1.      In DynamoDB, define table, item, and attribute what they are  
 - Table is a collection of data organized as key-value pairs. Serves as the top-level container for data.  
 - Item is a single record in a table. Each item is uniquely identifiable by a primary key.  
 - Attributes are individual pieces of data that make up an item. They can store different data types.

2.      List data types supported in DynamoDB  
 Null, Boolean, String, Number, Binary, Map, List, String Set, Number Set, Binary Set.

3.      Define what Lambda in AWS cloud is  
Lambda is a serverless compute service provided by AWS that enables you to run code without providing or managing servers. Uploaded code can execute in response to specific events, such as HTTP requests, database updates, or files uploaded to Amazon S3.

4.      Define what runtimes in Lambda are  
 Runtimes are the environments that allow functions to execute code written in specific programming languages. Each runtime includes a language-specific interpreter or compiler, libraries, and other dependencies needed to run the code. Example: Node.js, Python, Java.

5.      In Lambda, what is a handler function?  
 Entry point that Lambda invokes when it executes your function. It processes incoming requests, performs the specified logic, and returns a response. You specify the handler function when deploying a Lambda function.  
 1. Event – Contains details about the event that triggered the Lambda function (i.e. data from an API gateway)  
 2. Context (optional) – Provides information about the invocation, function configuration, and execution environment (i.e. functionName) **CHUPAME EL PITO**  
 3 – Callback (optional) – Callback function is used to signal completion with a result or error.

6.      List two methods in JSON with short descriptions  
 1. JSON.stringify(obj) – Converts a JS object into a JSON-formatted string  
 2. JSON.parse(string) – Converts a JSON-formatted string into a JS object

7.      What is a trigger in Lambda?  
 A trigger is a service or event source that automatically invokes a Lambda function in response to a specific action (i.e. data sent to POST /users). Triggers allow Lambda to operate in a fully event-driven manner, meaning the function runs only when needed. (i.e. API Gateway)

8.      In index.js, example code to read memo.html and send it back to the client  
 import \* as fs from 'node:fs';  
 export const handler = async (event) => {  
 const html = fs.readFileSync(‘memo.html’, { encoding: ‘utf-8’ });  
 const response = {  
     statusCode: 200,  
     headers: { ‘Content-type’: ‘text/html’, },  
     body: html,  
 };  
 return new Promise((res, rej) => { res(response) });  
 }

9.      Example code to list S3 buckets and return it  
 import { S3 } from ‘@aws-sdk/client-s3’;  
 let s3 = new S3();  
 export const handler = async (event) => {  
 return s3.listBuckets().promise();  
 }

10.   Example code to get the POST data from an HTTP request event object  
 import querystring from ‘node:querystring’;  
 export const handler = async (event) => {  
 if(event.requestContext.http.method == ‘POST’) {  
     const postData = querystring.parse(atob(event.body));  
 }  
 }

11.   Example code to get the GET data from an HTTP request event object  
 export const handler = async (event) => {  
 if(event.requestContext.http.method == ‘GET’) {  
     const queryParams = event.queryStringParameters;  
 }  
 }

12.   Example code to get/put an item from/to a DynamoDB table. Scan and delete as well.

let memo = req.body.memo;  
let command = new GetCommand({  
 TableName: “Memos”,  
 Key: { ID: -1,},  
});  
Let data = await docClient.send(command);

command = new PutCommand({  
 TableName: “Memos”,  
 Item: {

 ID: data.Item.NextID,

 Memo: memo,  
}),

data = await docClient.send(command);

13.   List of HTTP methods used in RESTful APIs  
 1. GET – Retrieve data from the server (GET /users)  
 2. POST – Submit/create data to be processed by the server (POST /users)  
 3. PUT – Update an existing resource or create a resource if it does not exist (PUT /users/{id})  
 4. DELETE – Delete a resource from the server (DELETE /users/{id})

14.   Example code to use Express app – methods and how to use req and res  
 import express from ‘express’;  
 const app = express();  
 app.use(express.urlencoded({ extended: true }));  
 app.use(express.json());  
  
 app.get(‘/’, async (request, response) => {  
 const html = fs.readFileSync('memo-api-dynamodb.html', { encoding: 'utf8' }) +  
           `<script>const API\_GATEWAY = "https://${request.get('host')}/"; </script>`;  
 response.send(html);  
 }

15.   Explain what CORS is  
Cross-Origin Resource Sharing – Security standard that allows client web applications to interact with resources from other domains/web servers. It uses HTTP headers to indicate which origins a browser can load resources from.

16.   Example code to use S3 buckets  
export const handler = async (event) => {

   const sourceBucket = event.Records[0].s3.bucket.name;

   const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, ' '));

   const response = await readFileFromS3Bucket(sourceBucket, key);

   if (response.result)

await uploadFileToS3Bucket(BACKUP\_BUCKET\_NAME, key, response.body);

   else console.log("File reading unsuccessful!");

};

17.   What is a route in REST APIs  
A route refers to the specific URL or endpoint that maps to a particular resource. It defines how clients interact with resources on the server, specifying what action to perform (read, create, update, delete).

18.   Explain what SOP is

Same-Origin Policy - a security feature implemented by web browsers to restrict how scripts from one origin can interact with resources from another origin.

19.   ES Module System  
Standardized system for modularizing JS code. You split code into reusable, isolated pieces called modules, which can export and import functionality from other modules.  
 // utils.js  
 export const add = (a, b) => a + b;  
 // main.js  
 import { add } from ‘./utils.js’;

20.   The handler in index.mjs  
Main function that serves as the entry point for the Lambda function. AWS automatically invokes this function when the function is triggered by an event source (API Gateway, DynamoDB, S3).

21.   Example code to get the next data from an HTTP request event object

a.      Query data  
 event.queryStringParameters → object  
 {

"name": "John",

"age": "30"

}  
  
 event.rawQueryString → string  
 name=John&age=30

b.      Host  
 event.headers[‘host’]

c.      Path  
 event.rawPath

22.   How to get query data and parameters from Express app  
app.get(‘/terms’, (req, res) => {

const queryData = req.query;  
 const params = req.params;

});

23.   How to access/change environment variables

Access:

const bucket = process.env.UploadBucket;  
Change:

process.env.UploadBucket = ‘NewUploadBucket’;