IoT Project

Instructions
Every screenshot requested in this workbook is compulsory and carries 8 marks.
Your AWS account ID must be clearly visible in every screenshot using the AWS console; missing id or using someone else's id is not permitted. Such cases will be considered as plagiarism and severe penalty will be imposed.
All screenshots must be in the order mentioned under "Expected Screenshots" for every step
DO NOT WAIT UNTIL THE LAST MINUTE. The program office will not extend the project submission deadline under any circumstances.
The file should be renamed in the format BATCH_FIRSTNAME_LASTNAME_PROJECT1. For example: ACSE_Batch_VIJAY_DWIVEDI_PROJECT1.docx

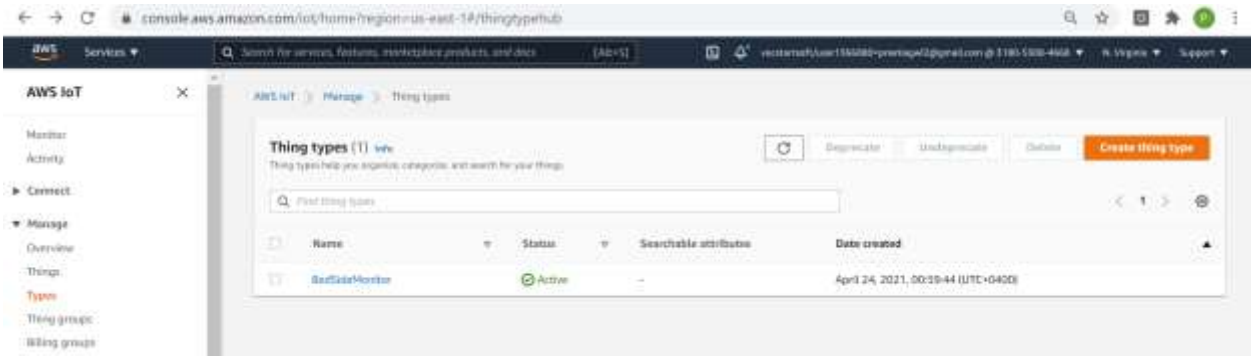
Resource Clean Up
Cloud is always a pay per use model and all resources/services that we consume are chargeable. Cleaning up when you've completed your lab or project is always necessary. This is true whether you're doing a lab or implementing a project at your workplace.
After completing the lab, make sure to delete each resource created in reverse chronological order.

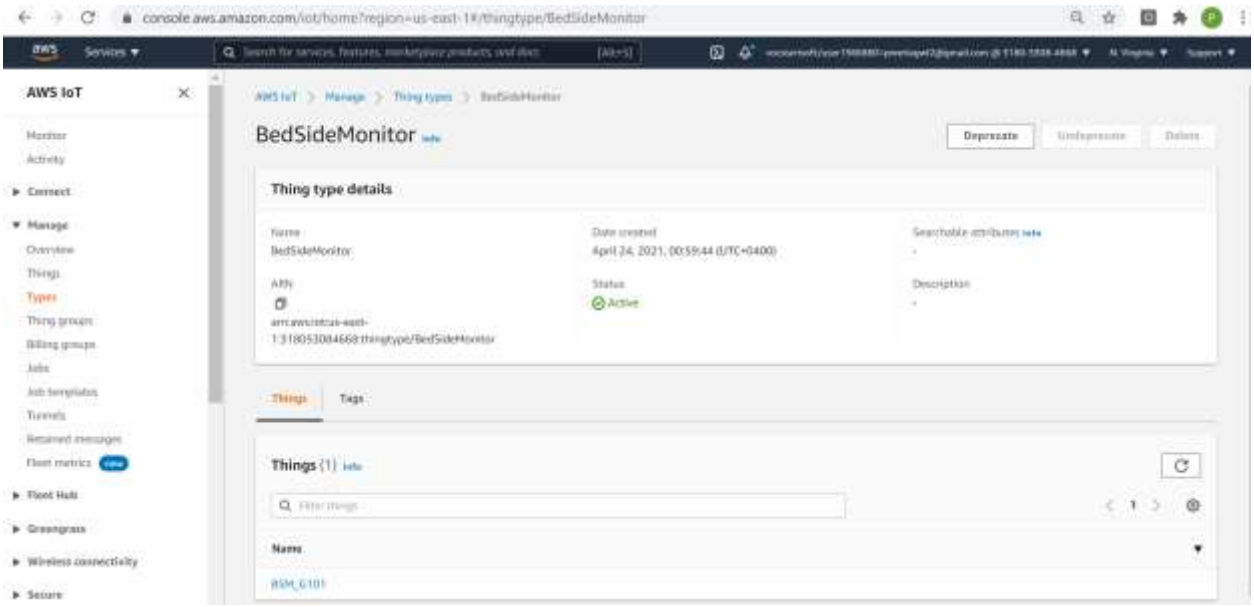
Creating Things, DynamoDB table

Step number	a
Step name	IoT Core and Things
Instructions	1) Goto IoT core and create the types of things you wish to create. 2) Create at least one group and parent group that will be attached to things. 3) Create a policy that will be attached with newly created IoT devices. 4) Create at least two devices on the IoT core to send and receive data.
Expected screenshots	1) Screenshot of successfully create thing type 2) Screenshot of successfully created group. 3) Screenshot of successfully created policy page. 4) Screenshot of successfully created thing.

