



AWS DYNAMODB SERVICE EXPIREMENT

CSCI 5410 – Serverless Data Processing
Assignment 1 – Part C

Dhrumil Amish Shah (B00857606)
dh416386@dal.ca

Screenshots of the DynamoDB service and operations

Figure 1 shows the screenshot of the **Super_Volcanos [1]** table structure setup to be created using the **Create table** functionality of the DynamoDB service.

The screenshot displays the AWS Management Console for the 'Create table' operation in DynamoDB. The interface is divided into several sections:

- Table details:**
 - Table name:** 'Super_Volcanos' (with a note: 'Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.)').
 - Partition key:** 'ID' (String type, with a note: '1 to 255 characters and case sensitive.').
 - Sort key - optional:** 'Enter the sort key name' (String type, with a note: '1 to 255 characters and case sensitive.').
- Settings:**
 - Default settings:** Selected. 'The fastest way to create your table. You can modify these settings now or after your table has been created.'
 - Customize settings:** Unselected. 'Use these advanced features to make DynamoDB work better for your needs.'
- Default settings:**
 - Read/write capacity:** 'Using provisioned capacity mode. Read and write capacity are set to 5 units each.'
 - Secondary indexes:** 'No secondary indexes have been created. Queries will be run by using the table's partition key and sort key only.'
 - Key management for encryption at rest:** 'Using the AWS owned customer master key. This key is managed by DynamoDB at no extra cost.'
- Tags:**
 - Key: 'owner' (Value: 'Dhrumil Amish shah') - Remove
 - Key: 'email' (Value: 'dh416386@dal.ca') - Remove
 - Key: 'university' (Value: 'Dalhousie University') - Remove
 - Add new tag:** (You can add 47 more tags.)

A blue information box at the bottom states: 'This table will be created with auto scaling disabled. You do not have permissions to enable auto scaling.'

At the bottom right, there are 'Cancel' and 'Create table' buttons.

Figure 1: Screenshot of Super_Volcanos table structure [2]

Figure 2 shows the screenshot of the **Super_Volcanos** table created successfully using the DynamoDB service linked to my Amazon account.

