# Real-time data management on AWS Cloud

## SITUATION

**Objectives**

To deploying infrastructure for real-time data management requirements on the AWS Cloud offers a scalable, flexible, and cost-effective solution for organizations aiming to harness the power of real-time data processing.

**Real-World Scenario:**

A company (TELEMAX) plans to build networks in rapidly growing, underserved markets around the world. The company offers innovative communications hardware that enables them to create many speed-efficient networking links with inexpensive hardware.

Presently, they need, to deploy an effective architecture for NoSQL based data warehousing built from real-time data being generated that can be analyzed in the future to optimize their topologies continuously. The team decided the AWS cloud is the perfect environment to support their solution needs and they will approach your organization for consultation on the same.

## TASKS

1. Create AWS Kinesis Stream
2. Create DynamoDB NoSQL table with partition key
3. Create IAM role that is to be assigned to the lambda function that has KinesisFullAccess (it’s possible to start with limited access as well) and putRecord & putRecords access
4. Create AWS Lambda function with trigger set as the Kinesis stream and write the data to the dynamoDB table created above and set IAM role as above.

5. Create an EC2 instance and download the Kinesis Agent on the instance to send data to the kinesis stream (Link)

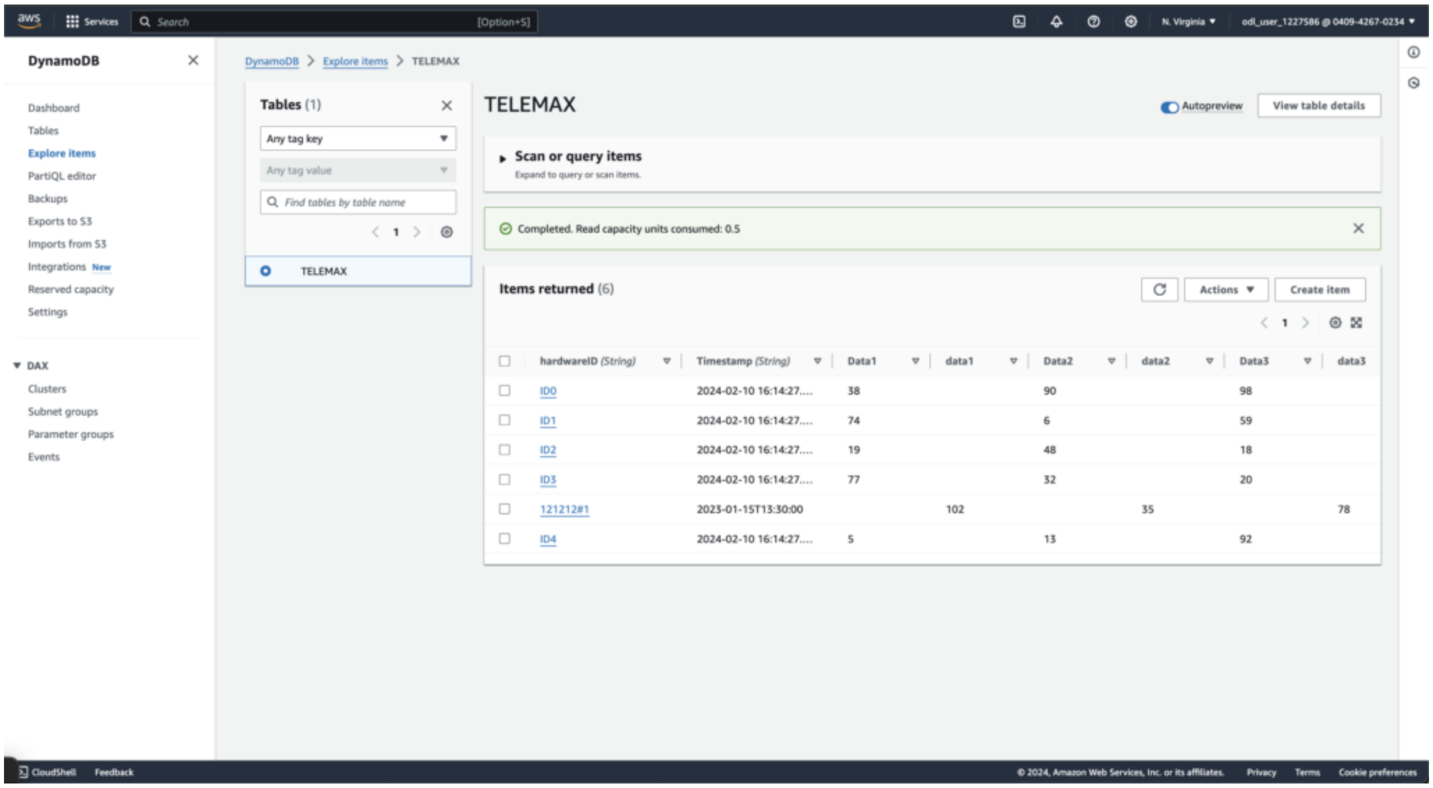
## ACTIONS

1. Created Kinesis Data Stream, configured its capacity (provisioned 2 shards), and gained insights into allocating shards for stream processing.
2. Established a structured table (TELEMAX), stored items with attributes
    - The table named TELEMAX contains hardwareID as the partition key and Timestamp of the data stream and 3 optional attributes based on domain requirements.
3. Created an IAM role with the policies mentioned in the tasks
4. Created lambda function and assigned the IAM role to it.
    - Tested the lambda function with python 3.9 code to send 200 Status Code
    - Updated the code to write synthetic data to DynamoDB and deployed
    - Updated the code to read records from the kinesis stream and deployed.
5. Created Amazon Linux EC2 instance with an IAM role with KinesisFullAccess
    - Update the /etc/aws-kinesis/agent.json file with the name of the kinesis stream
    - Update /tmp/app.log file with the data in json format to be sent to the stream
    - Started the Kinesis Agent

## RESULTS

Saw the data come through into DynamoDB.

The infrastructure setup is complete for Telemax to start writing their networking data to a NoSQL database and also leverage built-in functions like filter, aggregate, and transform data for advanced analytics with sub-second latencies. We can also setup events based on the incoming data.



DynamoDB > Explore items > TELEMAX

TELEMAX

Scan or query items

Expand to query or scan items.

Completed. Read capacity units consumed: 0.5

Items returned (6)

| hardwareID (String) | Timestamp (String) | Data1 | data1 | Data2 | data2 | Data3 | data3 |
|---|---|---|---|---|---|---|---|
| ID0 | 2024-02-10 16:14:27.... | 38 | | 90 | | 98 | |
| ID1 | 2024-02-10 16:14:27.... | 74 | | 6 | | 59 | |
| ID2 | 2024-02-10 16:14:27.... | 19 | | 48 | | 18 | |
| ID3 | 2024-02-10 16:14:27.... | 77 | | 32 | | 20 | |
| 121212#1 | 2023-01-15T13:30:00 | | 102 | | 35 | | 78 |
| ID4 | 2024-02-10 16:14:27.... | 5 | | 13 | | 92 | |