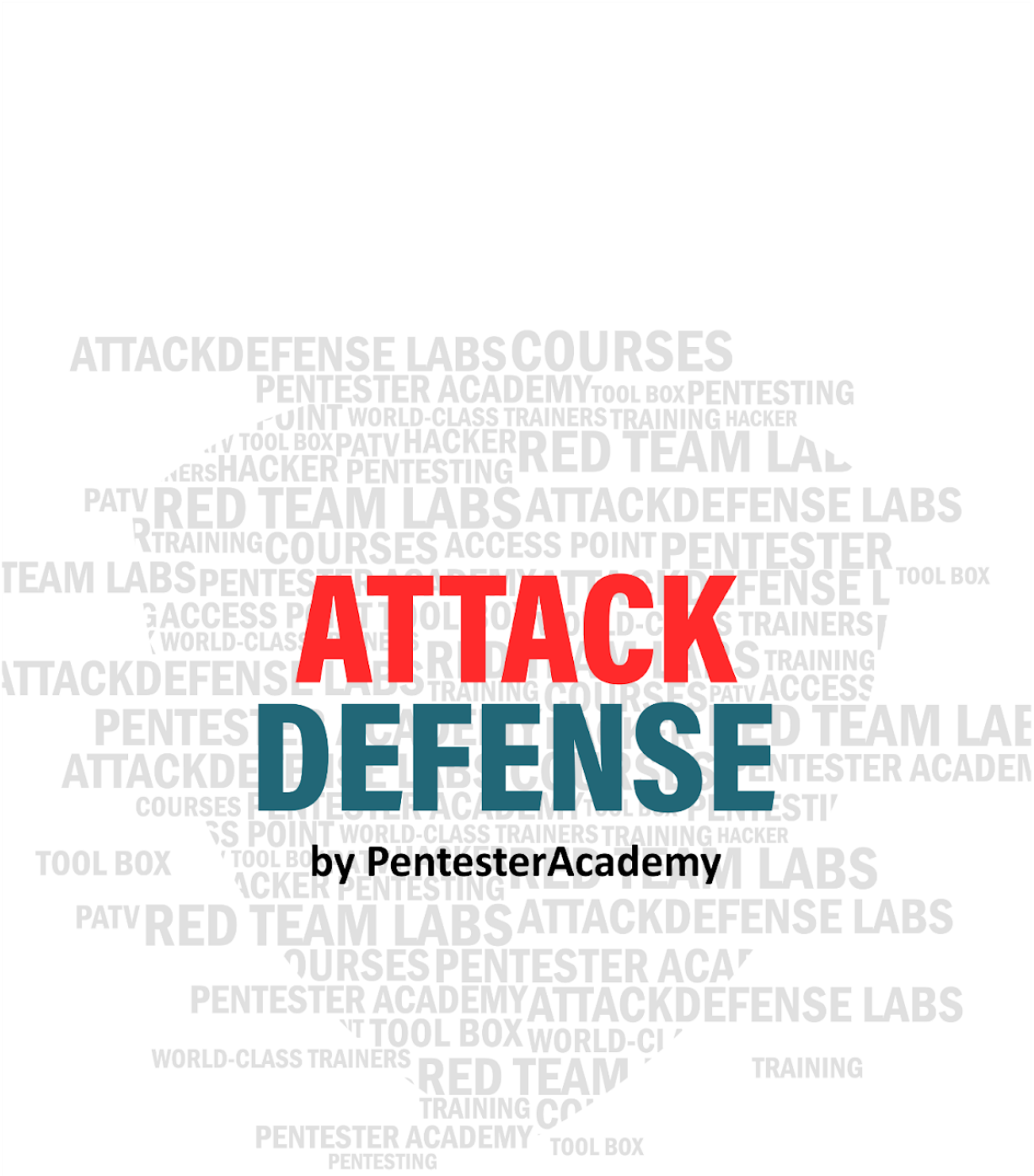
ATTACK
DEFENSE
by PentesterAcademy

Name	DynamoDB Basics II
URL	https://attackdefense.com/challengedetails?cid=1246
Type	Cloud Services : Amazon DynamoDB

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Objective: Interact with the DynamoDB endpoint and perform the following tasks:

1. Identify the data type of the attribute “features” in the items stored in table “products”?
2. Identify the partition key and sort key of table “cards”.
3. Find the value stored in attribute “NameOnCard” in table cards where the primary key has the value “36” and sort key has value “5188-9354-2769-1431”?
4. Find out the number of products which have a rating of more than 4.5?
5. Delete the order with OrderId “15” from the “orders” table.
6. Delete table “orders” from the DynamoDB server.

1) Identify the data type of the attribute “features” in the items stored in table “products”?

