Day 33 - 13th sept 2025

Creating Cluster using AWS cloud..

Done by

5 +

Carry forward from yesterday's doc - Juz FYI…

Task 01:

Creating a DAX cluster in AWS (using Console).

Requirements for resting cluster.

→ you need to have an AWs account … with permissions for DynamoDB and DAX.

—> create a table (old table also works

—> you need to create VPC and subnet

—> IAM Role will be used by DAX for Access to DynamoDB

Open DAX console

→ sign in to AWS management console

→ search for DynamoDB —> check for DAX cluster (left plane)

→ create cluster..

Configure Cluster settings:

→ Cluster name : give any name of your choice

→ Node type : dax.r5.large

→ cluster size : number of Nodes (1 to 3 for prod purpose)

→ IAM role:select the IAM role… (it has grants for AmazonDynamoDbFullAccess or create a custom role by giving the dynamoDb permissions..

Network settings:

VPC: plz select VPC the dynamoDB app is in

Subnets: you need to choose at least one subnet

Security groups : add security groups … which will allow the traffic from our app ECS

Or Lambda Functions.

Encryption and Security (this is not compulsory)

Encryption : we need to enable the encryption at rest ( not compulsory)

Parameter group : leave the default values as is..

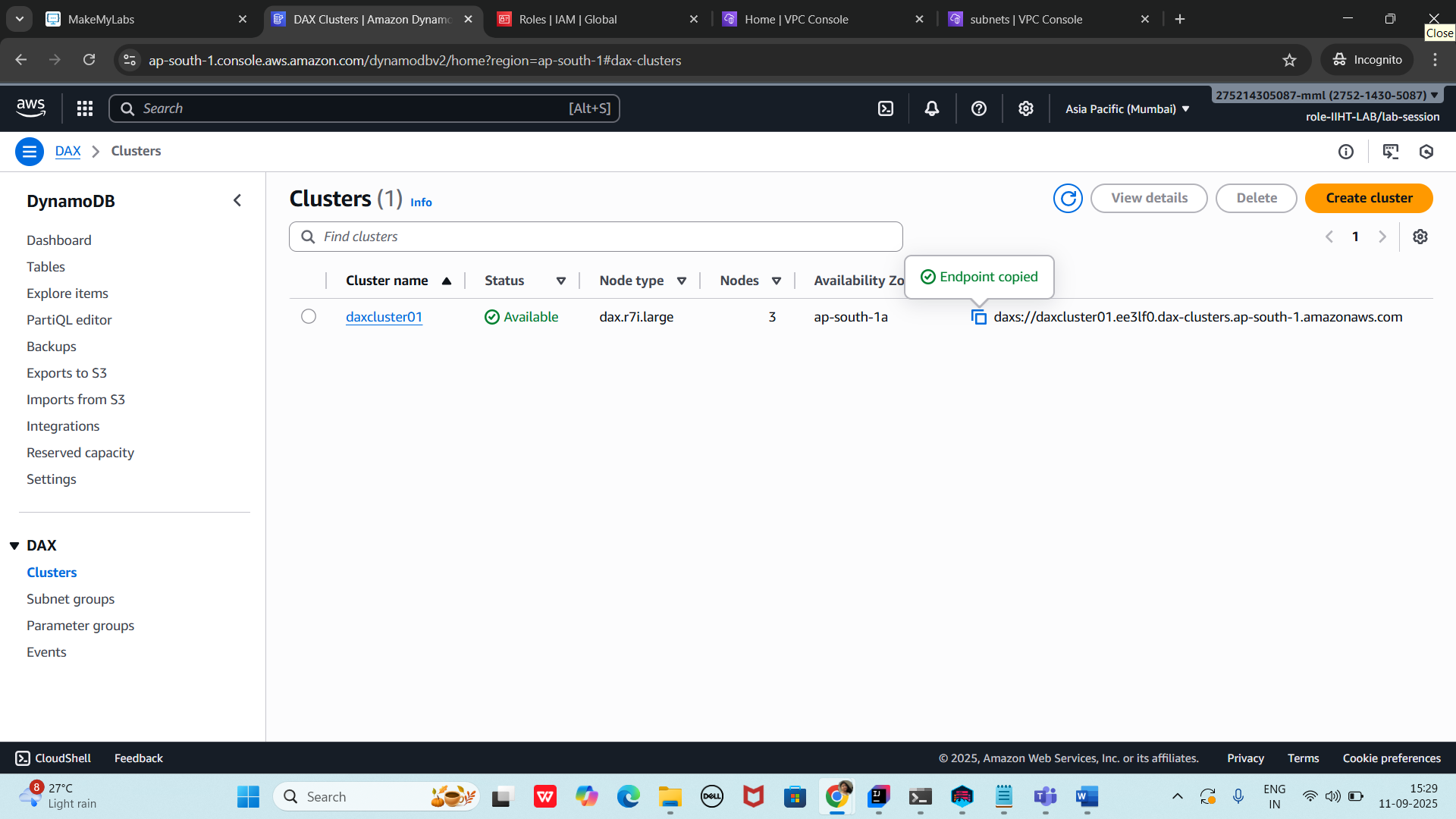
TTL

Cacheing ….

Create it…. Try to review all the settings that you have given …

In the below screen shot you can see the status of my cluster as available

And also DAx endpoint copy from here and paste it in your code… instead of standard DynamoDB SDk



Task 02:

package org.example.Accelerator\_DAX;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;

import software.amazon.awssdk.regions.Region;

import software.amazon.awssdk.services.dax.DaxClient;

import software.amazon.awssdk.services.dynamodb.model.AttributeValue;

import software.amazon.awssdk.services.dynamodb.model.GetItemRequest;

import software.amazon.awssdk.services.dynamodb.model.GetItemResponse;

import software.amazon.awssdk.services.dynamodb.model.PutItemRequest;

