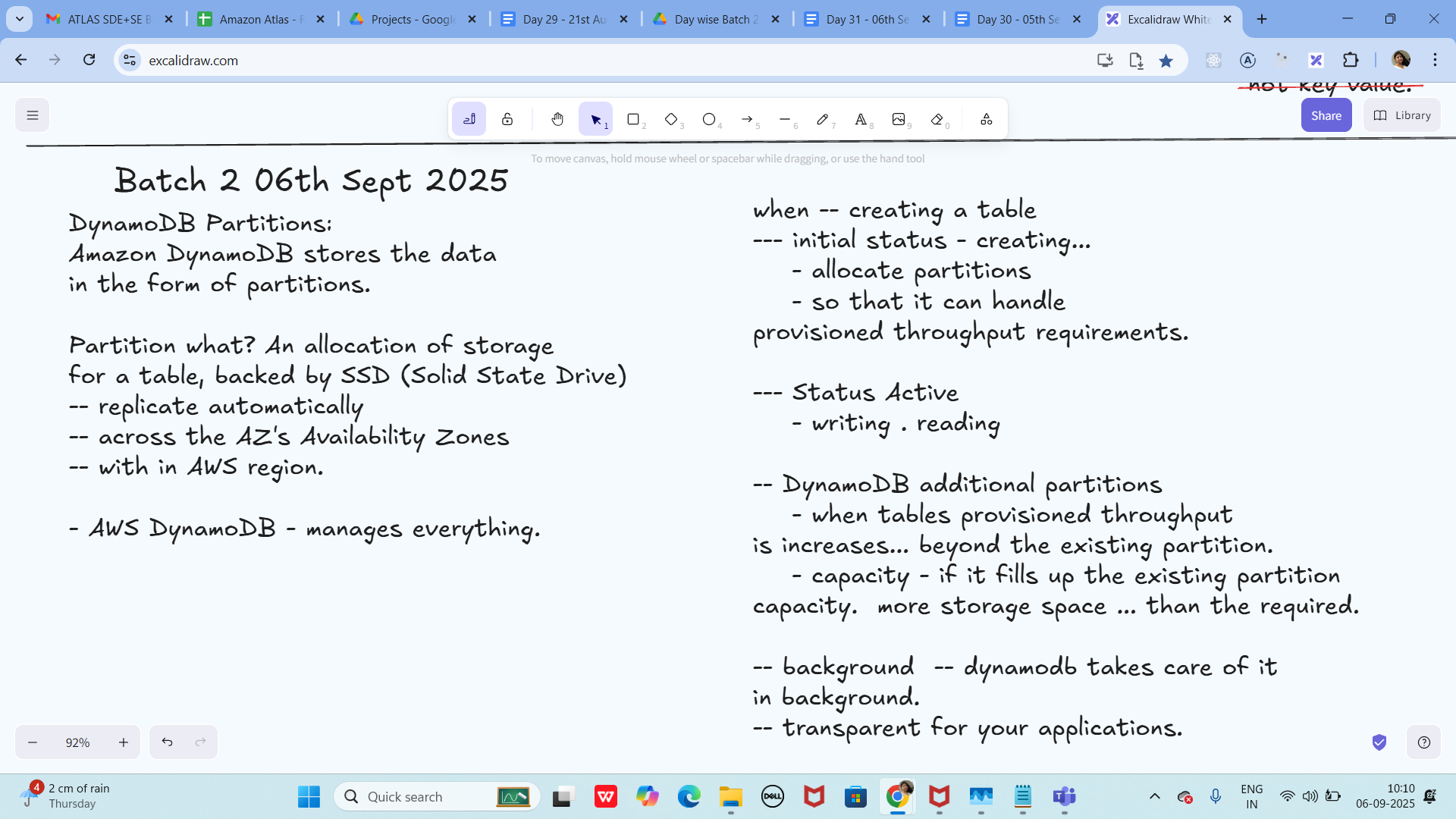
