Building a Real-Time IoT Data Pipeline
AWS Kinesis, Lambda, S3, Quick Sight
[Bootcamp - AWS Data Engineering Project - 4]

Project Overview: Build a real-time data pipeline to monitor factory machinery using IoT sensor data. The pipeline should ingest real-time sensor streams, detect anomalies, store data in a layered data lake (Bronze, Silver, Gold), and provide operational dashboards and alerts.

The pipeline provides:

- Real-time anomaly detection – Immediate alerts for abnormal machine conditions
- Historical analysis – Trend identification for predictive maintenance
- Operational efficiency – Automated data processing and storage
- Decision support – Visual dashboards for equipment health monitoring

The solution leverages a scalable, serverless architecture to process streaming IoT data while optimizing costs.

IAM Roles:

1. Managed Flink:
   a. AmazonKinesisFullAccess
   b. CloudWatchLogsFullAccess
   c. AmazonKinesisAnalyticsFullAccess
   d. AWSGlueConsoleFullAccess

2. **SNS:**
   a. AmazonKinesisFullAccess
   b. CloudWatchLogsFullAccess
   c. AmazonKinesisAnalyticsFullAccess

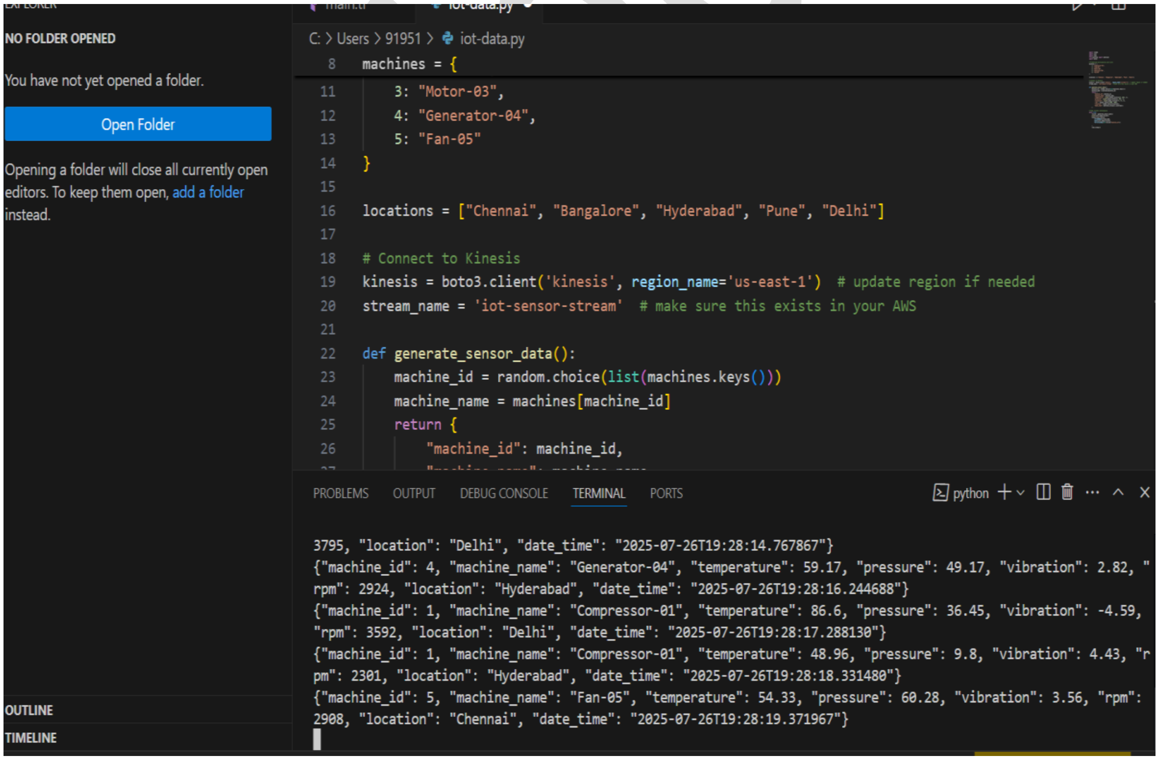
d. AmazonSNSFullAccess
e. AWSLambda_FullAccess
f. CloudWatchEventsFullAccess

3. **Glue**:
a. AmazonAthenaFullAccess
b. AmazonS3FullAccess
c. AWSGlueServiceRole
d. AWSQuicksightAthenaAccess
e. CloudWatchEventsFullAccess

Data Ingestion - Kinesis

Simulate IoT sensor data and send it to an AWS Kinesis Data Stream (`sensor-stream`) using a Python generator script. Each record includes temperature, pressure, vibration, rpm, location, and timestamp.

1. Create a S3 bucket and folders bronze, gold and silver.
2. “myp4-hema-b”, objects: silver, gold and bronze.
3. Generate a stream in your local machine and stream into kinesis.


NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
OUTLINE
TIMELINE
main.tf
iot-data.py
C: > Users > 91951 > iot-data.py
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
python

```python
machines = {
    3: "Motor-03",
    4: "Generator-04",
    5: "Fan-05"
}

locations = ["Chennai", "Bangalore", "Hyderabad", "Pune", "Delhi"]

# Connect to Kinesis
kinesis = boto3.client('kinesis', region_name='us-east-1')  # update region if needed
stream_name = 'iot-sensor-stream'  # make sure this exists in your AWS

def generate_sensor_data():
    machine_id = random.choice(list(machines.keys()))
    machine_name = machines[machine_id]
    return {
        "machine_id": machine_id,
```

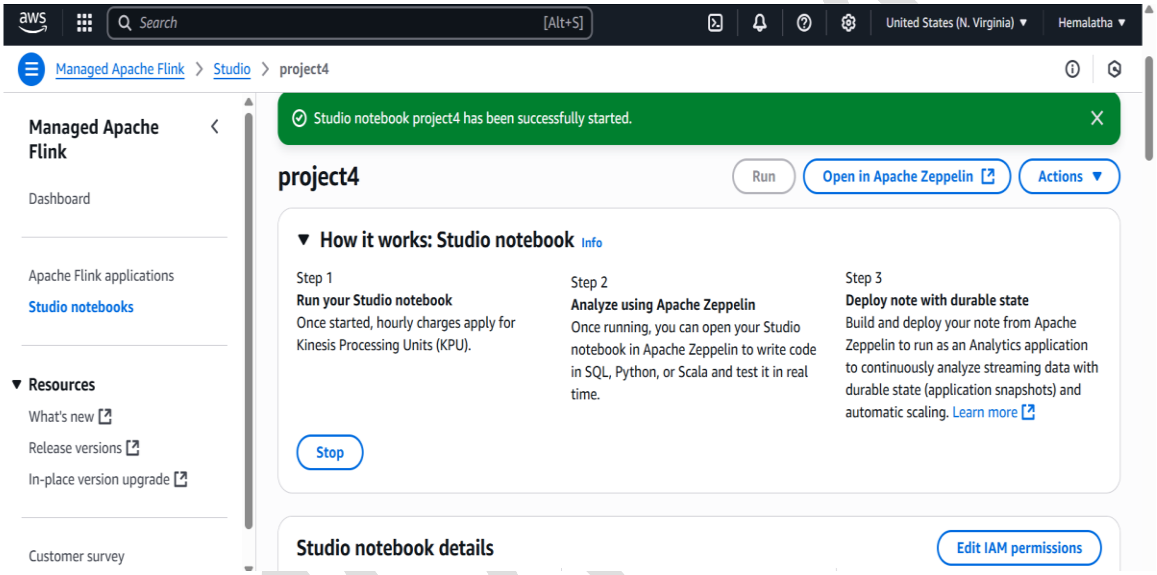
```
3795, "location": "Delhi", "date_time": "2025-07-26T19:28:14.767867"}
{"machine_id": 4, "machine_name": "Generator-04", "temperature": 59.17, "pressure": 49.17, "vibration": 2.82, "
rpm": 2924, "location": "Hyderabad", "date_time": "2025-07-26T19:28:16.244688"}
{"machine_id": 1, "machine_name": "Compressor-01", "temperature": 86.6, "pressure": 36.45, "vibration": -4.59,
"rpm": 3592, "location": "Delhi", "date_time": "2025-07-26T19:28:17.288130"}
{"machine_id": 1, "machine_name": "Compressor-01", "temperature": 48.96, "pressure": 9.8, "vibration": 4.43, "r
pm": 2301, "location": "Hyderabad", "date_time": "2025-07-26T19:28:18.331480"}
{"machine_id": 5, "machine_name": "Fan-05", "temperature": 54.33, "pressure": 60.28, "vibration": 3.56, "rpm":
2908, "location": "Chennai", "date_time": "2025-07-26T19:28:19.371967"}
```

Stream Processing - Apache Flink (Managed)

Use Apache Flink Studio Notebook to define real-time transformation on the incoming stream.

Key transformations:

- Calculate average metrics per minute
- Flag high temperatures
- Output streaming results to new Kinesis streams.



Studio notebook project4 has been successfully started.

project4

▼ How it works: Studio notebook

Step 1
Run your Studio notebook
Once started, hourly charges apply for Kinesis Processing Units (KPU).

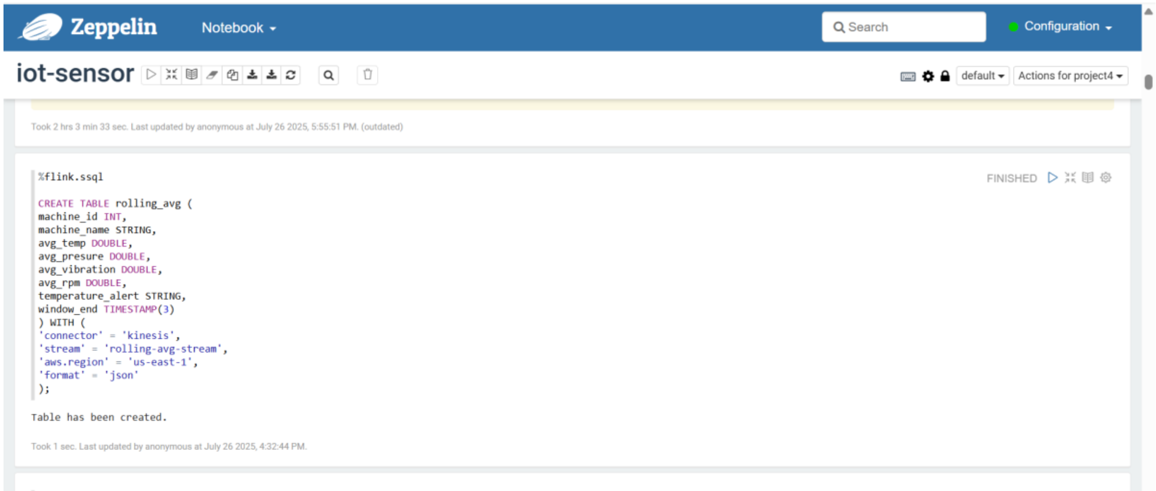
Step 2
Analyze using Apache Zeppelin
Once running, you can open your Studio notebook in Apache Zeppelin to write code in SQL, Python, or Scala and test it in real time.