import java.net.URI;

import java.util.HashMap;

import java.util.Map;

public class AcceeleratorDAXDemo {

public static void main(String[] args) {

//get this end point created using AWS cloud

// create a acluster..

String daxEndpoint = "daxs://daxcluster01.ee3lf0.dax-clusters.ap-south-1.amazonaws.com";

// we are creating DAx client

DaxClient daxClient = DaxClient.*builder*()

.endpointOverride(URI.*create*(daxEndpoint))

.region(Region.*AP\_SOUTH\_1*)

.credentialsProvider(DefaultCredentialsProvider.*create*())

.build();

String tableName = "Employees01";

String KeyName = "ID";

String KeyValue = "10001";

Map<String, AttributeValue> item = new HashMap<>();

item.put(KeyName, AttributeValue.*builder*().s(KeyValue).build());

item.put("msg", AttributeValue.*builder*().s("we are creating DAX ").build());

PutItemRequest request = PutItemRequest.*builder*()

.tableName(tableName)

.item(item)

.build();

daxClient.putItem(request);

Map<String, AttributeValue> getItem = new HashMap<>();

getItem.put(KeyName, AttributeValue.*builder*().build());

GetItemRequest request = GetItemRequest.*builder*()

.tableName(tableName)

.key(getItem)

.build();

GetItemResponse response = daxClient.getItem(request);

System.*out*.println("Dax working ..");

daxClient.close();

}

}

<dependency>

<groupId>software.amazon.awssdk</groupId>

<artifactId>dax</artifactId>

<version>2.20.100</version> <!-- Use the latest stable version of the SDK -->

</dependency>

—-----------------------------------------------------------------------------------------------------------

Task 03:

DAX code ….

Using micronaut … still I need to work on it…

package org.example;

// DAX demo...

//annotations @

//@DynamoDBTable

//@DynamoDBHashKey

//@DynamoDBRangeKey

//@Service

import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;

import software.amazon.awssdk.regions.Region;

import software.amazon.awssdk.services.dax.DaxClient;

import software.amazon.awssdk.services.dax.endpoints.internal.Value;

