Implementing a Streamlined Real-time Data Streaming and Analytics Pipeline with AWS Kinesis, Firehose, Data Streams, AWS kinesis Analytics Application, Glue Crawler, Glue ETL, and Athena for Querying S3-backed Databases.

Naveen Raju Sreerama Raju Govinda Raju
naveenraju100@gmail.com | +1 224 7067718 | +91 9886157101 |
https://www.linkedin.com/in/naveen-raju-s-g-bb1486124
https://naveenrajusg.github.io/Portfolio/

Table of Contents

9) Deleting resources

A) Stop the application created

B) Delete delivery stream (stream that delivers results to S3 buckets)

C) Delete delivery stream (ingest data from producers that are configured to send data to Kinesis Data FireHose)

D) Delete S3 buckets

E) Delete SQL application (Analytics applications)

F) Delete Crawler

G) Deelete AWS Glue Studio Job

H) Delete AWS Glue Databases

I) Delete S3 buckets

Please zoom in to view the screenshots clearly

1) Overview

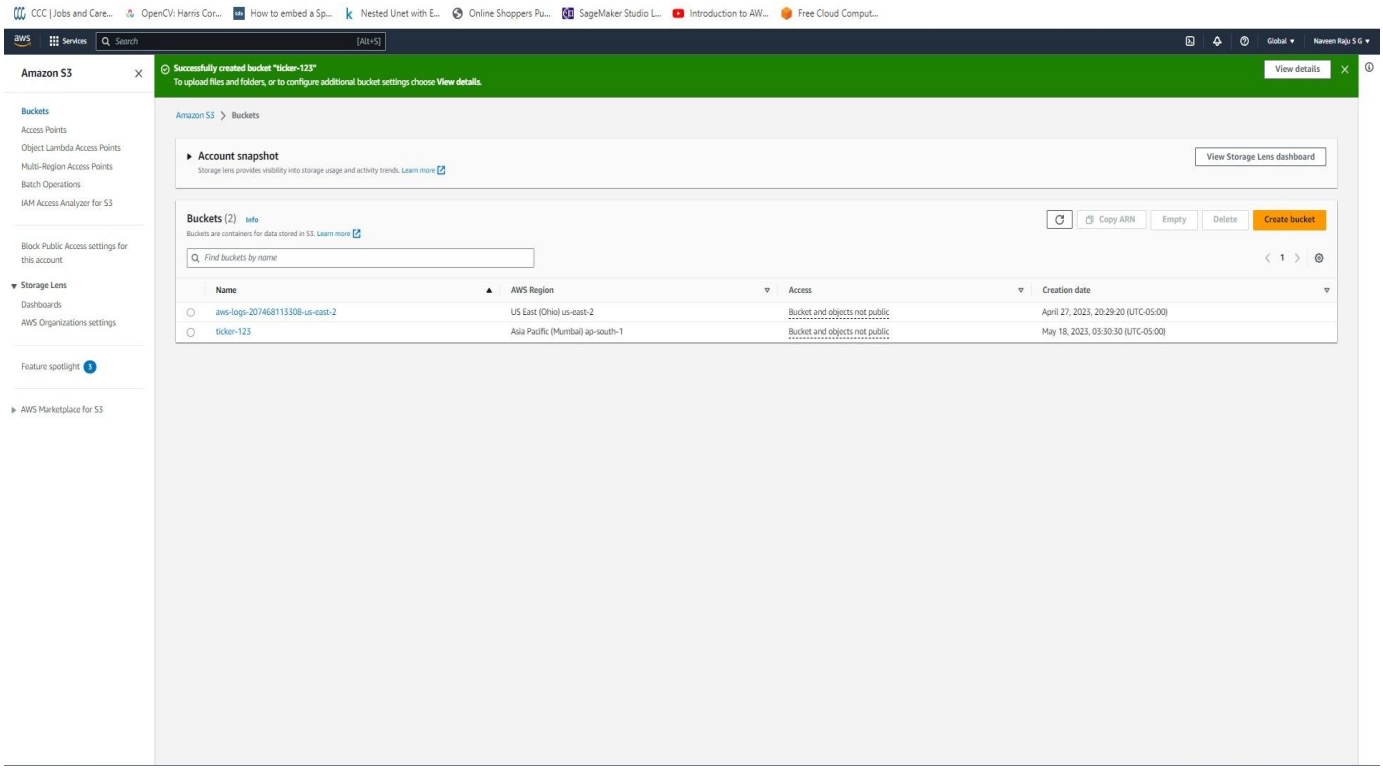
Amazon Kinesis is a suite of real-time data streaming services provided by Amazon Web Services (AWS). It is designed to make it easy for developers to collect, process, and analyze large amounts of streaming data in real-time. Core components of Amazon Kinesis include Amazon Kinesis Streams, Amazon Kinesis Firehose, Amazon Kinesis Analytics, Amazon Kinesis Video Streams.

AWS Glue is a fully managed Extract, Transform, and Load (ETL) service provided by Amazon Web Services (AWS). It is designed to help users efficiently and securely prepare and load their data for analytics and data-driven tasks. AWS Glue can handle both structured and semi-structured data, making it ideal for processing diverse data sources

Amazon Athena is an interactive query service that makes it easy to analyze data in Amazon S3 and other federated data sources using standard SQL.

2) S3 bucket creation

Create a s3 bucket named ticker-123.



3) Delivery stream creation for Data FireHose

Delivery stream ingest data from producers that are configured to send data to Kinesis Data FireHose. It can also optionally transform records using AWS lambda functions and finally it will deliver data to configured destination in our case it is S3 bucket.

In the AWS console navigate to Amazon Kinesis --> Data Firehose --> Create delivery stream.

Delivery stream configuration:

Source : Direct PUT (which means we have data source that generates data in real-time)

Destination : Amazon S3

Delivery stream name : ml-raju

Destination settings -> S3 bucket : s3://ticker-123 (path to s3 bucket created)

Dynamic partitioning : Not enabled
(Dynamic partitioning enables to create targeted datasets by partitioning streaming data based on partitioning keys. We can do this with help of AWS Lambda function.)

S3 bucket prefix : ticker/demo
(By default, kinesis Data Firehose appends prefix “YYYY/MM/dd/HH” (in UTC) to the data it delivers to Amazon S3. We can override this default by specifying a custom prefix.)

S3 bucket error output prefix : ticker/error

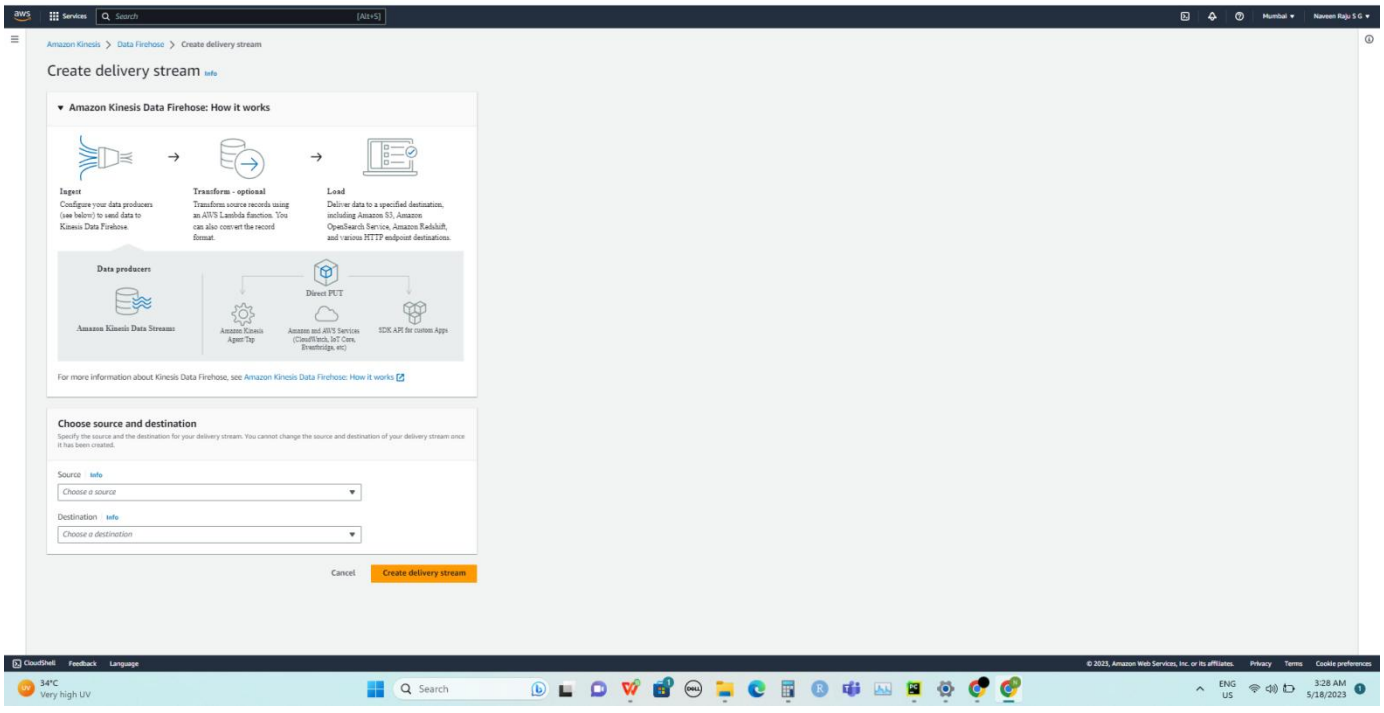
Buffer size : 1Mb
(Kinesis Data Firehouse buffers incoming records before delivering to S3 buckets. Record delivery is triggered once the set size limit is reached. Higher buffer size size may be lower in cost with high latency. The lower buffer size will be faster in delivery with higher cost and less latency.)

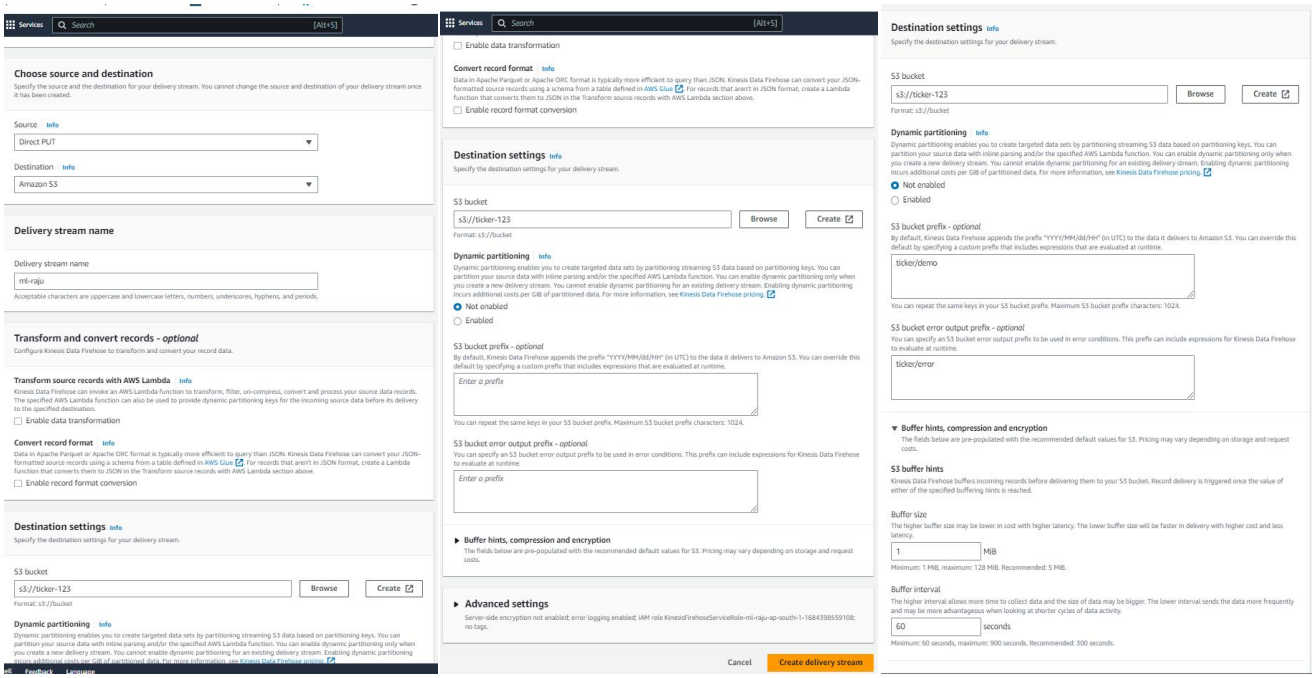
Buffer interval : 60 seconds
(The higher interval allows more time to collect data and size of data may be bigger. The lower interval sends the data more frequently and may be more advantageous when looking at shorter cycles of data activity.)

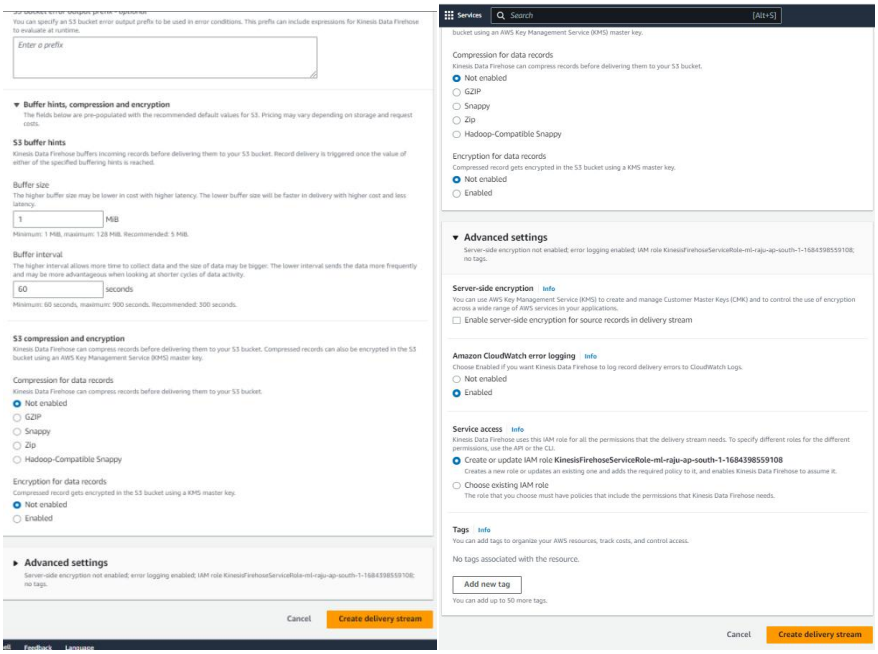
Compressionsn for Data records : Not enabled
(Kinesis Data Firehose can compress records before delivering them to S3 buckets.)

Encryptions for data records : Not enabled
(Compress record gets encrypted in the s3 bucket using KMS master key.)

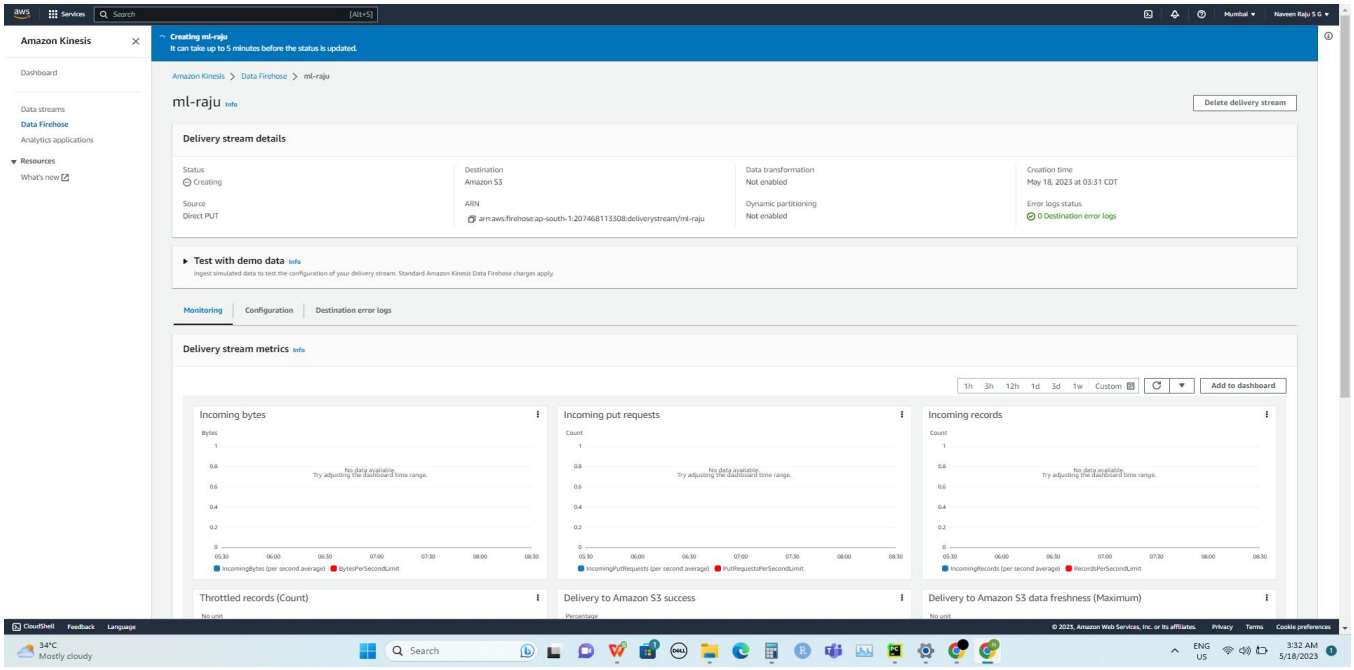
Amazon CloudWatch: Enabled
(Kinesis Data FireHose to log record delivery errors to CloudWatch Logs)