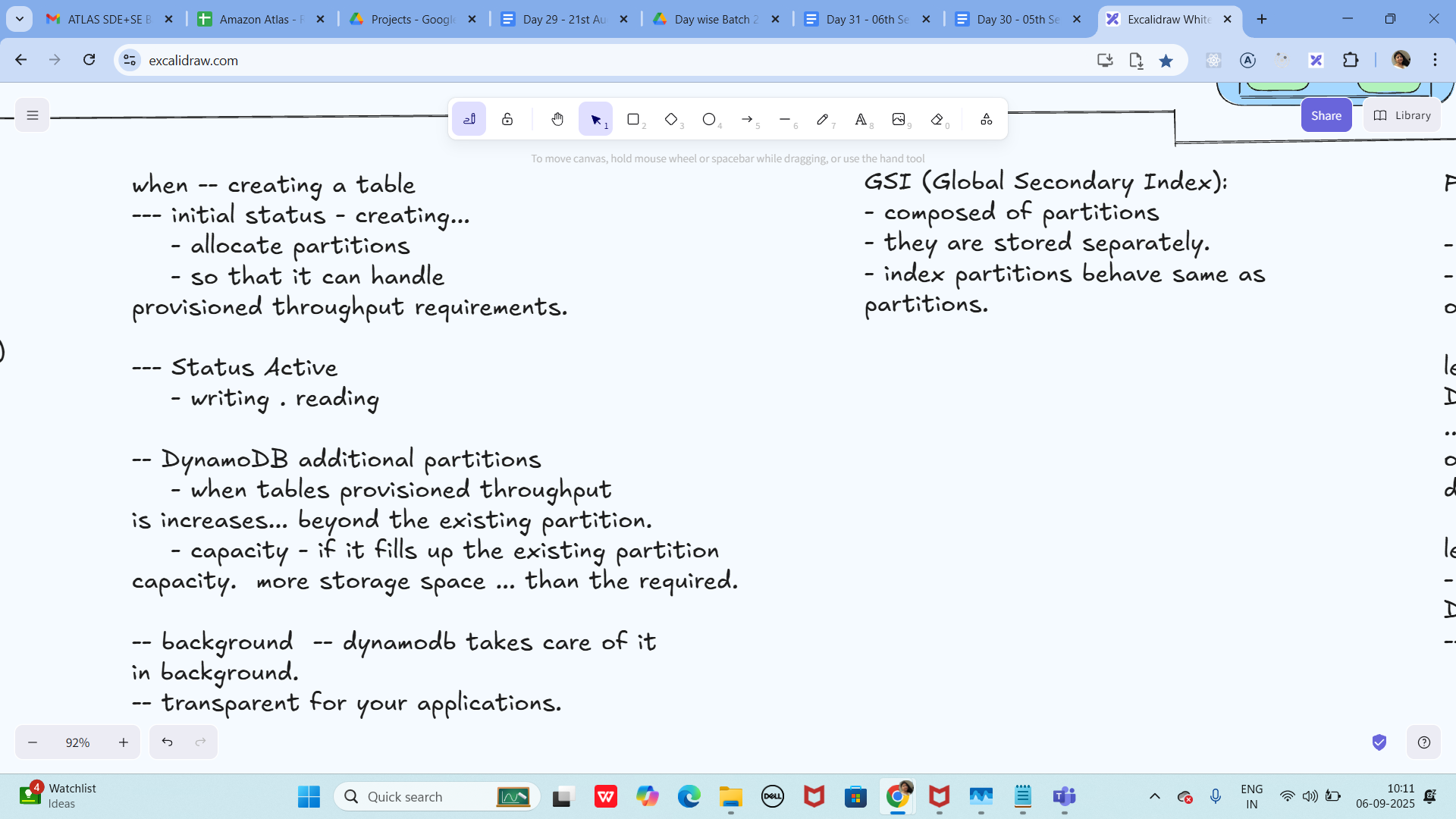
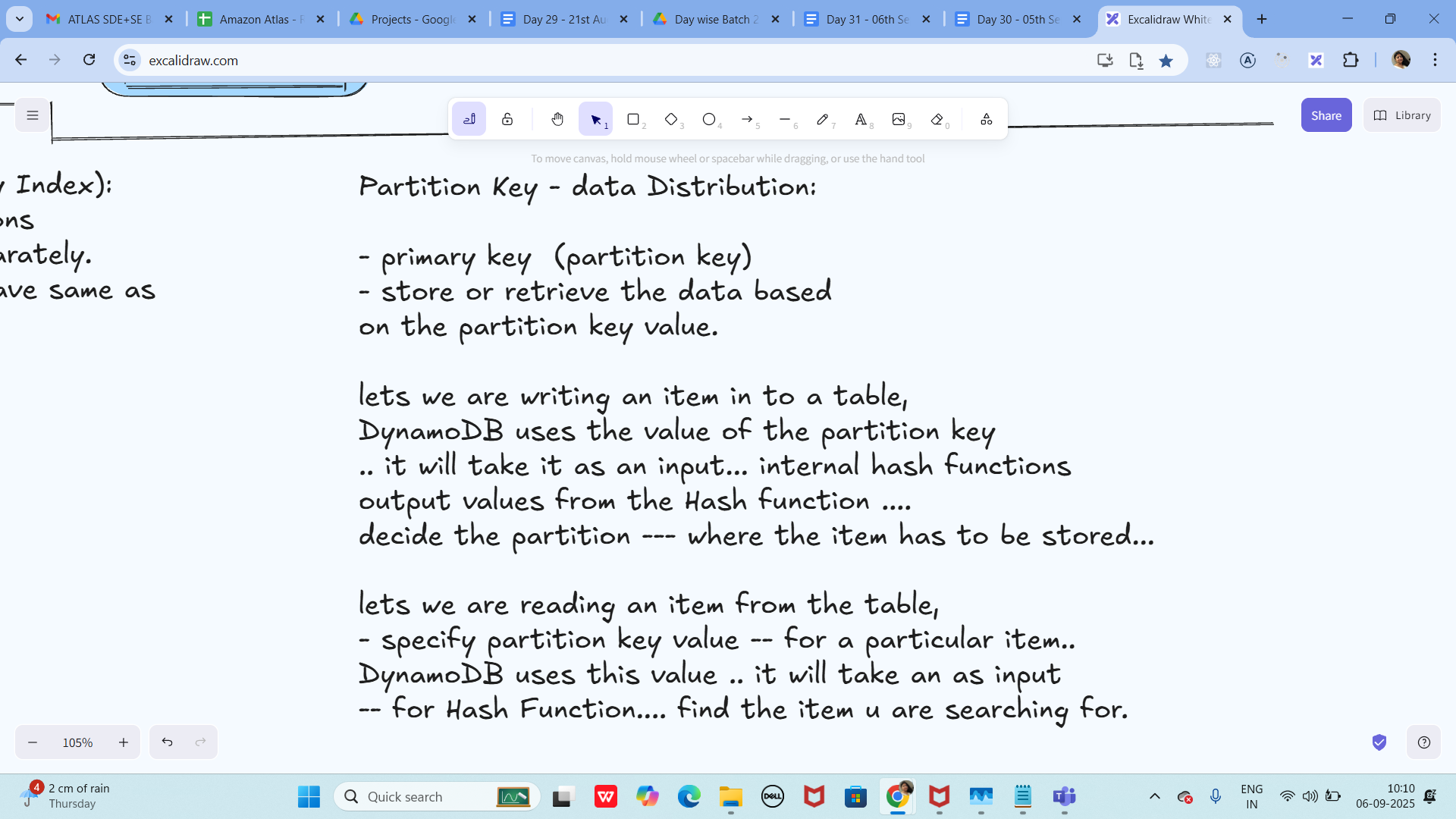
Day 31 - 06th Sept 2025