Step 3
Deploy note with durable state
Build and deploy your note from Apache Zeppelin to run as an Analytics application to continuously analyze streaming data with durable state (application snapshots) and automatic scaling. Learn more

Studio notebook details

Data Lake Design (Bronze - Silver - Gold)

- Bronze: Raw data is ingested into S3 via Kinesis Firehose (prefix: `bronze/`)
- Silver: Processed stream with aggregated averages (prefix: `silver/`)
- Gold: Final Delta Lake output written using AWS Glue jobs with PySpark.



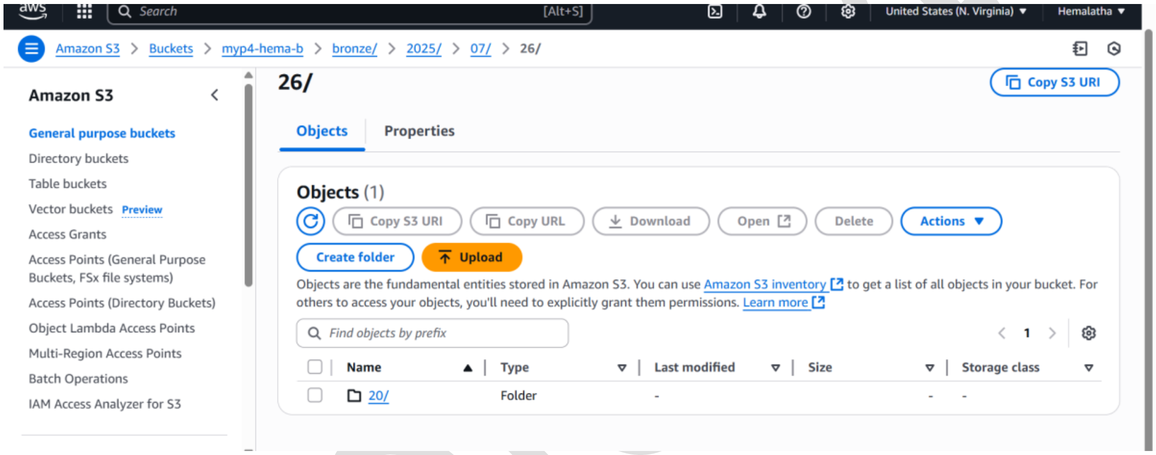
Took 2 hrs 3 min 33 sec. Last updated by anonymous at July 26 2025, 5:55:51 PM. (outdated)

```
%flink.ssql

CREATE TABLE rolling_avg (
machine_id INT,
machine_name STRING,
avg_temp DOUBLE,
avg_presure DOUBLE,
avg_vibration DOUBLE,
avg_rpm DOUBLE,
temperature_alert STRING,
window_end TIMESTAMP(3)
) WITH (
'connector' = 'kinesis',
'stream' = 'rolling-avg-stream',
'aws.region' = 'us-east-1',
'format' = 'json'
);
```

```
Table has been created.
```

Took 1 sec. Last updated by anonymous at July 26 2025, 4:32:44 PM.



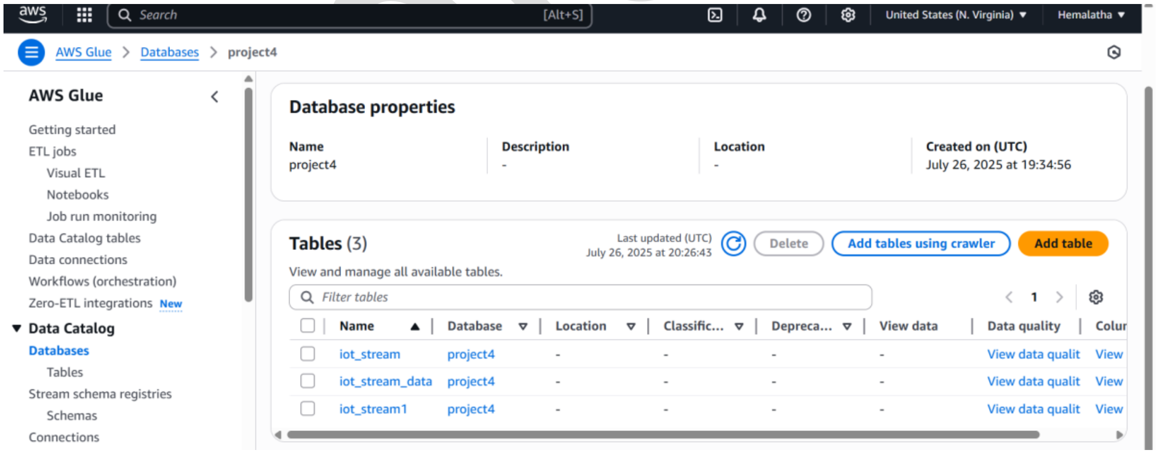
aws
Search
[Alt+S]
United States (N. Virginia)
Hemalatha
Amazon S3 > Buckets > myp4-hema-b > bronze/ > 2025/ > 07/ > 26/
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets Preview
Access Grants
Access Points (General Purpose Buckets, FSx file systems)
Access Points (Directory Buckets)
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Copy S3 URI
Objects
Properties
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix

26/

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
20/	Folder	-	-	-



Database properties

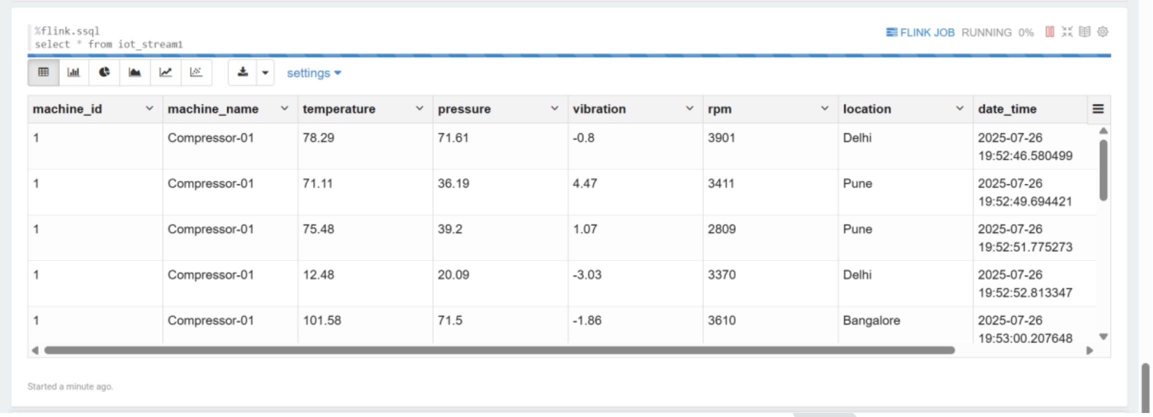
Name	Description	Location	Created on (UTC)
project4	-	-	July 26, 2025 at 19:34:56

Tables (3)

Last updated (UTC)
July 26, 2025 at 20:26:43

View and manage all available tables.

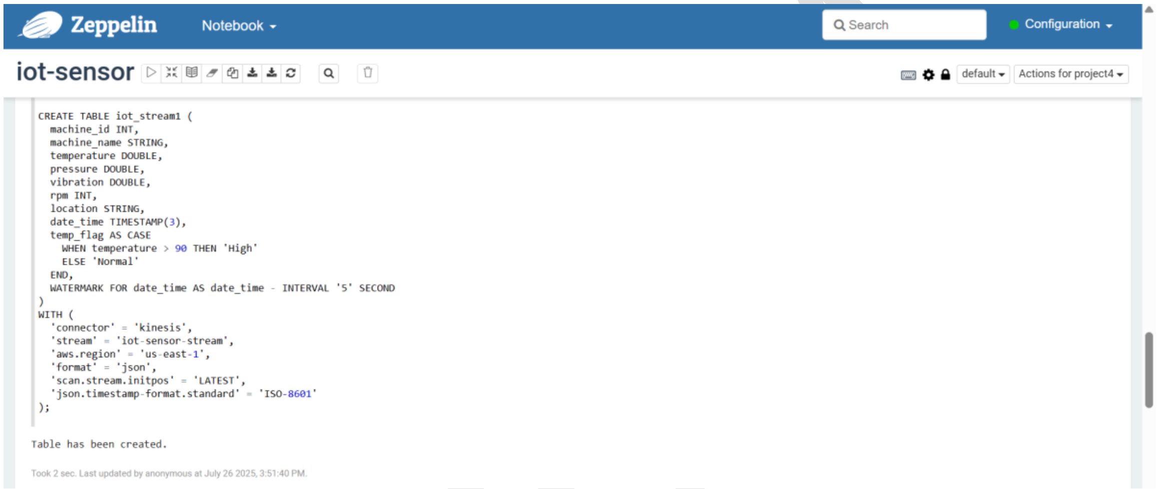
Name	Database	Location	Classific...	Depreca...	View data	Data quality	Colur
iot_stream	project4	-	-	-	-	View data qualit	View
iot_stream_data	project4	-	-	-	-	View data qualit	View
iot_stream1	project4	-	-	-	-	View data qualit	View



```
%flink.ssql
select * from iot_stream1
```

machine_id	machine_name	temperature	pressure	vibration	rpm	location	date_time
1	Compressor-01	78.29	71.61	-0.8	3901	Delhi	2025-07-26 19:52:46.580499
1	Compressor-01	71.11	36.19	4.47	3411	Pune	2025-07-26 19:52:49.694421
1	Compressor-01	75.48	39.2	1.07	2809	Pune	2025-07-26 19:52:51.775273
1	Compressor-01	12.48	20.09	-3.03	3370	Delhi	2025-07-26 19:52:52.813347
1	Compressor-01	101.58	71.5	-1.86	3610	Bangalore	2025-07-26 19:53:00.207648

