Project 2 - AWS CLI & DynamoDB

Objectives

- Install AWS CLI
- Run DynamoDB Locally
- Issue AWS CLI commands
- Execute DynamoDB commands

Project Overview

Your web application needs to have data. You have decided that you want to use a NoSQL database and specifically you have decided to use AWS DynamoDB. In order to minimize costs, you are setting up your development machine with a local instance of DynamoDB in order to avoid billing charges that would be incurred during the development phase if you were running your development database on AWS.

Project Details

Install DynamoDB Locally & Seed w/ Data

Install the AWS CLI. After installation, run the aws configure command to load valid credentials for the aws cli.

Install a local instance of DynamoDB. In order to complete this task, it may be necessary to install the pre-req components of the AWS CLI and a Java Runtime Environment (JRE)

Create the BaseballStats table (both locally and on AWS). List the commands you used below: (5 points)

```
aws dynamodb create-table --table-name BaseballStats --attribute-definitions AttributeName=TeamID,AttributeType=S AttributeName=SK,AttributeType=S --key-schema AttributeName=TeamID,KeyType=HASH AttributeName=SK,KeyType=RANGE --provisioned-throughput ReadCapacityUnits=5,WriteCapacityUnits=5 --endpoint-url http://localhost:8000

aws dynamodb create-table --table-name BaseballStats --attribute-definitions AttributeName=TeamID,AttributeType=S AttributeName=SK,AttributeType=S --key-schema AttributeName=TeamID,KeyType=HASH AttributeName=SK,KeyType=RANGE --provisioned-throughput ReadCapacityUnits=5,WriteCapacityUnits=5
```

Seed the BaseballStats local instance with teams, players, and games. List the commands you used below: (5 points)

```
aws dynamodb batch-write-item --request-items file://teams.json --endpoint-url http://localhost:8000
aws dynamodb batch-write-item --request-items file://playerss.json --endpoint-url http://localhost:8000
aws dynamodb batch-write-item --request-items file://games.json --endpoint-url http://localhost:8000
```

Scan the BaseballStats local instance. List the command you used below: (5 points)

```
aws dynamodb scan --table-name BaseballStats --endpoint-url http://localhost:8000
```

Query the BaseballStats local instance to see only the players. List the command you used below: (5 points)

```
aws dynamodb query --key-condition-expression "TeamID = :t" --expression-attribute-values '{":t": {"S":
"PLAYERINFO_LAA"}}' --table-name BaseballStats --endpoint-url http://localhost:8000
```

DynamoDB Planning and Structure

In addition to the BaseballStats data, you have been tasked with storing customer information. Customers are currently scheduled to consume the BaseballStats data, but your company is looking to expand into other sports as well. Ultimately, your company's goal is to have an equal distribution of customers across 5 different sports. You have made the decision that the Customers table will be a separate table from your sports data.

For your customers table, what would be an appropriate partition key and sort key? Please explain your choices: (5 points)

partition key - customer ID
Unique attribute that identifies each table item.

sort key - customer sport

allows querying of customer data by sport

Tests and Verification

Can I visually see the BaseballStats table on AWS?

Am I able to see the results from scanning and querying the BaseballStats table?

Do your AWS CLI commands work both locally and on the AWS site?

What did I learn?

NoSQL database structure

How to access NoSQL data

Use AWS CLI to issue AWS commands