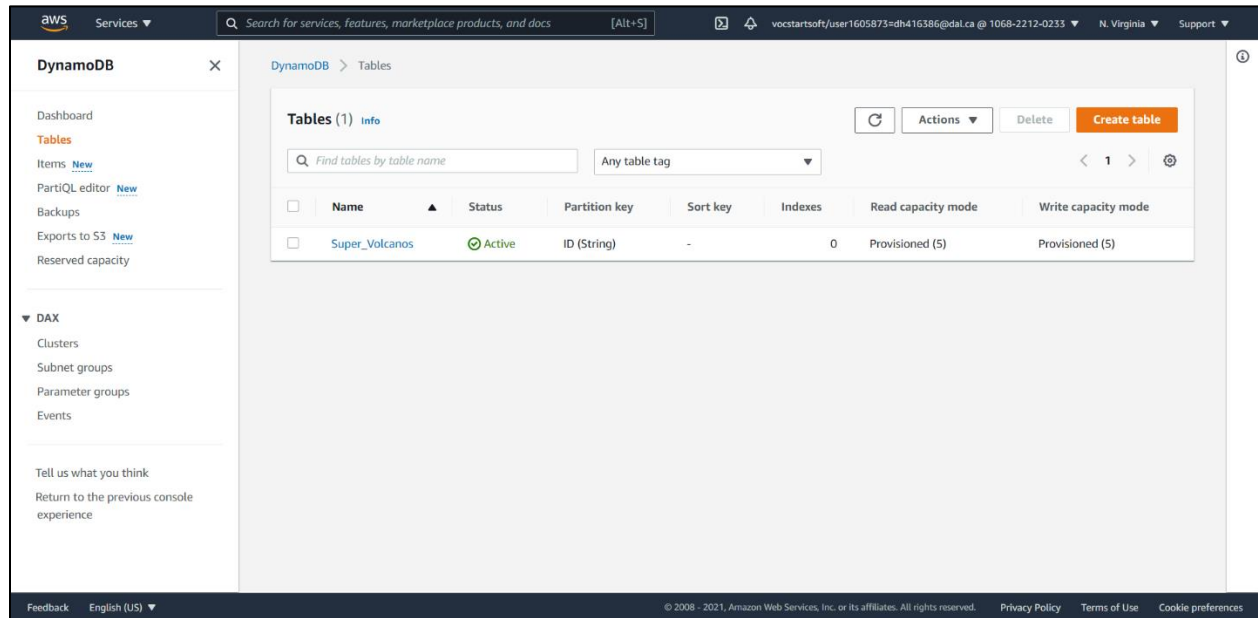


Figure 2: Screenshot of successful creation of table Super_Volcanos [2]

Figure 3 shows the screenshot of the empty **Super_Volcanos** table (i.e., zero items) linked to my Amazon account.

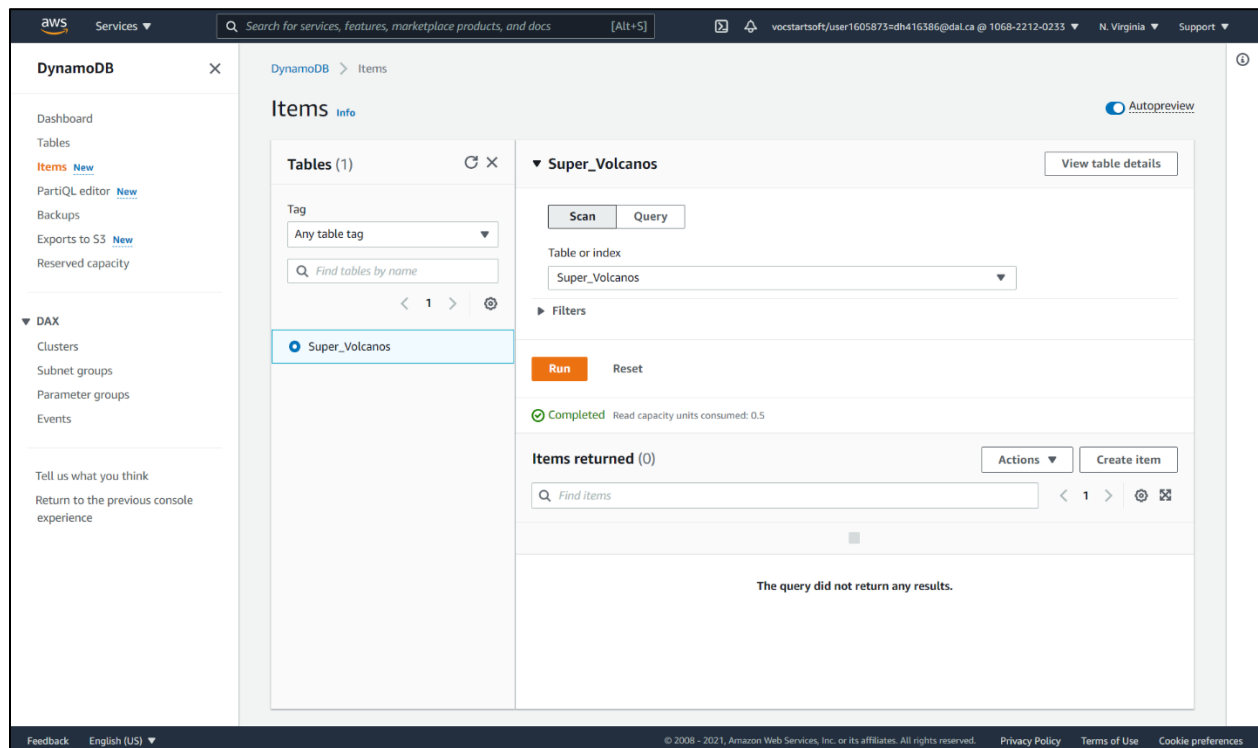


Figure 3: Screenshot of empty Super_Volcanos table [2]

Figure 4 shows the screenshot of the console when no entries in the table **Super_Volcanos** exists. (i.e., Empty DynamoDB table). – Output 1.