//import software.amazon.awssdk.services.dynamodb.DynamoDbClient;

import software.amazon.awssdk.enhanced.dynamodb.mapper.annotations.DynamoDbBean;

import software.amazon.awssdk.enhanced.dynamodb.mapper.annotations.DynamoDbPartitionKey;

import software.amazon.awssdk.enhanced.dynamodb.mapper.annotations.DynamoDbSortKey;

import software.amazon.awssdk.enhanced.dynamodb.mapper.annotations.DynamoDbAttribute;

import software.amazon.awssdk.enhanced.dynamodb.mapper.annotations.DynamoDbSecondaryPartitionKey;

import software.amazon.awssdk.enhanced.dynamodb.mapper.annotations.DynamoDbSecondarySortKey;

import software.amazon.awssdk.enhanced.dynamodb.mapper.annotations.DynamoDbTable; // This is the core annotation for table mapping

import software.amazon.awssdk.enhanced.dynamodb.DynamoDbEnhancedClient;

import software.amazon.awssdk.enhanced.dynamodb.DynamoDbTable;

import software.amazon.awssdk.enhanced.dynamodb.TableSchema;

import software.amazon.awssdk.enhanced.dynamodb.mapper.annotations.DynamoDbBean; // If using annotations

import software.amazon.awssdk.enhanced.dynamodb.mapper.annotations.DynamoDbPartitionKey; // If using annotations

import java.net.URI;

@DynamoDBTable(tableName = "DaxDynamoTable")

class AcceleratorDAXDemo04 {

@DynamoDBHashKey

private String hashKey;

@DynamoDBRangeKey

private String range;

public String getHashKey() {

return hashKey;

}

public void setHashKay(String hashKey) {

this.hashKey = hashKey;

}

public AcceleratorDAXDemo04(){

}

public AcceleratorDAXDemo04 UsingHashKey (String hashKey) {

this.hashKey=hashKey;

return this;

}

public String getRange() {

return range;

}

public void setRange(String range) {

this.range=range;

}

public AcceleratorDAXDemo04 usingRange(String range) {

this.range = range;

return this;

}

public static void main(String[] args) {

// DynamoDbClient

String daxEndpoint = "daxs://daxcluster01.ee3lf0.dax-clusters.ap-south-1.amazonaws.com";

DaxClient daxClient = DaxClient.*builder*()

.endpointOverride(URI.*create*(daxEndpoint))

.region(Region.*AP\_SOUTH\_1*)

.credentialsProvider(DefaultCredentialsProvider.*create*())

.build();

//table name

//key name

// kay value

System.*out*.println("helloo");

}

}

package org.example;

@Service(AcceleratorDAXDemo04.class)

public interface AcceleratorDAXDemo04Interface {

AcceleratorDAXDemo04 save(AcceleratorDAXDemo04 obj);

AcceleratorDAXDemo04 load(String hashKey, String rangeKey);

}

// implement ...

//public class AcceleratorDAXDemo04 {

//used AcceleratorDAXDemo04 instead of DaxTable class..

//}

package org.example;

import java.util.function.Supplier;

import software.amazon.awssdk.enhanced.dynamodb.mapper.annotations.DynamoDbBean;

@FunctionBean("dax-tester")

public class AcceleratorDAXDemo04Testing implements Supplier<String> {

private final AcceleratorDAXDemo04Interface testTableService;

public AcceleratorDAXDemo04Testing(AcceleratorDAXDemo04Interface service) {

this.testTableService= service;

}

@Override

public String get() {

long startTime = System.*currentTimeMillis*();

AcceleratorDAXDemo04 table = new AcceleratorDAXDemo04().UsingHashKey("hash")

.usingRange("range");

AcceleratorDAXDemo04Interface.save(table);

// running the loop for 1000 milli sec = 1 sec

for (int i = 0; i < 1000; i++) {

// check for the non null value

requireNotNull(AcceleratorDAXDemo04Interface.load("hash", "range"));

}

long endTime = System.*currentTimeMillis*();

return "table is loaded in " + endTime+" milli seconds ";

}

}

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>org.example</groupId>

<artifactId>Atlas</artifactId>

<version>1.0-SNAPSHOT</version>

<properties>

<maven.compiler.source>23</maven.compiler.source>

<maven.compiler.target>23</maven.compiler.target>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<!-- AWS SDK v2 for DynamoDB -->