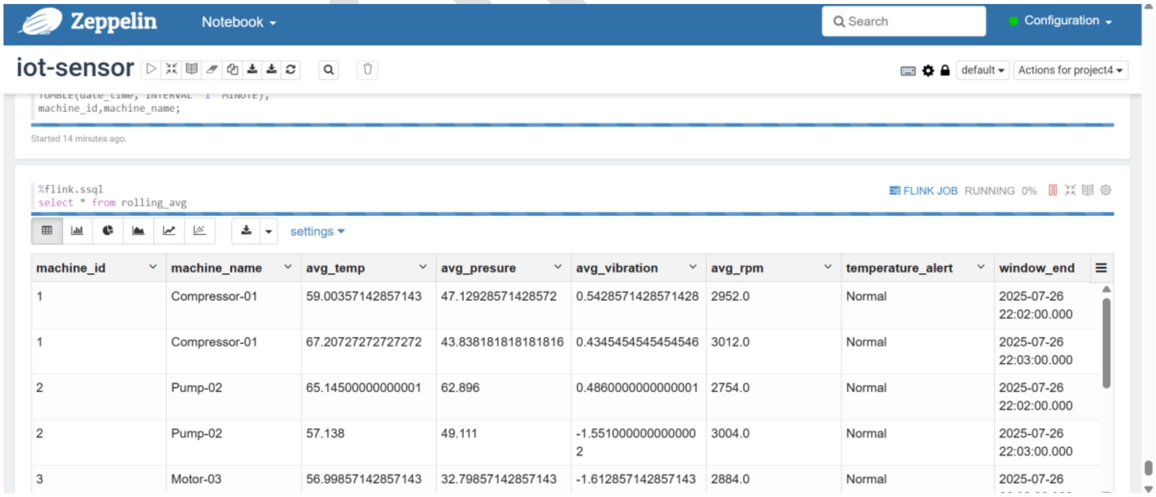
Started a minute ago.



```
CREATE TABLE iot_stream1 (
  machine_id INT,
  machine_name STRING,
  temperature DOUBLE,
  pressure DOUBLE,
  vibration DOUBLE,
  rpm INT,
  location STRING,
  date_time TIMESTAMP(3),
  temp_flag AS CASE
    WHEN temperature > 90 THEN 'High'
    ELSE 'Normal'
  END,
  WATERMARK FOR date_time AS date_time - INTERVAL '5' SECOND
)
WITH (
  'connector' = 'kinesis',
  'stream' = 'iot-sensor-stream',
  'aws.region' = 'us-east-1',
  'format' = 'json',
  'scan.stream.initpos' = 'LATEST',
  'json.timestamp-format.standard' = 'ISO-8601'
);
```

```
Table has been created.
```

Took 2 sec. Last updated by anonymous at July 26 2025, 3:51:40 PM.

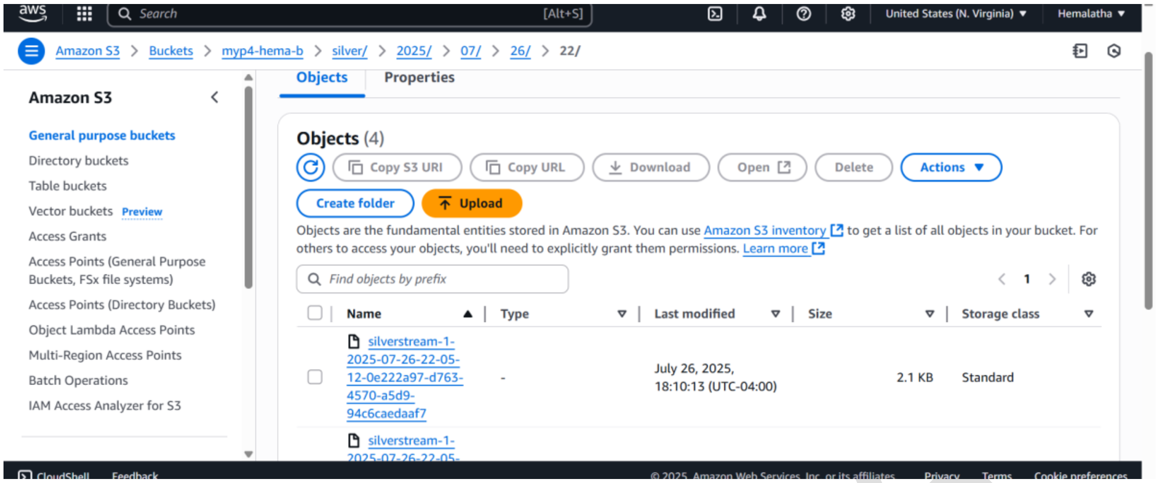


```
TUMBLE(date_time, INTERVAL '1' MINUTE),
machine_id,machine_name;
```

Started 14 minutes ago.

```
%flink.ssql
select * from rolling_avg
```

machine_id	machine_name	avg_temp	avg_presure	avg_vibration	avg_rpm	temperature_alert	window_end
1	Compressor-01	59.00357142857143	47.12928571428572	0.5428571428571428	2952.0	Normal	2025-07-26 22:02:00.000
1	Compressor-01	67.20727272727272	43.838181818181816	0.4345454545454546	3012.0	Normal	2025-07-26 22:03:00.000
2	Pump-02	65.14500000000001	62.896	0.4860000000000001	2754.0	Normal	2025-07-26 22:02:00.000
2	Pump-02	57.138	49.111	-1.5510000000000002	3004.0	Normal	2025-07-26 22:03:00.000
3	Motor-03	56.99857142857143	32.79857142857143	-1.612857142857143	2884.0	Normal	2025-07-26


aws
Search
[Alt+S]
United States (N. Virginia)
Hemalatha
Amazon S3 > Buckets > myp4-hema-b > silver/ > 2025/ > 07/ > 26/ > 22/
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Vector buckets Preview
Access Grants
Access Points (General Purpose Buckets, FSx file systems)
Access Points (Directory Buckets)
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Objects
Properties
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