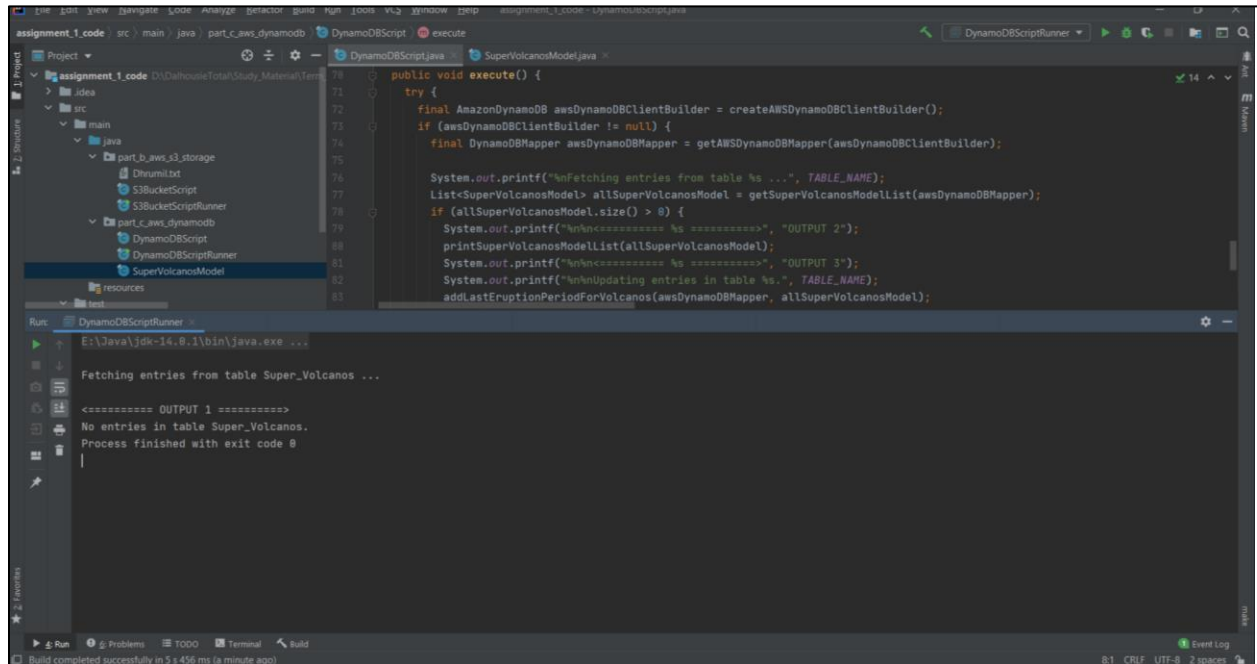


Figure 4: Screenshot of “No entries in the table Super_Volcanos.” message using AWS SDK for JAVA (Output 1)

Figure 5 shows the screenshot of inserted documents in the table **Super_Volcanos** linked to my Amazon account.