<Insert Screenshot a(1) here>

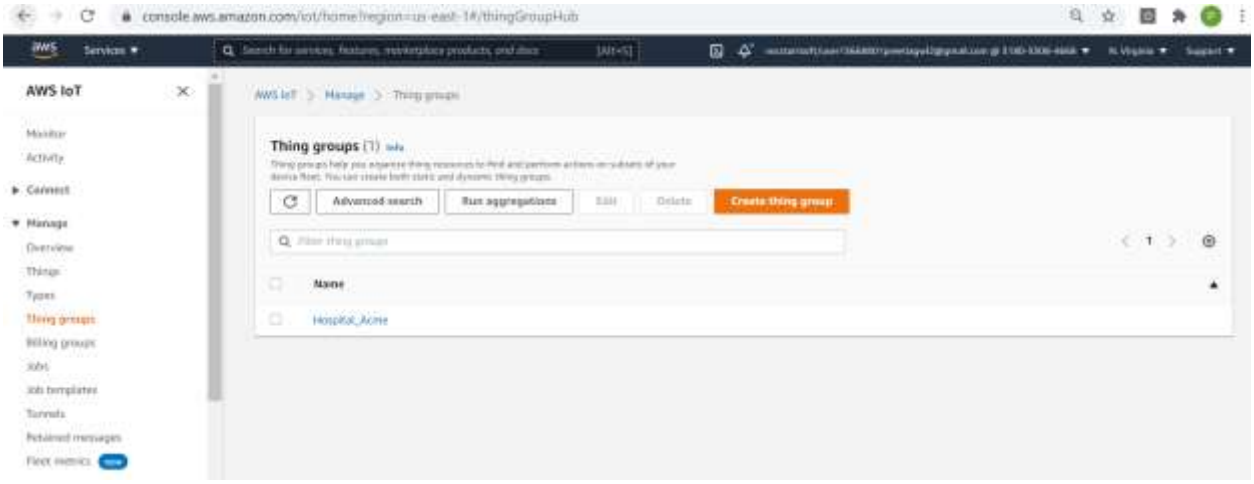
Thing Type



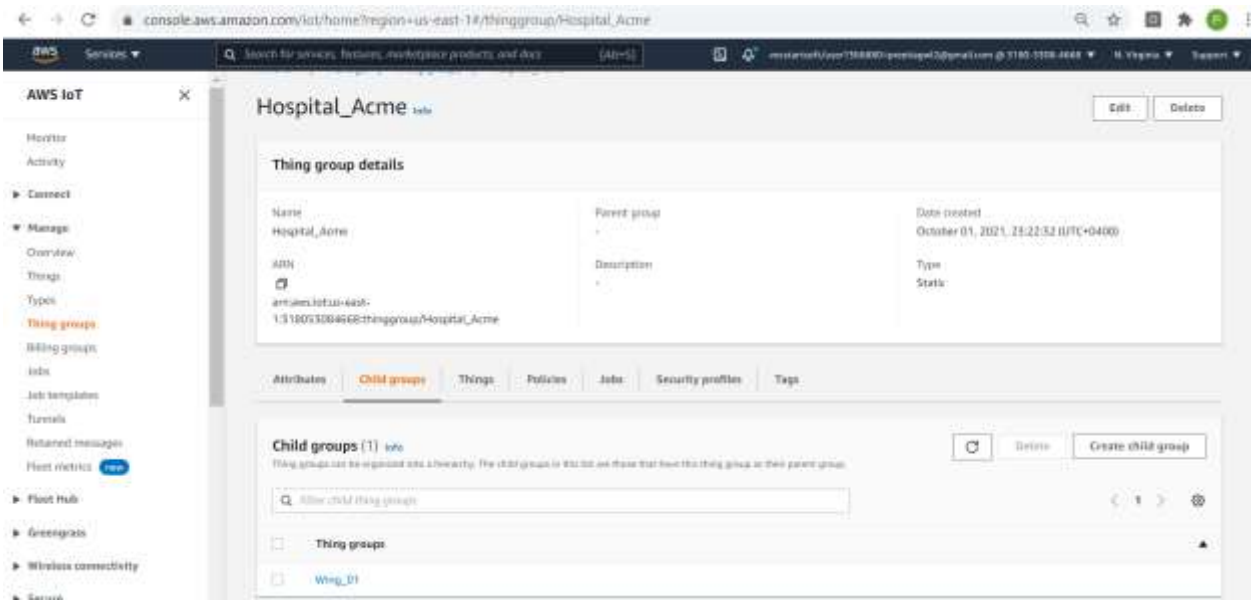


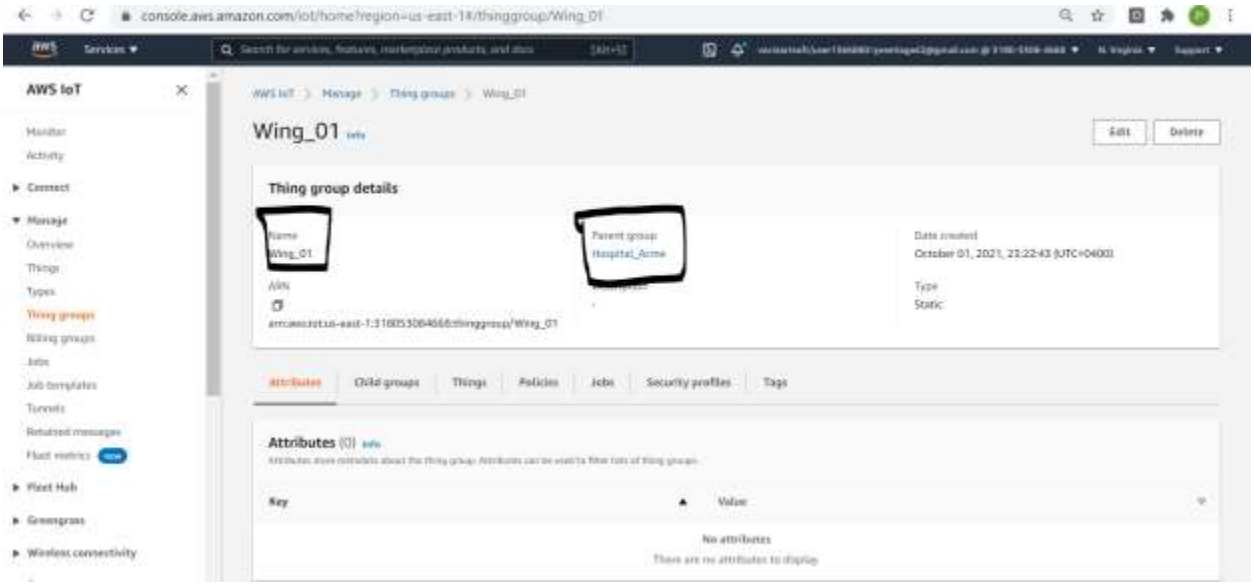
<Insert Screenshot a(2) here>

Parent group –



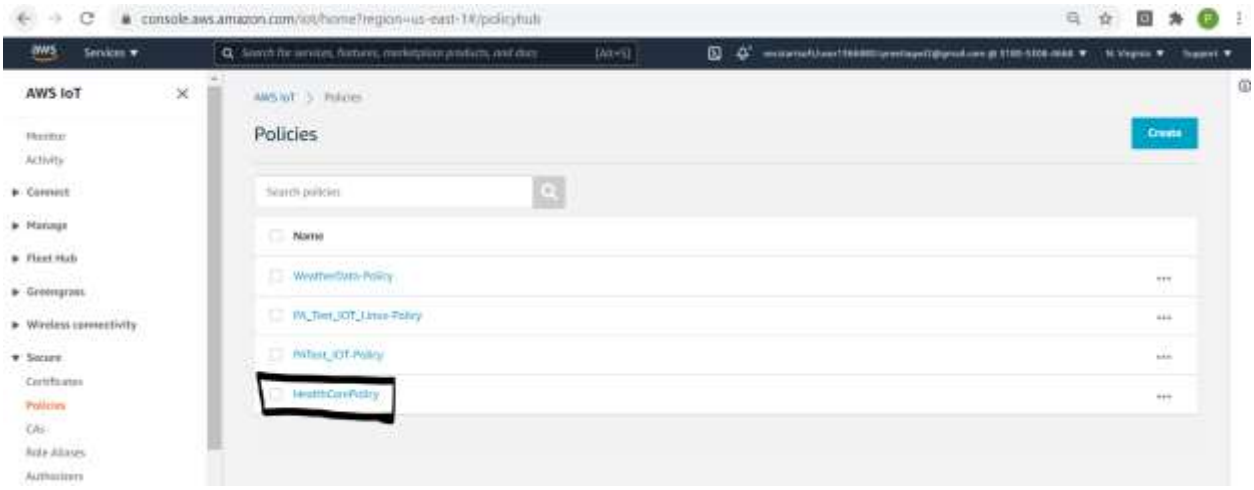
Child group –

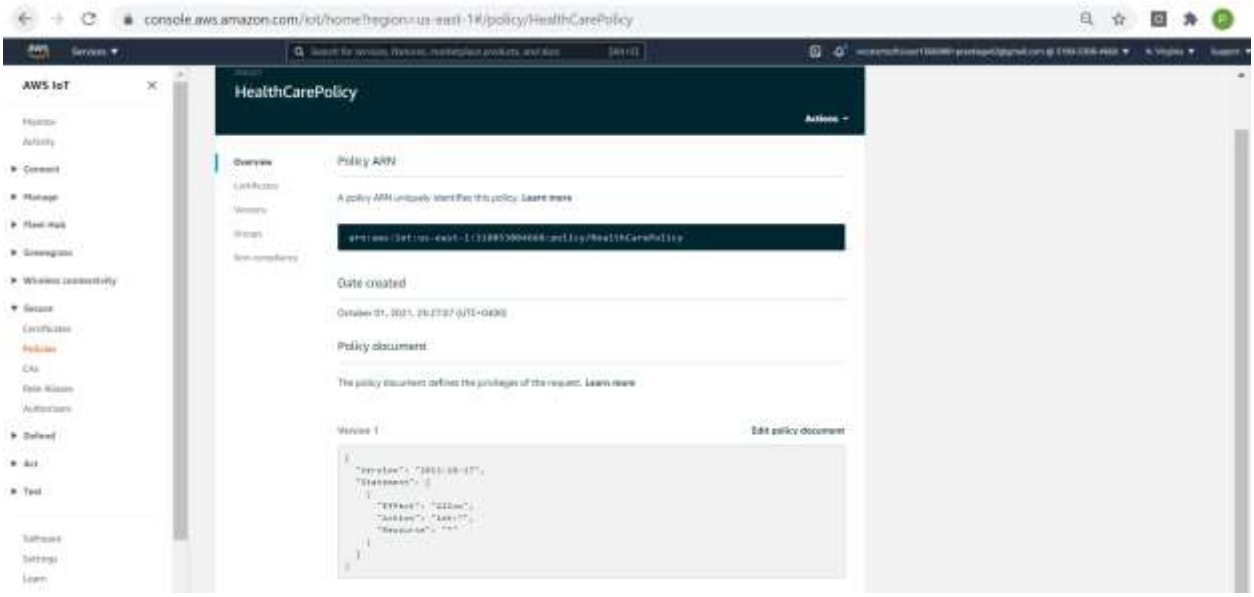



console.aws.amazon.com/iot/home?region=us-east-1#/thinggroup/Wing_01
AWS IoT
Monitor
Activity
Connect
Manage
Overview
Things
Types
Thing groups
Billing groups
Jobs
Job templates
Tunnels
Retained messages
Fleet metrics
Fleet Hub
Greengrass
Wireless connectivity
AWS IoT > Manage > Thing groups > Wing_01
Wing_01
Edit
Delete
Thing group details
Name
Wing_01
Parent group
Hospital_Acme
Data created
October 01, 2021, 23:22:43 (UTC+0400)
ARN
Type
Static
Attributes
Child groups
Things
Policies
Jobs
Security profiles
Tags
Attributes (0)
Key
Value
No attributes

