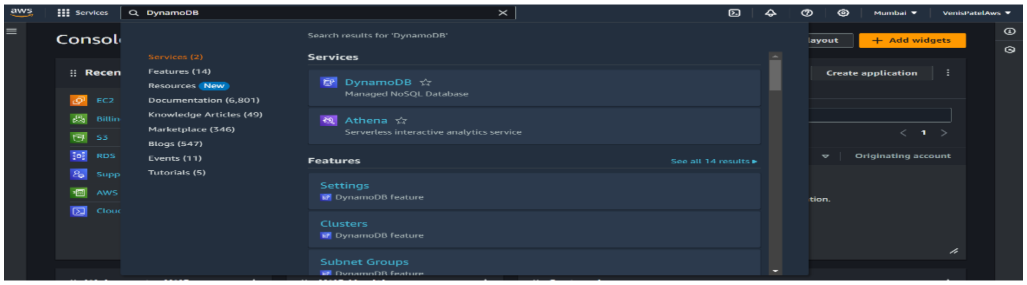
TASK 2: Create a DynamoDB Table:

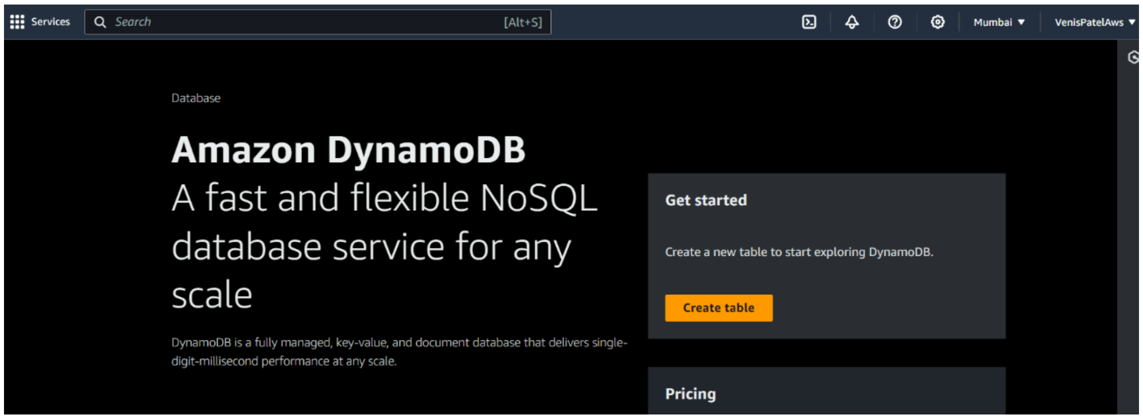
1. **Create a new DynamoDB table with a primary key of your choice.**
2. **Define the provisioned throughput for the table.**
3. **Run CRUD operation in the in DB table using Queries.**

➢ Steps to create a new DynamoDB Table with a primary key of your choice:

1. First go to AWS console and search for DynamoDB.



2. Inside DynamoDB dashboard click on "create table".


Database

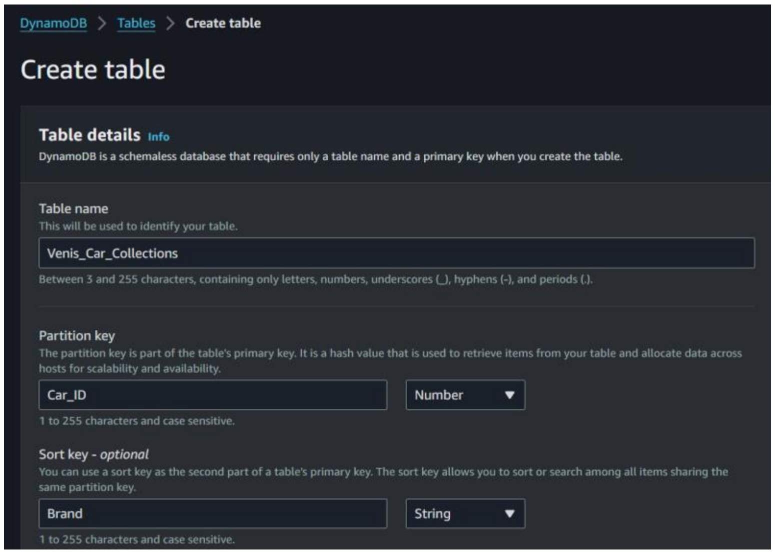
Amazon DynamoDB
A fast and flexible NoSQL database service for any scale

DynamoDB is a fully managed, key-value, and document database that delivers single-digit-millisecond performance at any scale.

Get started

Create a new table to start exploring DynamoDB.

3. Inside create table form, Enter the following detais.
    I. Table Name: VenisCarCollections
    II. Partition Key: Car_ID, Type: Number
    III. Sort Key: Brand, Type: String



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.

Venis_Car_Collections

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.

Car_ID — Number

1 to 255 characters and case sensitive.

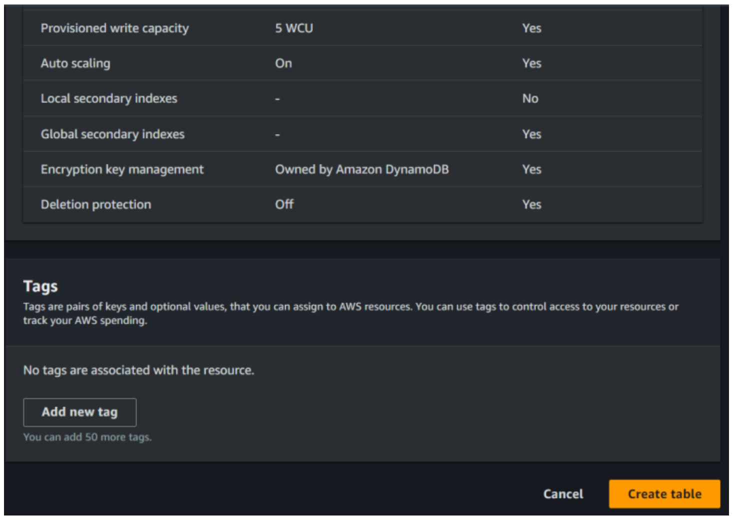
Sort key - *optional*

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.

Brand — String

1 to 255 characters and case sensitive.

