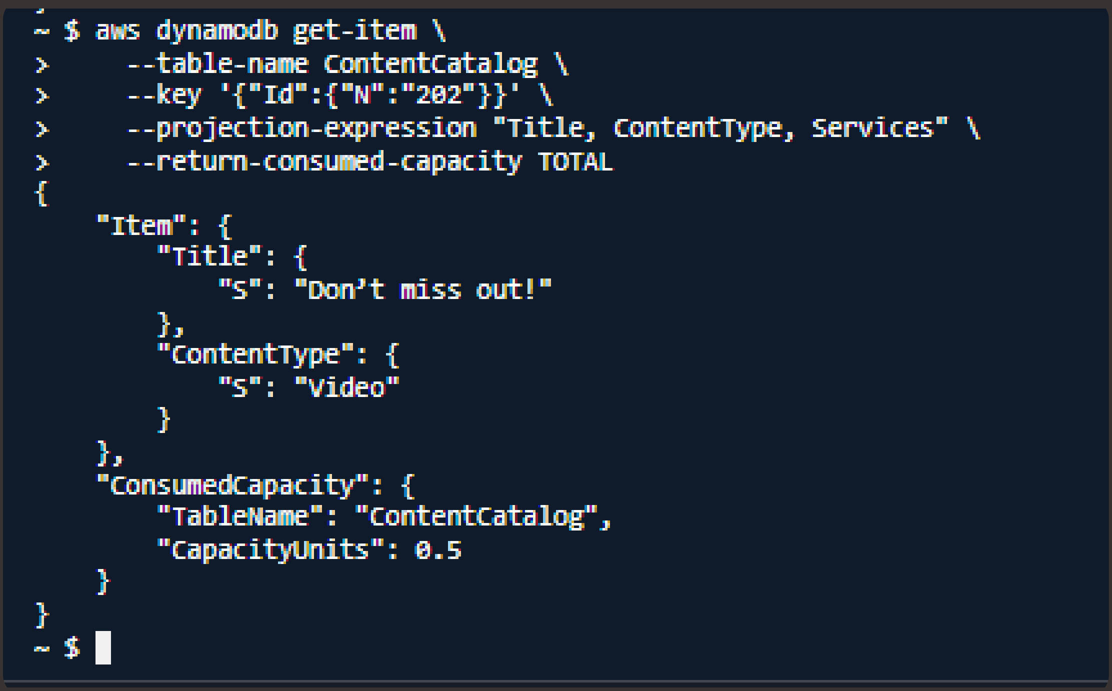
nextwork.org

# Query Data with DynamoDB

Nicolás Aversa



```
~ $ aws dynamodb get-item \
>     --table-name ContentCatalog \
>     --key '{"Id":{"N":"202"}}' \
>     --projection-expression "Title, ContentType, Services" \
>     --return-consumed-capacity TOTAL
{
    "Item": {
        "Title": {
            "S": "Don’t miss out!"
        },
        "ContentType": {
            "S": "Video"
        }
    },
    "ConsumedCapacity": {
        "TableName": "ContentCatalog",
        "CapacityUnits": 0.5
    }
}
~ $
```

# Introducing Today's Project!

## What is Amazon DynamoDB?

Amazon DynamoDB is a non-relational database service that lets you store and retrieve large amounts of data quickly, scale seamlessly to handle high traffic, and use a flexible schema where each item can have different attributes.

## How I used Amazon DynamoDB in this project

I used Amazon DynamoDB in today's project to query the data that was stored in the tables I had previously created.

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

One thing I didn't expect in this project was to find out transactions don't exist in the AWS Management Console, so you can only set them using AWS CLI.

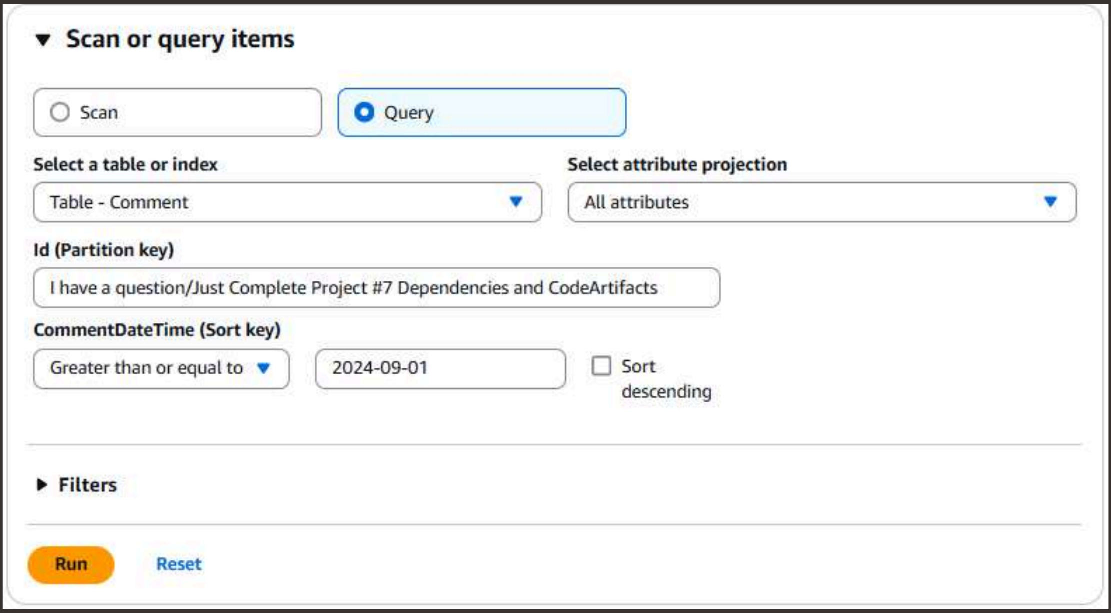
## This project took me...