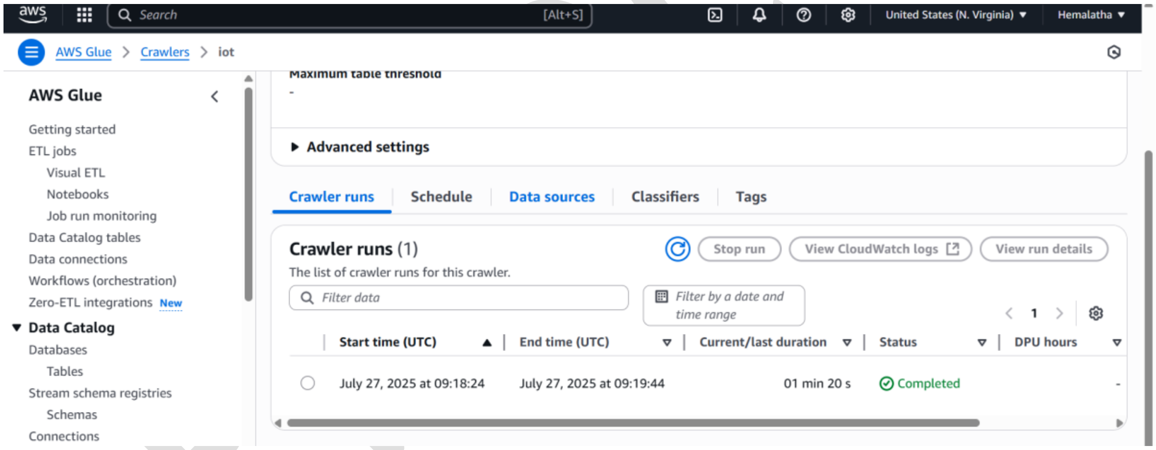
Objects (4)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
silverstream-1-2025-07-26-22-05-12-0e222a97-d763-4570-a5d9-94c6caedaaf7	-	July 26, 2025, 18:10:13 (UTC-04:00)	2.1 KB	Standard
silverstream-1-2025-07-26-22-05-				

ETL with AWS Glue

Configure Glue jobs to read from Silver S3 (Parquet format), perform transformations, and write Delta format output to gold layer in S3.

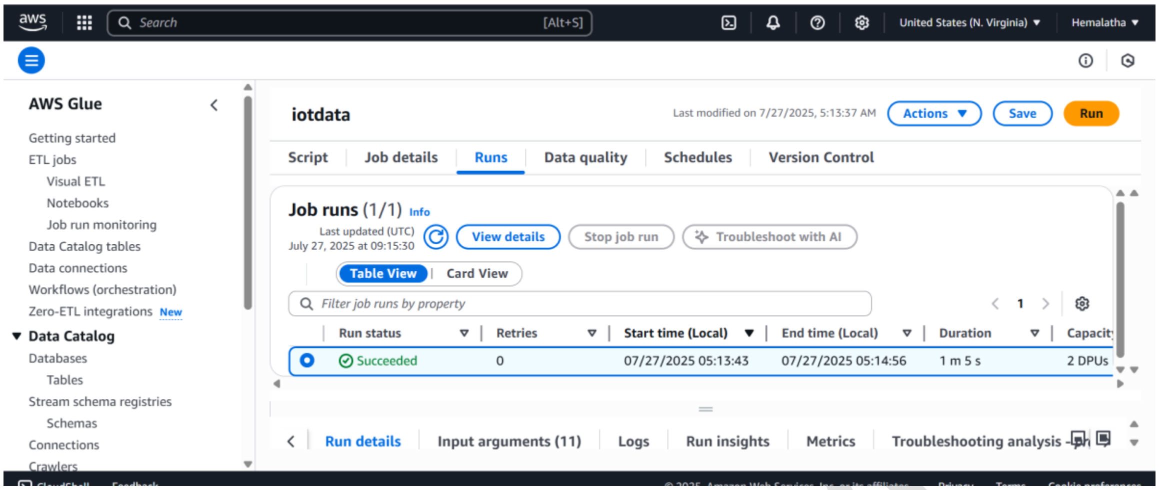


Maximum table threshold
-

Crawler runs (1)

The list of crawler runs for this crawler.

Start time (UTC)	End time (UTC)	Current/last duration	Status	DPU hours
July 27, 2025 at 09:18:24	July 27, 2025 at 09:19:44	01 min 20 s	Completed	-



iotdata

Last modified on 7/27/2025, 5:13:37 AM

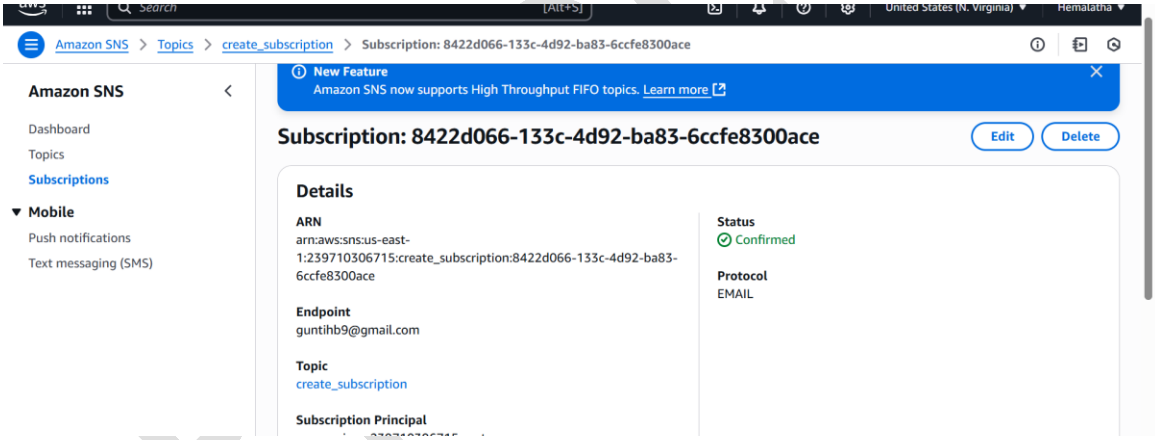
Job runs (1/1) Info

Last updated (UTC)
July 27, 2025 at 09:15:30

	Run status	Retries	Start time (Local)	End time (Local)	Duration	Capacity
●	✓ Succeeded	0	07/27/2025 05:13:43	07/27/2025 05:14:56	1 m 5 s	2 DPUs

