostedBy' attribute, which I used to query the data, was not the partition key.

Insights we can extract from our Comment table include all comments left on a single post, and even sorted filters by comment time and date. However, insights we can't easily extract from the Comment table include comments made by a specific user.

▼ Scan or query items

☐ Scan ☒ Query

Select a table or index: Table - Comment ▼

Select attribute projection: All attributes ▼

Id (Partition key)

Enter partition key value

⊗ The partition key filter cannot be empty.

CommentDateTime (Sort key)

Greater than ▼ Enter sort key value ☐ Sort descending

▼ Filters

Attribute name	Type	Condition	Value	
PostedBy	String	Equal to	User Abdulrahma	Remove

Add filter



Running Queries with CLI

A query I ran in CloudShell was `aws dynamodb get-item`. This query retrieves an item from the table using the partition key value specified in the command.

Query options I could add to my query include `consistent-read`, which provides a strongly consistent read, `projection-expression`, which returns only specified attributes, and `return-consumed-capacity`, which shows the number of capacity units consumed.

```
}  
}  
[cloudshell-user@ip-10-132-71-32 ~]$ aws dynamodb get-item \  
> --table-name ContentCatalog \  
> --key '{"Id":{"N":"101"}}' \  
> --consistent-read \  
> --projection-expression "Title, ContentType, Services" \  
> --return-consumed-capacity TOTAL  
{  
  "ConsumedCapacity": {  
    "TableName": "ContentCatalog",  
    "CapacityUnits": 1.0  
  }  
}  
[cloudshell-user@ip-10-132-71-32 ~]$  
[cloudshell-user@ip-10-132-71-32 ~]$
```



Transactions

A transaction is a group of operations that are bundled together and executed as one. For a transaction to succeed, all operations within that transaction must be processed successfully.

I executed a transaction using AWS CLI commands in AWS CloudShell. This transaction performed two actions: first, it added a new item to a table, and second, it updated an item in the Forum table by increasing the event's comment count by 1.

```
cloudshell-user@ip-10-132-71-32 ~]$  
cloudshell-user@ip-10-132-71-32 ~]$ aws dynamodb transact-write-items --client-request-token TRANSACTION1 --transact-items '[  
  {  
    "Put": {  
      "TableName" : "Comment",  
      "Item" : {  
        "Id" : {"S": "Events/Do a Project Together - NextWork Study Session"},  
        "CommentDateTime" : {"S": "2024-9-27T17:47:30Z"},  
        "Comment" : {"S": "Excited to attend!"},  
        "PostedBy" : {"S": "User Connor"}  
      }  
    },  
    {  
      "Update": {  
        "TableName" : "Forum",
```



NextWork.org

**Everyone
should be in a
job they love.**

Check out nextwork.org for
more projects