My Server is at C:\Program Files\DynamoDB\DynamoDB\_local&gt;

So open cmd at that location and run below command

Run the Server

java -Djava.library.path=./DynamoDBLocal\_lib -jar DynamoDBLocal.jar -sharedDb

Answer: ✅ (OK). Default port is 8000; you’ve chosen 8001—keep it consistent across SDK/CLI.

Pom.xml

&lt;dependencies&gt; &lt;!-- AWS SDK v2 for DynamoDB --&gt; &lt;!--

https://mvnrepository.com/artifact/software.amazon.awssdk/dynamodb --&gt; &lt;dependency&gt;

&lt;groupId&gt;software.amazon.awssdk&lt;/groupId&gt; &lt;artifactId&gt;dynamodb&lt;/artifactId&gt;

&lt;version&gt;2.33.4&lt;/version&gt; &lt;/dependency&gt; &lt;!-- Jackson (for JSON) --&gt; &lt;dependency&gt;

&lt;groupId&gt;com.fasterxml.jackson.core&lt;/groupId&gt; &lt;artifactId&gt;jackson-databind&lt;/artifactId&gt;

&lt;version&gt;2.17.2&lt;/version&gt; &lt;/dependency&gt; &lt;/dependencies&gt; &lt;dependency&gt;

&lt;groupId&gt;org.slf4j&lt;/groupId&gt; &lt;artifactId&gt;slf4j-simple&lt;/artifactId&gt; &lt;version&gt;1.7.36&lt;/version&gt;

&lt;!-- Use the latest stable version --&gt; &lt;/dependency&gt; \*\*Answer:\*\* Move `slf4j-simple`

\*\*inside\*\* `&lt;dependencies&gt;...&lt;/dependencies&gt;` (same level as others).

Create a table …

package org.example;

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

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

import software.amazon.awssdk.regions.Region;

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

import software.amazon.awssdk.services.dynamodb.model.\*;

import java.net.URI;

//create table in DynamoDB

public class Demo01 {

public static void main(String[] args) throws Exception {

System.out.println(&quot;hello create table in DynamoDB&quot;);

AwsBasicCredentials awsCreds = AwsBasicCredentials.create(&quot;fakeAccesskey&quot;,&quot;fakeSecretKey&quot;);

DynamoDbClient client = DynamoDbClient.builder()

.endpointOverride(URI.create(&quot;http://localhost:8001&quot;))

.region(Region.AP\_SOUTH\_1)

.credentialsProvider(StaticCredentialsProvider.create(awsCreds))

.build();

String tableName = &quot;Employees01&quot;;

try {

CreateTableRequest request = CreateTableRequest.builder()

.tableName(tableName)

.keySchema(KeySchemaElement.builder()

.attributeName(&quot;ID&quot;)

.keyType(KeyType.HASH)

.build())

.attributeDefinitions(AttributeDefinition.builder()

.attributeName(&quot;ID&quot;)

.attributeType(ScalarAttributeType.N)

.build())

.provisionedThroughput(ProvisionedThroughput.builder()

.readCapacityUnits(5L)

.writeCapacityUnits(5L)

.build())

.build();

client.createTable(request);

System.out.println(tableName + &quot; table is created. &quot;);

}catch (ResourceInUseException ex) {

System.out.println(&quot; plz choose different tablename as it already exists&quot;);

}

client.close();

}

}

Output:

Table created Employees01

Answer (expected console):

hello create table in DynamoDB

Employees01 table is created.

To see the description of the table ��

aws dynamodb describe-table --table-name Employees01 --endpoint-url http://localhost:8001

Answer (tip): Include --region ap-south-1 if your CLI requires it for Local:

aws dynamodb describe-table --table-name Employees01 --endpoint-url http://localhost:8001 -

-region ap-south-1

Loading data to the table // inserting data

package org.example;

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

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

import software.amazon.awssdk.regions.Region;

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

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

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

import com.fasterxml.jackson.databind.ObjectMapper;

import com.fasterxml.jackson.databind.JsonNode;

import java.io.InputStream;

import java.net.URI;

import java.util.HashMap;

import java.util.Iterator;

import java.util.Map;

//loading data to the table created

public class Demo02 {

public static void main(String[] args) throws Exception{

System.out.println(&quot;loading data to table &quot;);

AwsBasicCredentials awsCreds = AwsBasicCredentials.create(&quot;fakeAccesskey&quot;,&quot;fakeSecretKey&quot;);

DynamoDbClient client = DynamoDbClient.builder()

.endpointOverride(URI.create(&quot;http://localhost:8001&quot;))

.region(Region.AP\_SOUTH\_1)

.credentialsProvider(StaticCredentialsProvider.create(awsCreds))

.build();

String tableName = &quot;Employees01&quot;;

ObjectMapper mapper = new ObjectMapper();