Alerts via Lambda + SNS

Deploy a Lambda function triggered by Kinesis Data Stream to monitor `temperature > 90°C` and send real-time alerts to email via SNS.


aws
Search [Alt+S]
United States (N. Virginia) ▾
Hemalatha ▾
Amazon SNS > Topics > create_subscription > Subscription: 8422d066-133c-4d92-ba83-6ccfe8300ace
Amazon SNS
Dashboard
Topics
Subscriptions
▾ Mobile
Push notifications
Text messaging (SMS)
Edit
Delete

New Feature
Amazon SNS now supports High Throughput FIFO topics. Learn more

Subscription: 8422d066-133c-4d92-ba83-6ccfe8300ace

Details

ARN
arn:aws:sns:us-east-1:239710306715:create_subscription:8422d066-133c-4d92-ba83-6ccfe8300ace

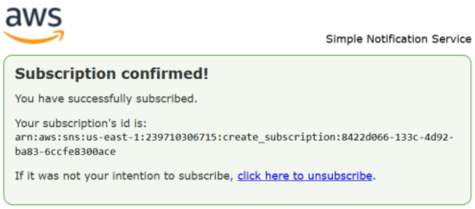
Endpoint
guntihb9@gmail.com

Topic
create_subscription

Subscription Principal

Status
✓ Confirmed

Protocol
EMAIL



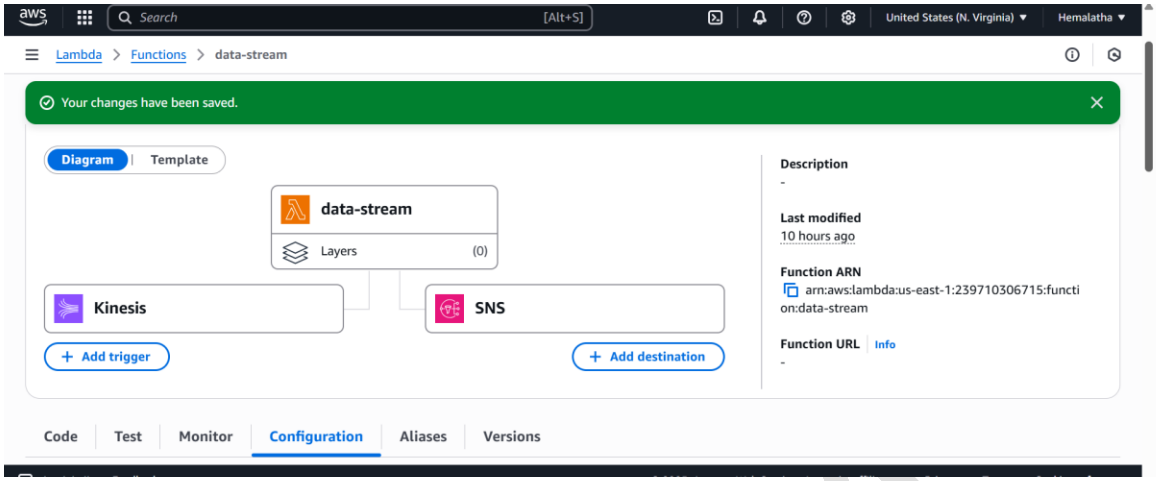
Simple Notification Service

Subscription confirmed!

You have successfully subscribed.

Your subscription's id is:
arn:aws:sns:us-east-1:239710306715:create_subscription:8422d066-133c-4d92-ba83-6ccfe8300ace

If it was not your intention to subscribe, click here to unsubscribe.



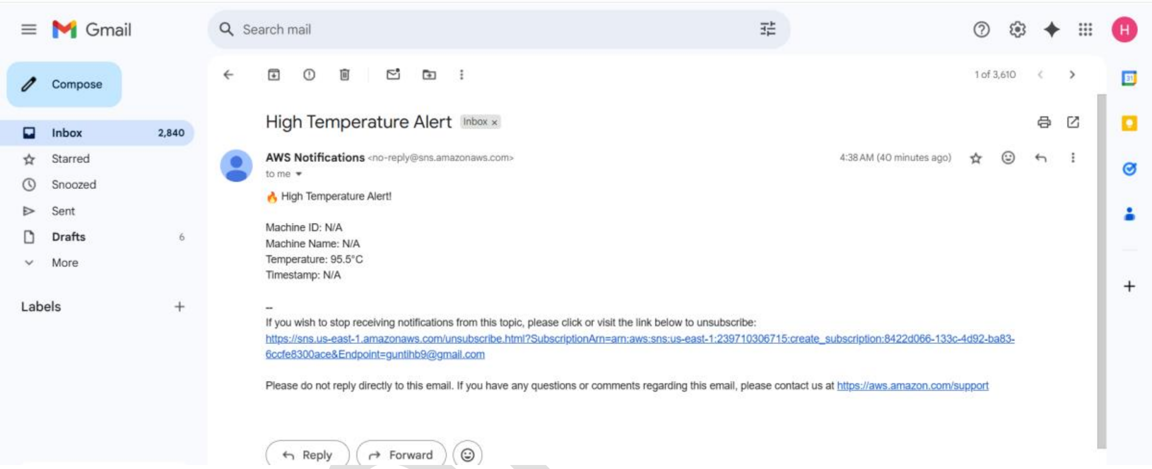
Your changes have been saved.