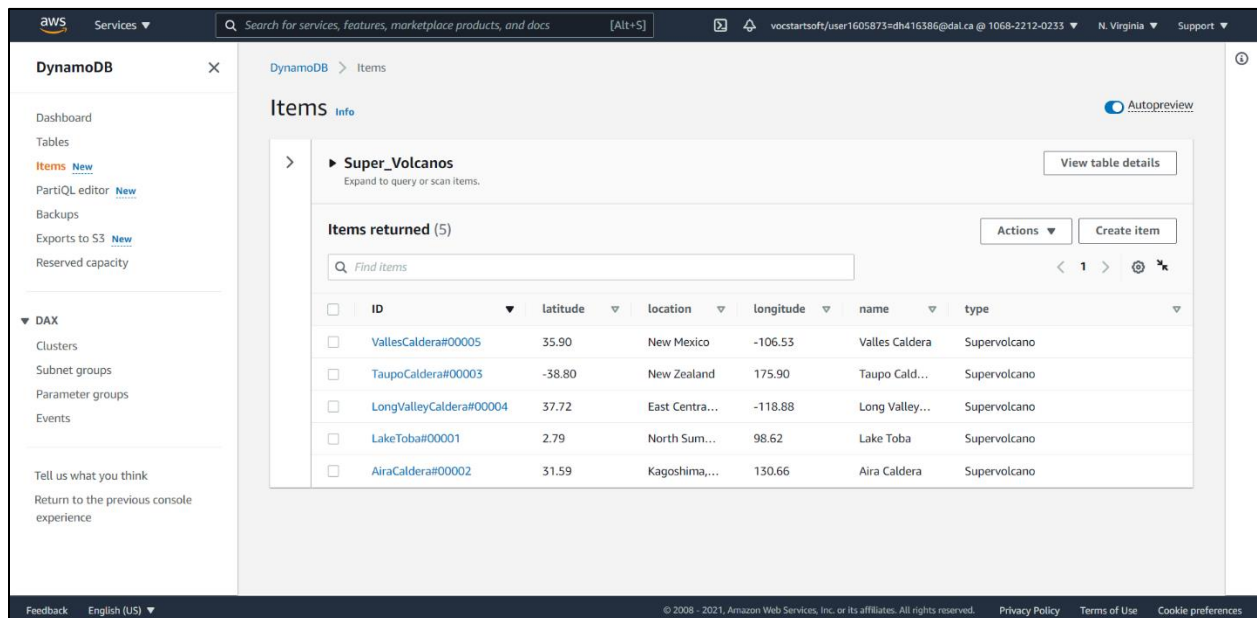
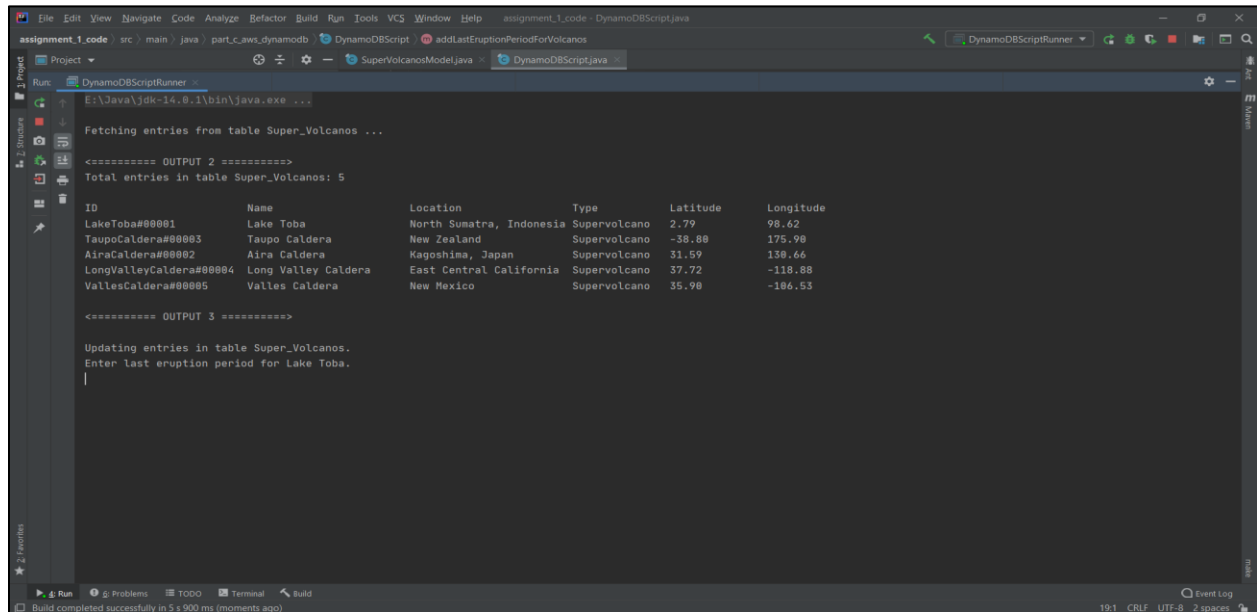


Figure 5: Screenshot of inserted documents in the table Super_Volcanos [2]