4. Then select Default settings under Table settings and click on Create Table



Provisioned write capacity	5 WCU	Yes
Auto scaling	On	Yes
Local secondary indexes	-	No
Global secondary indexes	-	Yes
Encryption key management	Owned by Amazon DynamoDB	Yes
Deletion protection	Off	Yes

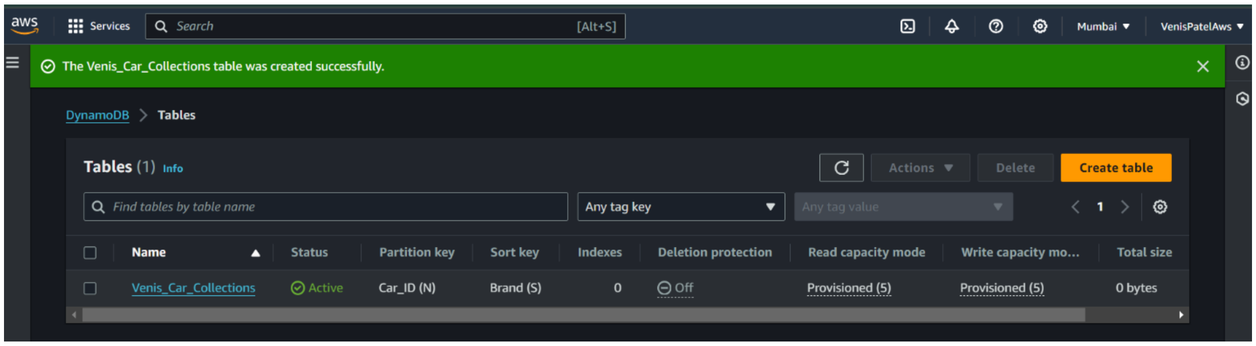
Tags

Tags are pairs of keys and optional values, that you can assign to AWS resources. You can use tags to control access to your resources or track your AWS spending.

No tags are associated with the resource.

You can add 50 more tags.

5. Now in DynamoDB Table Dashboard you will be able to see that Books Table has been created.


The Venis_Car_Collections table was created successfully.

DynamoDB > Tables

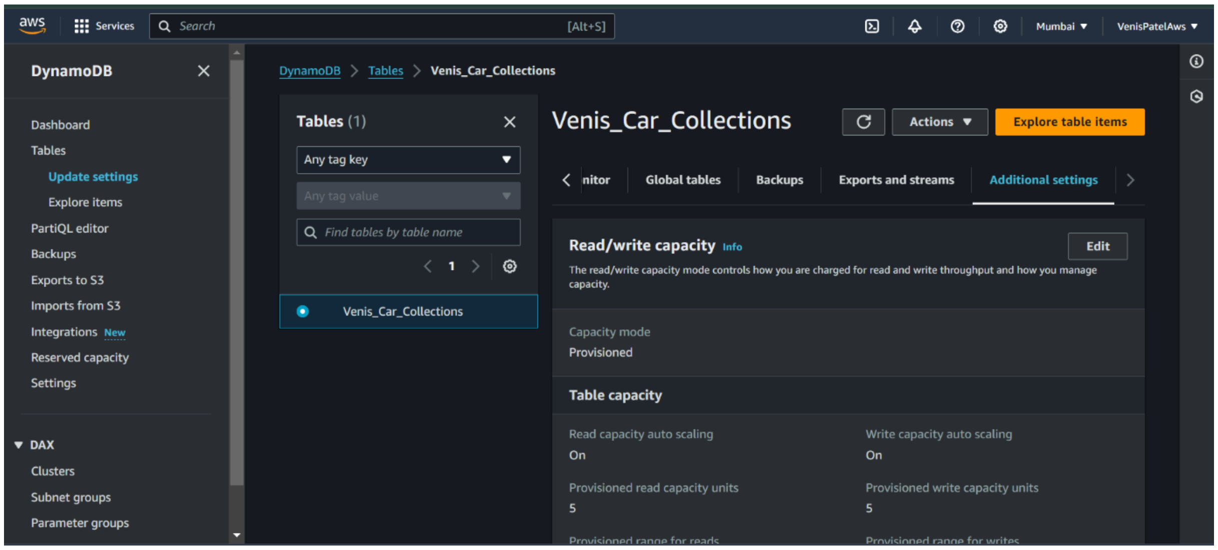
Tables (1) Info

	Name	Status	Partition key	Sort key	Indexes	Deletion protection	Read capacity mode	Write capacity mo...	Total size
☐	Venis_Car_Collections	Active	Car_ID (N)	Brand (S)	0	Off	Provisioned (5)	Provisioned (5)	0 bytes

➢ **Steps to define the provisioned throughput for the table:**

1. Go to Tables under DynaomoDB dashboard and then click on created table.

2. Now inside DynamoDB > Table > VenisCarCollections go to additional settings. Where we can see the provisioned capacity for the table.


DynamoDB > Tables > Venis_Car_Collections

Venis_Car_Collections

Read/write capacity Info

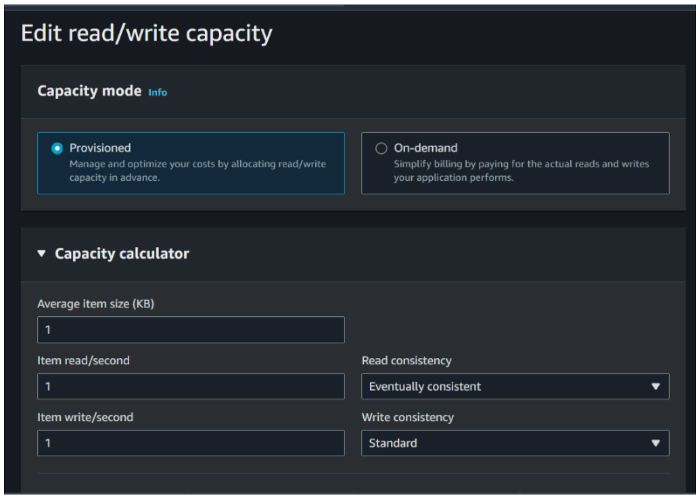
The read/write capacity mode controls how you are charged for read and write throughput and how you manage capacity.

Capacity mode
Provisioned

Table capacity

Read capacity auto scaling On	Write capacity auto scaling On
Provisioned read capacity units 5	Provisioned write capacity units 5
Provisioned range for reads	Provisioned range for writes

3. Click on edit read/write capacity, now we can change table provisioned read write capacity. After that turn off Auto Scaling. And then enter the provisioned capacity units. In my case I will keep it as 1 (because I don't want to incur any additional charges). And then click on save changes.