Solution:

Step 1: Find the ip address of the ubuntu machine

Command: ip addr

```
root@attackdefense:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
3442: eth0@if3443: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:01:01:08 brd ff:ff:ff:ff:ff:ff link-netnsid 0
```

```
    inet 10.1.1.8/24 brd 10.1.1.255 scope global eth0
       valid_lft forever preferred_lft forever
3445: eth1@if3446: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:f3:b5:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.243.181.2/24 brd 192.243.181.255 scope global eth1
       valid_lft forever preferred_lft forever
root@attackdefense:~#
```

The dynamodb server is running on port 4567 at the IP address 192.243.181.3

Step 2: Scan the products table.

Command: aws --endpoint http://192.243.181.3:4567 dynamodb scan --table products --projection-expression "features" --max-items 1

```
root@attackdefense:~# aws --endpoint http://192.243.181.3:4567 dynamodb scan --table products --projection-expression "features" --ma
x-items 1
{
    "Items": [
        {
            "features": {
                "M": {
                    "color": {
                        "S": "black"
                    },
                    "weight": {
                        "S": "750g"
                    },
                    "size": {
                        "S": "9"
                    }
                }
            }
        }
    ],
    "Count": 10,
    "ScannedCount": 10,
    "ConsumedCapacity": null,
    "NextToken": "eyJFeGNsdXNpdmVTdGFydEtleSI6IG51bGwsICJib3RvX3RydW5jYXRlX2Ftb3VudCI6IDF9"
}
root@attackdefense:~#
```

The data type of the attribute “features” is **Map**.

2) Identify the partition key and sort key of table “cards”.

Solution:

Command: aws --endpoint http://192.243.181.3:4567 dynamodb describe-table --table cards

```
root@attackdefense:~# aws --endpoint http://192.243.181.3:4567 dynamodb describe-table --table cards
{
    "Table": {
        "AttributeDefinitions": [
            {
                "AttributeName": "UserId",
                "AttributeType": "S"
            },
            {
                "AttributeName": "CardNumber",
                "AttributeType": "S"
            }
        ],
        "TableName": "cards",
        "KeySchema": [
            {
                "AttributeName": "UserId",
                "KeyType": "HASH"
            },
            {
                "AttributeName": "CardNumber",
                "KeyType": "RANGE"
            }
        ],
        "TableStatus": "ACTIVE",
        "CreationDateTime": 1569005223.031,
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 5,
```

The attribute "**UserId**" is the Primary key and the attribute "**CardNumber**" is the Range/Sort Key.

3) Find the value stored in attribute "NameOnCard" in table cards where the primary key has the value "36" and sort key has value "5188-9354-2769-1431"?

Solution:

Step 1: The attribute "UserId" is the primary key and the "CardNumber" is the sort key in the cards table. The table can be queried to retrieve the value stored in "NameOnCard" for the corresponding primary key and sort key. Create the JSON required to pass the "UserId" and "CardNumber" attribute as attribute values.

```
{
  ":value1": {"S": "36"},
```

":value2": {"S": "5188-9354-2769-1431"}
}
Save the above json as “attribute_values.json”.

```
root@attackdefense:~# cat attribute_values.json
{
  ":value1": {"S": "36"},
  ":value2": {"S": "5188-9354-2769-1431"}
}
root@attackdefense:~#
```

Step 2: Query the table and find the number of cards owned by the specific user.

Command: aws --endpoint http://192.243.181.3:4567 dynamodb query --table cards --key-condition-expression "UserId = :value1 and CardNumber = :value2" --expression-attribute-values file://attribute_values.json

```
root@attackdefense:~# aws --endpoint http://192.243.181.3:4567 dynamodb query --table cards --key-condition-expression "UserId = :val
ue1 and CardNumber = :value2" --expression-attribute-values file://attribute_values.json
{
    "Items": [
        {
            "UserId": {
                "S": "36"
            },
            "CardNumber": {
                "S": "5188-9354-2769-1431"
            },
            "NameOnCard": {
                "S": "Brynne Guthrie"
            },
            "Expiry": {
                "S": "20-05-17"
            },
            "RegisteredNumber": {
                "S": "0537416572"
            }
        }
    ],
    "Count": 1,
    "ScannedCount": 1,
    "ConsumedCapacity": null
}
root@attackdefense:~#
```

The NameOnCard attribute value is “Brynne Guthrie”.

4) Find out the number of products which have a rating of more than 4.5?

Solution:

Step 1: Check the datatype of rating attribute.