This project took me a bit more than an hour.

# Querying DynamoDB Tables

A partition key is the filter that DynamoDB will use to split up and find data. It is the key that referrences an item in a table.

A sort key is a secondary key used to filter the query results again. Sort keys work after the partition key i.e. we still have to use the partition key to split up the data first, and then the sort key partitions the data again.



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/Just Complete Project #7 Dependencies and CodeArtifacts

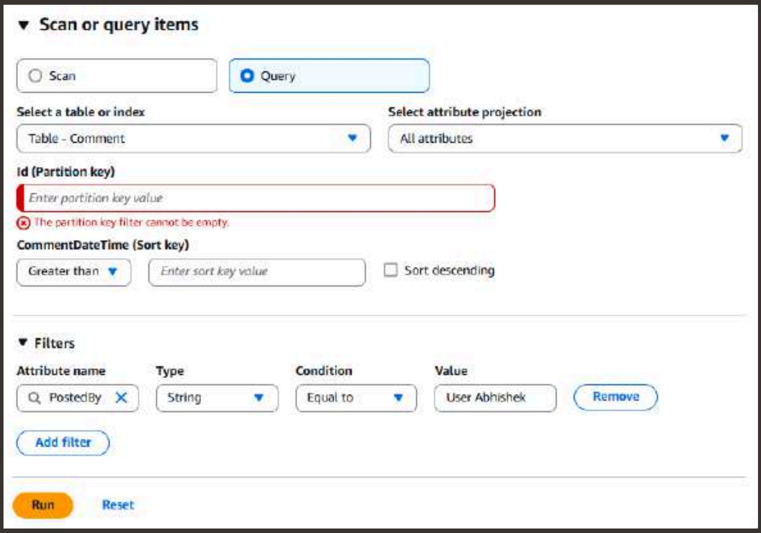
CommentDateTime (Sort key): Greater than or equal to 2024-09-01

☐ Sort descending

# Limits of Using DynamoDB

I ran into an error when I queried for finding all comments posted by User Abdulrahman. This was because you have to use the Id (Partition key) when you query items.

Insights we could extract from our Comment table includes comments to the post 'I have a question...' that were posted 2024-09-01 onwards. Insights we can't easily extract from the Comment table includes all comments posted by User Abdulrahman.