InputStream jsonStream = Demo02.class.getClassLoader()

.getResourceAsStream((&quot;Employee.json&quot;));

if(jsonStream == null) {

System.err.println(&quot;plz provide json file&quot;);

System.exit(1);

}

JsonNode rootNode = mapper.readTree(jsonStream);

Iterator&lt;JsonNode&gt; iterator = rootNode.elements();

//now inserting data to your table at port no 800\*

while(iterator.hasNext()) {

JsonNode node = iterator.next();

Map&lt;String, AttributeValue&gt; item = new HashMap&lt;&gt;();

item.put(&quot;ID&quot;, AttributeValue.builder().n(node.get(&quot;ID&quot;).asText()).build());

item.put(&quot;Name&quot;, AttributeValue.builder().s(node.get(&quot;Name&quot;).asText()).build());

item.put(&quot;Address&quot;, AttributeValue.builder().s(node.get(&quot;Address&quot;).asText()).build());

PutItemRequest putRequest = PutItemRequest.builder()

.tableName(tableName)

.item(item)

.build();

client.putItem(putRequest);

System.out.println(&quot;inserted data &quot;+ node.get(&quot;ID&quot;).asInt()+ &quot; \*\*\*\*\* &quot; +

node.get(&quot;Name&quot;).asText()+&quot; $$$ &quot;+node.get(&quot;Address&quot;).asText());

}

client.close();

}

}

This is my json file Employee.json

[

{

&quot;ID&quot;: 1001,

&quot;Name&quot;: &quot;Prasunamba&quot;,

&quot;Address&quot;: &quot;India&quot;

},

{

&quot;ID&quot;: 1002,

&quot;Name&quot;: &quot;Meher&quot;,

&quot;Address&quot;: &quot;Australia&quot;

},

{

&quot;ID&quot;: 1003,

&quot;Name&quot;: &quot;K&quot;,

&quot;Address&quot;: &quot;USA&quot;

}

]

Output:

Answer (expected console):

loading data to table

inserted data 1001 \*\*\*\*\* Prasunamba $$$ India

inserted data 1002 \*\*\*\*\* Meher $$$ Australia

inserted data 1003 \*\*\*\*\* K $$$ USA

Scanning data from the table // getting data // loading data

package org.example;

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

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

import software.amazon.awssdk.regions.Region;

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

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

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

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

import java.net.URI;

import java.util.Map;

public class Demo03 {

public static void main(String[] args) {

System.out.println(&quot;Scanning data from table&quot;);

AwsBasicCredentials awsCreds = AwsBasicCredentials.create(&quot;fakeAccesskey&quot;,&quot;fakeSecretKey&quot;);

DynamoDbClient client = DynamoDbClient.builder()

.endpointOverride(URI.create(&quot;http://localhost:8001&quot;))

.region(Region.AP\_SOUTH\_1)

.credentialsProvider(StaticCredentialsProvider.create(awsCreds))

.build();

String tableName = &quot;Employees01&quot;;

ScanRequest req = ScanRequest.builder().tableName(tableName).build();

ScanResponse resp = client.scan(req);

for(Map&lt;String, AttributeValue&gt; dbitem : resp.items()) {

System.out.println(dbitem);

}

client.close();

}

}

Output:

Answer (example printout):

Scanning data from table

{ID=N:1001, Name=S:Prasunamba, Address=S:India}

{ID=N:1002, Name=S:Meher, Address=S:Australia}

{ID=N:1003, Name=S:K, Address=S:USA}

Home Task:

Plz work on

Task 01:

Delete a record operation

Task 02:

Delete a table

Task 03:

Update a record in a table

Answer (minimal SDK v2 snippets):

Task 01 – Delete a record

client.deleteItem(b -&gt; b.tableName(&quot;Employees01&quot;)

.key(Map.of(&quot;ID&quot;, AttributeValue.builder().n(&quot;1002&quot;).build())));

Task 02 – Delete a table

client.deleteTable(b -&gt; b.tableName(&quot;Employees01&quot;));

Task 03 – Update a record (e.g., change Address)

client.updateItem(b -&gt; b

.tableName(&quot;Employees01&quot;)