Command: aws --endpoint http://192.243.181.3:4567 dynamodb scan --table products --max-items 1

```
root@attackdefense:~# aws --endpoint http://192.243.181.3:4567 dynamodb scan --table products --max-items 1
{
    "Items": [
        {
            "ProductId": {
                "S": "2"
            },
            "PublishOn": {
                "S": "2018-01-01"
            },
            "ProductName": {
                "S": "Puma Sneakers"
            },
            "ProductDescription": {
                "S": "Puma Shoes at cheap Price"
            },
            "Category": {
                "S": "Footwear"
            },
            "Rating": {
                "N": "4"
            },
            "Price": {
                "S": "145$"
            },
```

The name of the attribute is “Rating” and the datatype is Number.

Step 2: Create a JSON file to store the value of Rating attribute.

```
{
":value":{
        "N":"4.5"
        }
}
```

Save the JSON file as “attribute_value.json”.

```
root@attackdefense:~# cat attribute_value.json
{
        ":value":{
                "N":"4.5"
        }
}
root@attackdefense:~#
```

Step 3: Use the scan subcommand and retrieve the items where the rating is greater than 4.5

Command: aws --endpoint http://192.243.181.3:4567 dynamodb scan --table products --filter-expression "Rating > :value" --expression-attribute-values file://attribute_value.json

```
root@attackdefense:~# aws --endpoint http://192.243.181.3:4567 dynamodb scan --table products --filter-expression "Rating > :value" --
-expression-attribute-values file://attribute_value.json
{
    "Items": [
        {
            "ProductId": {
                "S": "8"
            },
            "PublishOn": {
                "S": "2018-01-01"
            },
            "ProductName": {
                "S": "Gucci Wallet"
            },
            "ProductDescription": {
                "S": "Limited Edition Gucci Wallet"
            },
            "Category": {
                "S": "Purse"
            },
            "Rating": {
                "N": "5"
            },
            "Price": {
                "S": "745$"
            },
            "Reviews": {
                "M": {}
            },
            "SellerId": {
                "S": "2"
            },
            "features": {
                "M": {
                    "weight": {
                        "S": "170g"
                    },
                    "color": {
                        "S": "red"
                    },
                    "warranty": {
                        "S": "5 years"
                    }
                }
            }
```

There are 4 products which have rating greater than 4.5

5) Delete the order with OrderId “15” from the “orders” table.

Solution:

Step 1: Create the JSON required to delete the order with OrderId ‘15’

```
{
    "OrderId": {
        "S": "15"
    }
}
```

Save the above JSON as key.json

```
root@attackdefense:~# cat key.json
{
        "OrderId": {
                "S": "15"
        }
}
root@attackdefense:~#
```

Step 2: Delete the item

Command: aws --endpoint http://192.243.181.3:4567 dynamodb delete-item --table orders --key file://key.json

```
root@attackdefense:~# aws --endpoint http://192.243.181.3:4567 dynamodb delete-item --table orders --key file://key.json
root@attackdefense:~#
root@attackdefense:~#
```

Step 3: Verify that the item was deleted.

Command: aws --endpoint http://192.243.181.3:4567 dynamodb get-item --table orders --key file://key.json

```
root@attackdefense:~# aws --endpoint http://192.243.181.3:4567 dynamodb get-item --table orders --key file://key.json
root@attackdefense:~#
root@attackdefense:~#
```

Since no output was returned, it can be concluded that the item was deleted.

6) Delete table “orders” from the DynamoDB server.

Solution:

Step 1: Delete the table.

Command: aws --endpoint http://192.243.181.3:4567 dynamodb delete-table --table orders

```
root@attackdefense:~# aws --endpoint http://192.243.181.3:4567 dynamodb delete-table --table orders
{
    "TableDescription": {
        "AttributeDefinitions": [
            {
                "AttributeName": "OrderId",
                "AttributeType": "S"
            }
        ],
        "TableName": "orders",
        "KeySchema": [
            {
                "AttributeName": "OrderId",
                "KeyType": "HASH"
            }
        ],
        "TableStatus": "DELETING",
        "CreationDateTime": 1569005086.548,
        "ProvisionedThroughput": {
            "NumberOfDecreasesToday": 0,
            "ReadCapacityUnits": 5,
            "WriteCapacityUnits": 5
        },
        "TableSizeBytes": 0,
        "ItemCount": 0,
        "TableArn": "arn:aws:dynamodb:us-east-1:000000000000:table/orders",
        "TableId": "9fc66df4-d2c8-4d6d-3aab-4bfd4d21"
    }
}
```

Step 2: Check whether the table was deleted.

Command: aws --endpoint http://192.243.181.3:4567 dynamodb list-tables

```
root@attackdefense:~# aws --endpoint http://192.243.181.3:4567 dynamodb list-tables
{
    "TableNames": [
        "cards",
        "products",
        "sellers",
        "users"
    ]
}
root@attackdefense:~#
```

The table “orders” is no longer listed. Therefore, the table was deleted successfully.

References:

1. AWS CLI Reference DynamoDB (https://docs.aws.amazon.com/cli/latest/reference/dynamodb/index.html)