<Insert Screenshot a(3) here>

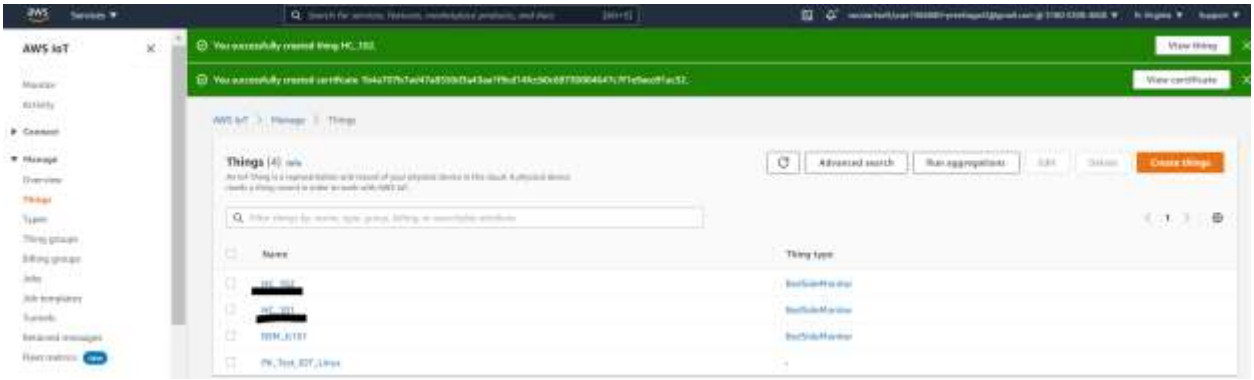
Policy-

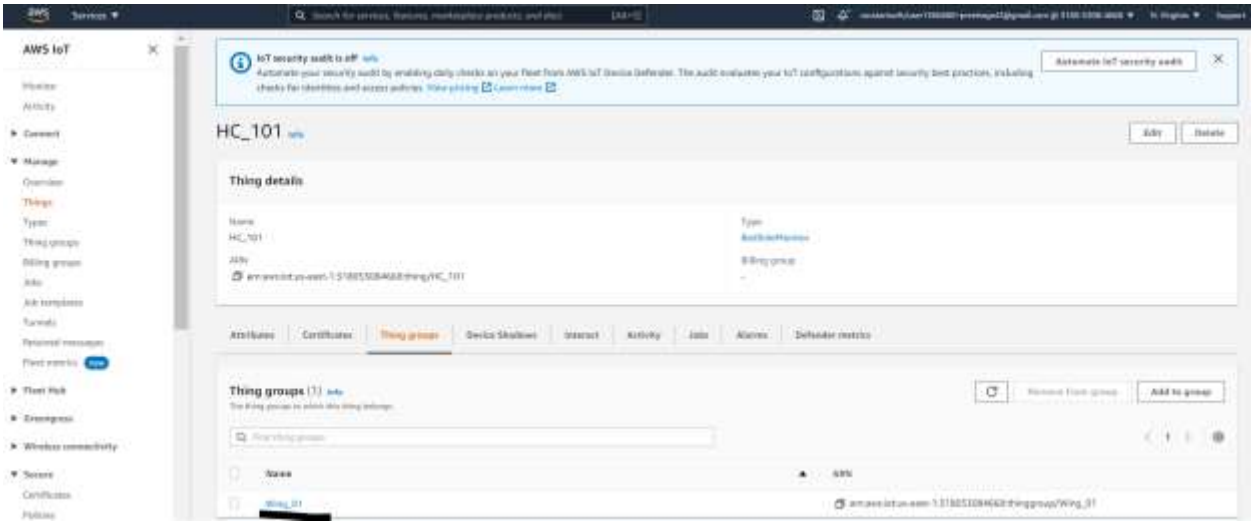


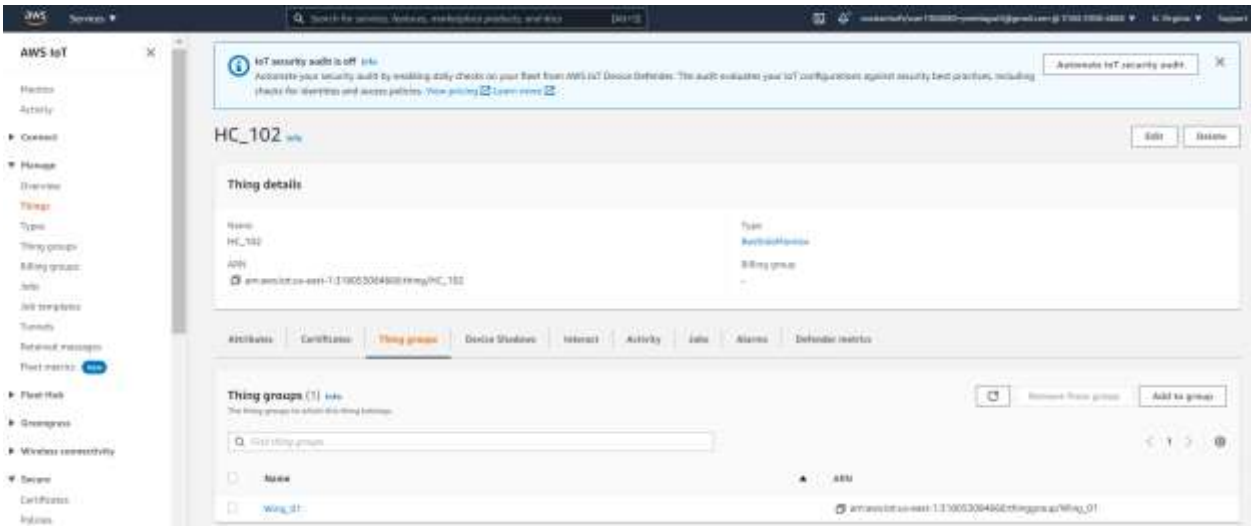


<Insert Screenshot a(4) here>

Things –

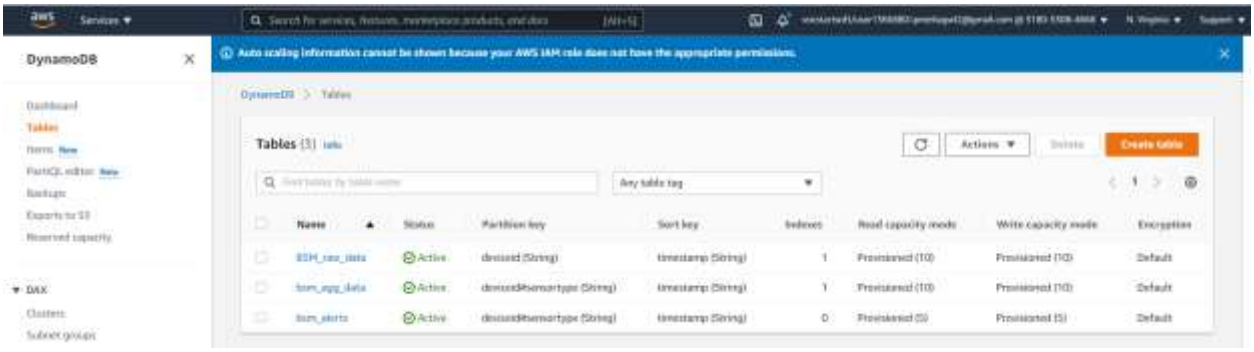


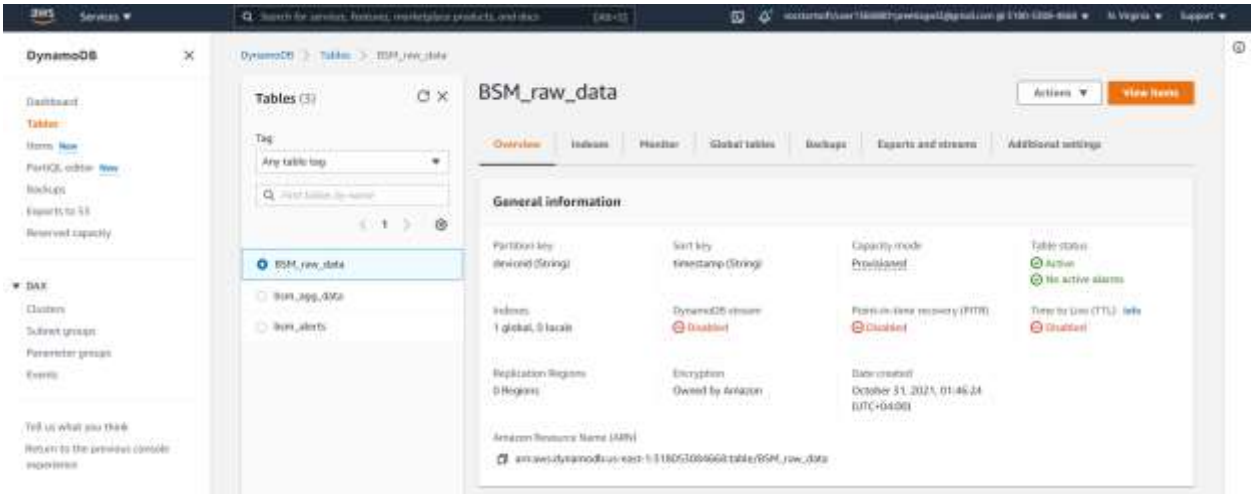




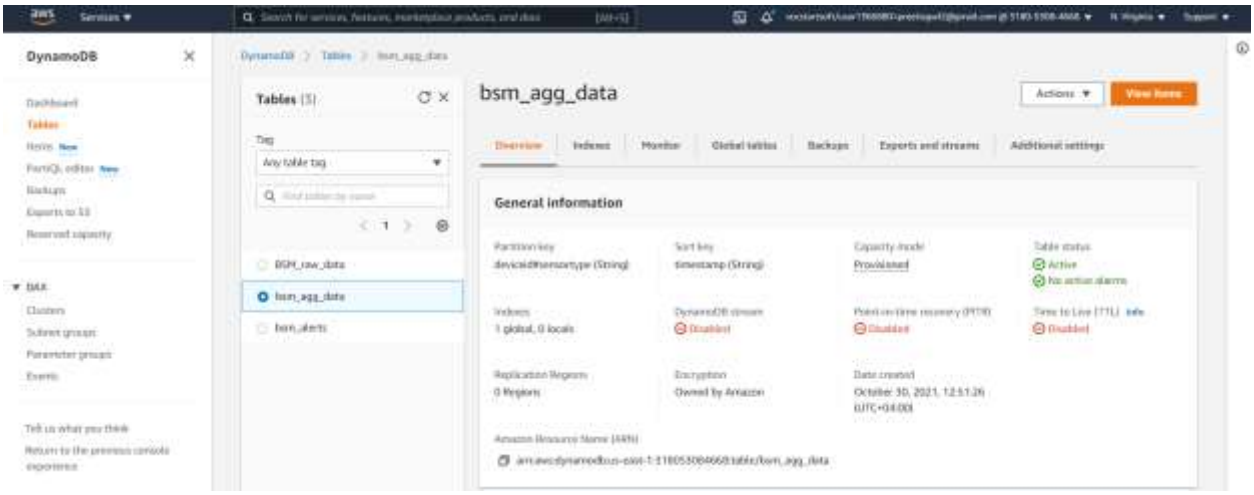
Step number	b
Step name	DynamoDB
Instructions	1) Goto DynamoDB console or use python script to create the table to store the raw_data, aggregate_data and anomaly-data with appropriate partition key and sort key 2) Assign the name to the table to store the raw data 3) The name of the tables should be clearly visible in the screenshot.
Expected screenshots	1. Screenshot of successfully created table to hold the raw data. (Empty table) 2. Screenshot of successfully created table to hold the aggregated/ data. (Empty table) 3. Screenshot of successfully created table to hold the anomaly data. (Empty table)