.key(Map.of(&quot;ID&quot;, AttributeValue.builder().n(&quot;1003&quot;).build()))

.updateExpression(&quot;SET #A = :addr&quot;)

.expressionAttributeNames(Map.of(&quot;#A&quot;, &quot;Address&quot;))

.expressionAttributeValues(Map.of(&quot;:addr&quot;, AttributeValue.builder().s(&quot;UK&quot;).build()))

);

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

Delete operation

package org.example;

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

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

import software.amazon.awssdk.regions.Region;

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

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

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

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

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

import java.net.URI;

import java.util.HashMap;

import java.util.Map;

public class DeleteItem {

public static void main(String[] args) {

System.out.println(&quot;hello Delete item by id from table in DynamoDB&quot;);

AwsBasicCredentials awsCreds = AwsBasicCredentials.create(&quot;fakeAccesskey&quot;,&quot;fakeSecretKey&quot;);

DynamoDbClient client = DynamoDbClient.builder()

.endpointOverride(URI.create(&quot;http://localhost:8001&quot;))

.region(Region.AP\_SOUTH\_1)

.credentialsProvider(StaticCredentialsProvider.create(awsCreds))

.build();

String tableName = &quot;Employees01&quot;;

int delId = 1002;

Map&lt;String, AttributeValue&gt; item = new HashMap&lt;&gt;();

item.put(&quot;ID&quot;, AttributeValue.builder().n(String.valueOf(delId)).build());

GetItemResponse getResponse =client.getItem(builder -&gt;

builder.tableName(tableName).key(item));

client.deleteItem(builder -&gt; builder.tableName(tableName).key(item));

System.out.println(&quot;items after deletion&quot;);

ScanRequest req = ScanRequest.builder().tableName(tableName).build();

ScanResponse resp = client.scan(req);

for(Map&lt;String, AttributeValue&gt; dbitem : resp.items()) {

System.out.println(dbitem);

}

client.close();

}

}

Output:

Answer (example after deleting ID=1002):

hello Delete item by id from table in DynamoDB

items after deletion

{ID=N:1001, Name=S:Prasunamba, Address=S:India}

{ID=N:1003, Name=S:K, Address=S:USA}

16th Sept 2025: 18.47

Add ons:

Displaying list of all tables in server:

Code:

package org.example;

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

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

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

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

import java.util.List;

public class Demo03DynamoDBl\_TablesList {

void tablesList(DynamoDbClient client){

System.out.println(&quot;display existing tables in the DynamoDB server&quot;);

boolean tables = true;

String lastName = null;

while (tables) {

try {

ListTablesResponse res = null;

if (lastName == null) {

ListTablesRequest req = ListTablesRequest.builder().build();

res = client.listTables(req);

} else {

ListTablesRequest req = ListTablesRequest.builder()

.exclusiveStartTableName(lastName).build();

res = client.listTables(req);

}

List&lt;String&gt; namesOfTables = res.tableNames();

if (namesOfTables.size() &gt; 0) {

for (String currentName : namesOfTables) {

System.out.format(&quot;\* %s\n&quot;, currentName);

}

} else {

System.out.println(&quot;No tables found!&quot;);

System.exit(0);

}

lastName = res.lastEvaluatedTableName();

if (lastName == null) {

tables = false;

}

} catch (DynamoDbException ex) {

System.err.println(ex.getMessage());

System.exit(1);

}

}

System.out.println(&quot;\nDone!&quot;);

}

public static void main(String[] args) {

demo02DynamoDBconnection obj = new demo02DynamoDBconnection();

DynamoDbClient client = obj.dynamoDBConnection();

Demo03DynamoDBl\_TablesList obj2 = new Demo03DynamoDBl\_TablesList();

obj2.tablesList(client);

}

}

Answer: ✅ Works. Ensure demo02DynamoDBconnection returns a DynamoDbClient configured

with endpointOverride(http://localhost:8001) and Region.AP\_SOUTH\_1. Example output:

display existing tables in the DynamoDB server

\* Employees01

Done!

Info Box:

Excalidraw updated at 10.12

<https://excalidraw.com/#json=NwMX_ny4Q1qCV5DM1yAGu,uuB0ScTcUq0RTVkO_l7kzg>