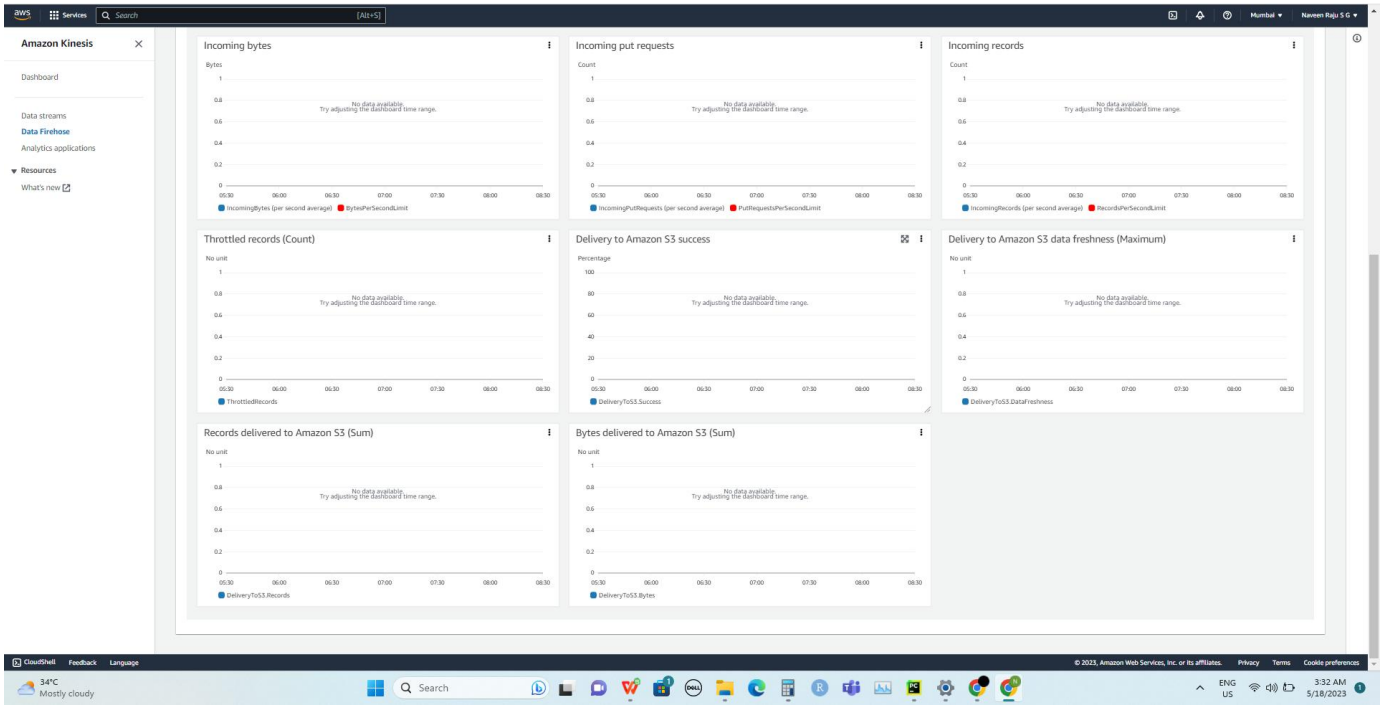


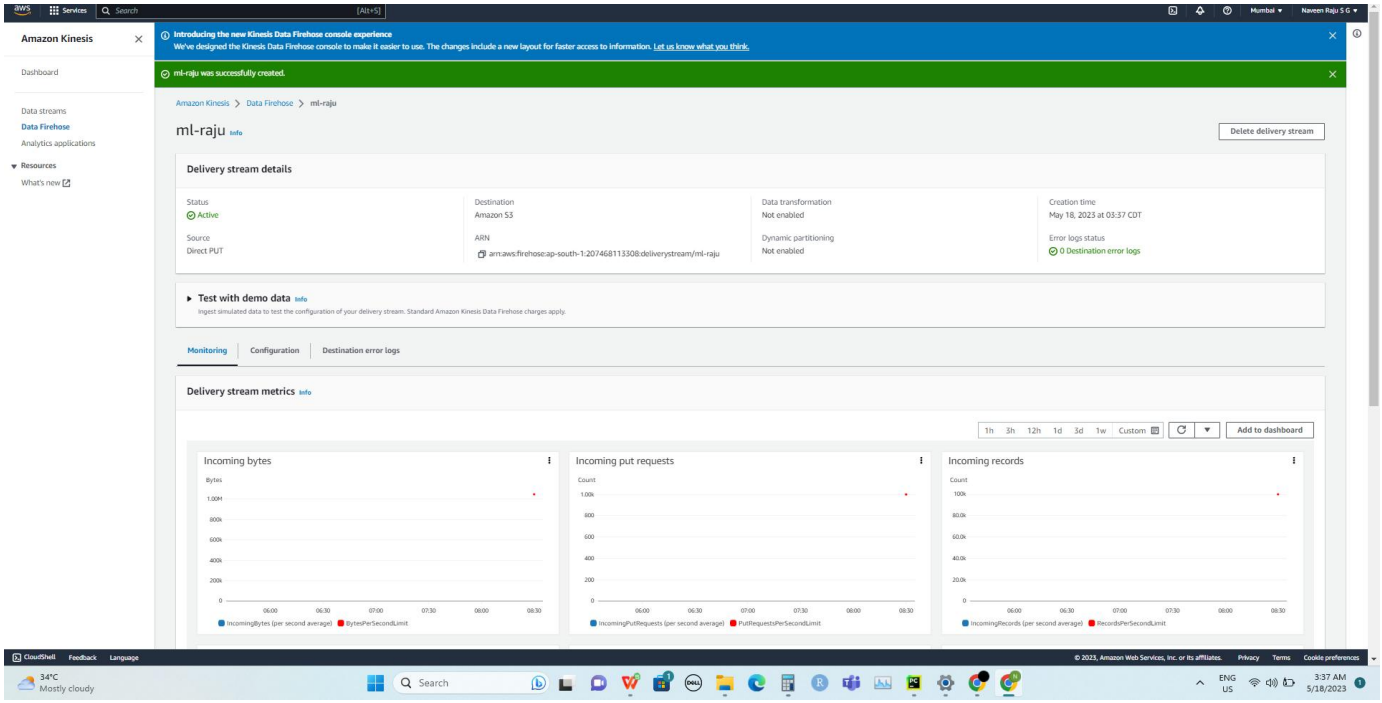

Choose source and destination
Source
Direct PUT
Destination
Amazon S3
Delivery stream name
Delivery stream name
ml-raju
Transform and convert records - optional
Transform source records with AWS Lambda
Enable data transformation
Convert record format
Enable record format conversion
Destination settings
S3 bucket
s3://ticker-123
Browse
Create
Dynamic partitioning
Enable data transformation
Convert record format
Enable record format conversion
Destination settings
S3 bucket
s3://ticker-123
Browse
Create
Dynamic partitioning
Not enabled
Enabled
S3 bucket prefix - optional
Enter a prefix
S3 bucket error output prefix - optional
Enter a prefix
Buffer hints, compression and encryption
Advanced settings
Cancel
Create delivery stream
Destination settings
Specify the destination settings for your delivery stream.
S3 bucket
s3://ticker-123
Browse
Create
Format: s3://bucket
Dynamic partitioning
Not enabled
Enabled
S3 bucket prefix - optional
ticker/demo
S3 bucket error output prefix - optional
ticker/error
Buffer hints, compression and encryption
S3 buffer hints
Buffer size
1
MiB
Buffer interval
60
seconds


Buffer hints, compression and encryption
S3 buffer hints
Buffer size
1
MiB
Buffer interval
60
seconds
S3 compression and encryption
Compression for data records
Not enabled
GZIP
Snappy
Zip
Hadoop-Compatible Snappy
Encryption for data records
Not enabled
Enabled
Advanced settings
Cancel
Create delivery stream
Compression for data records
Not enabled
GZIP
Snappy
Zip
Hadoop-Compatible Snappy
Encryption for data records
Not enabled
Enabled
Advanced settings
Server-side encryption
Enable server-side encryption for source records in delivery stream
Amazon CloudWatch error logging
Not enabled
Enabled
Service access
Create or update IAM role KinesisFirehoseServiceRole-ml-raju-ap-south-1-1684398559108
Choose existing IAM role
Tags
No tags associated with the resource.
Add new tag
Cancel
Create delivery stream

Delivery stream has been created :

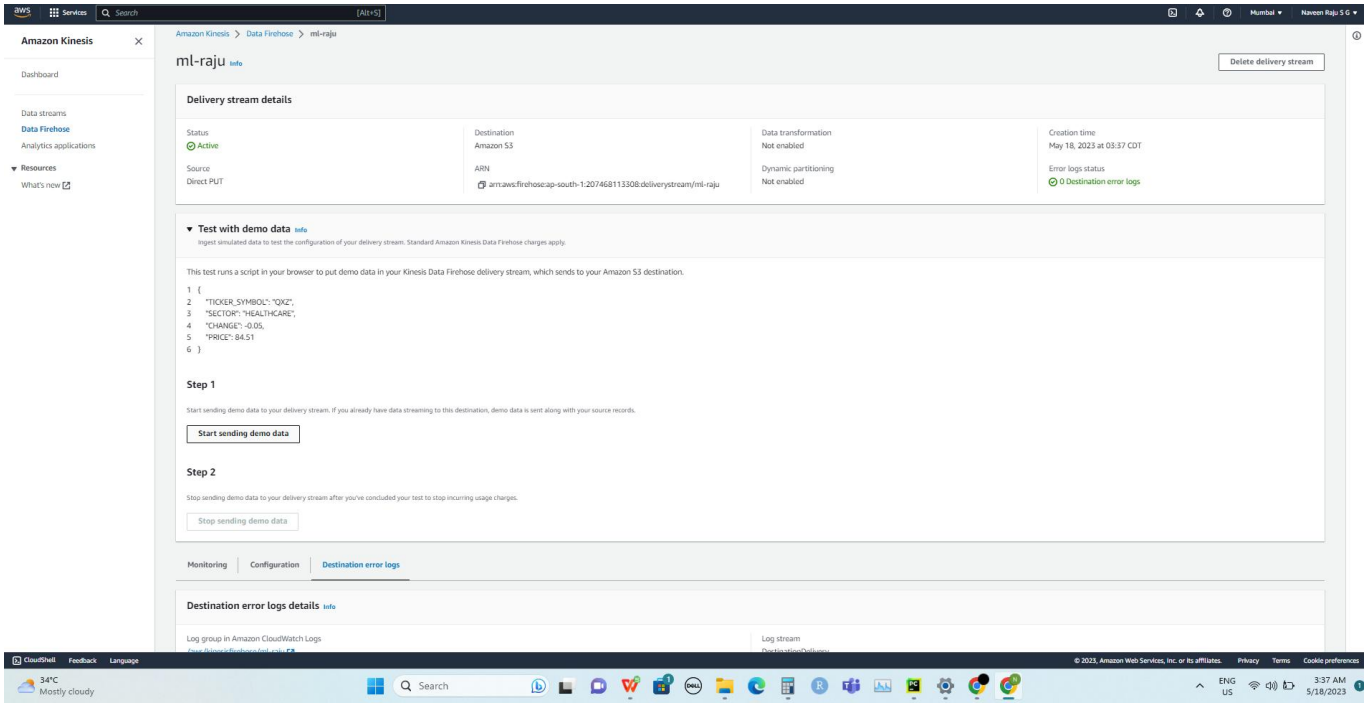




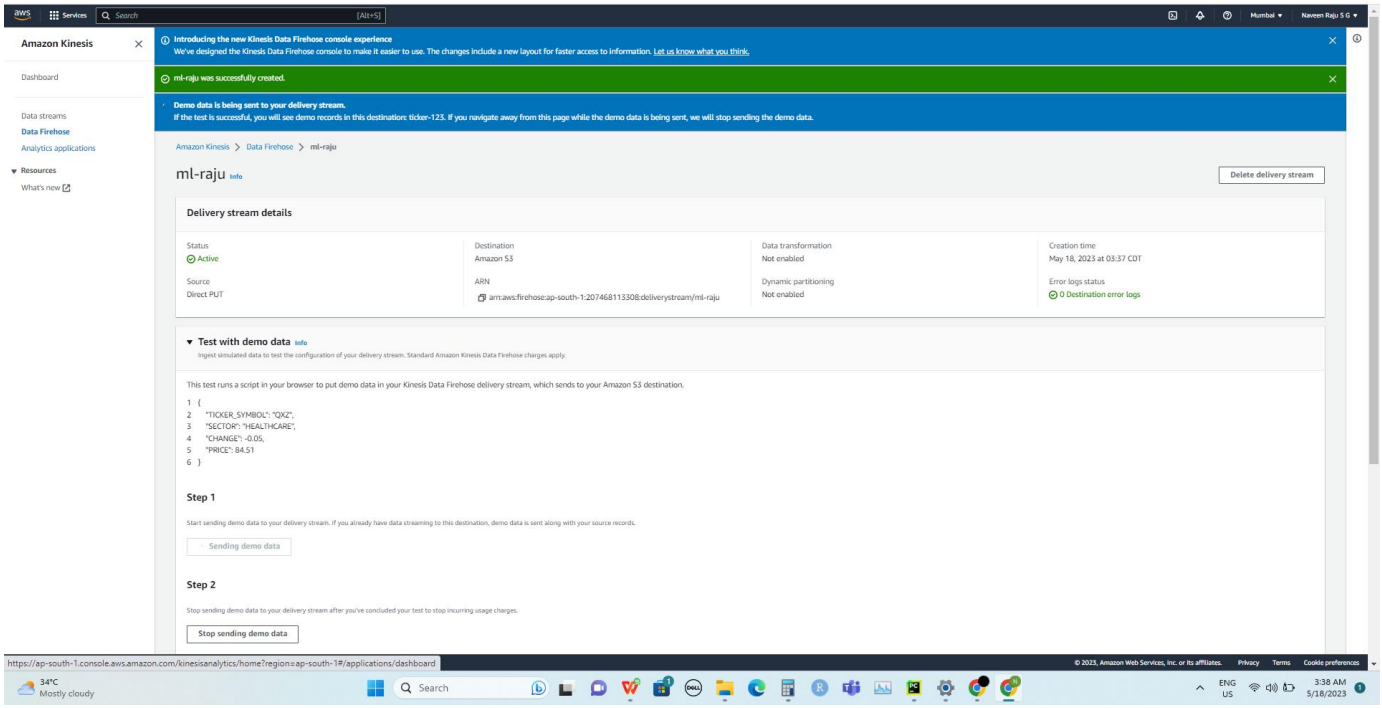


4) <u>Testing delivery stream by sending demo data</u>

After delivery stream is created click on drop down “Test with demo data”.
This runs a script in the browser to put demo data in Kinesis Data FireHose delivery stream, which then sends to the Amazon S3 destination.



Click on start sending demo data.



5) Create and configure Amazon kinesis Analytics applications

A) Create SQL application

Create and configure Amazon kinesis Analytics applications

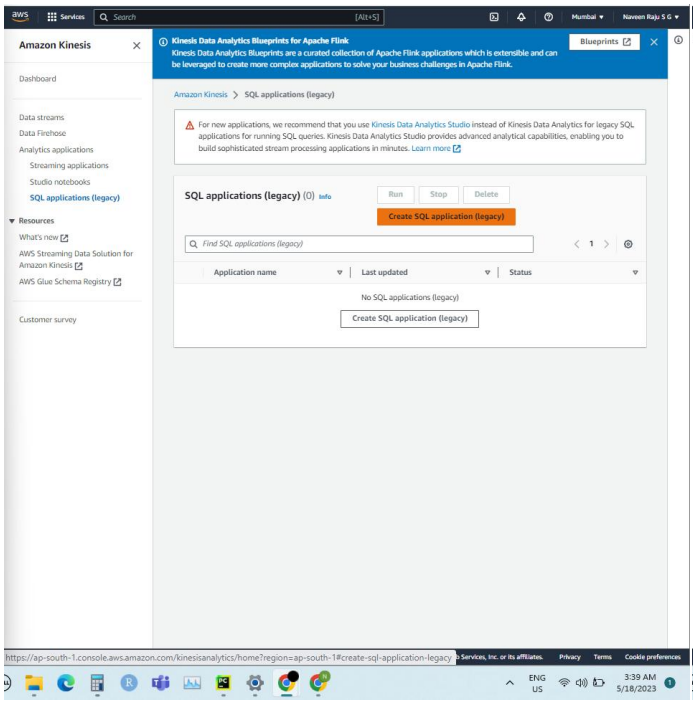
In AWS console navigate to Amazon kinesis -> Analytics applications -> SQL applications

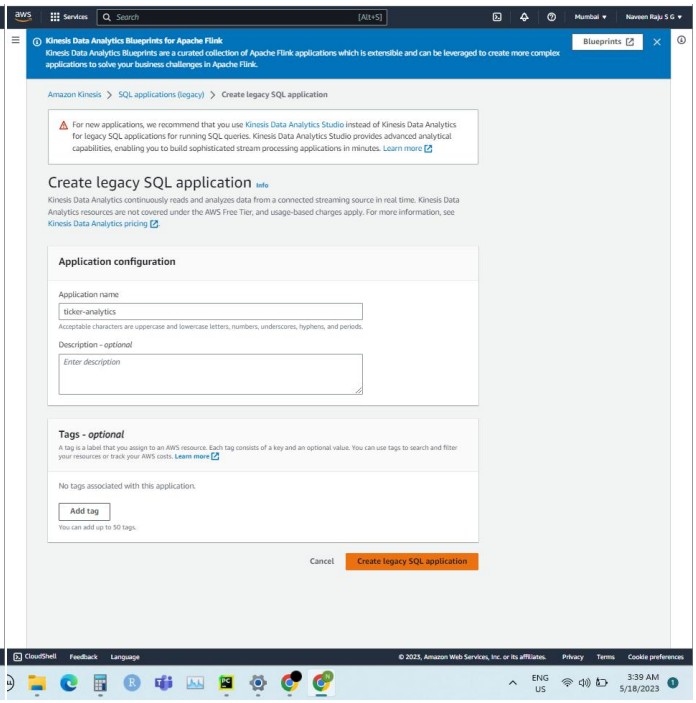
Click on Create SQL application.

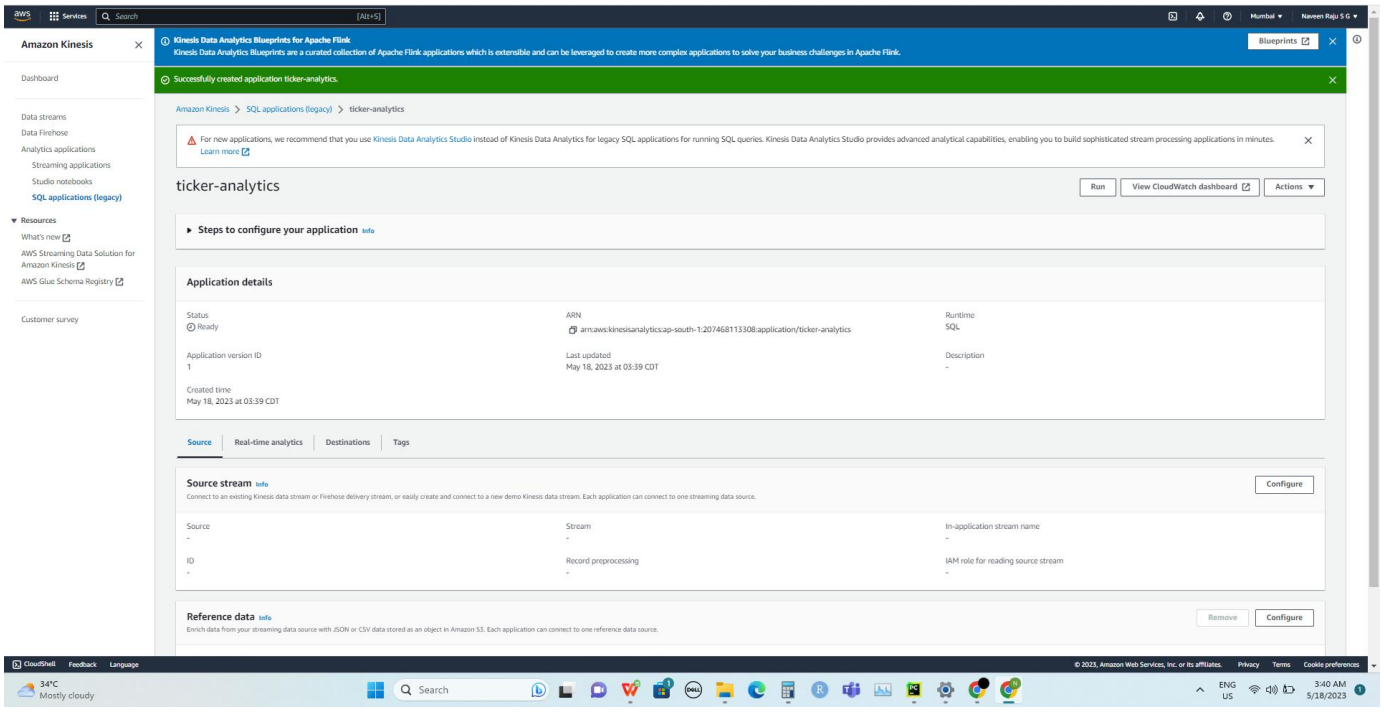
Config : Application name : tiger-analytics.