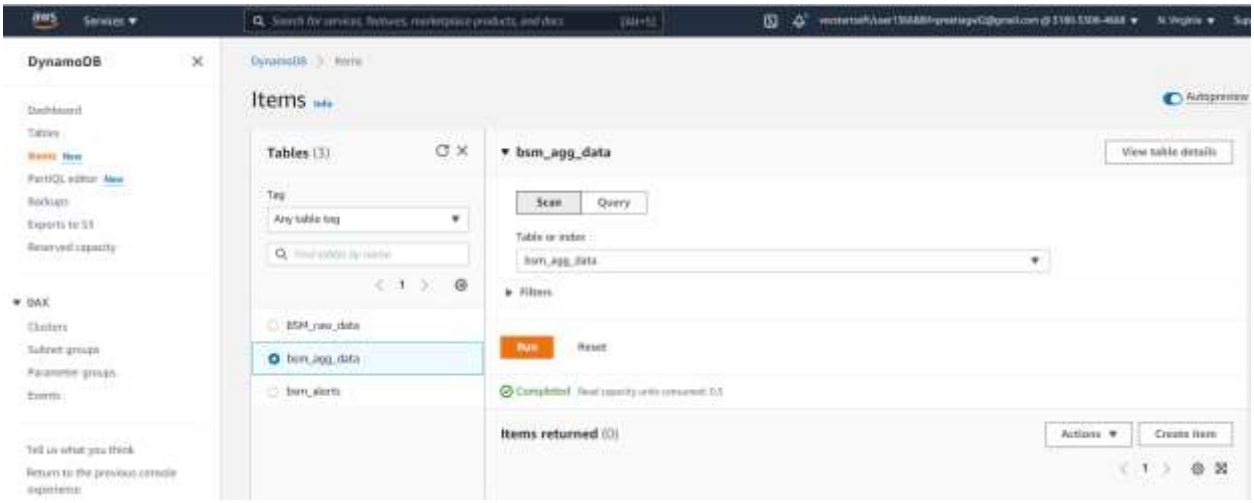
<Insert Screenshot b(1) here>



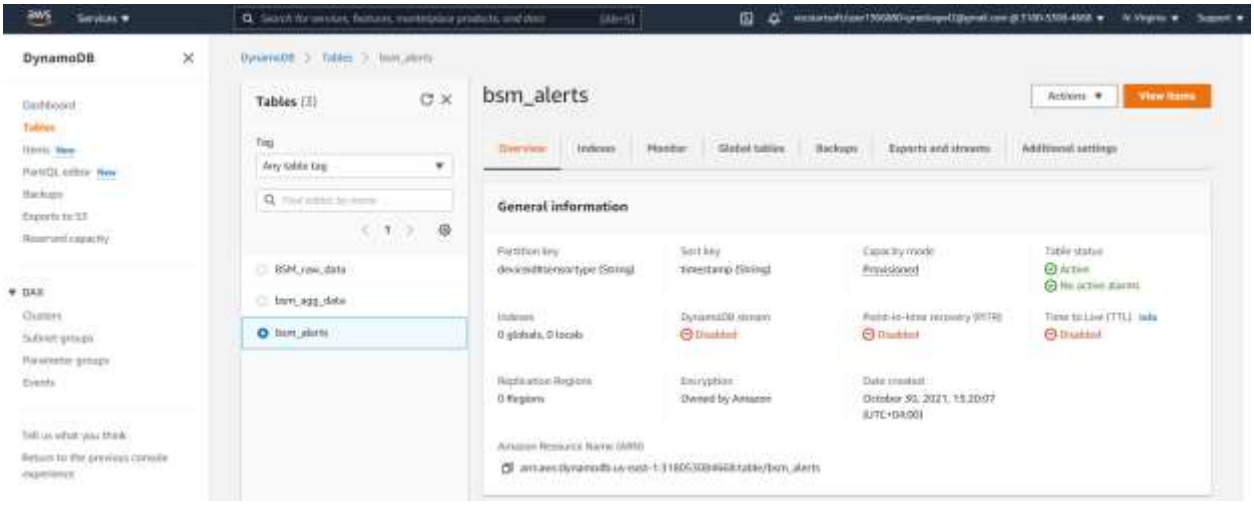


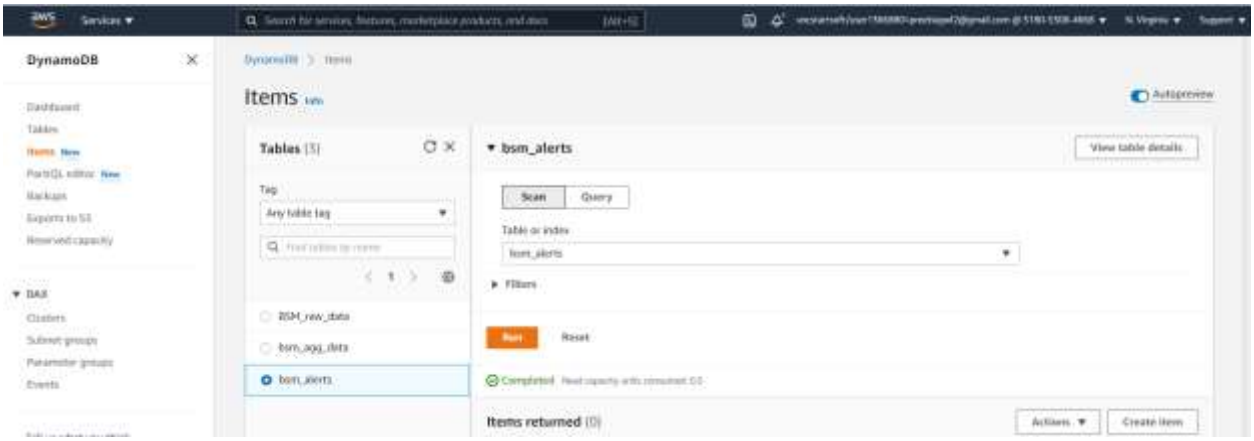
<Insert Screenshot b(2) here>





<Insert Screenshot b(3) here>

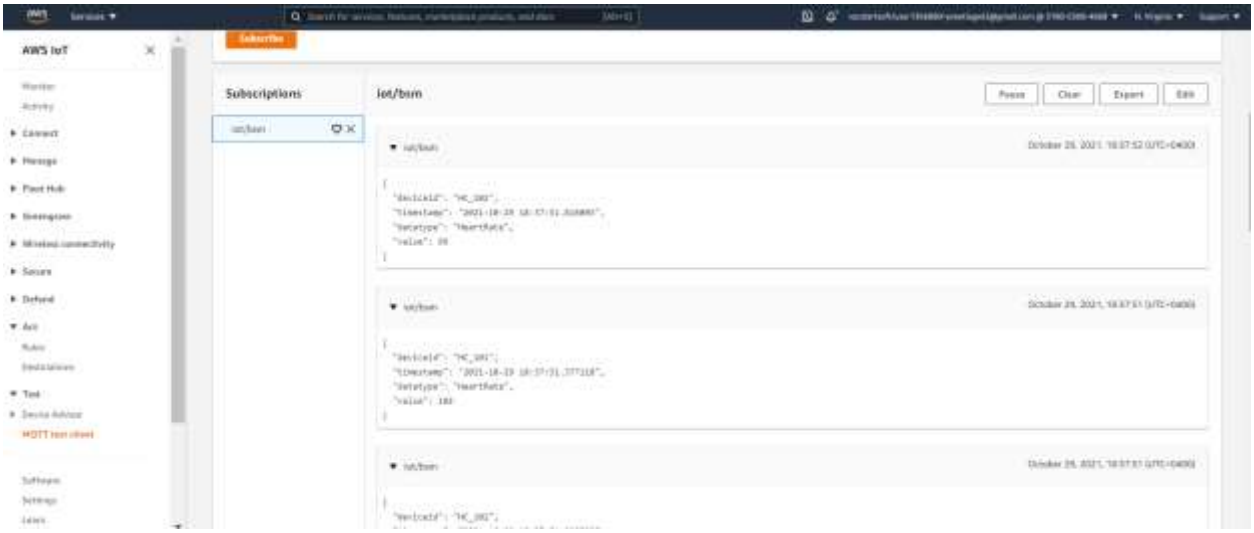




Step number	c
Step name	Rule creation to push raw data
Instructions	1) Goto IoT core and create rule to push simulated data in the raw data table created in the above step 2) Once the rule is created, run the python script to push data on AWS MQTT. If the created rule is valid then data will be available in the previously created table.
Expected screenshots	1. Screenshot of successfully created rule to push the raw data 2. Screenshot of data available in the AWS MQTT