Edit read/write capacity

Capacity mode Info

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

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

Capacity calculator

Average item size (KB): 1

Item read/second: 1

Read consistency: Eventually consistent

Item write/second: 1

Write consistency: Standard

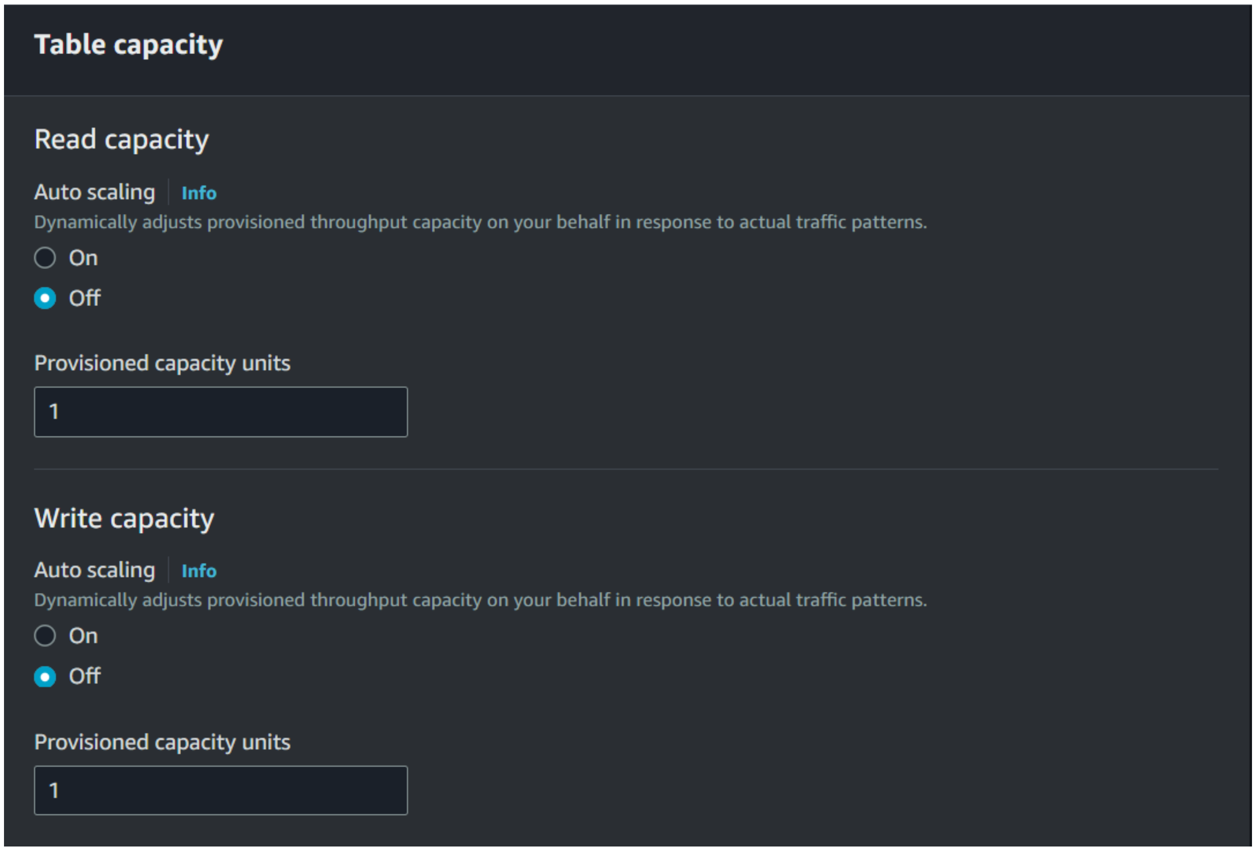


Table capacity

Read capacity

Auto scaling Info

Dynamically adjusts provisioned throughput capacity on your behalf in response to actual traffic patterns.

On [✓ Off]

Provisioned capacity units: 1

Write capacity

Auto scaling Info

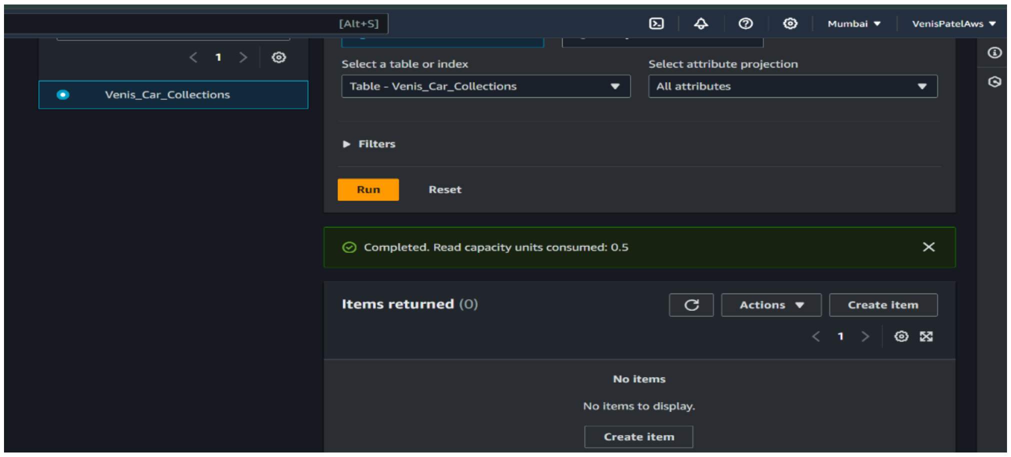
Dynamically adjusts provisioned throughput capacity on your behalf in response to actual traffic patterns.

On [✓ Off]

Provisioned capacity units: 1

➢ Steps to run CRUD operation in the in DB table using Queries:

1. Go to DynamoDB > Table > Books and then click on "Explore table items".



Select a table or index: Table - Venis_Car_Collections

Select attribute projection: All attributes

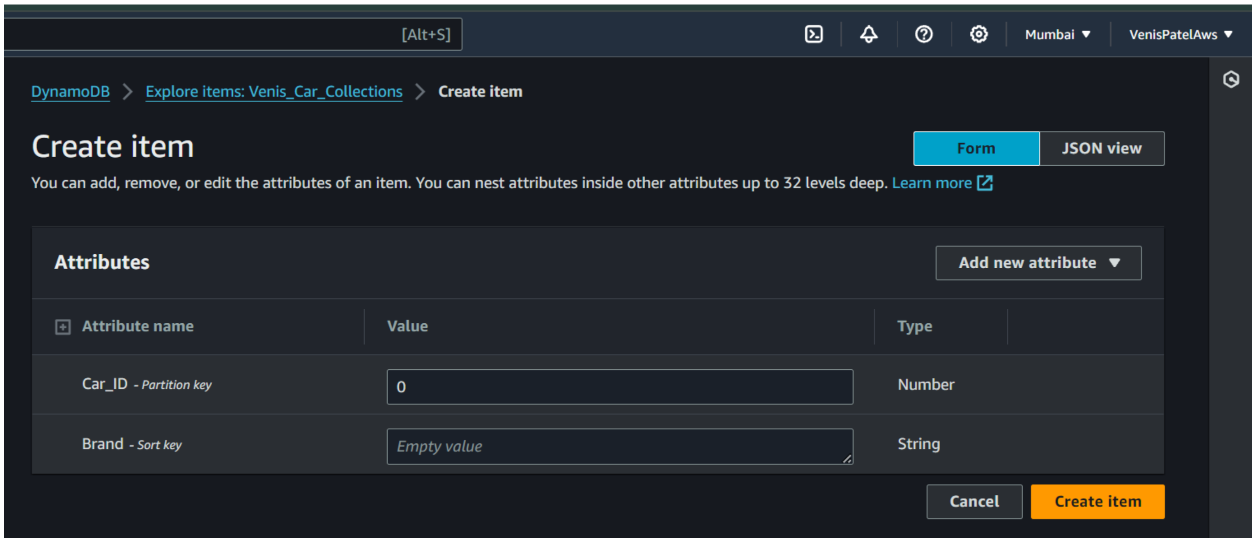
Completed. Read capacity units consumed: 0.5

Items returned (0)

No items

No items to display.

2. Then scroll down and click on create item.



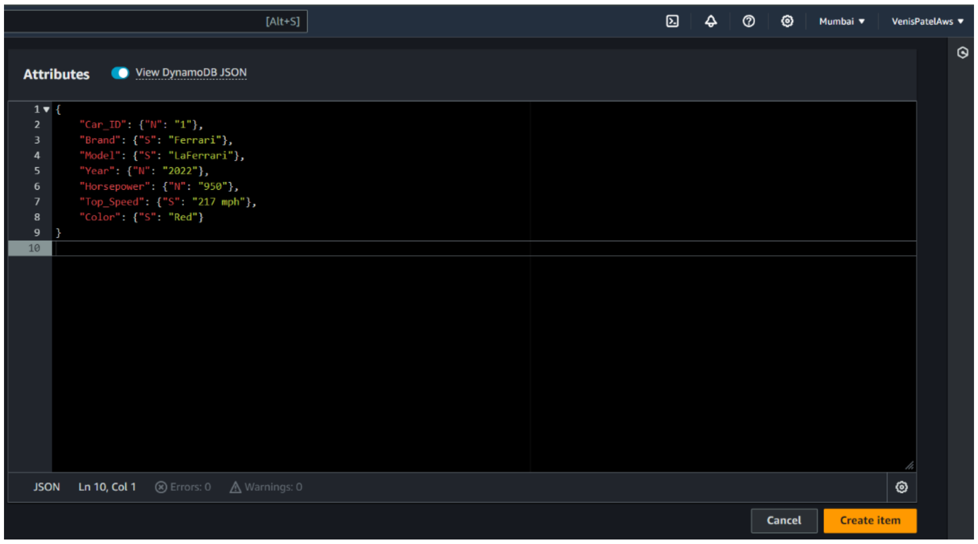
Create item

You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep.

Attributes

Attribute name	Value	Type
Car_ID - *Partition key*	0	Number
Brand - *Sort key*	*Empty value*	String

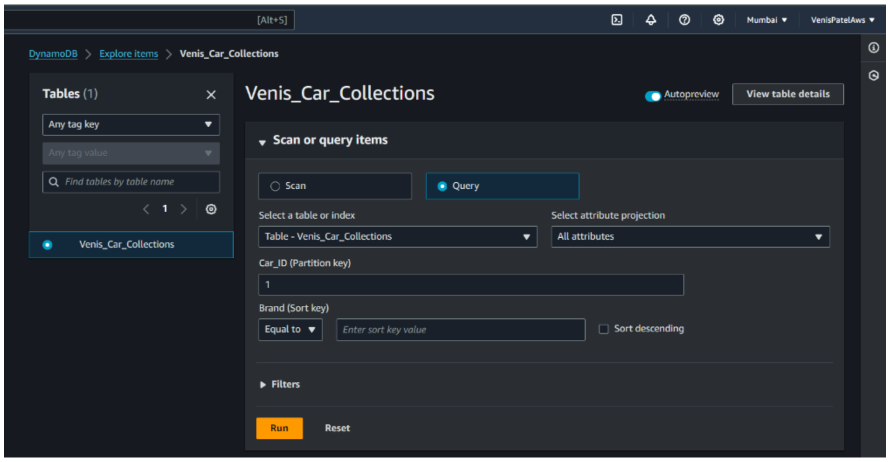
3. Now select JSON View, and then enter the item details. Then click on create item.



```
{
    "Car_ID": {"N": "1"},
    "Brand": {"S": "Ferrari"},
    "Model": {"S": "LaFerrari"},
    "Year": {"N": "2022"},
    "Horsepower": {"N": "950"},
    "Top_Speed": {"S": "217 mph"},
    "Color": {"S": "Red"}
}
```

➢ **To Read Item using AWS console:**

1. Inside DynamoDB > Explore Items > Venis_Car_Collections. Click on query and enter the Car_ID = 1. You will be able to see book details with Car_Id = 1.



Venis_Car_Collections

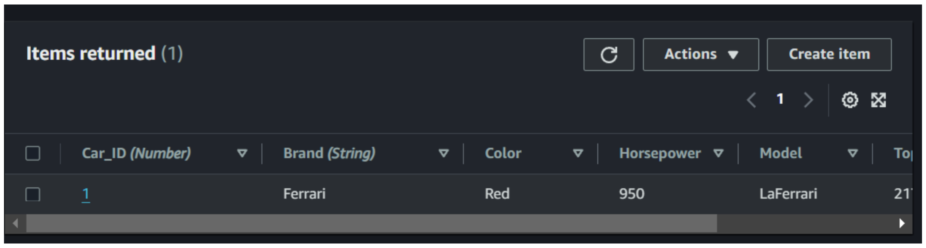
Scan [✓ Query]