Click on Create Legacy SQL application.

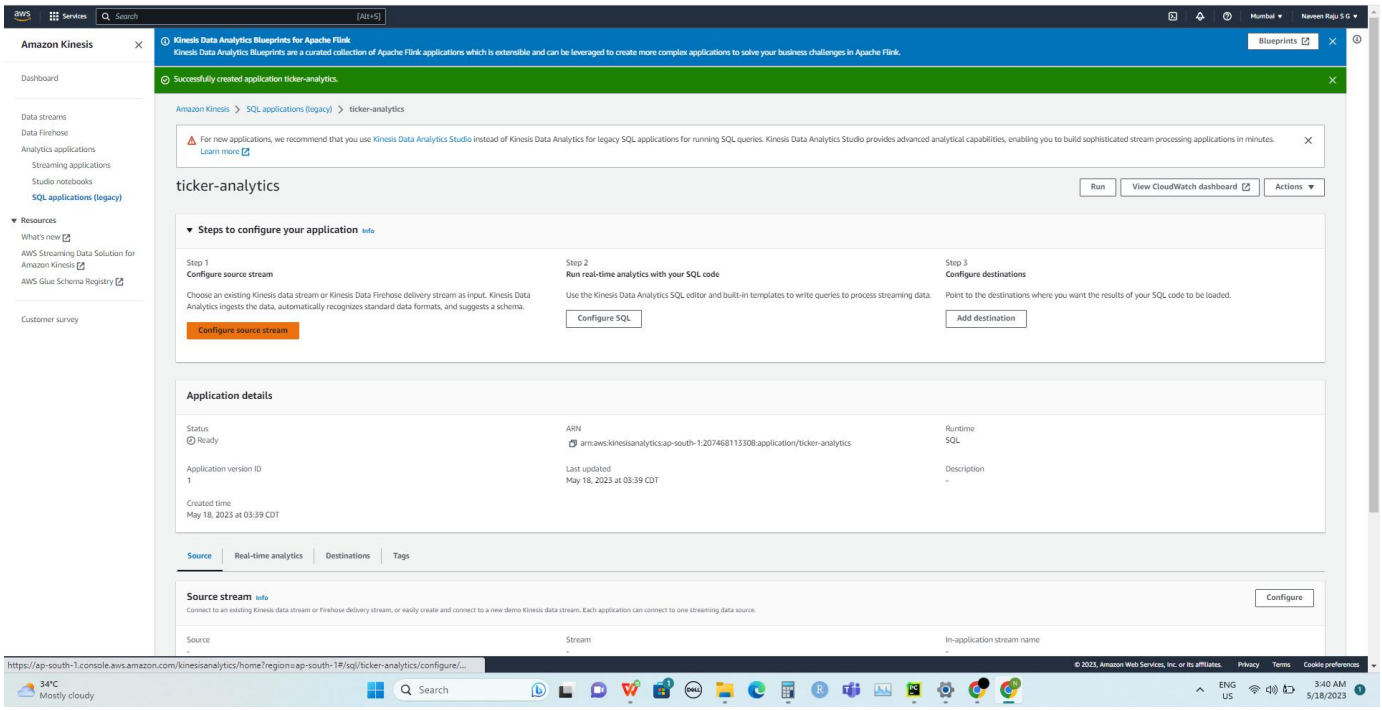
Kinesis Data Analytics continuously reads and analyzes data from a connected streaming source in real time.


Amazon Kinesis
Kinesis Data Analytics Blueprints for Apache Flink
Kinesis Data Analytics Blueprints are a curated collection of Apache Flink applications which is extensible and can be leveraged to create more complex applications to solve your business challenges in Apache Flink.
Blueprints
Dashboard
Data streams
Data Firehose
Analytics applications
Streaming applications
Studio notebooks
SQL applications (legacy)
Resources
What's new
AWS Streaming Data Solution for Amazon Kinesis
AWS Glue Schema Registry
Customer survey
Amazon Kinesis > SQL applications (legacy)
For new applications, we recommend that you use Kinesis Data Analytics Studio instead of Kinesis Data Analytics for legacy SQL applications for running SQL queries. Kinesis Data Analytics Studio provides advanced analytical capabilities, enabling you to build sophisticated stream processing applications in minutes. Learn more
SQL applications (legacy) (0) Info
Run
Stop
Delete
Create SQL application (legacy)
Find SQL applications (legacy)
Application name
Last updated
Status
No SQL applications (legacy)
Create SQL application (legacy)
https://ap-south-1.console.aws.amazon.com/kinesisanalytics/home?region=ap-south-1#create-sql-application-legacy
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
US
3:39 AM
5/18/2023


Amazon Kinesis
Kinesis Data Analytics Blueprints for Apache Flink
Kinesis Data Analytics Blueprints are a curated collection of Apache Flink applications which is extensible and can be leveraged to create more complex applications to solve your business challenges in Apache Flink.
Blueprints
Amazon Kinesis > SQL applications (legacy) > Create legacy SQL application
For new applications, we recommend that you use Kinesis Data Analytics Studio instead of Kinesis Data Analytics for legacy SQL applications for running SQL queries. Kinesis Data Analytics Studio provides advanced analytical capabilities, enabling you to build sophisticated stream processing applications in minutes. Learn more
Create legacy SQL application Info
Kinesis Data Analytics continuously reads and analyzes data from a connected streaming source in real time. Kinesis Data Analytics resources are not covered under the AWS Free Tier, and usage-based charges apply. For more information, see Kinesis Data Analytics pricing
Application configuration
Application name
ticker-analytics
Acceptable characters are uppercase and lowercase letters, numbers, underscores, hyphens, and periods.
Description - optional
Enter description
Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs. Learn more
No tags associated with this application.
Add tag
You can add up to 50 tags.
Cancel
Create legacy SQL application
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
US
3:39 AM
5/18/2023


Amazon Kinesis
Kinesis Data Analytics Blueprints for Apache Flink
Kinesis Data Analytics Blueprints are a curated collection of Apache Flink applications which is extensible and can be leveraged to create more complex applications to solve your business challenges in Apache Flink.
Blueprints
Successfully created application ticker-analytics.
Dashboard
Data streams
Data Firehose
Analytics applications
Streaming applications
Studio notebooks
SQL applications (legacy)
Resources
What's new
AWS Streaming Data Solution for Amazon Kinesis
AWS Glue Schema Registry
Customer survey
Amazon Kinesis > SQL applications (legacy) > ticker-analytics
For new applications, we recommend that you use Kinesis Data Analytics Studio instead of Kinesis Data Analytics for legacy SQL applications for running SQL queries. Kinesis Data Analytics Studio provides advanced analytical capabilities, enabling you to build sophisticated stream processing applications in minutes. Learn more
ticker-analytics
Run
View CloudWatch dashboard
Actions
Steps to configure your application Info
Application details
Status
Ready
ARN
arn:aws:kinesisanalytics:ap-south-1:207468113308:application/ticker-analytics
Runtime
SQL
Application version ID
1
Last updated
May 18, 2023 at 03:39 CDT
Description
-
Created time
May 18, 2023 at 03:39 CDT
Source
Real-time analytics
Destinations
Tags
Source stream Info
Connect to an existing Kinesis data stream or Firehose delivery stream, or easily create and connect to a new demo Kinesis data stream. Each application can connect to one streaming data source.
Configure
Source
Stream
In-application stream name
ID
Record preprocessing
IAM role for reading source stream
Reference data Info
Enrich data from your streaming data source with JSON or CSV data stored as an object in Amazon S3. Each application can connect to one reference data source.
Remove
Configure
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
34°C
Mostly cloudy
Search
ENG
US
3:40 AM
5/18/2023

After application is created click on “Steps to configure the application” drop down and then select “Configure source stream”

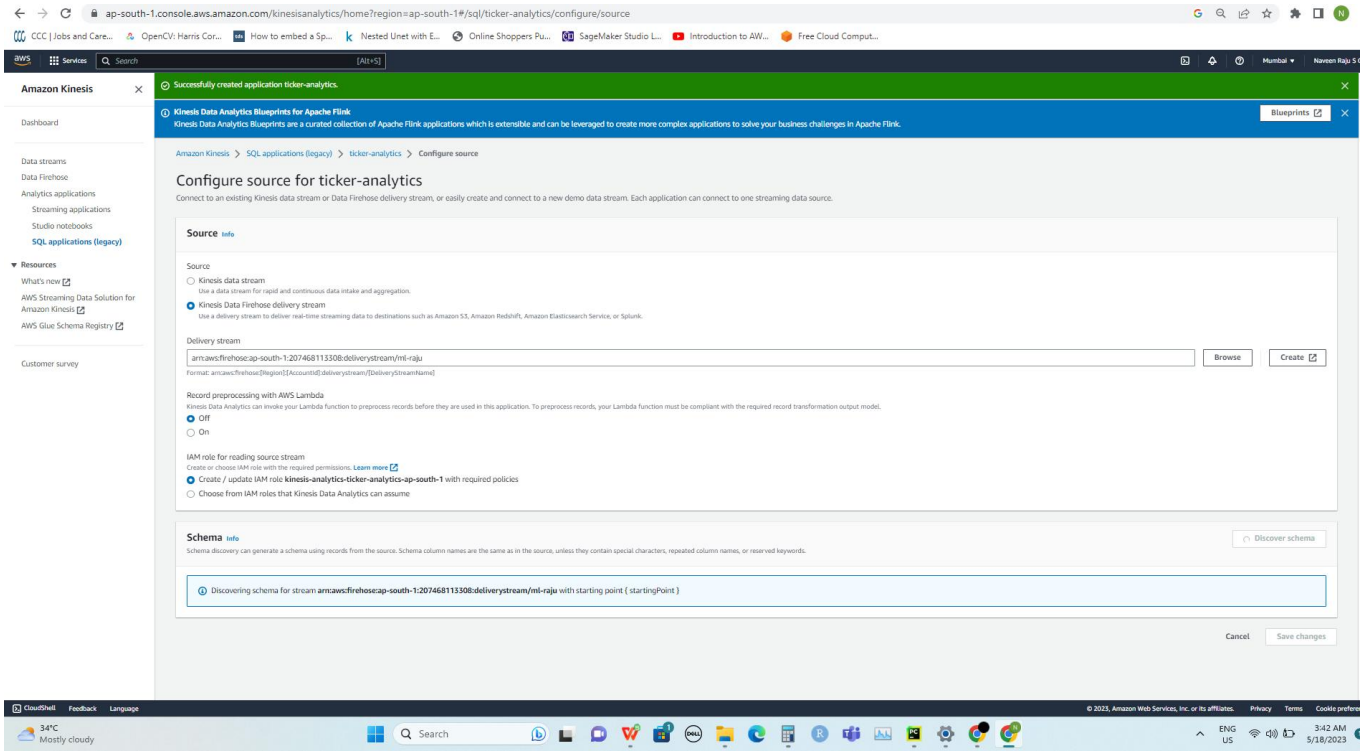


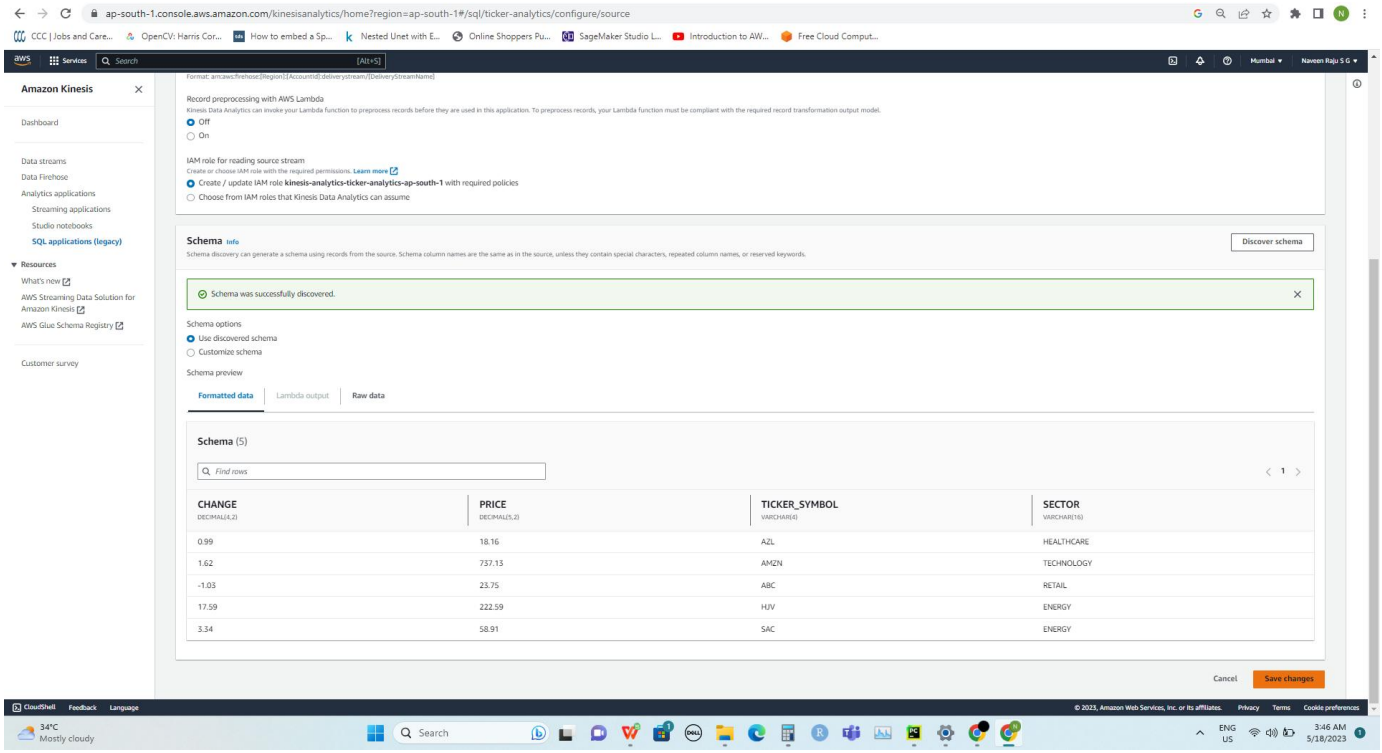
I) <u>Configure Source stream</u>

Now we should configure data source for the application

Source : Kinesis DataFirehose delivery stream
Delivery stream : ml-raju

Schema -> click on Discover schema (Schema discovery can generate a schema using records from the source. Schema column names are the same as in source, unless they contain special characters, repeated column names, or reserved keywords.)

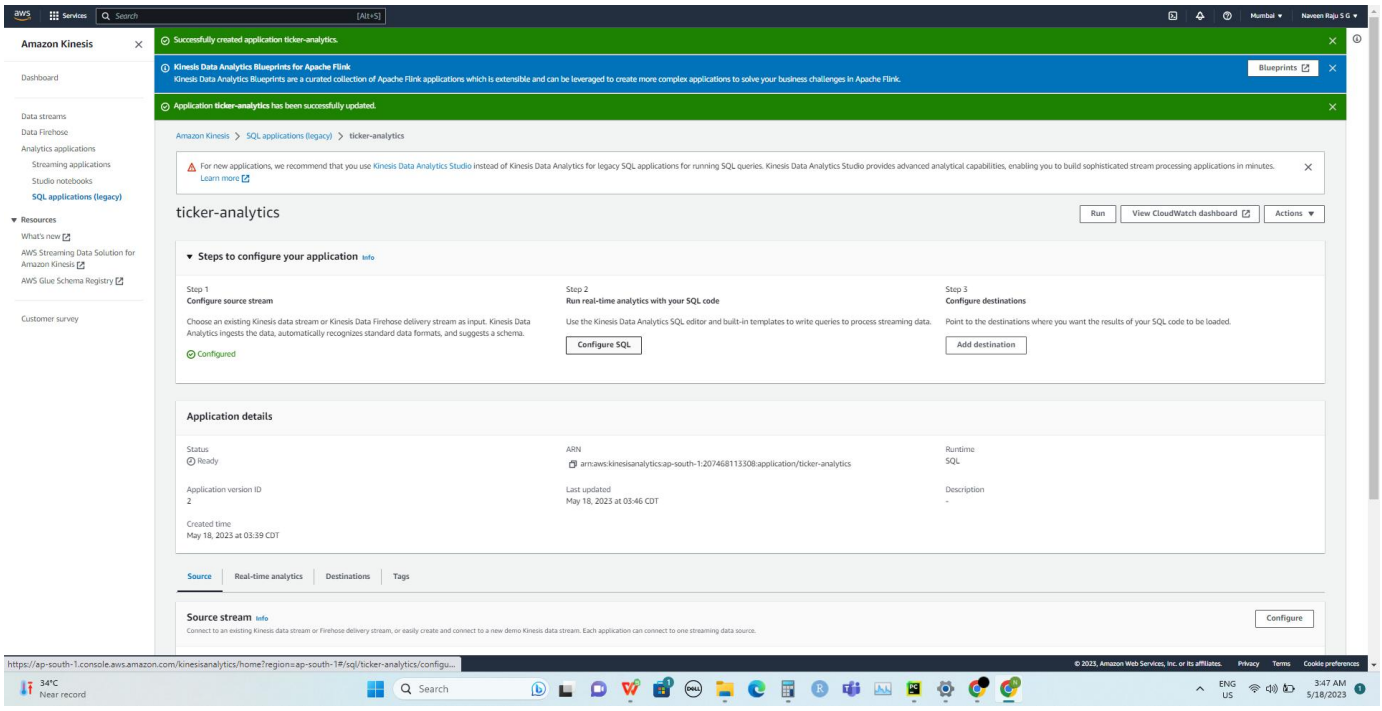




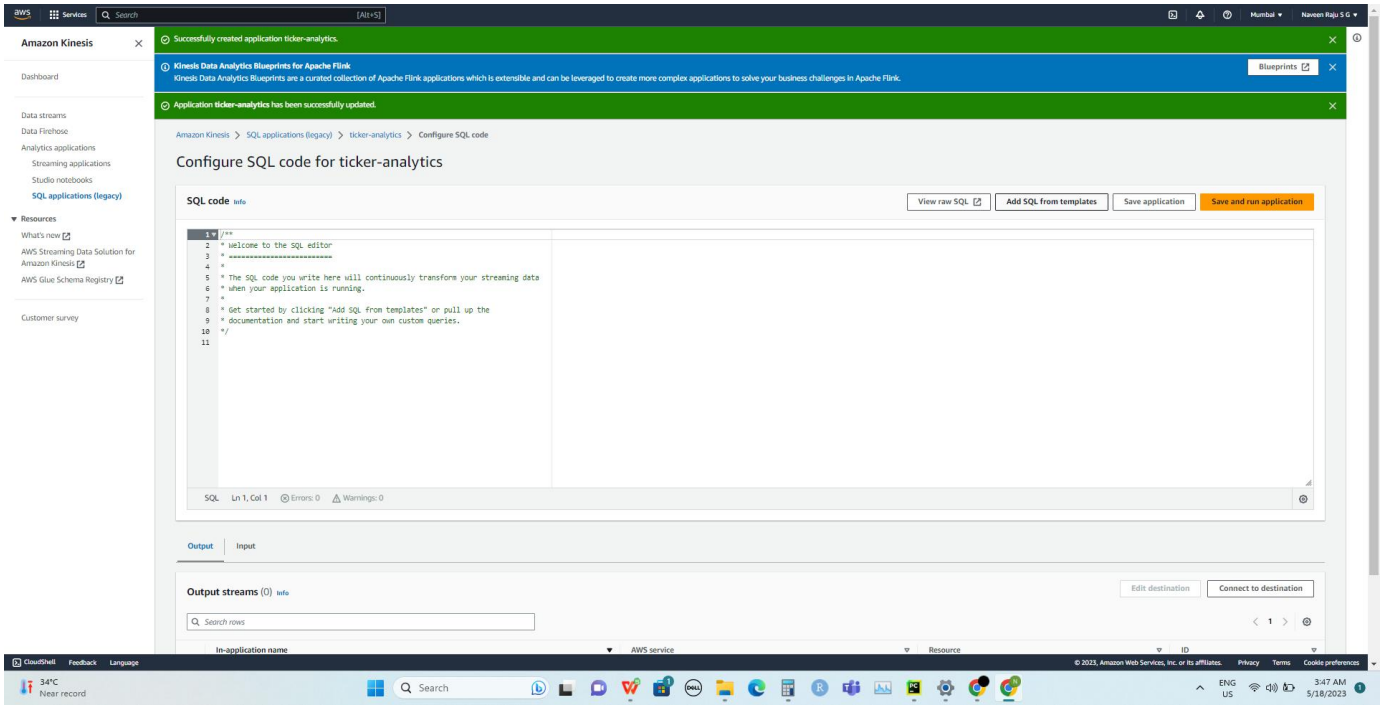
II) Configure Run real time analytics with SQL code