Figure 6 shows the screenshot of the console when documents are fetched from the table **Super_Volcanos** and displayed in a tabular manner. – Output 2



```

Fetch entries from table Super_Volcanos ...
===== OUTPUT 2 =====
Total entries in table Super_Volcanos: 5

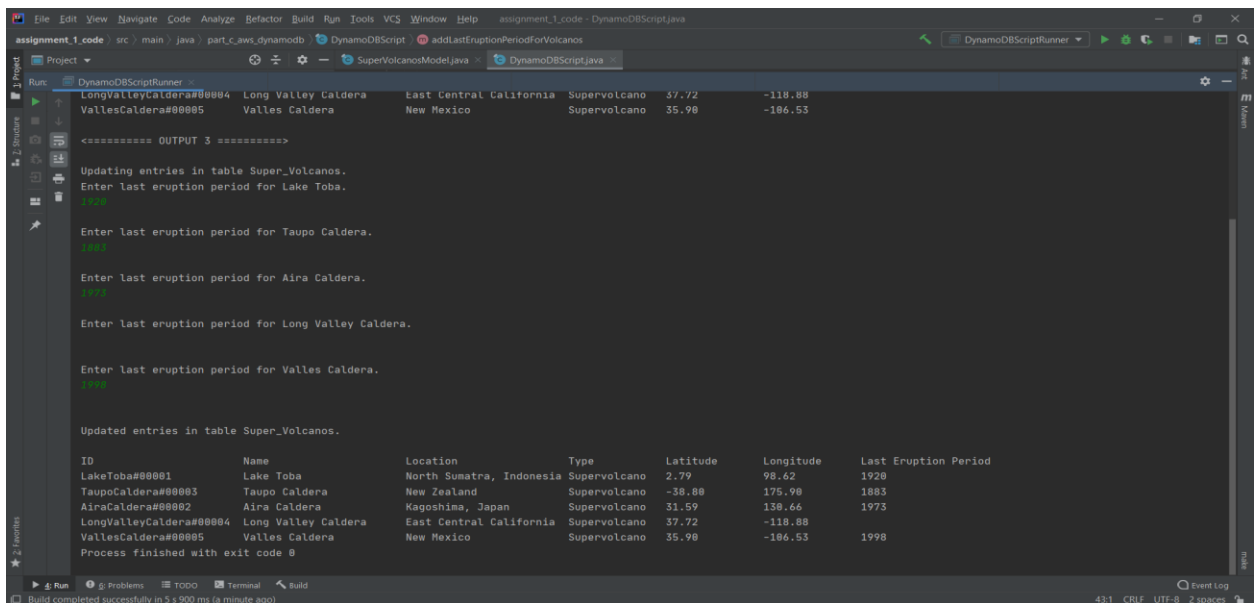
ID          Name          Location          Type          Latitude  Longitude
LakeToba#00001  Lake Toba    North Sumatra, Indonesia  Supervolcano  2.79      98.62
TaupoCaldera#00003  Taupo Caldera  New Zealand    Supervolcano  -38.88    175.98
AiraCaldera#00002  Aira Caldera   Kagoshima, Japan  Supervolcano  31.59     130.66
LongValleyCaldera#00004  Long Valley Caldera  East Central California  Supervolcano  37.72     -118.88
VallesCaldera#00005  Valles Caldera  New Mexico      Supervolcano  35.98     -106.53

===== OUTPUT 3 =====
Updating entries in table Super_Volcanos.
Enter last eruption period for Lake Toba.
|

```

*Figure 6: Screenshot of the documents fetched from the table **Super_Volcanos** and displayed in the console (Output 2)*