Select a table or index: Table - Venis_Car_Collections

Select attribute projection: All attributes

Car_ID (Partition key): 1

Brand (Sort key): Equal to | Enter sort key value | ☐ Sort descending

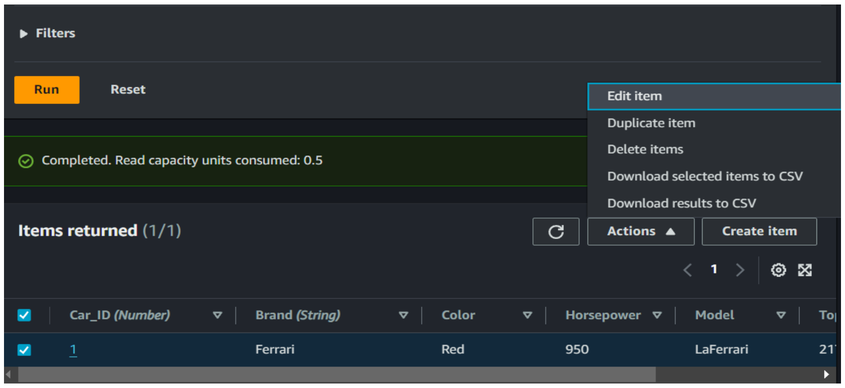


Items returned (1)

☐	Car_ID (Number)	Brand (String)	Color	Horsepower	Model	To
☐	1	Ferrari	Red	950	LaFerrari	21

➢ **Update Items using AWS Console:**

1. Select the item you want to update and then click on action, under action select edit item.

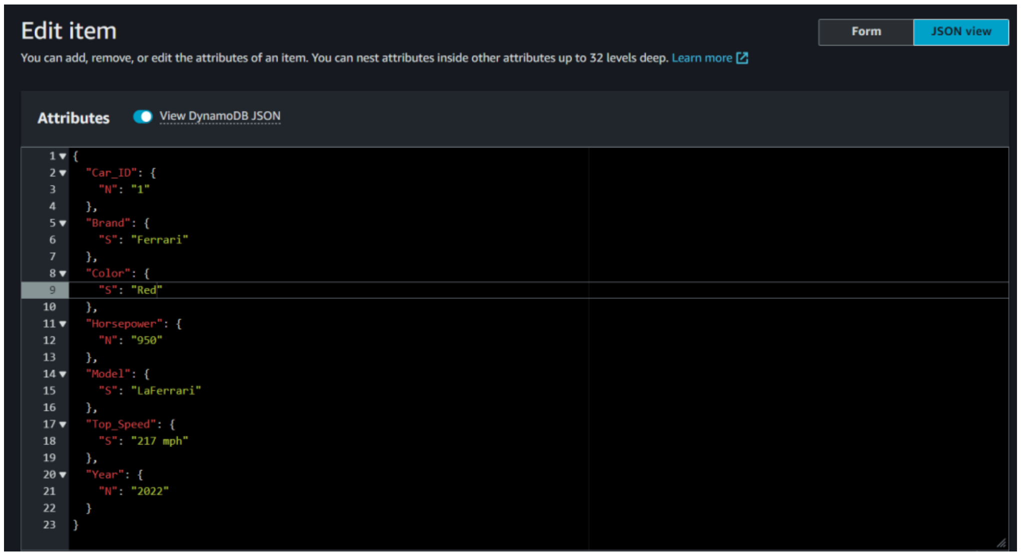


Completed. Read capacity units consumed: 0.5

Items returned (1/1)

	Car_ID (Number)	Brand (String)	Color	Horsepower	Model	To
✓	1	Ferrari	Red	950	LaFerrari	21

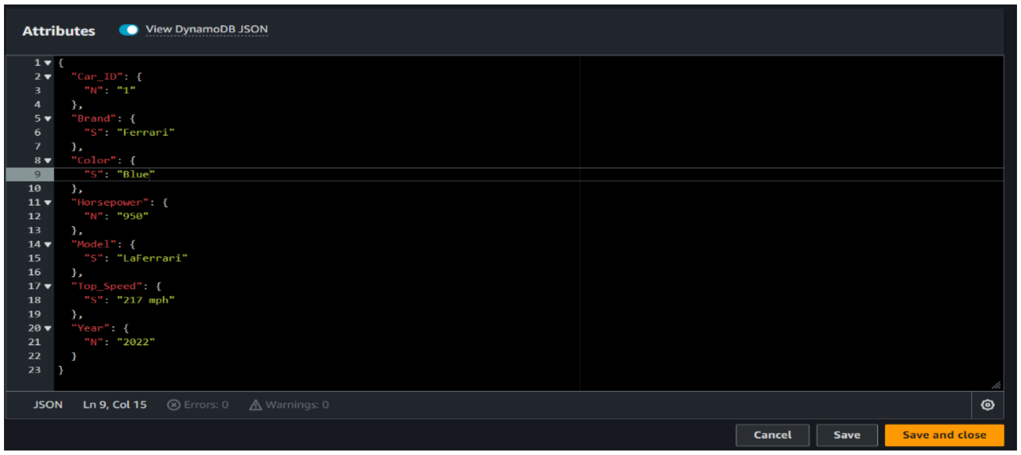
2. Now select JSON View, and change the details that needs to be updated, In my case I change color to Blue from Red. After that click on save and close to update the item.