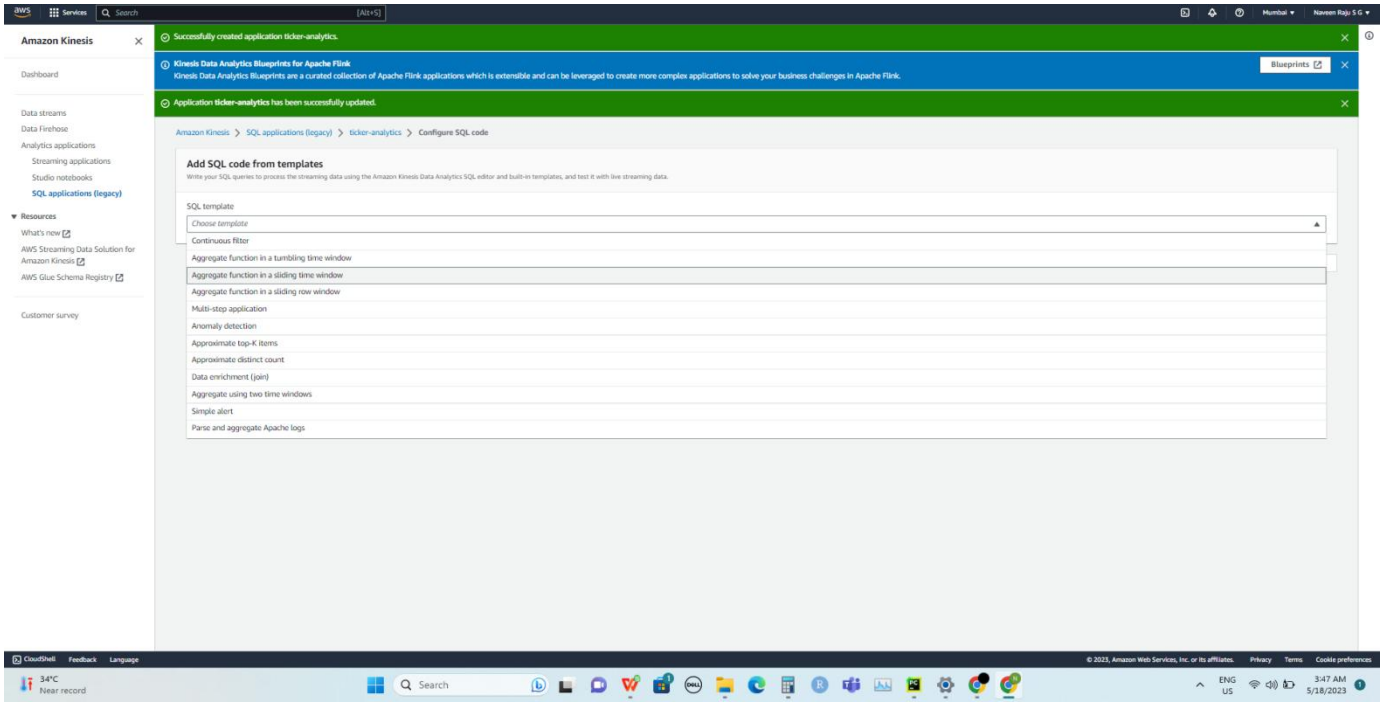
On the same web page navigate to “Steps to configure your application” -> “Run real time analytics with your SQL code”

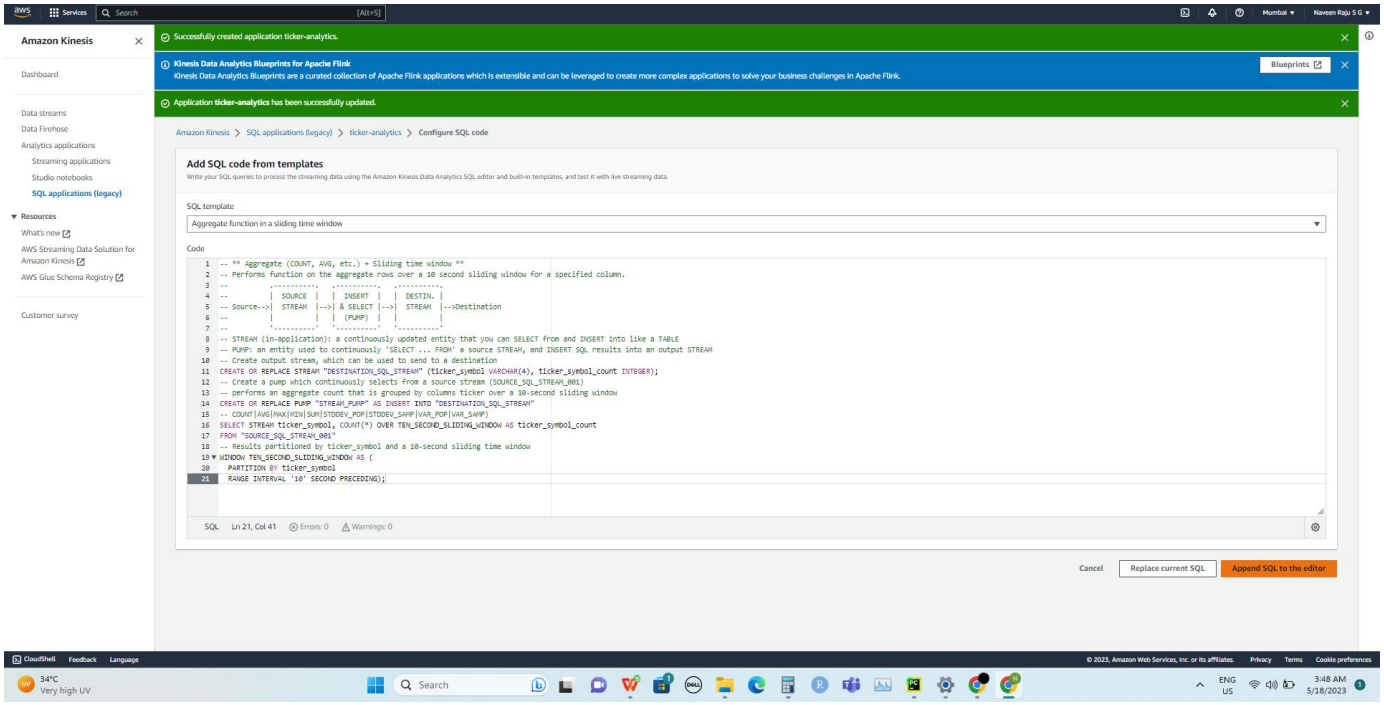
Click on Configure SQL



Click on the button “Add SQL from templates” - > “Aggregate function in sliding time window”







Description of SQL code:

This template performs aggregate on rows over a 10 seconds sliding window.

Basically the application has 3 parts : Source stream, Select and Insert (Pump), create a output stream which can be sent to a destination.

STREAM(in-application): a continuously updated entity that we select from and insert into Table

PUMP : an entity used to continuously ‘ SELECT … FROM’ a source STREAM, and insert SQL results into like a TABLE

Create a output stream, which can be used to send to a destination

[Below line - Create pump which continuously selects from a source stream (SOURCE_SQL_STREAM_001) and performs an aggregate count that is grouped by columns ticker over a 10-second sliding window]

CREATE OR REPLACE STREAM “DESTINATION_SQL_STREAM” (ticker symbol VARCHAR(4), ticker_symbol_count INTEGER);

[Below line does COUNT | AVG | MAX | MIN | SUM | STDDEV_POP | STDDEV_SAMP | VAR_POP | VAR_SAMP]

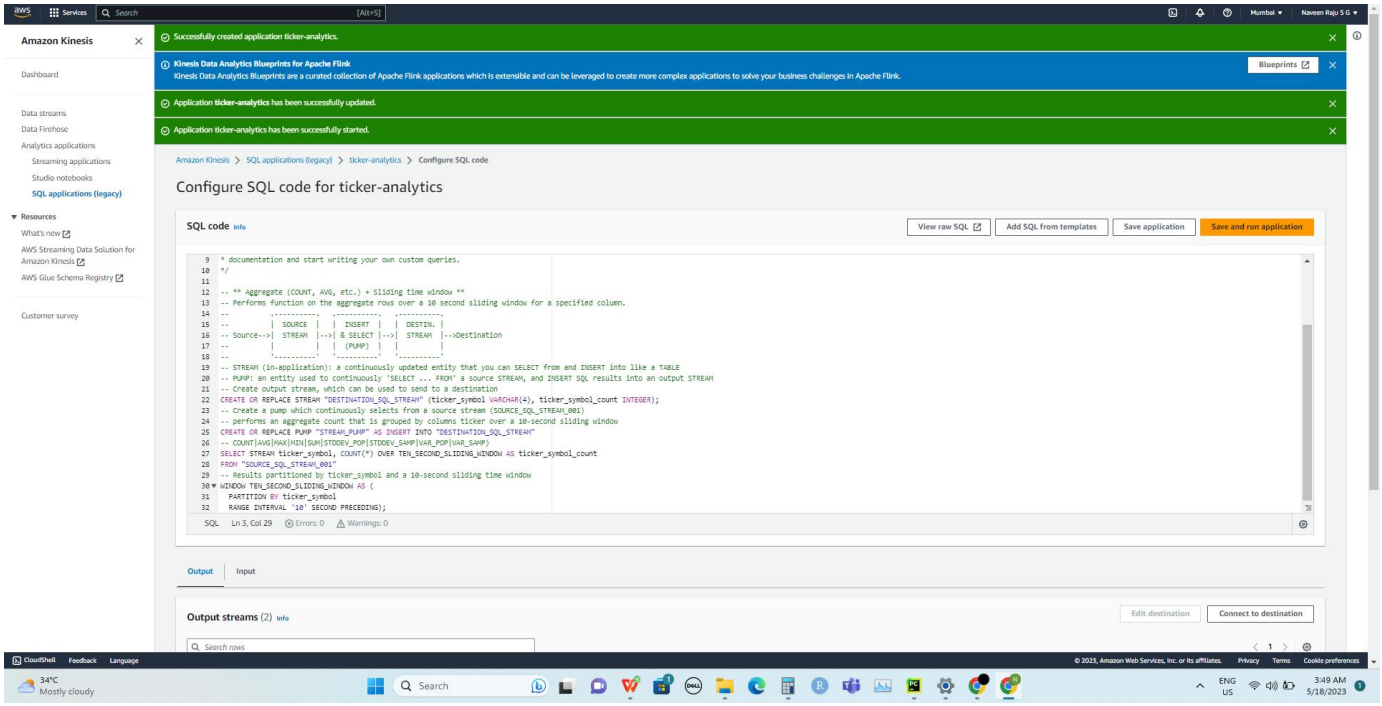
CREATE OR REPLACE PUMP “STREAM_PUMP” AS INSERT INTO “DESTINATION_SQL_STREAM”

SELECT STREAM ticker_symbol, COUNT(*) OVER TEN_SECONDS_SLIDING_WINDOW AS ticker_symbol_count
FROM “SOURCE_SQL_STREAM_001”

[The below code create results partitioned by ticker_symbol and a 10-second sliding time window]

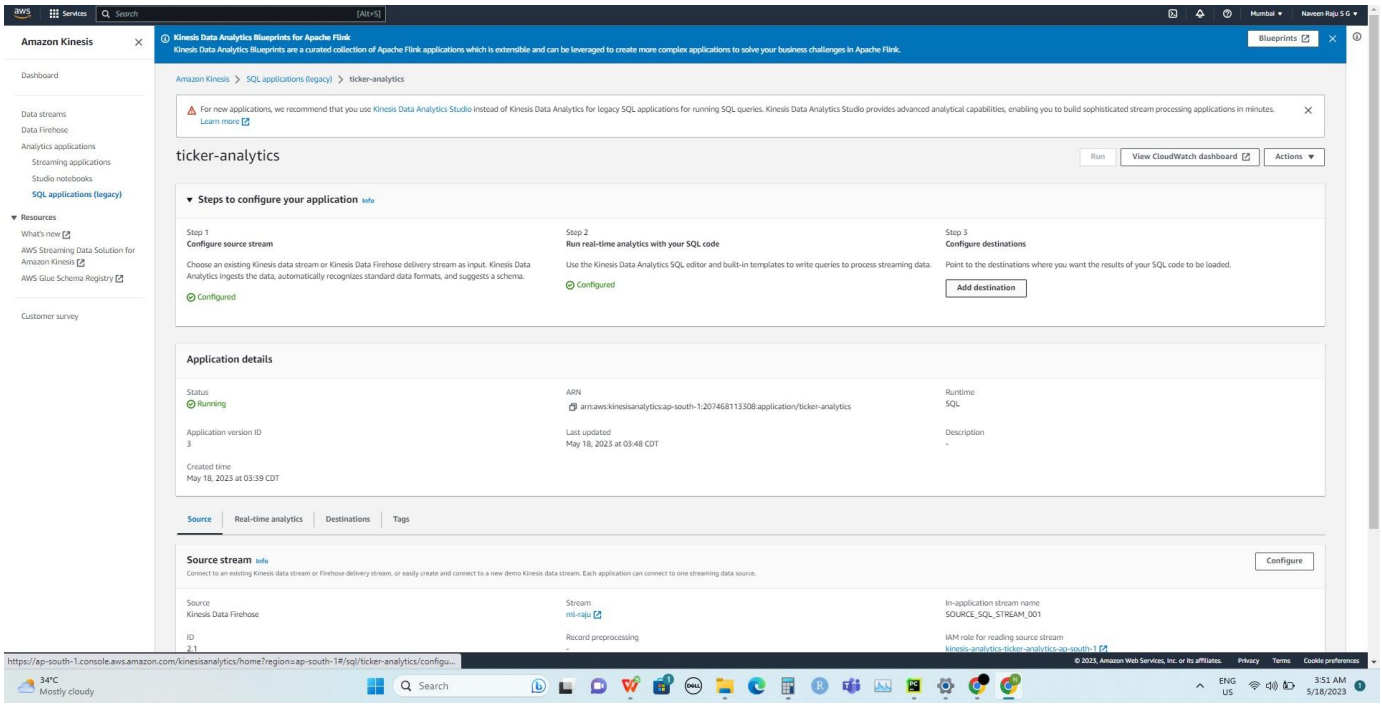
```
WINDOW TEN_SECOND_SLIDING_WINDOW AS (
        PARTITION BY ticker_symbol
        RANGE INTERVAL '10' SECOND PRECEDING);
```

Now click on “Save and run application”



III) <u>Configure destination</u>

On the same web page navigate to “Steps to configure your application” -> “Configure destination”



IV) Create delivery stream:

Navigate through Amazon Kinesis -> Data Firehose -> Create delivery stream

Source : Direct PUT (which means we have data source that generates data in real-time)

Destination : Amazon S3

Delivery stream name : tiger-analytics

Destination settings -> S3 bucket : s3://ticker-123 (path to s3 bucket created)

S3 bucket prefix : tiger_analytics/
(By default, kinesis Data Firehose appends prefix “YYYY/MM/dd/HH” (in UTC) to the data it delivers to Amazon S3. We can override this default by specifying a custom prefix.)