Description
-

Last modified
10 hours ago

Function ARN
arn:aws:lambda:us-east-1:239710306715:function:data-stream

Function URL Info
-



High Temperature Alert Inbox ×

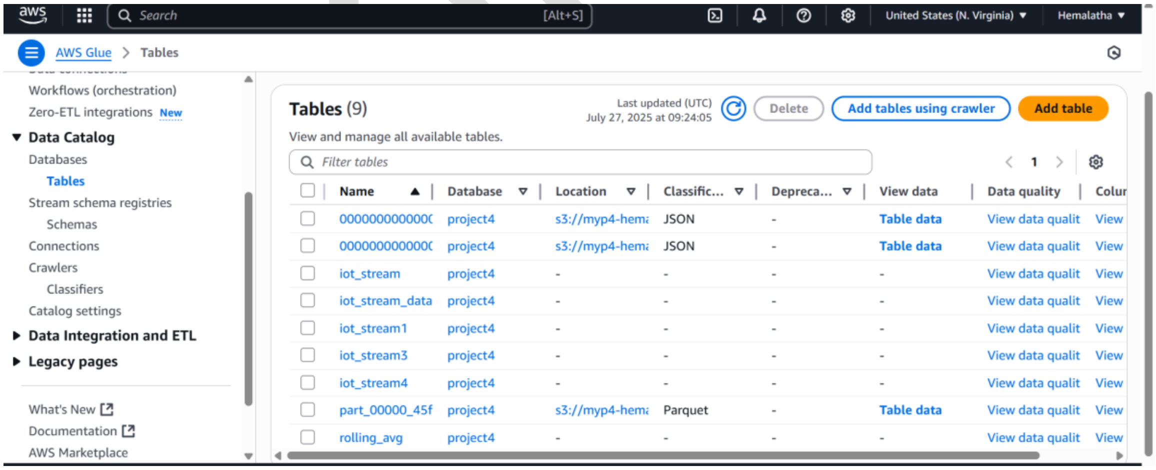
AWS Notifications <no-reply@sns.amazonaws.com>
to me
4:38 AM (40 minutes ago)

🔥 High Temperature Alert!

Machine ID: N/A
Machine Name: N/A
Temperature: 95.5°C
Timestamp: N/A

--
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:239710306715:create_subscription:8422d066-133c-4d92-ba83-6ccfe8300ace&Endpoint=guntihb9@gmail.com

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support



Tables (9)

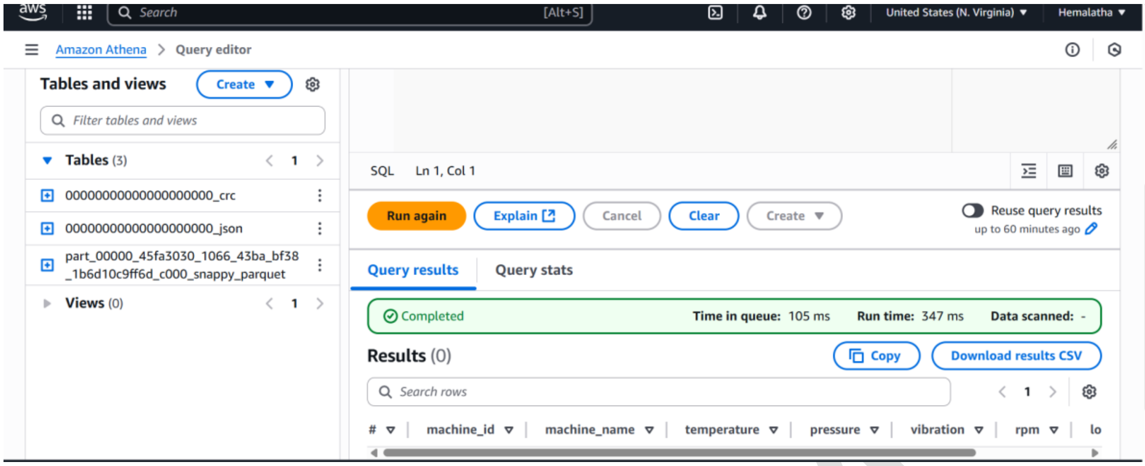
Last updated (UTC)
July 27, 2025 at 09:24:05

View and manage all available tables.

□	Name	Database	Location	Classific...	Depreca...	View data	Data quality	Colur
□	000000000000(	project4	s3://myp4-hemꞔ	JSON	-	Table data	View data qualit	View
□	000000000000(	project4	s3://myp4-hemꞔ	JSON	-	Table data	View data qualit	View
□	iot_stream	project4	-	-	-	-	View data qualit	View
□	iot_stream_data	project4	-	-	-	-	View data qualit	View
□	iot_stream1	project4	-	-	-	-	View data qualit	View
□	iot_stream3	project4	-	-	-	-	View data qualit	View
□	iot_stream4	project4	-	-	-	-	View data qualit	View
□	part_00000_45f	project4	s3://myp4-hemꞔ	Parquet	-	Table data	View data qualit	View
□	rolling_avg	project4	-	-	-	-	View data qualit	View

Visualization with Athena + QuickSight

Use Glue Crawlers to catalog Delta Lake outputs. Connect Athena tables to QuickSight for building visual dashboards (e.g., average temp, RPM, anomaly trends).



Tables (3)

- 00000000000000000000_crc
- 00000000000000000000_json
- part_00000_45fa3030_1066_43ba_bf38_1b6d10c9ff6d_c000_snappy_parquet

Completed   Time in queue: 105 ms   Run time: 347 ms   Data scanned: -

Results (0)

#	machine_id	machine_name	temperature	pressure	vibration	rpm	lo

Summary

The end-to-end pipeline enables real-time monitoring, alerting, storage, and insight generation for smart factory operations. It demonstrates scalable design using cloud-native tools and Delta Lake architecture.