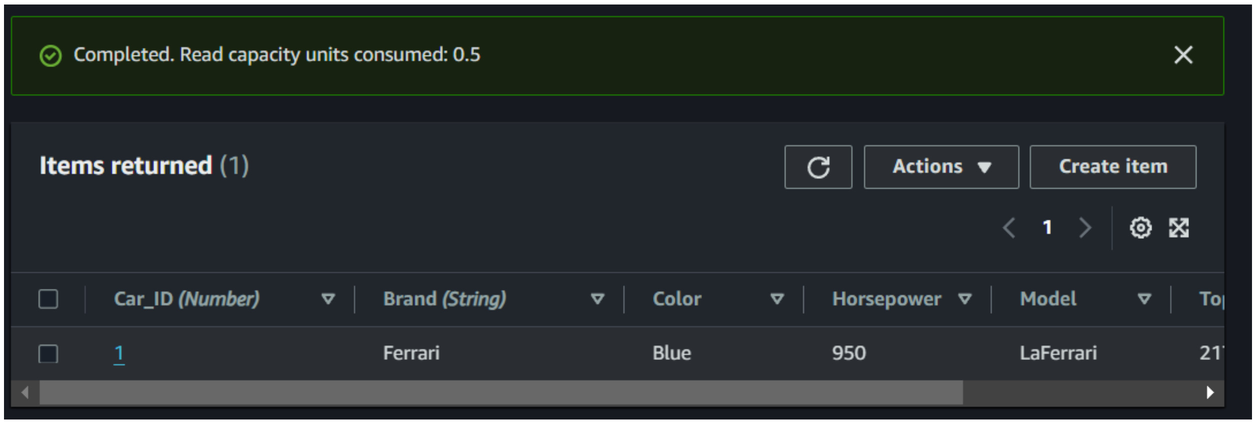
Edit item

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

```
{
  "Car_ID": {
    "N": "1"
  },
  "Brand": {
    "S": "Ferrari"
  },
  "Color": {
    "S": "Red"
  },
  "Horsepower": {
    "N": "950"
  },
  "Model": {
    "S": "LaFerrari"
  },
  "Top_Speed": {
    "S": "217 mph"
  },
  "Year": {
    "N": "2022"
  }
}
```



```
{
  "Car_ID": {
    "N": "1"
  },
  "Brand": {
    "S": "Ferrari"
  },
  "Color": {
    "S": "Blue"
  },
  "Horsepower": {
    "N": "950"
  },
  "Model": {
    "S": "LaFerrari"
  },
  "Top_Speed": {
    "S": "217 mph"
  },
  "Year": {
    "N": "2022"
  }
}
```



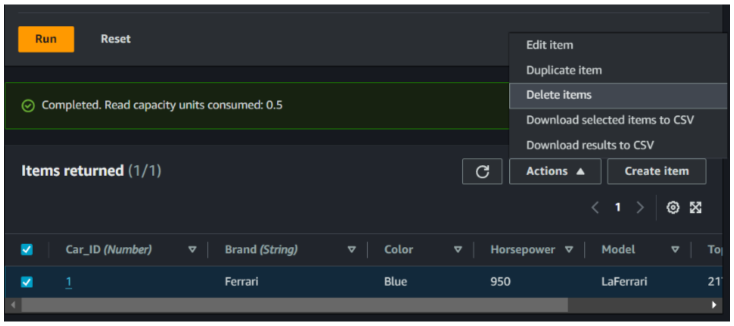
Completed. Read capacity units consumed: 0.5

Items returned (1)

Car_ID (Number)	Brand (String)	Color	Horsepower	Model	To
1	Ferrari	Blue	950	LaFerrari	21

➢ Delete item using AWS Console:

1. Select the item you want to delete, and then click on action and then select delete items to delete selected item.

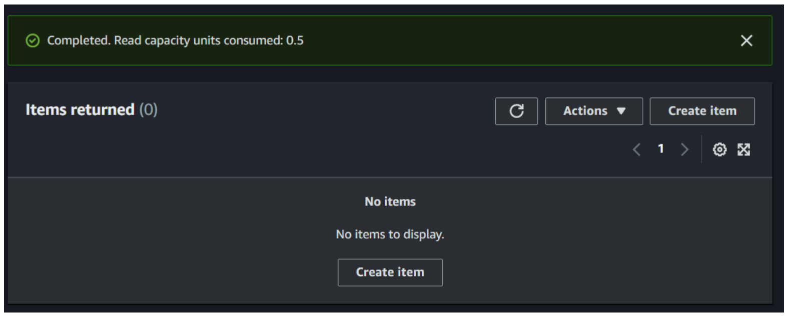


Completed. Read capacity units consumed: 0.5

Items returned (1/1)

Car_ID (Number)	Brand (String)	Color	Horsepower	Model	To
1	Ferrari	Blue	950	LaFerrari	21

2. After that you will be able to see that my item with BookID=1 is deleted.



Completed. Read capacity units consumed: 0.5

Items returned (0)

No items

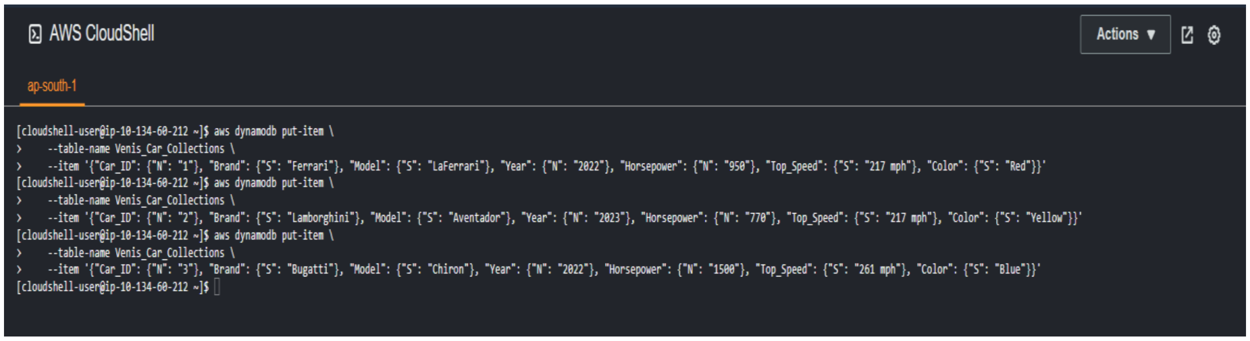
No items to display.

➢ Steps for CRUD Operations using AWS CLI.

- ### 1. Insert:

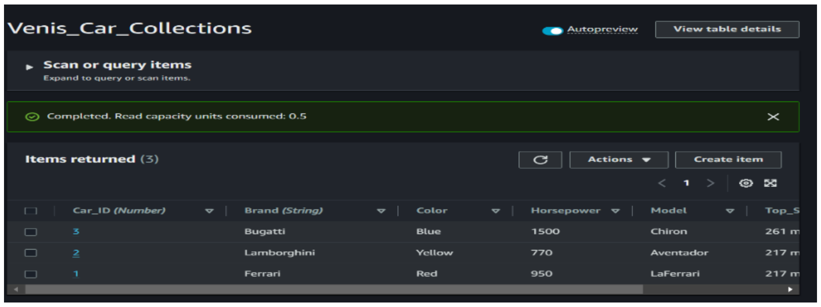
1. Open AWS Cloudshell and run this command .

```
aws dynamodb put-item \
    --table-name Venis_Car_Collections \
    --item '{"Car_ID": {"N": "1"}, "Brand": {"S": "Ferrari"}, "Model": {"S": "LaFerrari"}, "Year": {"N": "2022"}, "Horsepower": {"N": "950"}, "Top_Speed": {"S": "217 mph"}, "Color": {"S": "Red"}}'

aws dynamodb put-item \
    --table-name Venis_Car_Collections \
    --item '{"Car_ID": {"N": "2"}, "Brand": {"S": "Lamborghini"}, "Model": {"S": "Aventador"}, "Year": {"N": "2023"}, "Horsepower": {"N": "770"}, "Top_Speed": {"S": "217 mph"}, "Color": {"S": "Yellow"}}'

aws dynamodb put-item \
    --table-name Venis_Car_Collections \
    --item '{"Car_ID": {"N": "3"}, "Brand": {"S": "Bugatti"}, "Model": {"S": "Chiron"}, "Year": {"N": "2022"}, "Horsepower": {"N": "1500"}, "Top_Speed": {"S": "261 mph"}, "Color": {"S": "Blue"}}'
```



```
[cloudshell-user@ip-10-134-60-212 ~]$ aws dynamodb put-item \
>     --table-name Venis_Car_Collections \
>     --item '{"Car_ID": {"N": "1"}, "Brand": {"S": "Ferrari"}, "Model": {"S": "LaFerrari"}, "Year": {"N": "2022"}, "Horsepower": {"N": "950"}, "Top_Speed": {"S": "217 mph"}, "Color": {"S": "Red"}}'
[cloudshell-user@ip-10-134-60-212 ~]$ aws dynamodb put-item \
>     --table-name Venis_Car_Collections \
>     --item '{"Car_ID": {"N": "2"}, "Brand": {"S": "Lamborghini"}, "Model": {"S": "Aventador"}, "Year": {"N": "2023"}, "Horsepower": {"N": "770"}, "Top_Speed": {"S": "217 mph"}, "Color": {"S": "Yellow"}}'
[cloudshell-user@ip-10-134-60-212 ~]$ aws dynamodb put-item \
>     --table-name Venis_Car_Collections \
>     --item '{"Car_ID": {"N": "3"}, "Brand": {"S": "Bugatti"}, "Model": {"S": "Chiron"}, "Year": {"N": "2022"}, "Horsepower": {"N": "1500"}, "Top_Speed": {"S": "261 mph"}, "Color": {"S": "Blue"}}'
[cloudshell-user@ip-10-134-60-212 ~]$
```

2. Check that the item is created.



Completed. Read capacity units consumed: 0.5

Items returned (3)

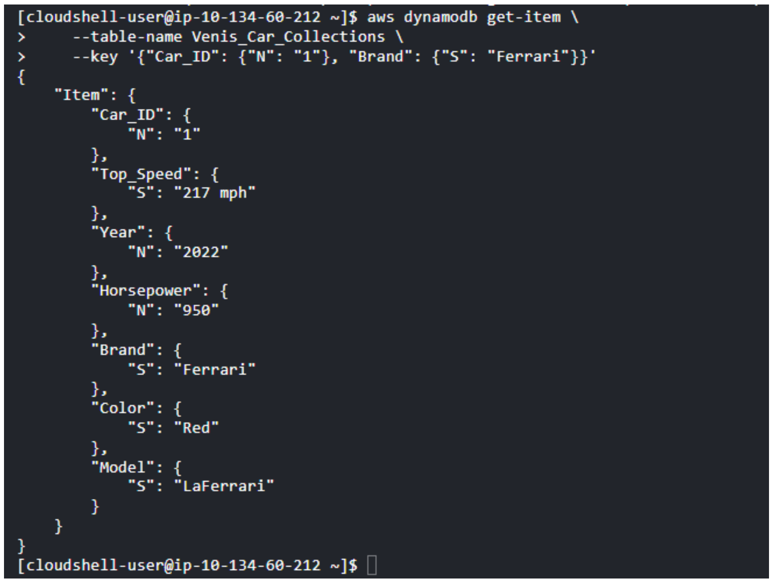
Car_ID (Number)	Brand (String)	Color	Horsepower	Model	Top_S
3	Bugatti	Blue	1500	Chiron	261 m
2	Lamborghini	Yellow	770	Aventador	217 m
1	Ferrari	Red	950	LaFerrari	217 m

- **2. READ:**
  1. Enter the following code in the command line

     aws dynamodb get-item \

     --table-name Venis_Car_Collections \

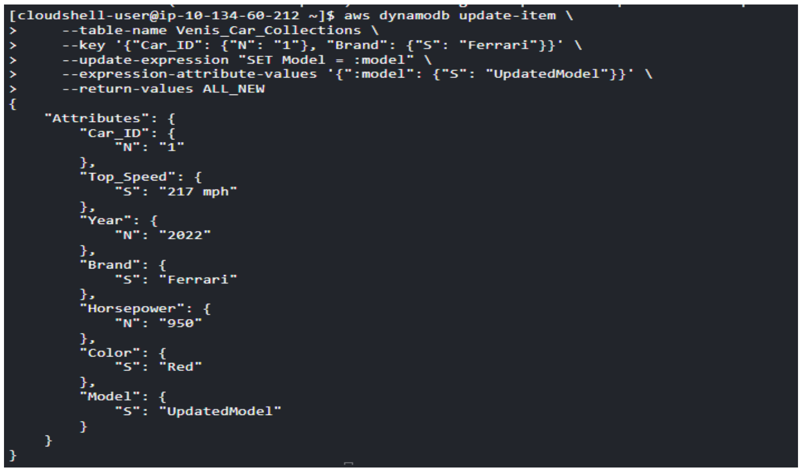
     --key '{"Car_ID": {"N": "1"}, "Brand": {"S": "Ferrari"}}'



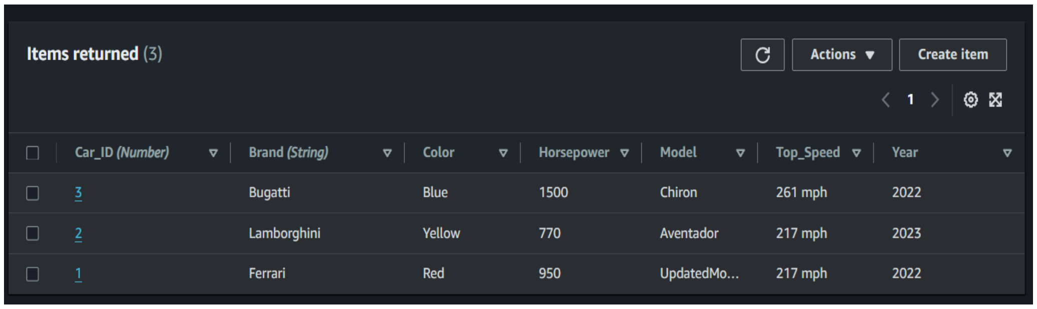
```
[cloudshell-user@ip-10-134-60-212 ~]$ aws dynamodb get-item \
>     --table-name Venis_Car_Collections \
>     --key '{"Car_ID": {"N": "1"}, "Brand": {"S": "Ferrari"}}'
{
    "Item": {
        "Car_ID": {
            "N": "1"
        },
        "Top_Speed": {
            "S": "217 mph"
        },
        "Year": {
            "N": "2022"
        },
        "Horsepower": {
            "N": "950"
        },
        "Brand": {
            "S": "Ferrari"
        },
        "Color": {
            "S": "Red"
        },
        "Model": {
            "S": "LaFerrari"
        }
    }
}
[cloudshell-user@ip-10-134-60-212 ~]$
```