S3 bucket error output prefix : tiger_analytics_error/

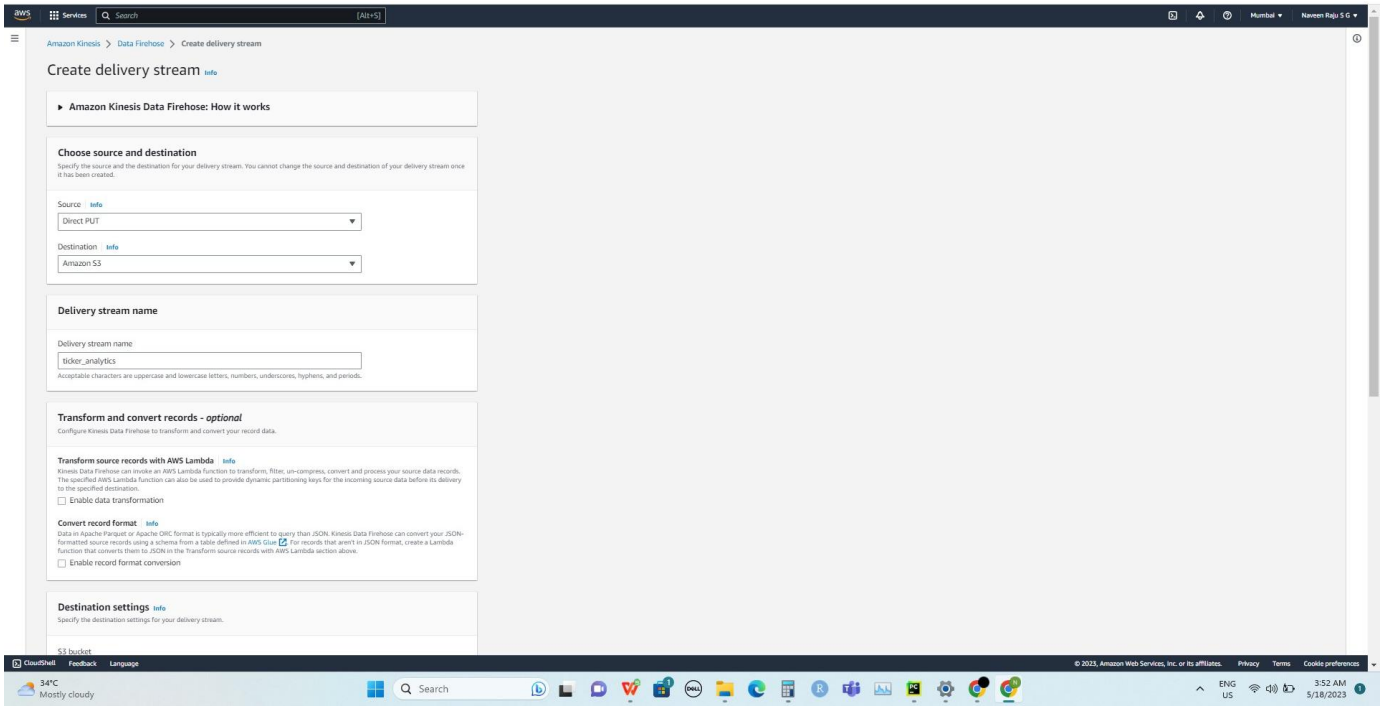
Buffer size : 1Mb

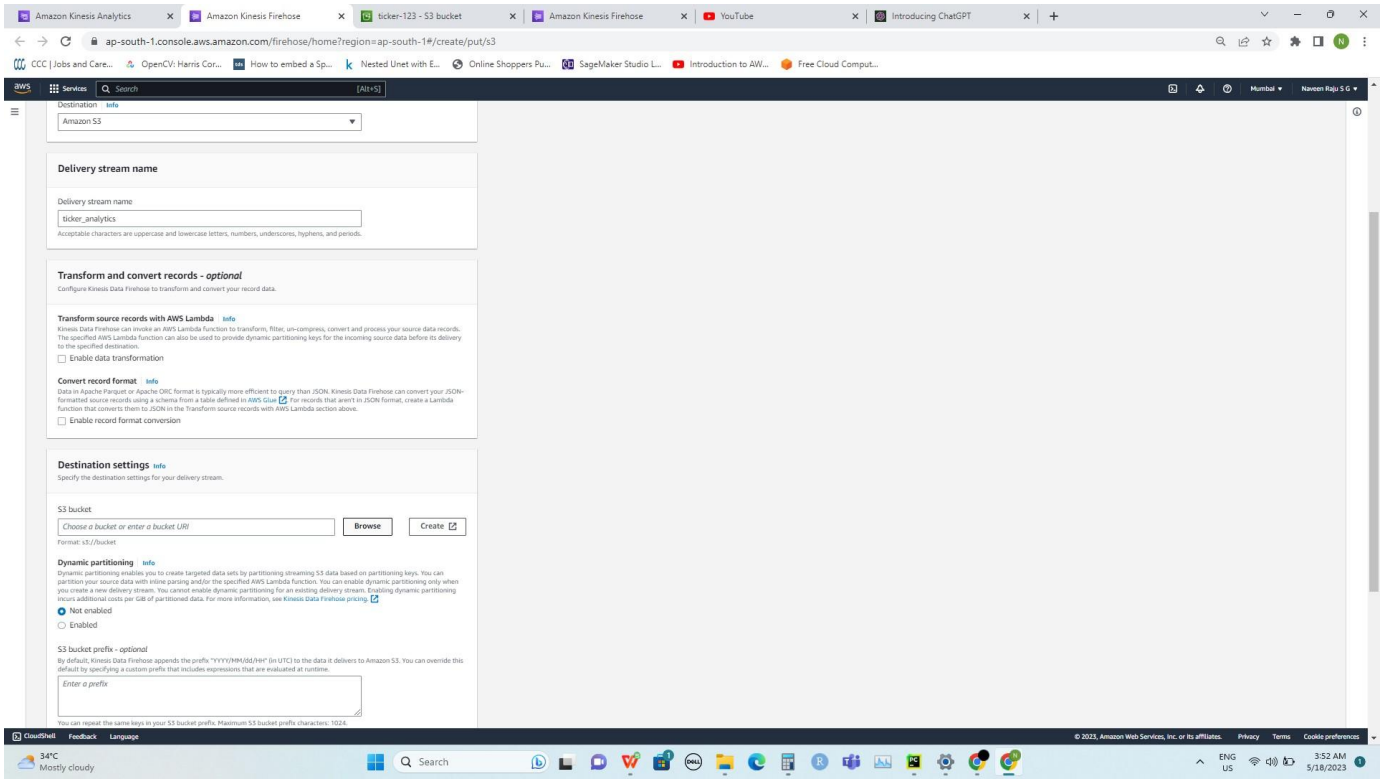
Buffer interval : 60 seconds

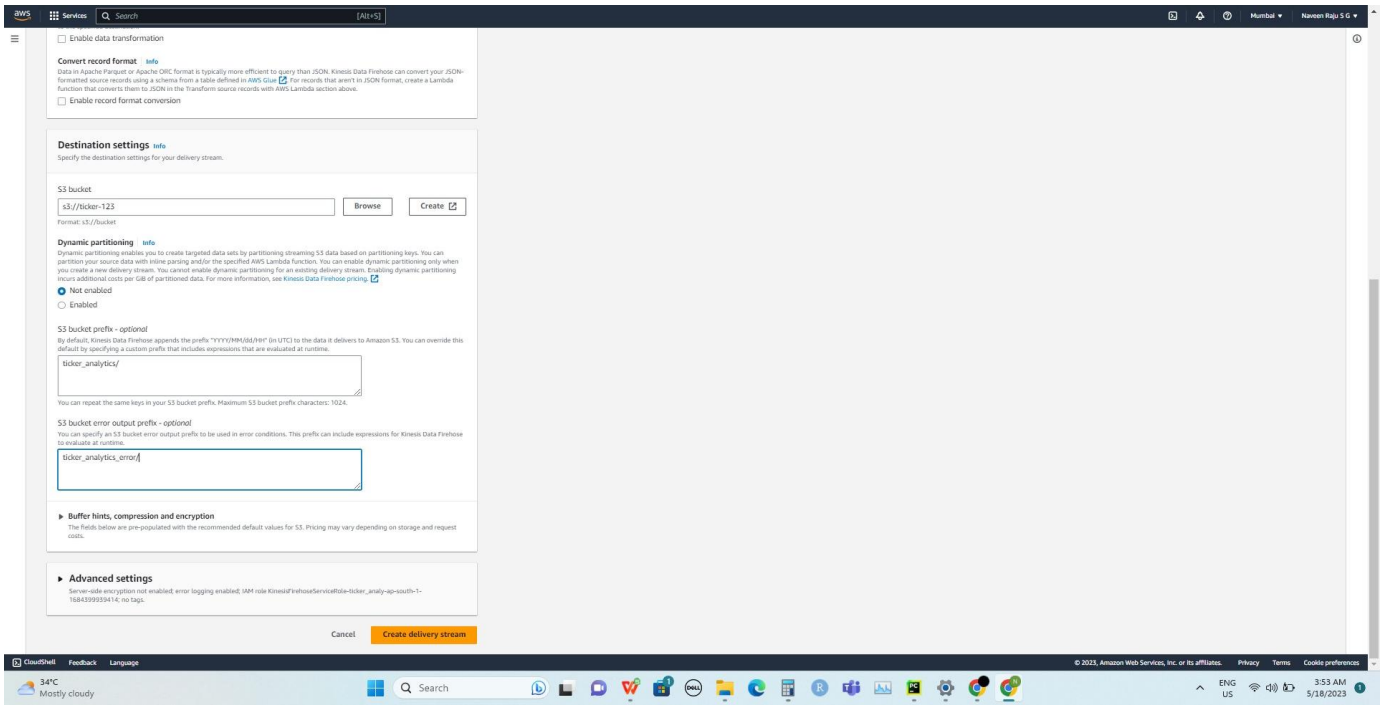
Compressionsn for Data records : Not enabled

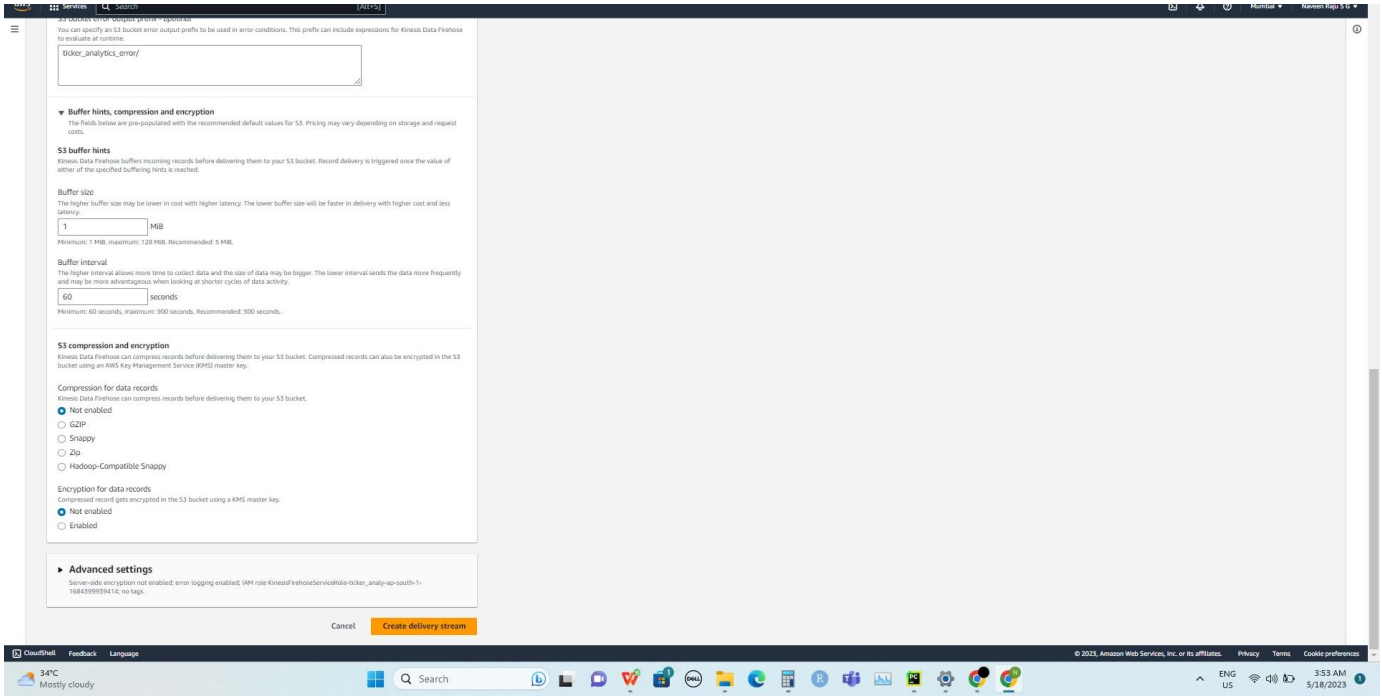
Encryptions for data records : Not enabled

Amazon CloudWatch: Enabled
(Kinesis Data FireHose to log record delivery errors to CloudWatch Logs)




Amazon Kinesis Analytics
Amazon Kinesis Firehose
ticker-123 - S3 bucket
Amazon Kinesis Firehose
YouTube
Introducing ChatGPT
ap-south-1.console.aws.amazon.com/firehose/home?region=ap-south-1#/create/put/s3
Destination Info
Amazon S3
Delivery stream name
Delivery stream name
ticker_analytics
Acceptable characters are uppercase and lowercase letters, numbers, underscores, hyphens, and periods.
Transform and convert records - optional
Configure Kinesis Data Firehose to transform and convert your record data.
Transform source records with AWS Lambda Info
Kinesis Data Firehose can invoke an AWS Lambda function to transform, filter, un-compress, convert and process your source data records. The specified AWS Lambda function can also be used to provide dynamic partitioning keys for the incoming source data before its delivery to the specified destination.
Enable data transformation
Convert record format Info
Data in Apache Parquet or Apache ORC format is typically more efficient to query than JSON. Kinesis Data Firehose can convert your JSON-formatted source records using a schema from a table defined in AWS Glue. For records that aren't in JSON format, create a Lambda function that converts them to JSON in the Transform source records with AWS Lambda section above.
Enable record format conversion
Destination settings Info
Specify the destination settings for your delivery stream.
S3 bucket
Choose a bucket or enter a bucket URI
Browse
Create
Format: s3://bucket
Dynamic partitioning Info
Dynamic partitioning enables you to create targeted data sets by partitioning streaming S5 data based on partitioning keys. You can partition your source data with inline parsing and/or the specified AWS Lambda function. You can enable dynamic partitioning only when you create a new delivery stream. You cannot enable dynamic partitioning for an existing delivery stream. Enabling dynamic partitioning incurs additional costs per GiB of partitioned data. For more information, see Kinesis Data Firehose pricing.
Not enabled
Enabled
S3 bucket prefix - optional
By default, Kinesis Data Firehose appends the prefix "YYYY/MM/dd/HH/" (in UTC) to the data it delivers to Amazon S3. You can override this default by specifying a custom prefix that includes expressions that are evaluated at runtime.
Enter a prefix
You can repeat the same keys in your S3 bucket prefix. Maximum S3 bucket prefix characters: 1024.
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
34°C
Mostly cloudy
Search
ENG
US
3:52 AM
5/18/2023


Enable data transformation
Convert record format Info
Enable record format conversion
Destination settings Info
Specify the destination settings for your delivery stream.
S3 bucket
s3://ticker-123
Browse
Create
Format: s3://bucket
Dynamic partitioning Info
Not enabled
Enabled
S3 bucket prefix - optional
ticker_analytics/
You can repeat the same keys in your S3 bucket prefix. Maximum S3 bucket prefix characters: 1024.
S3 bucket error output prefix - optional
You can specify an S3 bucket error output prefix to be used in error conditions. This prefix can include expressions for Kinesis Data Firehose to evaluate at runtime.
ticker_analytics_error/
Buffer hints, compression and encryption
The fields below are pre-populated with the recommended default values for S3. Pricing may vary depending on storage and request costs.
Advanced settings
Server-side encryption not enabled; error logging enabled; IAM role KinesisFirehoseServiceRole-ticker_analy-ap-south-1-168439939414; no tags.
Cancel
Create delivery stream
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
34°C
Mostly cloudy
Search
ENG
US
3:53 AM
5/18/2023



V) <u>Configure destination</u>

Continue “Steps to configure your application” -> “Configure destination”

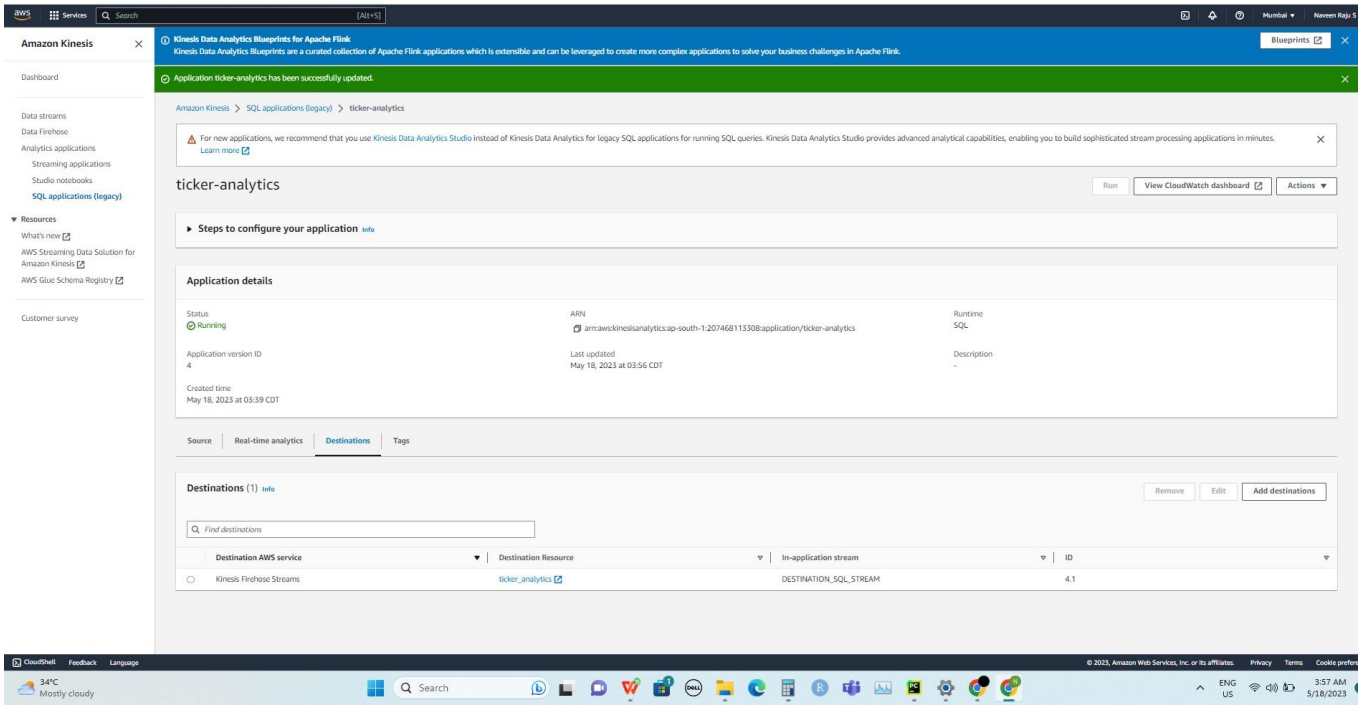
Set delivery stream to tiger_analytics stream that we created

Under In_Application stream name section :

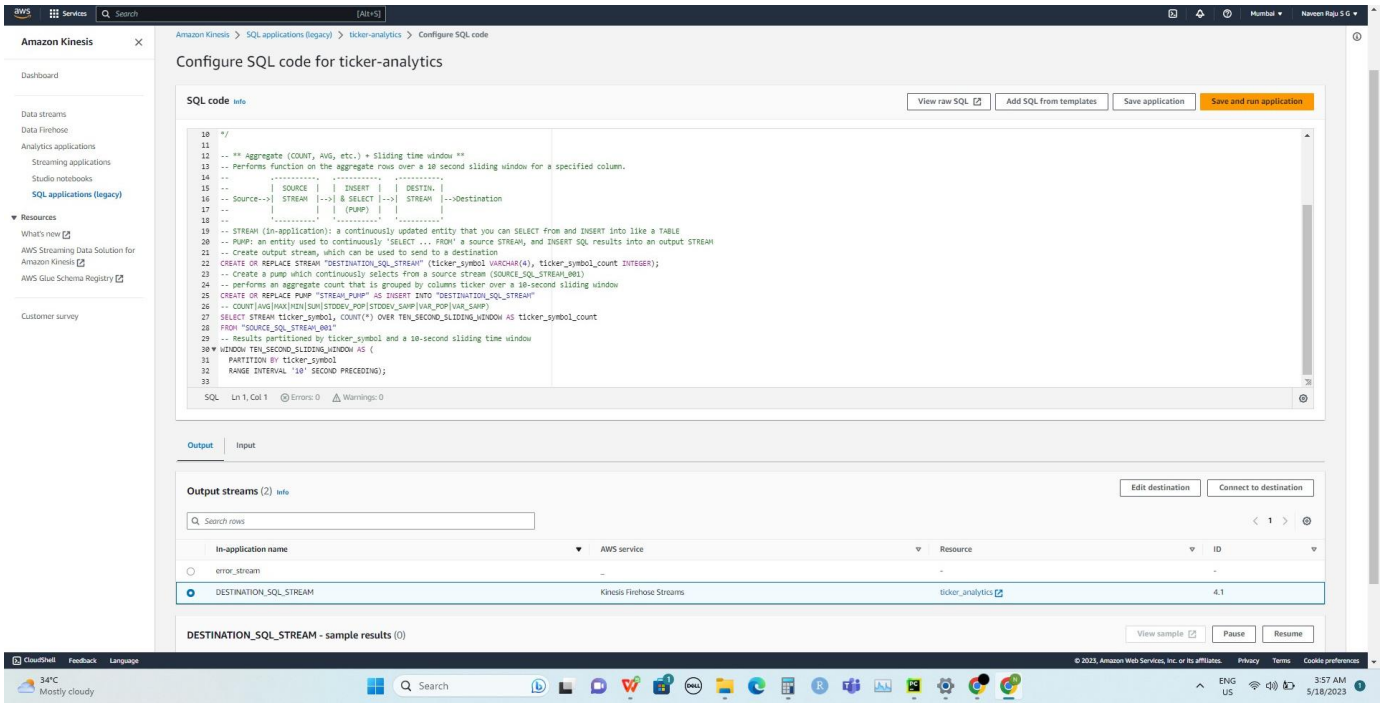
Select Choose an existing in-application stream -> In_application stream name -> DESTINATION_SQL_STREAM