<!-- https://mvnrepository.com/artifact/software.amazon.awssdk/dynamodb -->

<dependency>

<groupId>software.amazon.awssdk</groupId>

<artifactId>dynamodb</artifactId>

<version>2.33.4</version>

</dependency>

<!-- Jackson (for JSON) -->

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

<version>2.17.2</version>

</dependency>

<!-- &lt;!&ndash; SLF4J Simple Logger &ndash;&gt;-->

<!-- <dependency>-->

<!-- <groupId>org.slf4j</groupId>-->

<!-- <artifactId>slf4j-simple</artifactId>-->

<!-- <version>2.0.0</version>-->

<!-- </dependency>-->

<!-- https://mvnrepository.com/artifact/org.slf4j/slf4j-simple -->

<!-- <dependency>-->

<!-- <groupId>org.slf4j</groupId>-->

<!-- <artifactId>slf4j-simple</artifactId>-->

<!-- <version>1.7.30</version>-->

<!-- <scope>test</scope>-->

<!-- </dependency>-->

<!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.40</version>

</dependency>

<!-- https://mvnrepository.com/artifact/software.amazon.awssdk/qapps -->

<!-- <dependency>-->

<!-- <groupId>software.amazon.awssdk</groupId>-->

<!-- <artifactId>qapps</artifactId>-->

<!-- <version>2.33.5</version>-->

<!-- </dependency>-->

<!-- &lt;!&ndash; https://mvnrepository.com/artifact/com.amazonaws/aws-java-sdk &ndash;&gt;-->

<!-- <dependency>-->

<!-- <groupId>com.amazonaws</groupId>-->

<!-- <artifactId>aws-java-sdk</artifactId>-->

<!-- <version>1.12.791</version>-->

<!-- </dependency>-->

<!-- https://mvnrepository.com/artifact/software.amazon.awssdk/dynamodb-enhanced -->

<dependency>

<groupId>software.amazon.awssdk</groupId>

<artifactId>dynamodb-enhanced</artifactId>

<version>2.25.60</version>

</dependency>

<!-- for accelerator DAX -->

<dependency>

<groupId>software.amazon.awssdk</groupId>

<artifactId>dax</artifactId>

<version>2.20.100</version> <!-- Use the latest stable version of the SDK -->

</dependency>

<!-- https://mvnrepository.com/artifact/software.amazon.dax/amazon-dax-client -->

<dependency>

<groupId>software.amazon.dax</groupId>

<artifactId>amazon-dax-client</artifactId>

<version>2.0.5</version>

<scope>runtime</scope>

</dependency>

<!-- DAX SDK -->

<!-- <dependency>-->

<!-- <groupId>com.amazonaws</groupId>-->

<!-- <artifactId>amazon-dax-client</artifactId>-->

<!-- <version>2</version> &lt;!&ndash; Use latest &ndash;&gt;-->

<!-- </dependency>-->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-simple</artifactId>

<version>2.0.7</version>

</dependency>

</dependencies>

<!-- <dependency>-->

<!-- <groupId>software.amazon.awssdk</groupId>-->

<!-- <artifactId>dynamodb</artifactId>-->

<!-- <version>2.32.2</version> &lt;!&ndash; use latest stable version while you are trying&ndash;&gt;-->