Figure 7 shows the screenshot of the console where a new item is added to all the documents in the table **Super_Volcanos**, which is **last_eruption_period**. For super volcanos with no last eruption period, the field is kept empty. Documents are fetched with the newly inserted field – last_eruption_period. – Output 3



```

LongValleyCaldera#00004  Long Valley Caldera  East Central California  Supervolcano  37.72     -118.88
VallesCaldera#00005  Valles Caldera      New Mexico              Supervolcano  35.98     -106.53

===== OUTPUT 3 =====
Updating entries in table Super_Volcanos.
Enter last eruption period for Lake Toba.
1920

Enter last eruption period for Taupo Caldera.
1883

Enter last eruption period for Aira Caldera.
1973

Enter last eruption period for Long Valley Caldera.
1988

Enter last eruption period for Valles Caldera.
1998

Updated entries in table Super_Volcanos.

ID          Name          Location          Type          Latitude  Longitude  Last Eruption Period
LakeToba#00001  Lake Toba    North Sumatra, Indonesia  Supervolcano  2.79      98.62      1920
TaupoCaldera#00003  Taupo Caldera  New Zealand    Supervolcano  -38.88    175.98     1883
AiraCaldera#00002  Aira Caldera   Kagoshima, Japan  Supervolcano  31.59     130.66     1973
LongValleyCaldera#00004  Long Valley Caldera  East Central California  Supervolcano  37.72     -118.88    1988
VallesCaldera#00005  Valles Caldera  New Mexico      Supervolcano  35.98     -106.53    1998
Process finished with exit code 0

```

Figure 7: Screenshot of the console where a new item - last_eruption_period is added to all the documents. (Output – 3)

Figure 8 shows the screenshot of the final Super_Volcanos table linked to my Amazon account.

The screenshot shows the AWS DynamoDB console interface. On the left is a navigation menu with options like Dashboard, Tables, Items, PartiQL editor, Backups, Exports to S3, and Reserved capacity. The main area displays the 'Items' view for the 'Super_Volcanos' table. It shows a list of 5 items with columns for ID, last eruption year, latitude, longitude, location, name, and type. Each item is a supervolcano with a unique ID and specific geographical and historical data.

ID	last_eru...	latitude	location	longitude	name	type
LakeToba#00001	1920	2.79	North Sum...	98.62	Lake Toba	Supervolcano
TaupoCaldera#00003	1883	-38.80	New Zealand	175.90	Taupo Cald...	Supervolcano
AiraCaldera#00002	1973	31.59	Kagoshima,...	130.66	Aira Caldera	Supervolcano
LongValleyCaldera#00004	37.72		East Centra...	-118.88	Long Valley...	Supervolcano
VallesCaldera#00005	1998	35.90	New Mexico	-106.53	Valles Caldera	Supervolcano

Figure 8: Screenshot of the final Super_Volcanos table [2]

Program Script

The program code consists of three Java files namely SuperVolcanosModel.java, DyanamoDBScript.java, and DynamoDBScriptRunner.java.

Filename: SuperVolcanosModel.java [3]

This JAVA program is a model(structure) for an individual document in the table Super_Volcanos. Below is the code for this file.

```
package part_c_aws_dynamodb;

import com.amazonaws.services.dynamodbv2.datamodeling.DynamoDBAttribute;
import com.amazonaws.services.dynamodbv2.datamodeling.DynamoDBHashKey;
import com.amazonaws.services.dynamodbv2.datamodeling.DynamoDBTable;

@DynamoDBTable(tableName = "Super_Volcanos")
public final class SuperVolcanosModel {
    private String id;
    private String name;
    private String location;
    private String type;
    private String lat;
    private String lng;
    private String lastEruptionPeriod;

    @DynamoDBHashKey(attributeName = "ID")
```

```
public String getId() {
    return id;
}

public void setId(final String id) {
    this.id = id;
}

@DynamoDBAttribute(attributeName = "name")
public String getName() {
    return name;
}

public void setName(final String name) {
    this.name = name;
}

@DynamoDBAttribute(attributeName = "location")
public String getLocation() {
    return location;
}

public void setLocation(final String location) {
    this.location = location;
}

@DynamoDBAttribute(attributeName = "type")
public String getType() {
    return type;
}

public void setType(final String type) {
    this.type = type;
}

@DynamoDBAttribute(attributeName = "latitude")
public String getLat() {
    return lat;
}

public void setLat(String lat) {
    this.lat = lat;
}

@DynamoDBAttribute(attributeName = "longitude")
public String getLng() {
    return lng;
}

public void setLng(String lng) {
    this.lng = lng;
}

@DynamoDBAttribute(attributeName = "last_eruption_period")
public String getLastEruptionPeriod() {
    return lastEruptionPeriod;
}
```

```

}

public void setLastEruptionPeriod(final String lastEruptionPeriod) {
    this.lastEruptionPeriod = lastEruptionPeriod;
}
}

