Exercise: DynamoDB - Java

In this exercise you will:

1. Create a DynamoDB table named “follows” that keeps track of which users are following each other (as in Twitter).
2. Create a secondary index named “follows_index” on the “follows” table.
3. Write a program that uses the AWS SDK to:
    a. Insert, update, and delete items in the “follows” table
    b. Query the “follows” table with and without pagination
    c. Query the “follows_index” index with and without pagination

Assumptions

You have already configured your AWS credentials on your computer. This should have been done when you installed the AWS CLI on your computer.

Intellij is installed on your computer.

Steps

1. Go the the AWS DynamoDB web console

2. Create a table named “follows”.
    a. Partition Key: a string named “follower_handle”. This attribute stores the Twitter handle of the user who is following another user.
    b. Sort Key: a string named “followee_handle”. This attribute stores the Twitter handle of the user who is being followed.
    c. This Primary Key lets you query the table by “follower_handle” and sort the results by “followee_handle”.

3. Select the “Indexes” tab and create an index named “follows_index”.
    a. Partition Key: a string named “followee_handle. This is the same “followee_handle” attribute you created in the previous step.
    b. Check “Add sort key”
    c. Sort Key: a string named “follower_handle”. This is the same “follower_handle” attribute you created in the previous step.
    d. This Primary Key lets you query the index by “followee_handle” and sort the results by “follower_handle”.
    e. For “Projected attributes”, keep the “All” setting.

4. Next, you will write a Java program that performs operations on the “follows” table.
5. Create a directory for this exercise

6. Create a new Intellij project

7. In the new project, add dependencies on the AWS SDK Java library
    a. Select the File -> Project Structure menu
    b. Select “Modules” on the left side of the “Project Structure” dialog
    c. Select the “Dependencies” tab
    d. Click the “plus” icon in the top-right

e. Select “Library”
f. Select “From Maven”
g. Enter “com.amazonaws:aws-java-sdk-core:1.11.547”, and click OK. This will add a dependency on the core AWS Java SDK library.
h. Using the same process, add a Maven dependency on “com.amazonaws:aws-java-sdk-dynamodb:1.11.547”. This will add a dependency on the DynamoDB portion of the AWS Java SDK library.

8. Write Java code to put, get, update, and delete items in the “follows” table. The examples on this page demonstrate how to put, get, update, and delete items in a DynamoDB table: https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GettingStarted.Java.03.html. Also, the complete online documentation for the AWS Java SDK can be found here: https://docs.aws.amazon.com/AWSJavaSDK/latest/javadoc/index.html

    a. “Put” 100 items into the “follows” table with the following attributes:
        i. “follower_handle”: string handle of the user who is following (e.g., “@FredFlintstone”)
        ii. “follower_name”: string name of the user who is following (e.g., “Fred Flintstone”)
        iii. “followee_handle”: string handle of the user being followed (e.g., “@ClintEastwood”)
        iiii. “followee_name”: string name of the user being followed (e.g., “Clint Eastwood”)
        v. In the items that you create, make sure that each user is following many other users, and that each user is being followed by many other users.

    b. “Get” one of the items from the “follows” table using its primary key

    c. “Update” the “follower_name” and “followee_name” attributes of one of the items in the “follows” table

    d. “Delete” one of the items in the “follows” table using its primary key

9. Write Java code to query the “follows” table and the “follows_index” index. The example on this page demonstrates how to query a table: https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GettingStarted.Java.04.html.

    a. “Query” the “follows” table to return all of the users being followed by a user, sorted by “followee_handle”
        i. Hint: Call QuerySpec.withScanIndexForward(true) to sort the result in ascending order

    b. “Query” the “follows_index” index to return all of the users following a user, reverse sorted by “follower_handle”
        i. Hint: Call Table.getIndex to get a reference to the Index object so the index can be queried
        ii. Hint: Call QuerySpec.withScanIndexForward(false) to sort the result in descending order

10. Make copies of the previous two queries, and modify them to return the query results in pages of size ten (i.e., ten at a time)
    a. This means you will need to re-execute the query multiple times in order to fetch all of the results.
    b. Hint: Call withMaxPageSize() on the QuerySpec to tell it how many items to return in each page
    c. Hint: Call withExclusiveStartKey() on the QuerySpec to tell DynamoDB the

c. Hint: Call withExclusiveStartKey() on the QuerySpec to tell DynamoDB the primary key value of the last item returned in the previous page (this is not necessary for the first page). This is how DynamoDB knows where the next page should start.
d. Hint: Here's how you get the primary key value for the last item returned in the current page:
    i. On the ItemCollection<QueryOutcome> object returned by the query, call getLastLowLevelResult() to get a reference to the QueryOutcome.
    ii. Call getQueryResult() on the QueryOutcome to get a reference to the QueryResult.
    iii. Call getLastEvaluatedKey() on the QueryResult to get the last item's primary key value. If this value is non-null, there is more data to retrieve.

11. Submit your Java code through Canvas.

More In formation

The full AWS SDK documentation can be found here in the SDKs section:
https://aws.amazon.com/tools/