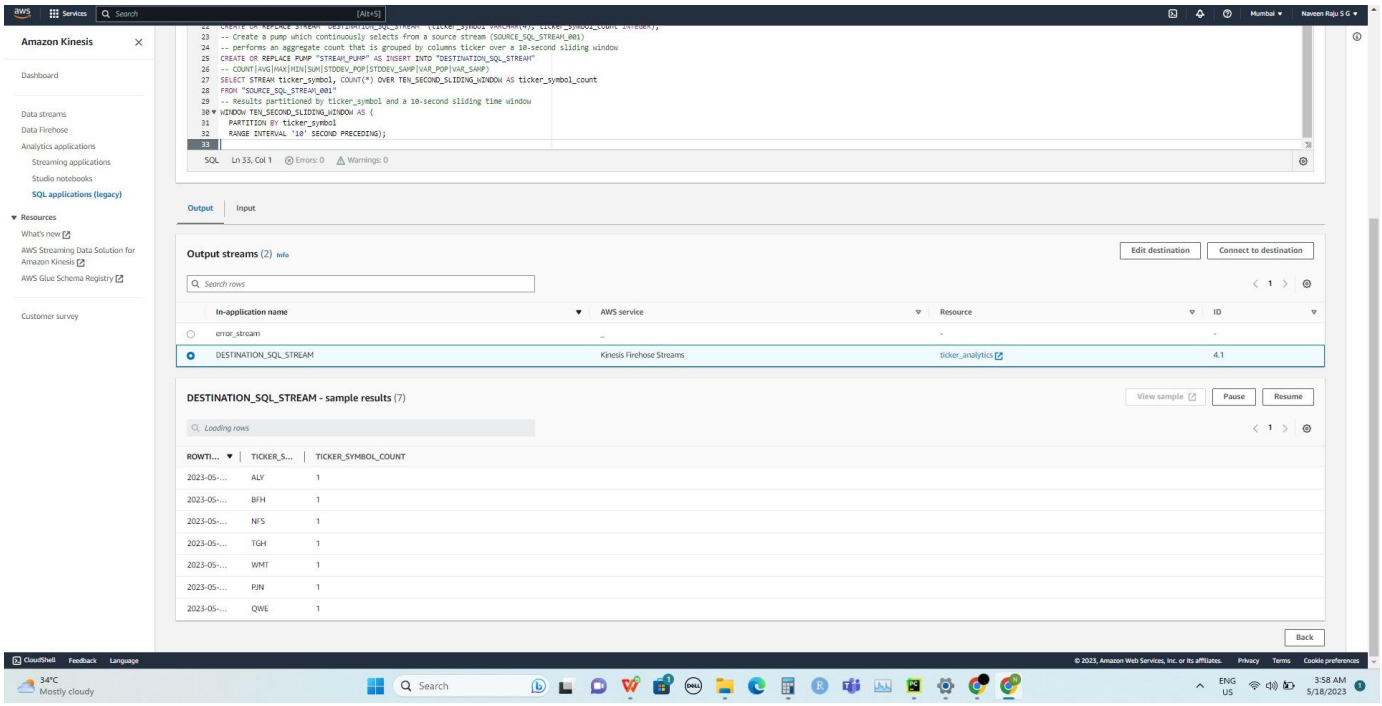
Output format : JSON

Now a Analytics application has been created:



Navigate to application that we created then in Configure SQL code section select Destination SQL stream in the output section, here we can see the results being generated.

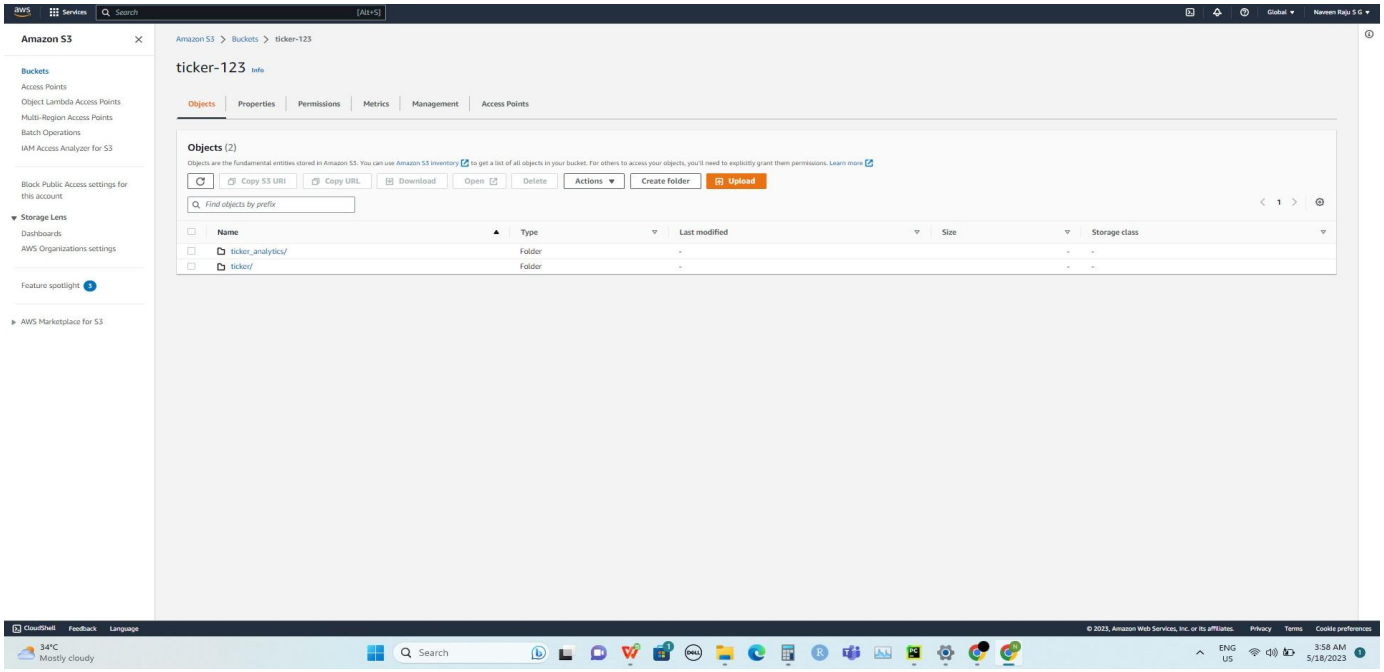


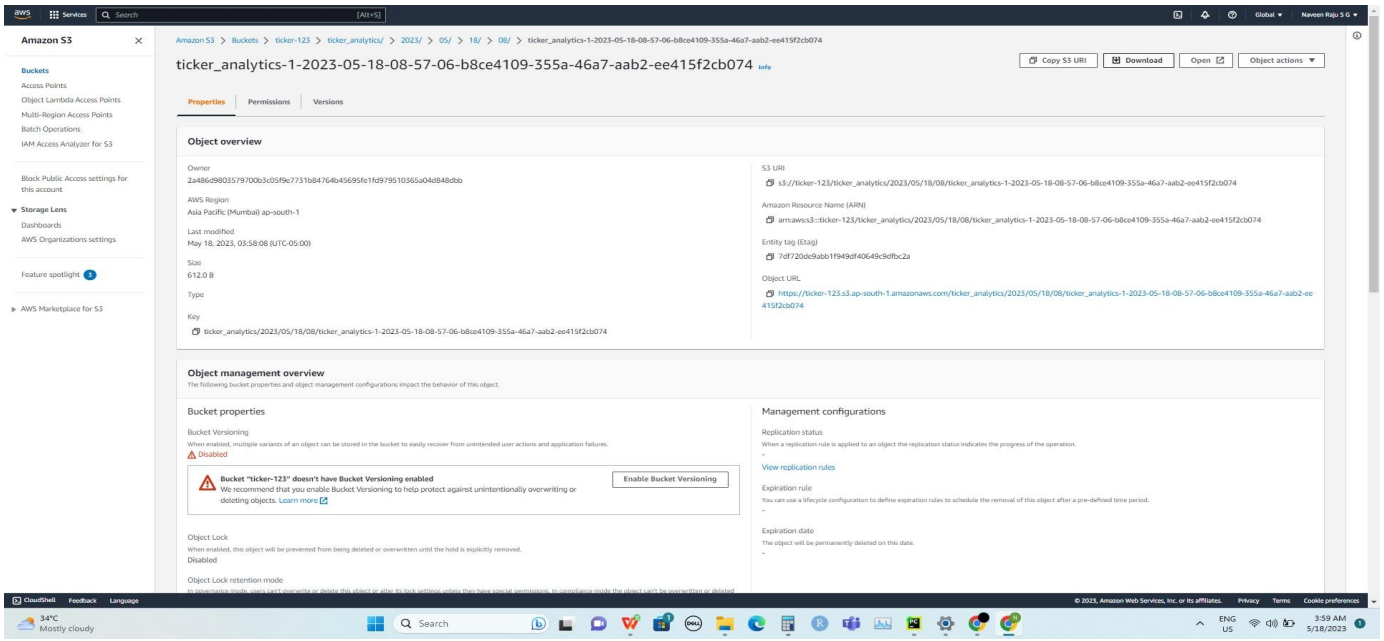


B) <u>View results in S3 buckets</u>

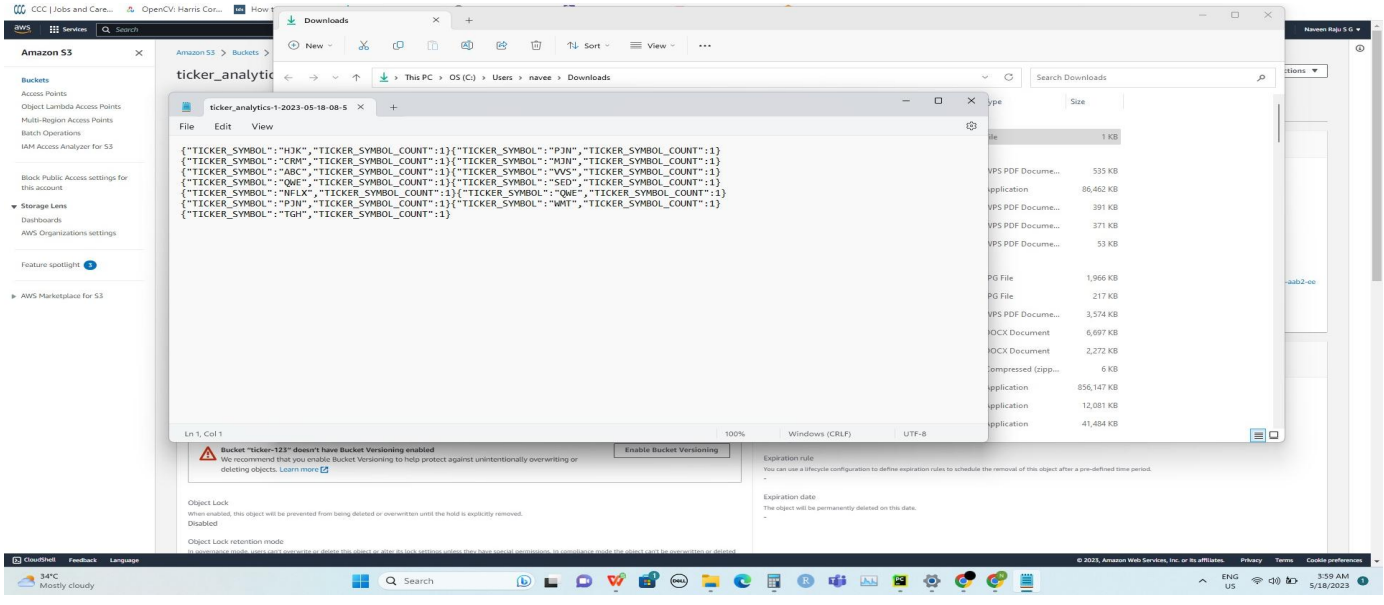
Navigate to S3 buckets to view the results being saved there:

Amazon S3 ->Buckets -> ticker-123 ->ticker_analytics/

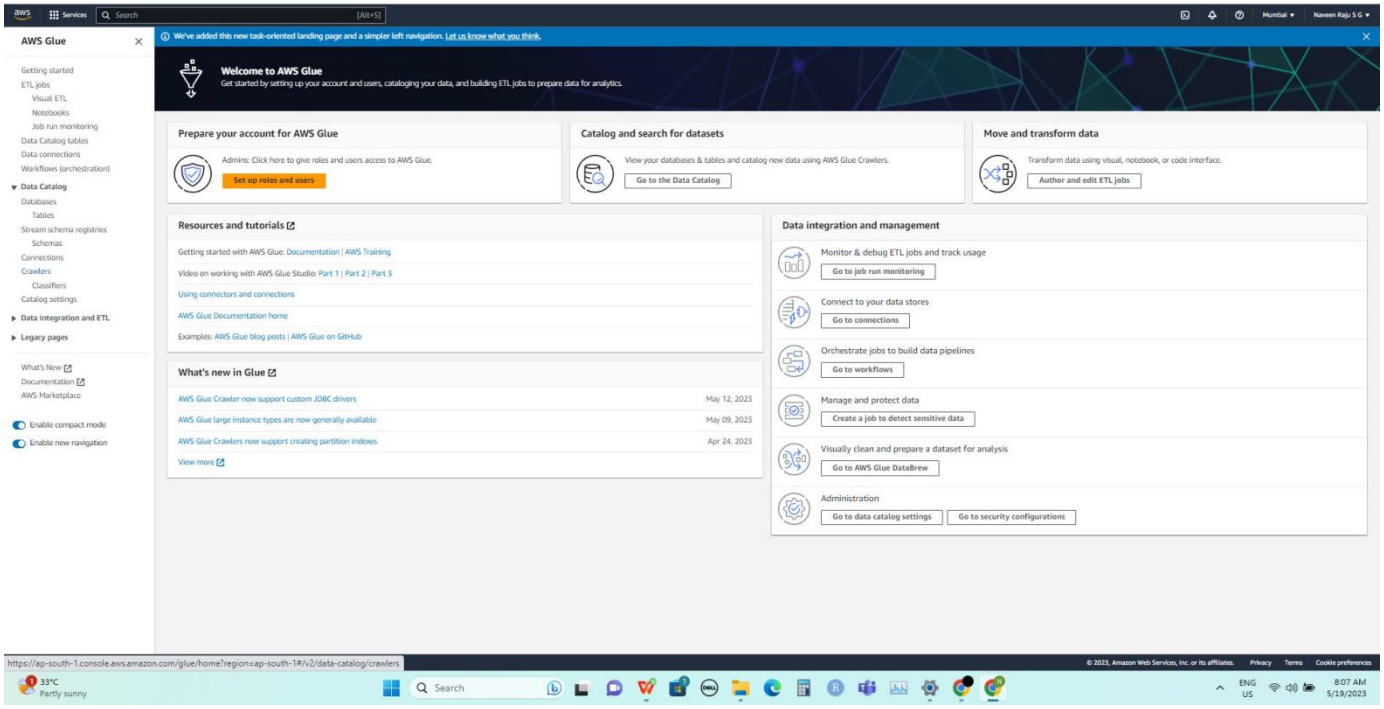




Dowload the results to local system:



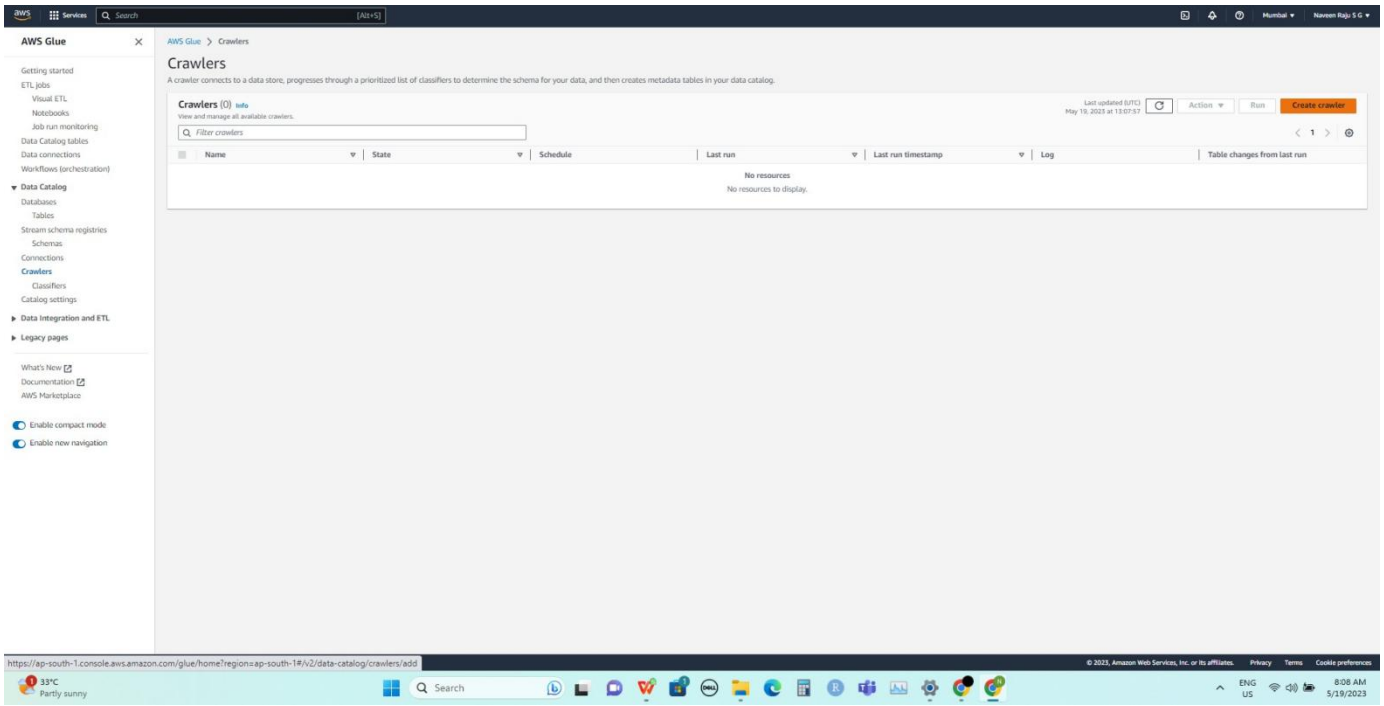
6) Configure, create and run AWS Glue crawler



Navigate to AWS Glue - > Data Catalog - > Crawlers - > Create Crawler

Crawlers

Crawlers connects to your data stores, examines the data, and creates metadata tables in the AWS Glue Data Catalog. This metadata is then used by AWS Glue for ETL (Extract, Transform, Load) jobs and other data processing tasks.