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

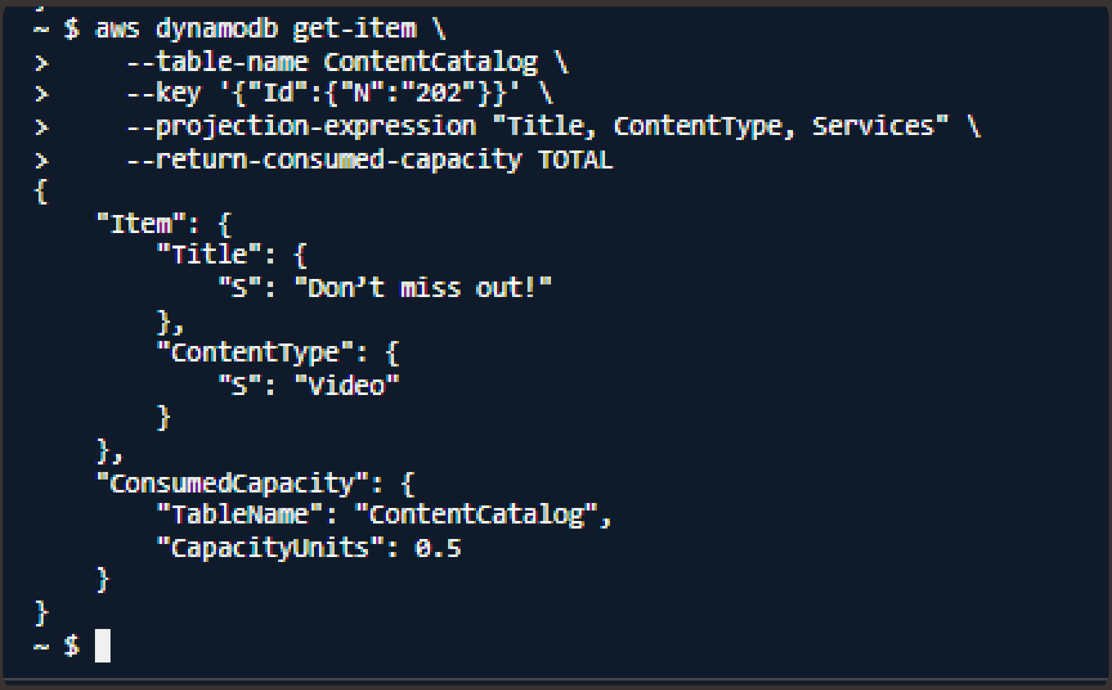
Filters

| Attribute name | Type | Condition | Value |
|---|---|---|---|
| PostedBy | String | Equal to | User Abhishek |

# Running Queries with CLI

A query I ran in CloudShell was to get a single item from a table, with the key 'Id' with a value of 202, only displaying some attributes, and the capacity it consumed. This query will consume 0.5 capacity units, since it's not a consistent read.

Query options I could add to my query were '--consistent-read', which would increase the capacity units to 1.

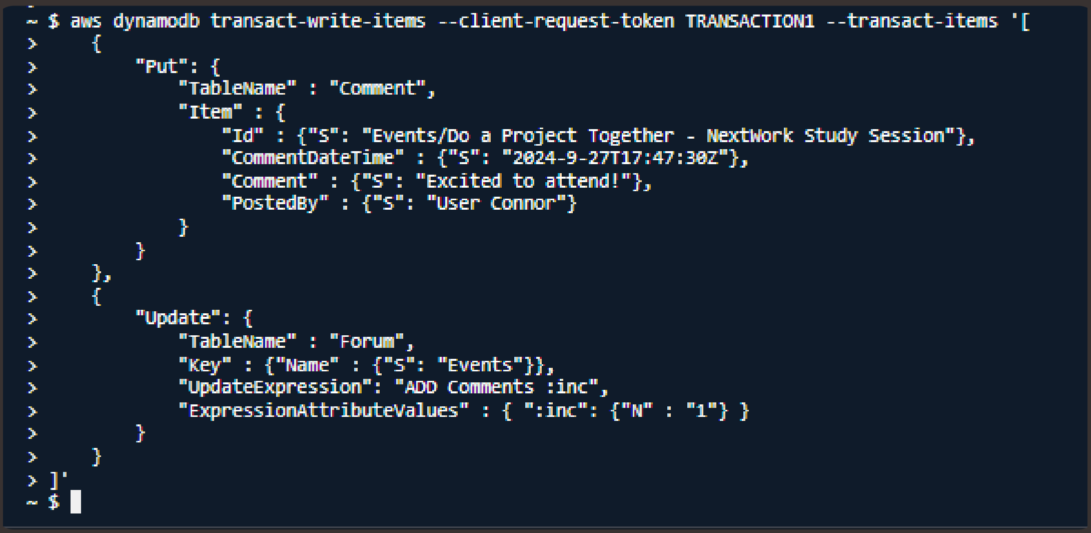


```
~ $ aws dynamodb get-item \
>     --table-name ContentCatalog \
>     --key '{"Id":{"N":"202"}}' \
>     --projection-expression "Title, ContentType, Services" \
>     --return-consumed-capacity TOTAL
{
    "Item": {
        "Title": {
            "S": "Don't miss out!"
        },
        "ContentType": {
            "S": "Video"
        }
    },
    "ConsumedCapacity": {
        "TableName": "ContentCatalog",
        "CapacityUnits": 0.5
    }
}
~ $
```

# Transactions

A transaction is a group of operations that all have to succeed - if any of the operations in the group fails, none of the changes get applied. This makes sure that any change to the database is consistent across all the tables.

'I ran a transaction using AWS CLI. This transaction did two things: 1. Record a new comment made by User Connor into the 'Comment' table. 2. Update 'Forum' table, since the comment was made on a post in the Events forum, so the counter should +1.



```
~ $ 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",
>             "Key" : {"Name" : {"S": "Events"}},
>             "UpdateExpression": "ADD Comments :inc",
>             "ExpressionAttributeValues" : { ":inc": {"N" : "1"} }
>         }
>     }
> ]'
~ $ 
```

# Everyone should be in a job they love.

Check out nextwork.org for more projects