```

Filename: DyanmoDBScript.java [3]

This JAVA program fetches all the documents from the table Super_Volcanos linked to my Amazon account and inserts a new item to all the documents in the table Super_Volcanos which is last_eruption_period. For achieving this, I made use of AWS SDK for JAVA. Below is the code for this file.

```

package part_c_aws_dynamodb;

import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.dynamodbv2.AmazonDynamoDB;
import com.amazonaws.services.dynamodbv2.AmazonDynamoDBClientBuilder;
import com.amazonaws.services.dynamodbv2.datamodeling.DynamoDBMapper;
import com.amazonaws.services.dynamodbv2.datamodeling.DynamoDBMapperConfig;
import com.amazonaws.services.dynamodbv2.datamodeling.DynamoDBScanExpression;

import java.util.List;
import java.util.Scanner;

public final class DynamoDBScript {
    private static final String AWS_ACCESS_KEY = "<AWS_ACCESS_KEY>";
    private static final String AWS_SECRET_KEY = "<AWS_SECRET_KEY>";
    private static final String AWS_SESSION_TOKEN = "<AWS_SESSION_TOKEN>";
    private static final BasicSessionCredentials AWS_CREDENTIALS = new
BasicSessionCredentials(AWS_ACCESS_KEY, AWS_SECRET_KEY, AWS_SESSION_TOKEN);
    private static final String TABLE_NAME = "Super_Volcanos";

    private void addLastEruptionPeriodForVolcanos(final DynamoDBMapper awsDynamoDBMapper,
        final List<SuperVolcanosModel> allSuperVolcanosModel) {
        final Scanner scanner = new Scanner(System.in);
        for (final SuperVolcanosModel superVolcanosModel : allSuperVolcanosModel) {
            System.out.printf("%nEnter last eruption period for %s.%n", superVolcanosModel.getName());
            final String lastEruptionPeriod = scanner.nextLine();
            superVolcanosModel.setLastEruptionPeriod(lastEruptionPeriod);
            awsDynamoDBMapper.save(superVolcanosModel);
        }
    }

    private void printSuperVolcanosModelList(final List<SuperVolcanosModel> allSuperVolcanosModel) {
        if (allSuperVolcanosModel.size() > 0) {
            System.out.printf("%nTotal entries in table %s: %d", TABLE_NAME, allSuperVolcanosModel.size());
            System.out.printf("%n%-25s%-25s%-25s%-15s%-15s%-15s", "ID", "Name", "Location", "Type",
"Latitude", "Longitude");
            for (final SuperVolcanosModel superVolcanosModel : allSuperVolcanosModel) {
                final String id = superVolcanosModel.getId();
                final String name = superVolcanosModel.getName();