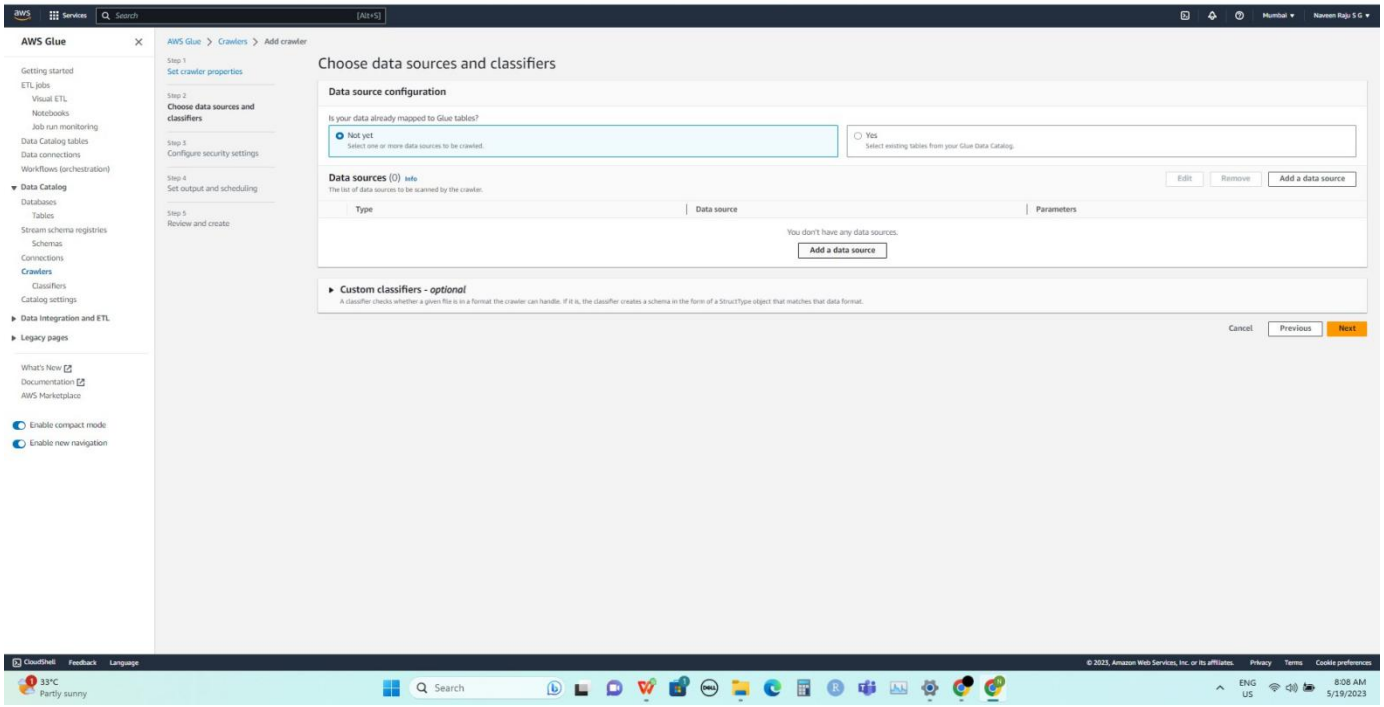
I) Set Crawler properties

Name : DemoCrawler

II)Choose data sources and classifiers

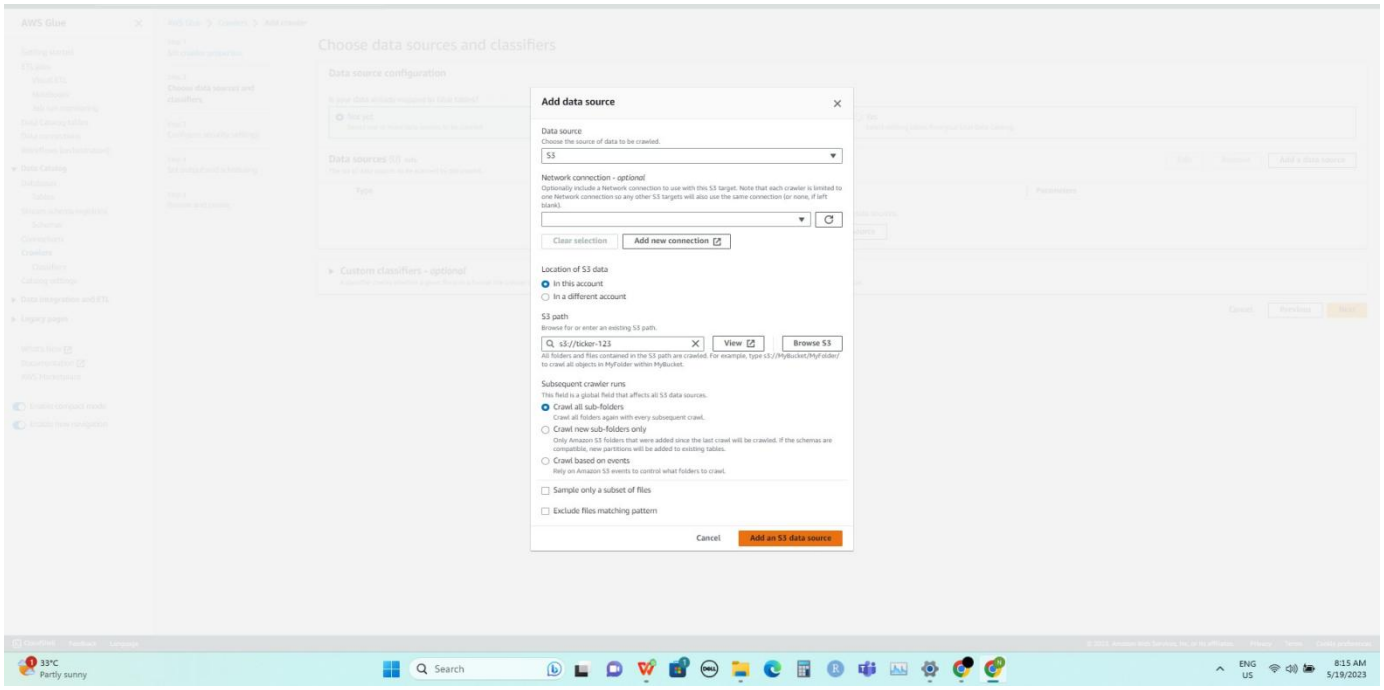
Is your data already mapped to Glue tables : No

Click on Add a Data Source



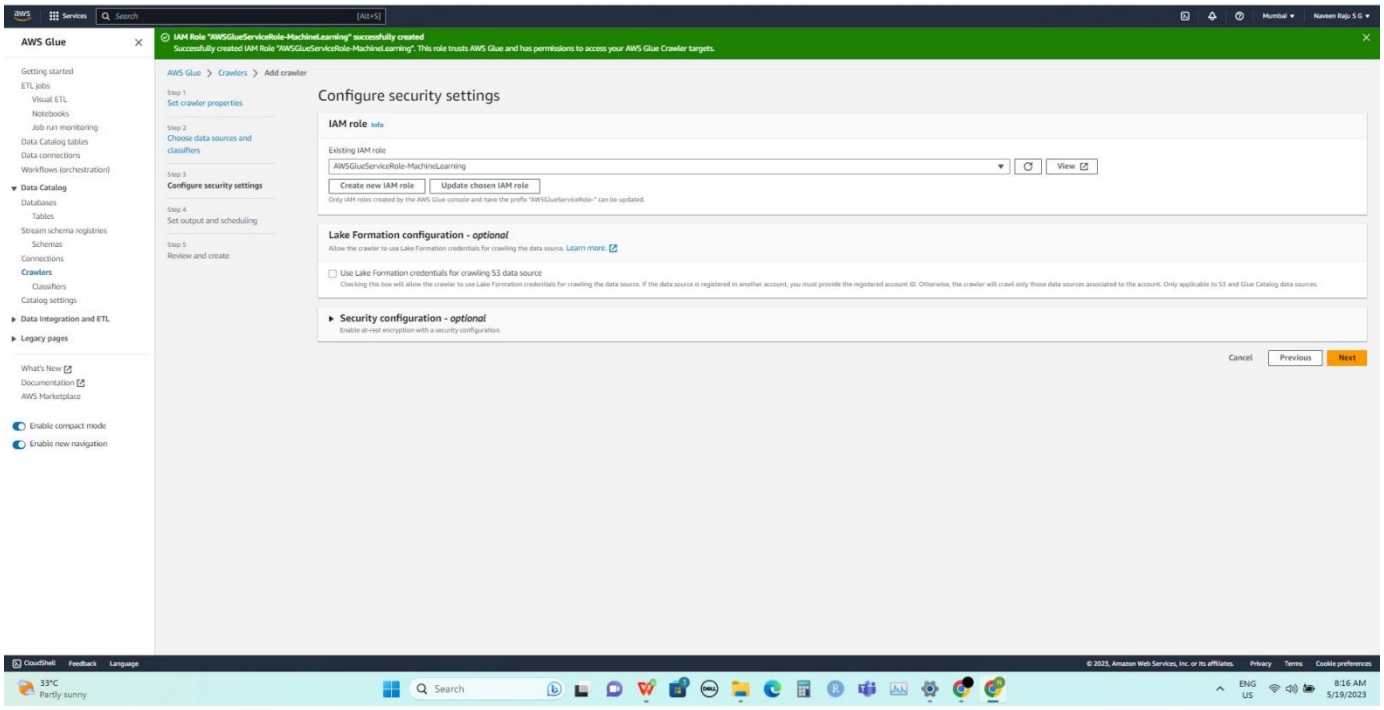
Choose S3 bucket ticker-123 which was created by AWS Kinesis FireHose and AWS Kinesis Analytics application

Subsequent crawler runs : Crawl all sub-folders

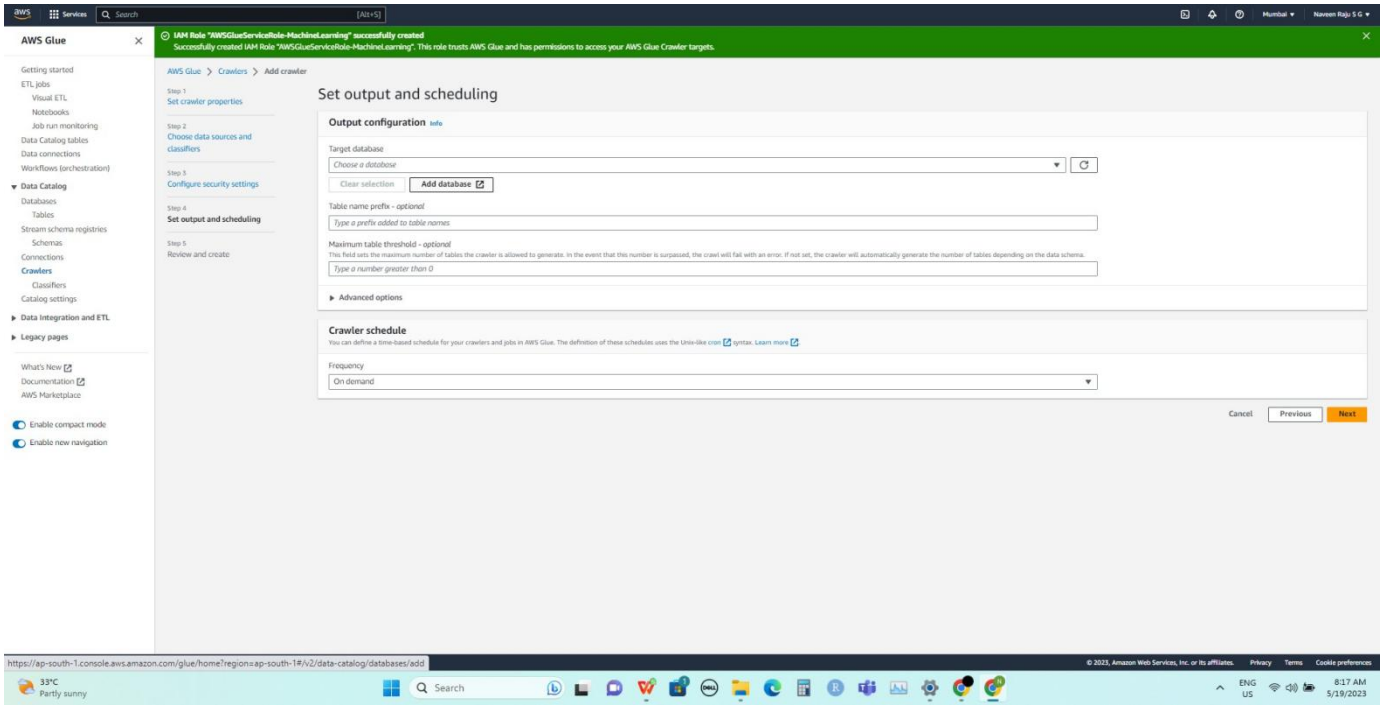


III) Configure security settings

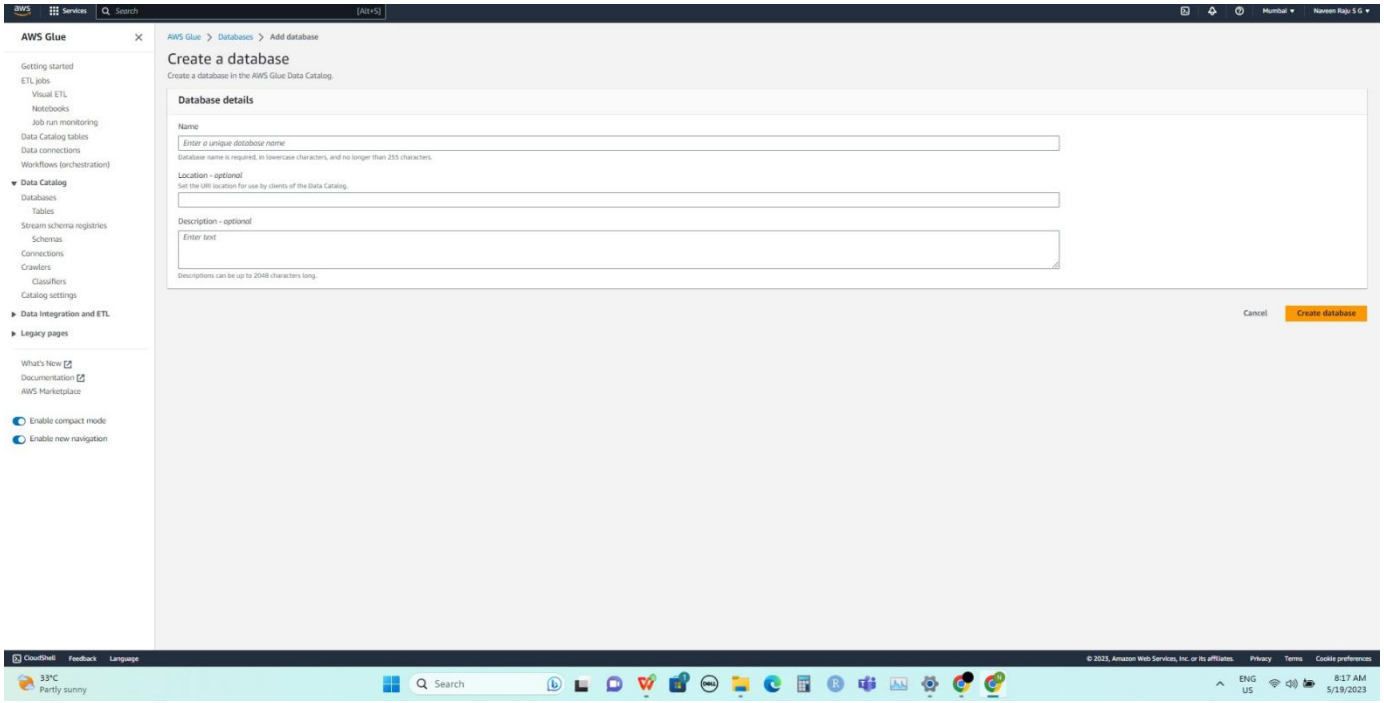
Create a new IAM role : AWSGlueServiceRole-MachineLearning which has following permissions: “s3:GetBucketLocation”,"s3:ListBucket", "s3:ListAllMyBuckets", "s3:GetBucketAcl", "ec2:DescribeVpcEndpoints", "ec2:DescribeRouteTables","ec2:CreateNetworkInterface", "ec2:DeleteNetworkInterface","ec2:DescribeNetworkInterfaces", "ec2:DescribeSecurityGroups", "ec2:DescribeSubnets","ec2:DescribeVpcAttribute", "iam:ListRolePolicies","iam:GetRole", "iam:GetRolePolicy","cloudwatch:PutMetricData", “s3:CreateBucket”, “s3:GetObject”, “s3:PutObject”, “s3:DeleteObject”, “logs:CreateLogGroup”, “logs:CreateLogStream”, “logs:PutLogEvents”, "ec2:CreateTags", "ec2:DeleteTags".