<!-- </dependency>-->

<!-- <dependency>-->

<!-- <groupId>software.amazon.awssdk</groupId>-->

<!-- <artifactId>dynamodb</artifactId>-->

<!-- <version>2.33.4</version>-->

<!--&lt;!&ndash; <version>2.32.24</version>&ndash;&gt;-->

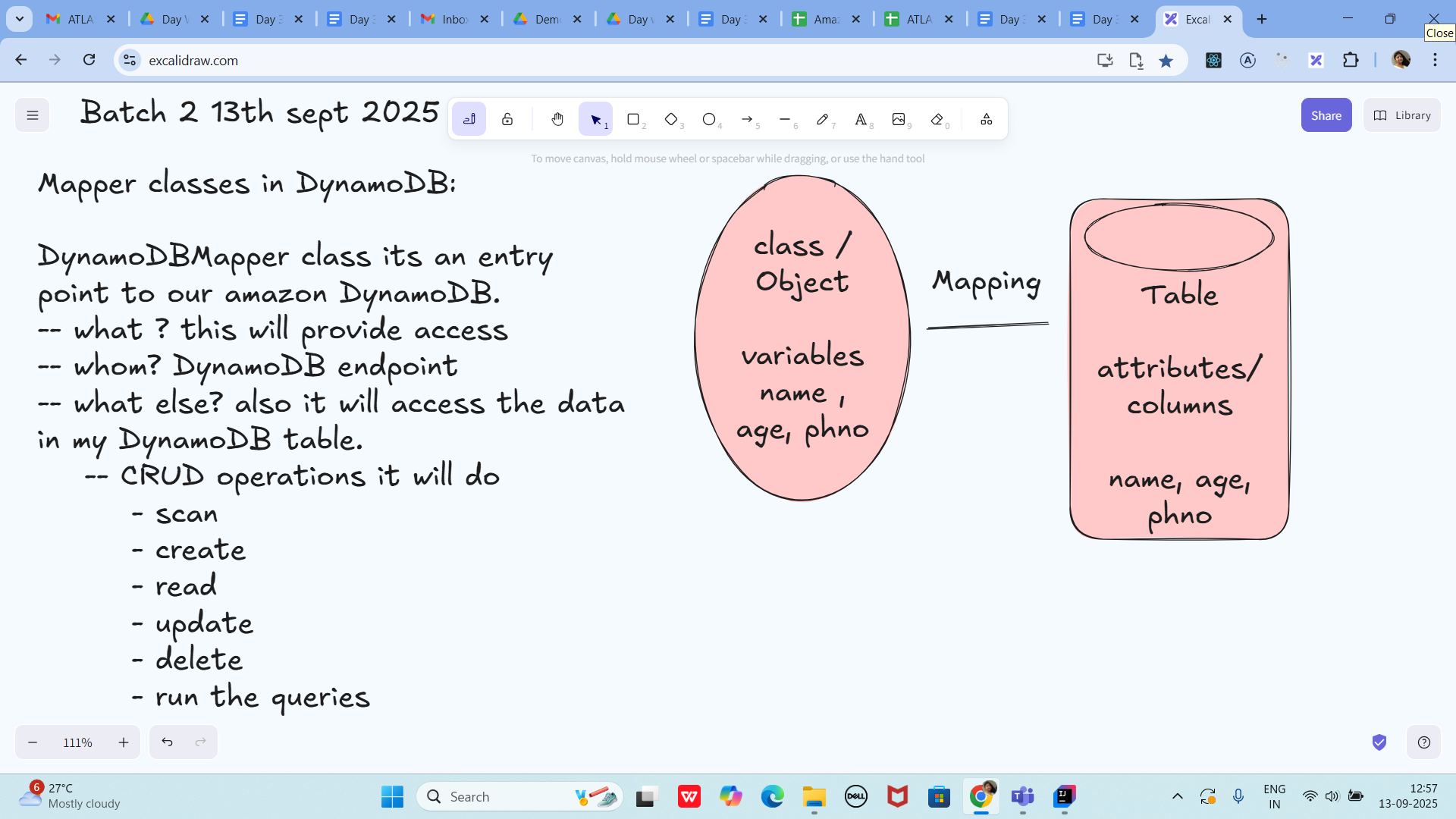
<!-- </dependency>-->

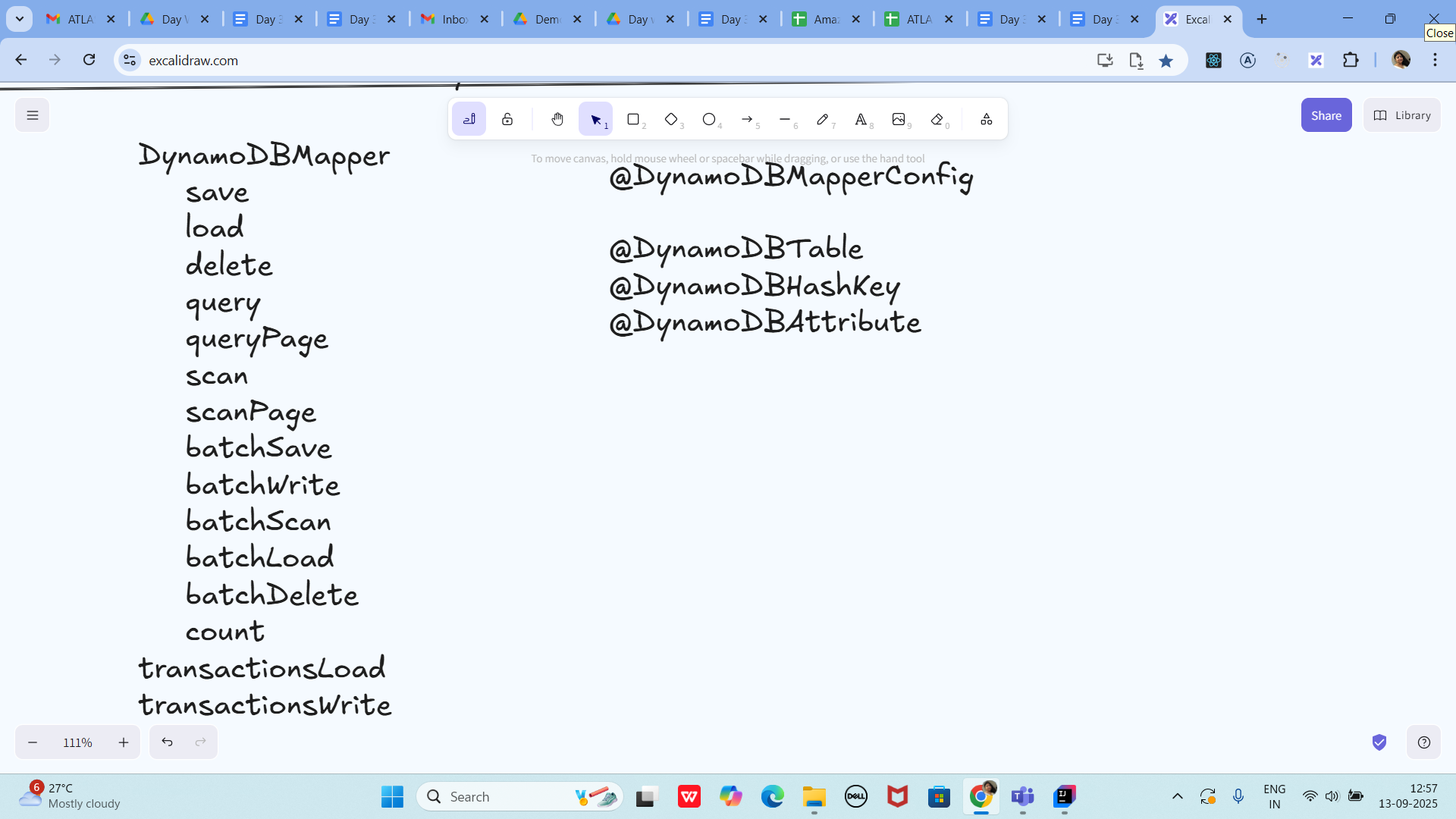
</project>

—---------------------------------------------------------------------------------------------------------------------------

—--------------------------------------------------------------------------------------------------------------------------

DynamoDbMapper Class:





Task 04 :

package org.example;

import lombok.Data;

import java.sql.Struct;

import java.util.List;

@Data

@DynamoDBTable(tableName="Students")

class Students {

@DynamoDBHashKay(attributeName="ID")

private String ID;

@DynamoDBAttribute(attributeName="SName")

private String Name;

}

//DynamoDB Access

class DynamoDBAccess {

private DynamoDBMapper Mapper = new DynamoDBMapper((AmazonDynamoDB) AmazonDynamoDBClientBuilder.standard().build());

public Students getItem(Students item) {

return Mapper.load(Students.class, item.getID());

}

public void saveItem(Students item) {

Mapper.save(Item);

}

public List<Students> getItems(Students item) {

return Mapper.query(Students.class, new DynamoDBQueryExpression().withHashKeyValues(item));

}

}

public class DynamoDBMapperClass {

public static void main(String[] args) {

}

}

Lombok dependency

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.20</version>

<scope>provided</scope>

</dependency>

============================================================================================================================================

Info Box:

Excalidraw updated at 12.56

<https://excalidraw.com/#json=RPROxQ8EuBIRuHvGsahCA,iTT3ZgQ2B60UqAMx90k_mA>