```



```

        final String location = superVolcanosModel.getLocation();
        final String type = superVolcanosModel.getType();
        final String latitude = superVolcanosModel.getLat();
        final String longitude = superVolcanosModel.getLng();
        System.out.printf("%n%-25s%-25s%-25s%-15s%-15s%-15s", id, name, location, type, latitude, longitude);
    }
} else {
    System.out.printf("%nNo entries in table %s.", TABLE_NAME);
}
}

private List<SuperVolcanosModel> getSuperVolcanosModelList(final DynamoDBMapper
awsDynamoDBMapper) {
    final DynamoDBScanExpression scanExpression = new DynamoDBScanExpression();
    return awsDynamoDBMapper.scan(SuperVolcanosModel.class, scanExpression);
}

private DynamoDBMapper getAWSDynamoDBMapper(final AmazonDynamoDB awsDynamoDBClientBuilder)
{
    final DynamoDBMapperConfig mapperConfig = new DynamoDBMapperConfig
        .Builder()
        .withTableNameOverride(DynamoDBMapperConfig
            .TableNameOverride
            .withTableNameReplacement(TABLE_NAME)).build();
    return new DynamoDBMapper(awsDynamoDBClientBuilder, mapperConfig);
}

private AmazonDynamoDB createAWSDynamoDBClientBuilder() {
    return AmazonDynamoDBClientBuilder.standard()
        .withCredentials(new AWSStaticCredentialsProvider(AWS_CREDENTIALS))
        .withRegion(Regions.US_EAST_1)
        .build();
}

public void execute() {
    try {
        final AmazonDynamoDB awsDynamoDBClientBuilder = createAWSDynamoDBClientBuilder();
        if (awsDynamoDBClientBuilder != null) {
            final DynamoDBMapper awsDynamoDBMapper =
getAWSDynamoDBMapper(awsDynamoDBClientBuilder);

            System.out.printf("%nFetching entries from table %s ...", TABLE_NAME);
            final List<SuperVolcanosModel> allSuperVolcanosModel =
getSuperVolcanosModelList(awsDynamoDBMapper);
            if (allSuperVolcanosModel.size() > 0) {
                System.out.printf("%n%n<===== %s =====>", "OUTPUT 2");
                printSuperVolcanosModelList(allSuperVolcanosModel);
                System.out.printf("%n%n<===== %s =====>", "OUTPUT 3");
                System.out.printf("%n%nUpdating entries in table %s.", TABLE_NAME);
                addLastEruptionPeriodForVolcanos(awsDynamoDBMapper, allSuperVolcanosModel);

                System.out.printf("%n%nUpdated entries in table %s.", TABLE_NAME);
                System.out.printf("%n%n%-25s%-25s%-25s%-15s%-15s%-15s%-30s", "ID", "Name", "Location", "Type",
"Latitude", "Longitude", "Last Eruption Period");
                for (final SuperVolcanosModel superVolcanosModel : allSuperVolcanosModel) {
                    final String id = superVolcanosModel.getId();

```

```

        final String name = superVolcanosModel.getName();
        final String location = superVolcanosModel.getLocation();
        final String type = superVolcanosModel.getType();
        final String latitude = superVolcanosModel.getLat();
        final String longitude = superVolcanosModel.getLng();
        final String lastErrPeriod = superVolcanosModel.getLastEruptionPeriod();
        System.out.printf("%n%-25s%-25s%-25s%-15s%-15s%-15s%-30s", id, name, location, type, latitude,
longitude, lastErrPeriod);
    }
    } else {
        System.out.printf("%n%n<===== %s =====>", "OUTPUT 1");
        printSuperVolcanosModelList(allSuperVolcanosModel);
    }
    } else {
        System.err.println("\nError creating AWS DynamoDB client builder instance!");
    }
    } catch (final Exception e) {
        System.err.println(e.getMessage());
    }
    }
}

```

Filename: DynamoDBScriptRunner.java

This JAVA program is a runner program for the DyanmoDBScript.JAVA file. Below is the code for this file.

```

package part_c_aws_dynamodb;

public final class DynamoDBScriptRunner {
    public static void main(String[] args) {
        final DynamoDBScript dynamoDBScript = new DynamoDBScript();
        dynamoDBScript.execute();
    }
}

```

Link to the Gitlab Repository

https://git.cs.dal.ca/dashah/csci-5410-f2021-b00857606-dhrumil-amish-shah/-/tree/main/assignment_1_code

Link to package part_c_aws_dynamodb:

https://git.cs.dal.ca/dashah/csci-5410-f2021-b00857606-dhrumil-amish-shah/-/tree/main/assignment_1_code/src/main/java/part_c_aws_dynamodb

References

- [1] arcgis, "Supervolcanoes and Notable Volcanic Eruptions in History," [Online]. Available: <https://www.arcgis.com/apps/MapJournal/index.html?appid=a546b46a7fb942008455e072c69ea767>. [Accessed 23 September 2021].
- [2] Amazon and AWS, "Cloud Services - Amazon Web Services (AWS)," Amazon, [Online]. Available: <https://aws.amazon.com/>. [Accessed 23 September 2021].
- [3] Amazon and AWS, "DynamoDB Examples Using the AWS SDK for Java," [Online]. Available: <https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/examples-dynamodb.html>. [Accessed 18 September 2021].