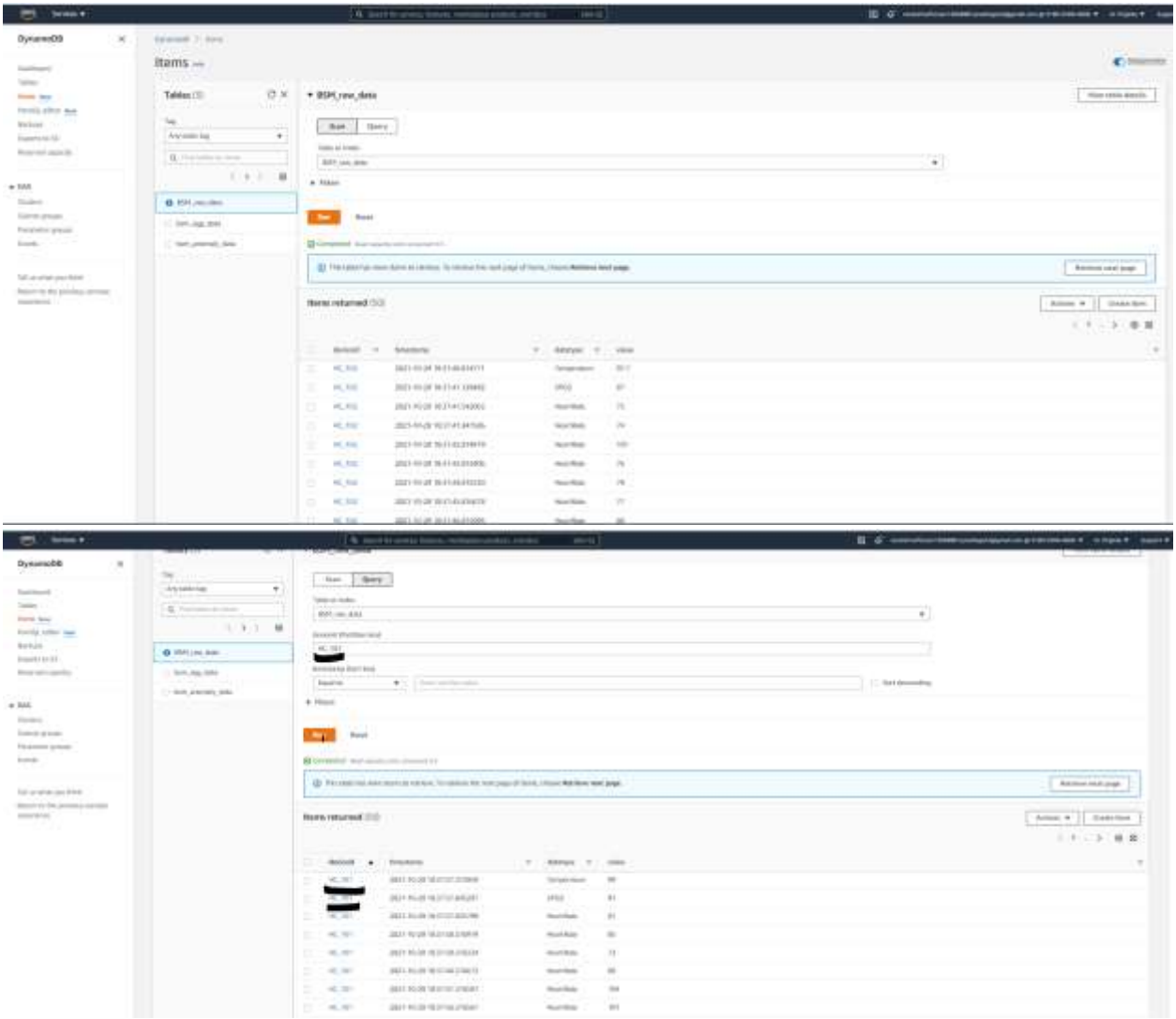
<Insert Screenshot c(1) here>

<Insert Screenshot c(2) here>



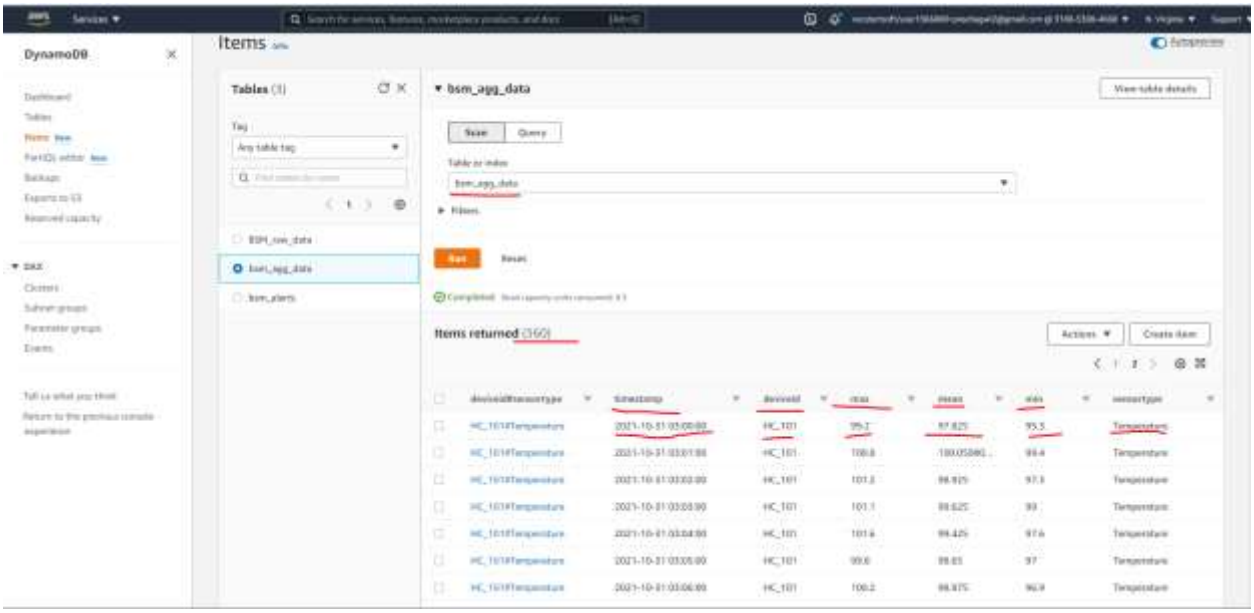
Step number	d
Step name	Data in the DynamoDB table
Instructions	1) Once the tables are created and raw data rule is created to push the data 2) Perform aggregation as mentioned in Task-2 of the problem statement, once done the data should be there in the newly created table. 3) Perform anomaly as per the task-3 available in the problem statement and then push the data in the newly created table.