- **3. Update:**

  1. Enter the following command to update BookName.
     aws dynamodb update-item \
        --table-name Venis_Car_Collections \
        --key '{"Car_ID": {"N": "1"}, "Brand": {"S": "Ferrari"}}' \
        --update-expression "SET Model = :model" \
        --expression-attribute-values '{":model": {"S": "UpdatedModel"}}' \
        --return-values ALL_NEW



```
[cloudshell-user@ip-10-134-60-212 ~]$ aws dynamodb update-item \
>     --table-name Venis_Car_Collections \
>     --key '{"Car_ID": {"N": "1"}, "Brand": {"S": "Ferrari"}}' \
>     --update-expression "SET Model = :model" \
>     --expression-attribute-values '{":model": {"S": "UpdatedModel"}}' \
>     --return-values ALL_NEW
{
    "Attributes": {
        "Car_ID": {
            "N": "1"
        },
        "Top_Speed": {
            "S": "217 mph"
        },
        "Year": {
            "N": "2022"
        },
        "Brand": {
            "S": "Ferrari"
        },
        "Horsepower": {
            "N": "950"
        },
        "Color": {
            "S": "Red"
        },
        "Model": {
            "S": "UpdatedModel"
        }
    }
}
```

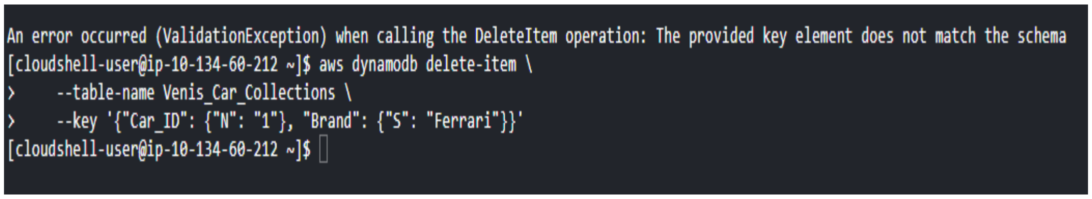


Items returned (3)

Car_ID (Number)	Brand (String)	Color	Horsepower	Model	Top_Speed	Year
3	Bugatti	Blue	1500	Chiron	261 mph	2022
2	Lamborghini	Yellow	770	Aventador	217 mph	2023
1	Ferrari	Red	950	UpdatedMo...	217 mph	2022

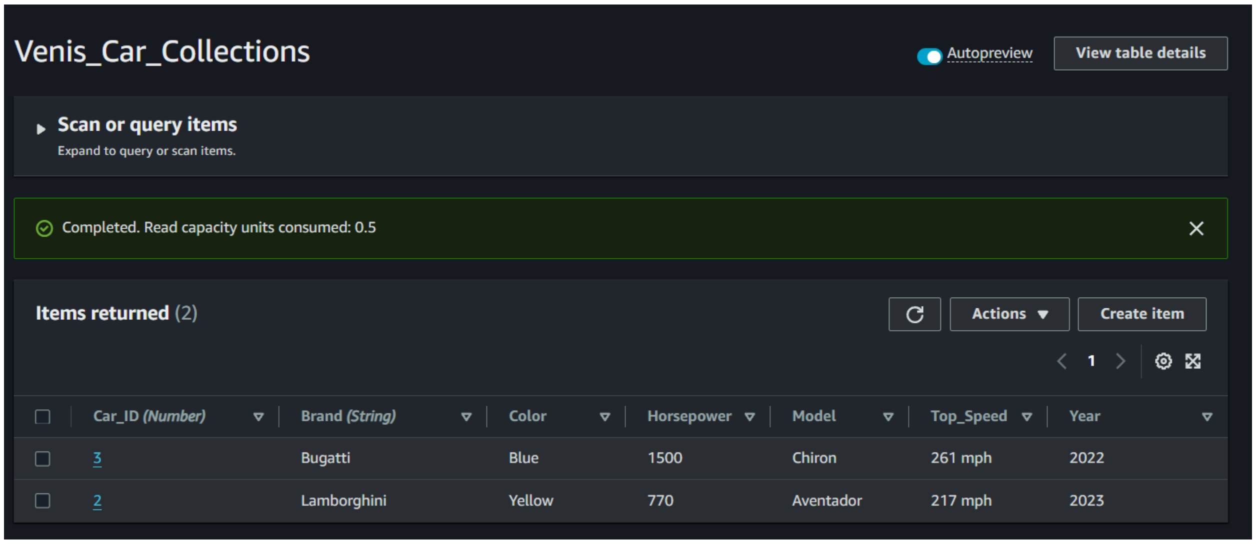
- **4. Delete: Here we are trying to delete car tuple which has Car_ID=1 and Brand=Ferrari .**

```
aws dynamodb delete-item \
    --table-name Venis_Car_Collections \
    --key '{"Car_ID": {"N": "1"}, "Brand": {"S": "Ferrari"}}'
```



```
An error occurred (ValidationException) when calling the DeleteItem operation: The provided key element does not match the schema
[cloudshell-user@ip-10-134-60-212 ~]$ aws dynamodb delete-item \
>     --table-name Venis_Car_Collections \
>     --key '{"Car_ID": {"N": "1"}, "Brand": {"S": "Ferrari"}}'
[cloudshell-user@ip-10-134-60-212 ~]$
```

Here we can see that tuple has been deleted



Venis_Car_Collections

Scan or query items
Expand to query or scan items.

Completed. Read capacity units consumed: 0.5

Items returned (2)

Car_ID (Number)	Brand (String)	Color	Horsepower	Model	Top_Speed	Year
3	Bugatti	Blue	1500	Chiron	261 mph	2022
2	Lamborghini	Yellow	770	Aventador	217 mph	2023