Expected screenshots	1. Screenshot of populated raw data in the table 2. Screenshot of populated aggregated data in the table 3. Screenshot of populated anomaly data in the table

<Insert Screenshot d(1) here>

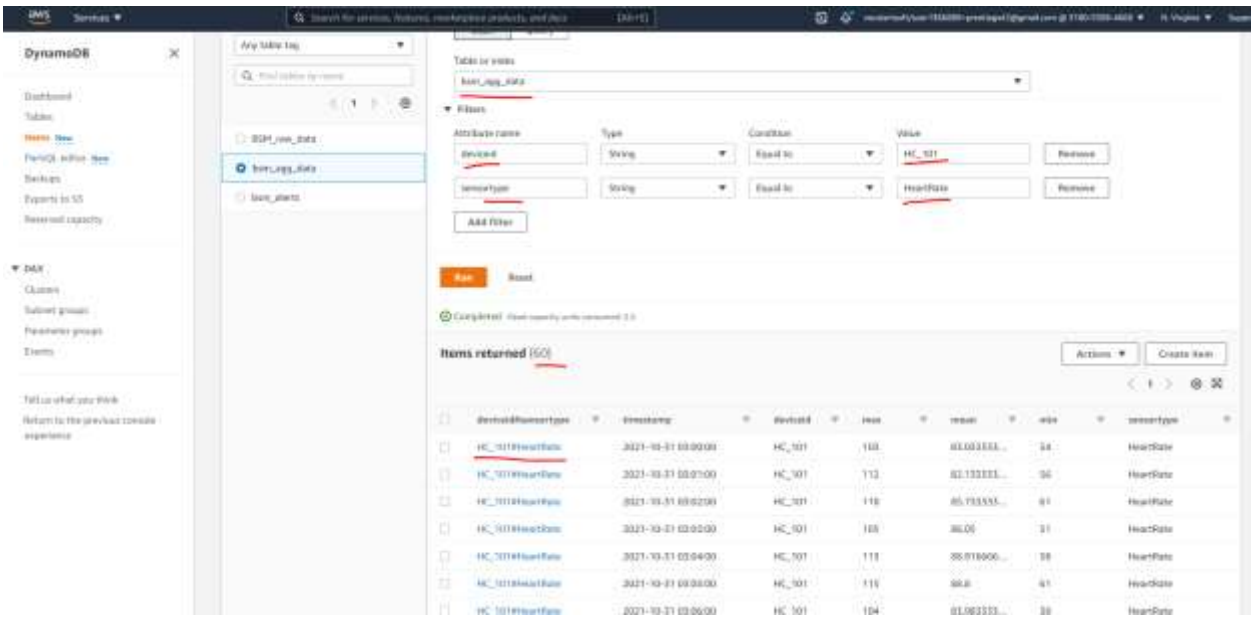


<Insert Screenshot d(2) here>

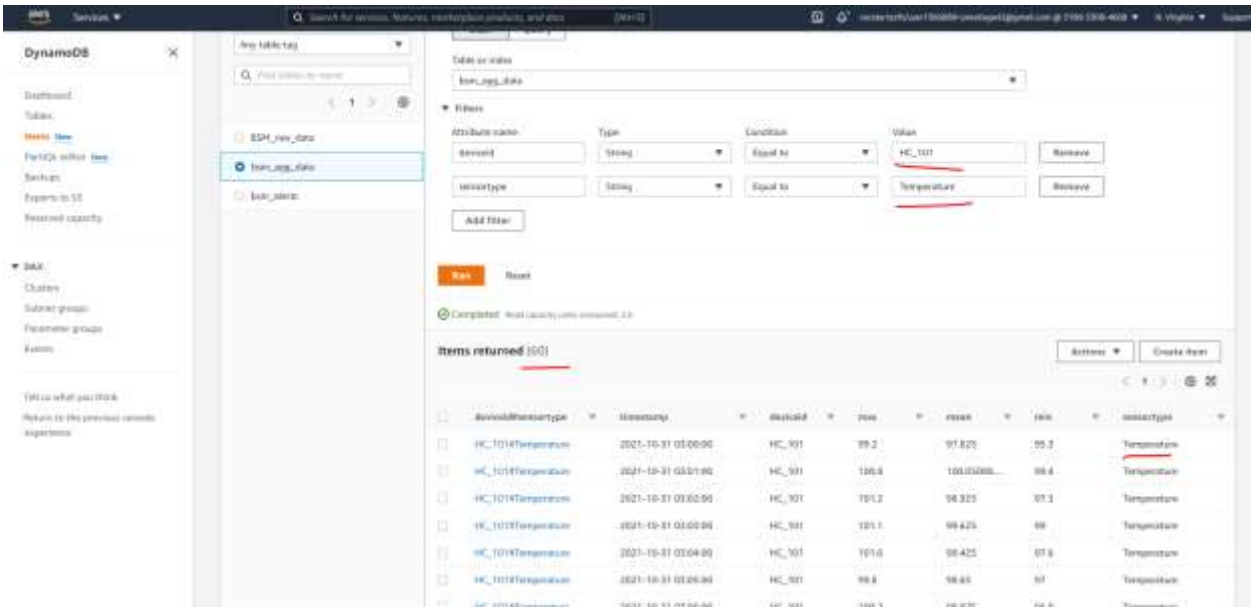
Total 360 records inserted into bsm_agg_data table for 2 devices HC_101 and HC_102 for the 3 sensor types – Heartrate, Temperature and SPO2 for 1 hour.
The table has composite partition key of 'deviceid#sensortype' and sort key of 'timestamp'



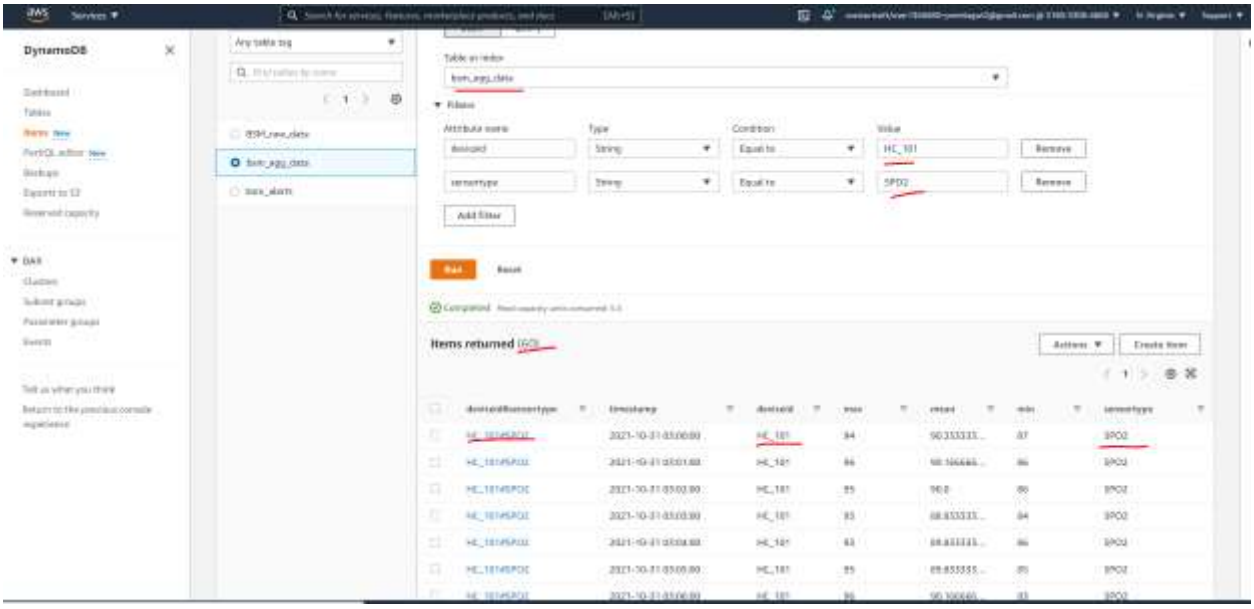
HeartRate data – 60 records for HeartRate aggregation inserted into the bsm_agg_data table as calculated with start time '10/31/2021 03:00:00' and end time '10/31/2021 04:00:00' for device HC_101. Similar created for HC_102.



Temperature data – 60 records for Temperature aggregation inserted into the bsm_agg_data table as calculated with start time '10/29/2021 18:37:00' and end time '10/29/2021 19:37:00' for device HC_101. Similar created for HC_102.



SPO2 data – 60 records for SPO2 aggregation inserted into the bsm_agg_data table as calculated with start time '10/29/2021 18:37:00' and end time '10/29/2021 19:37:00' for device HC_101. Similar created for HC_102.

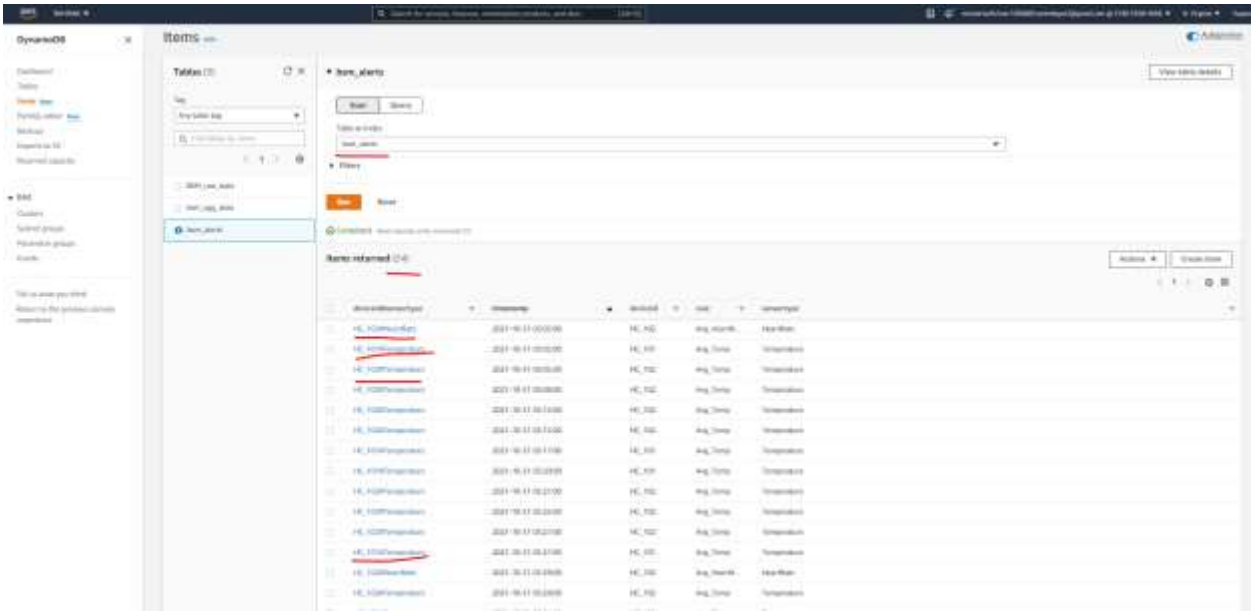


Whole aggregation data for all sensor types for 1 hour from bsm_agg_data table is fetched into CSV from AWS and attached below.

bsm_agg_data.csv

<Insert Screenshot d(3) here>

Bsm_alerts
24 records inserted into the table as anomalies according to the rules defined in rules.json file.



Bsm_alerts data for all sensor types for 1 hour from bsm_alerts table is fetched into CSV from AWS and attached below.

bsm_alerts.csv