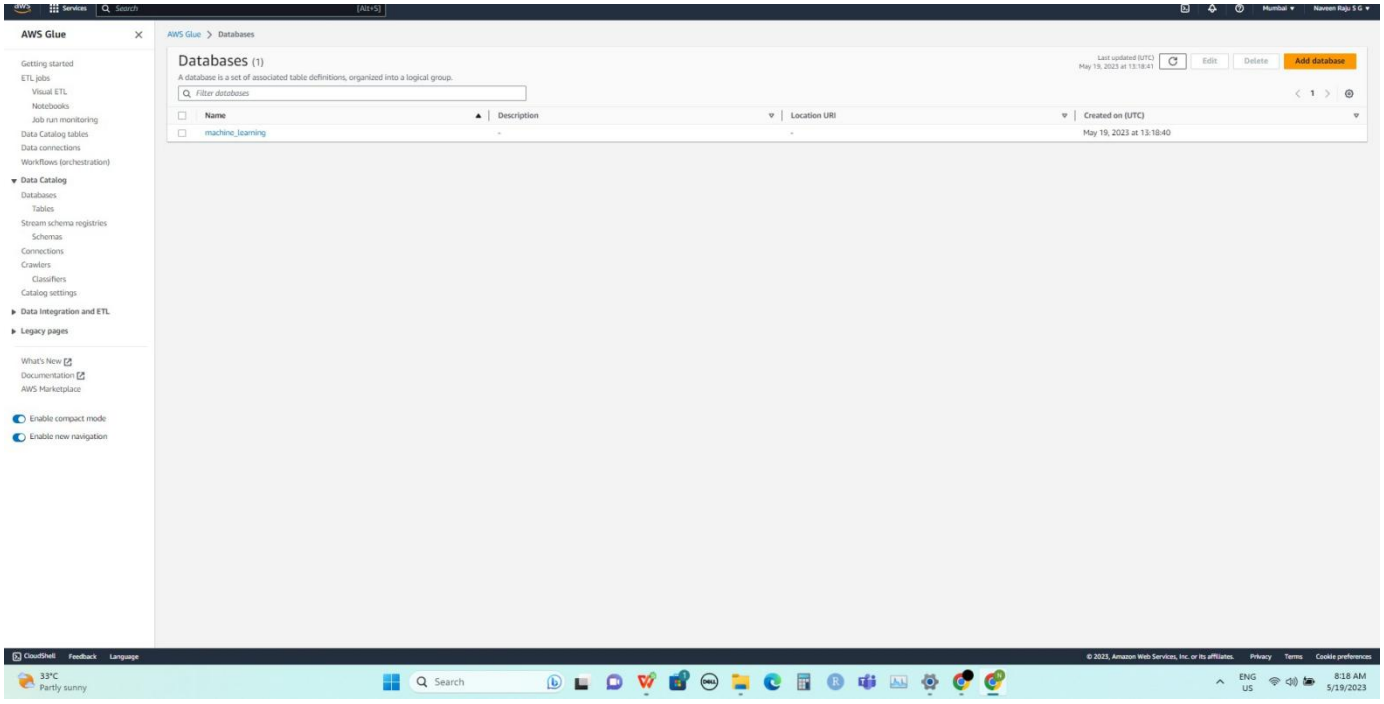


IV)Set output and Scheduling



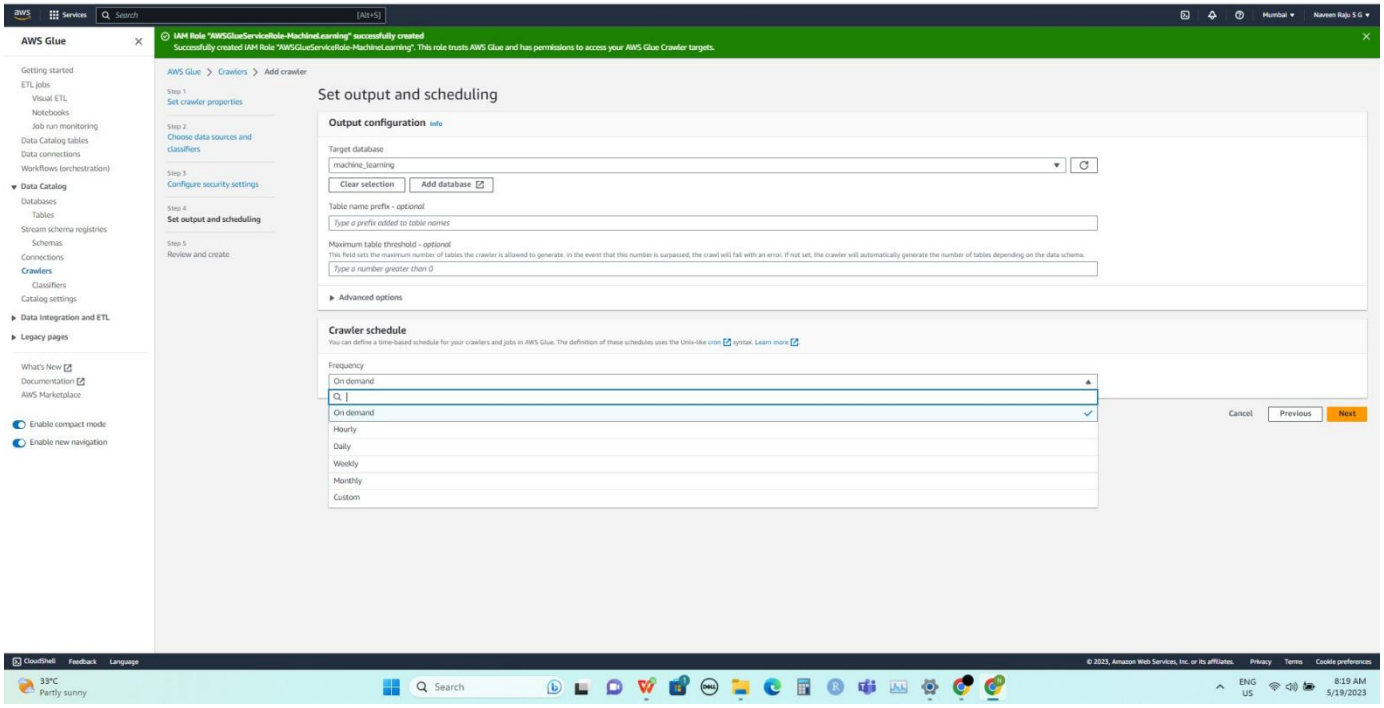
Click on Add Database





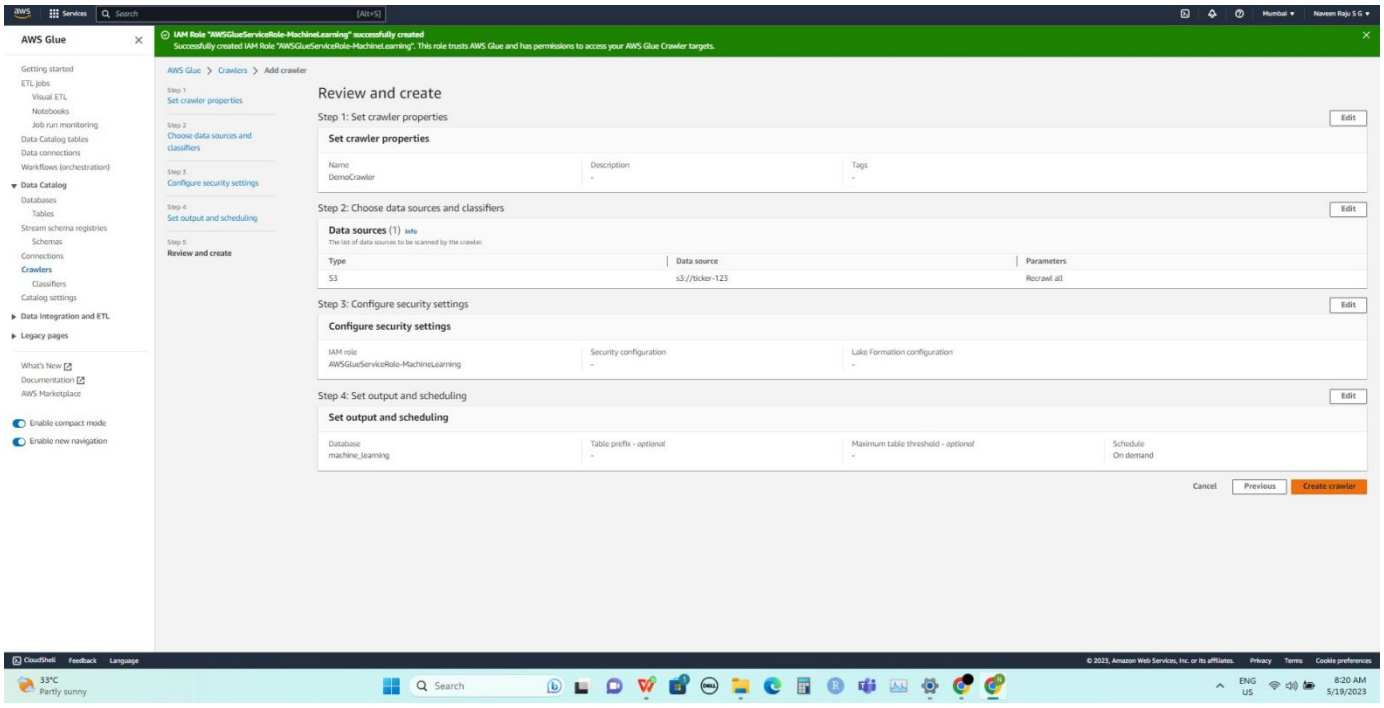
Continue configuring “Set output and Scheduling”

Target database : machine_learning (previously created for AWS Kinesis)
Crawler Schedule : On demand

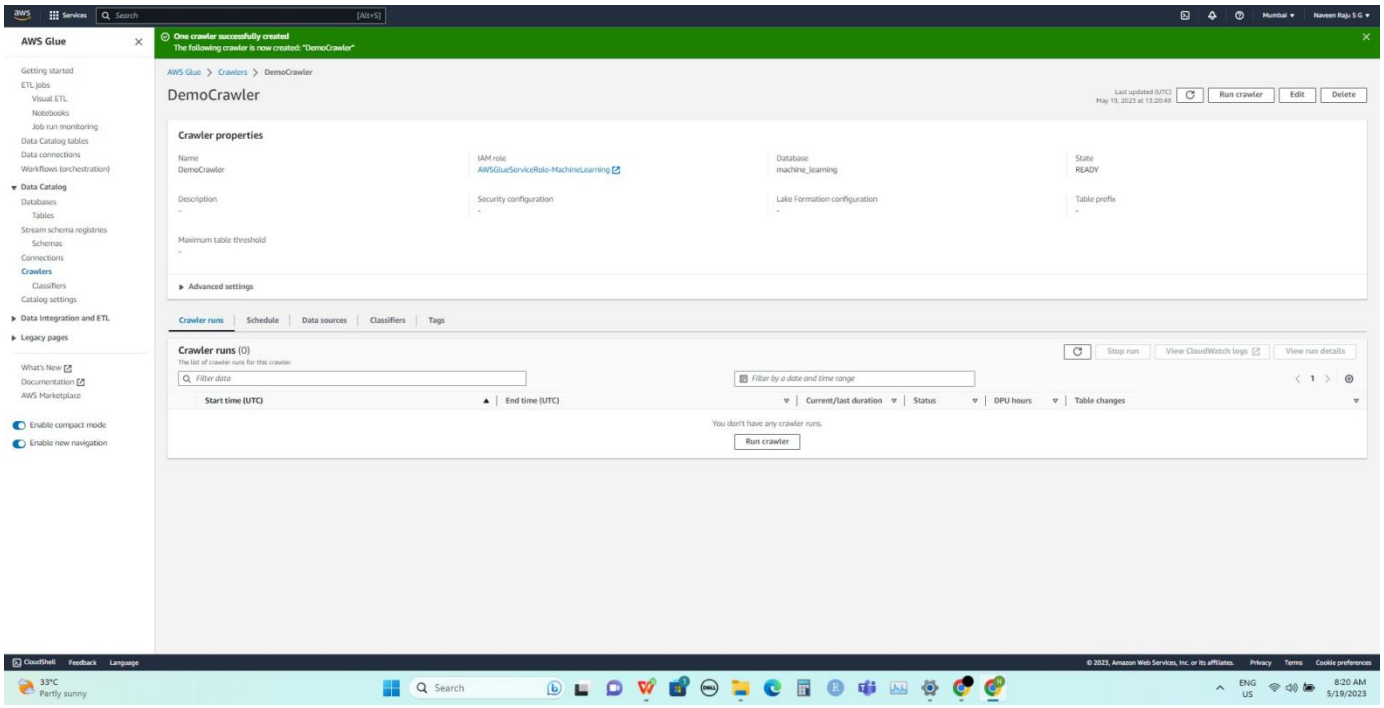


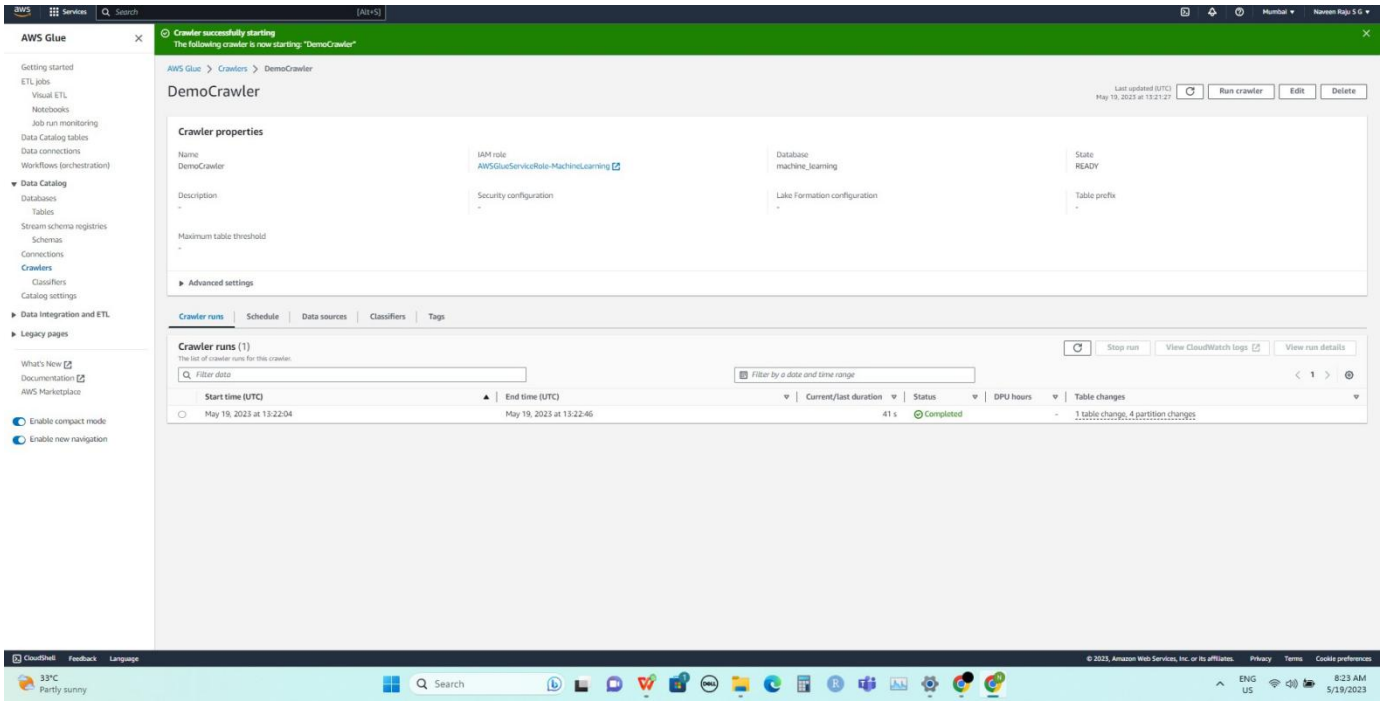
V)Create and run Crawler

Click on Create Crawler



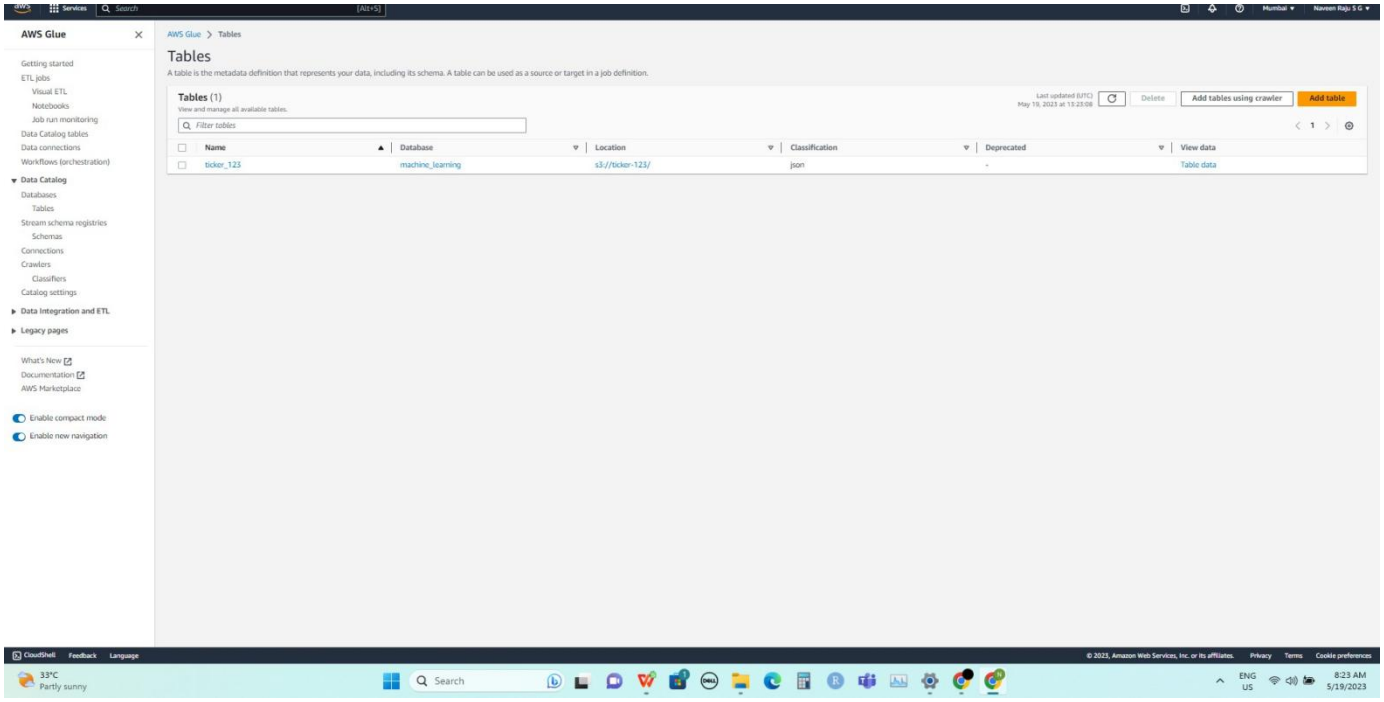
Click on Run Crawler





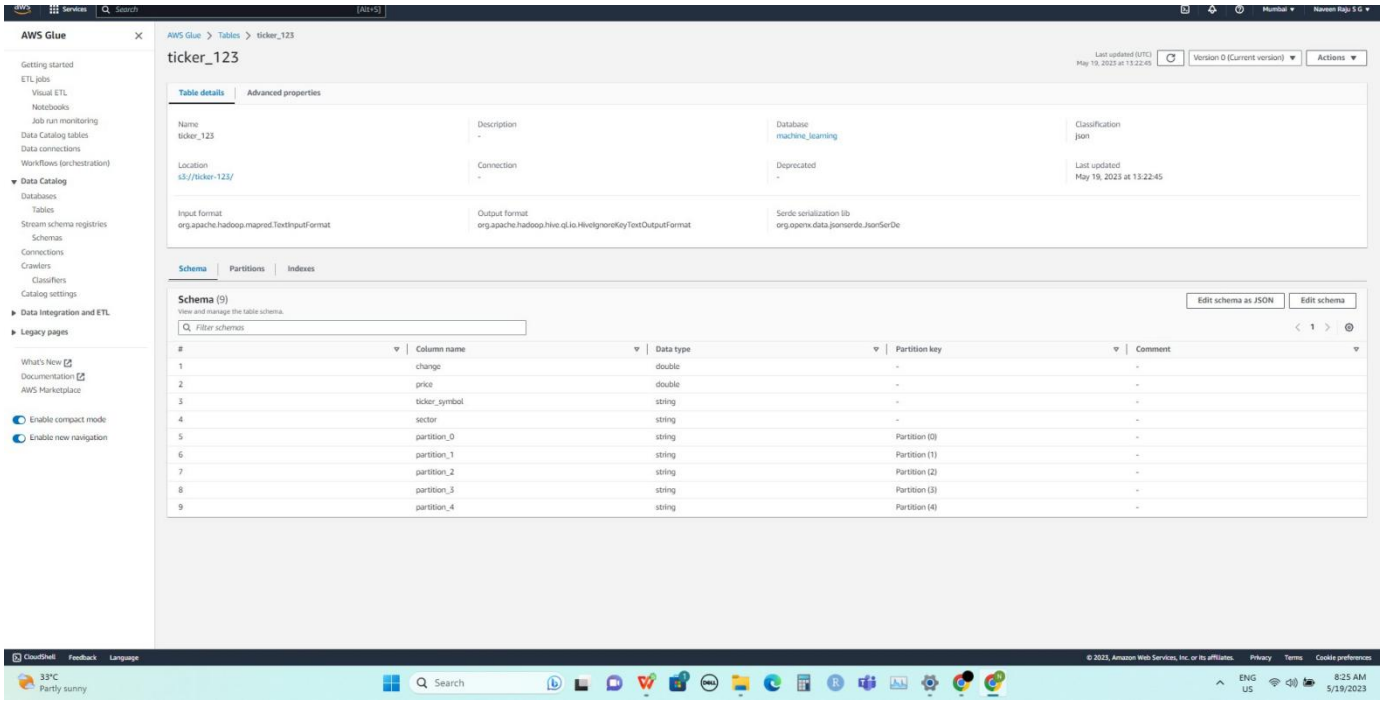
VI)View results in AWS Glue tables

Navigate to AWS Glue -> Tables
We can see a table has been created in the database.

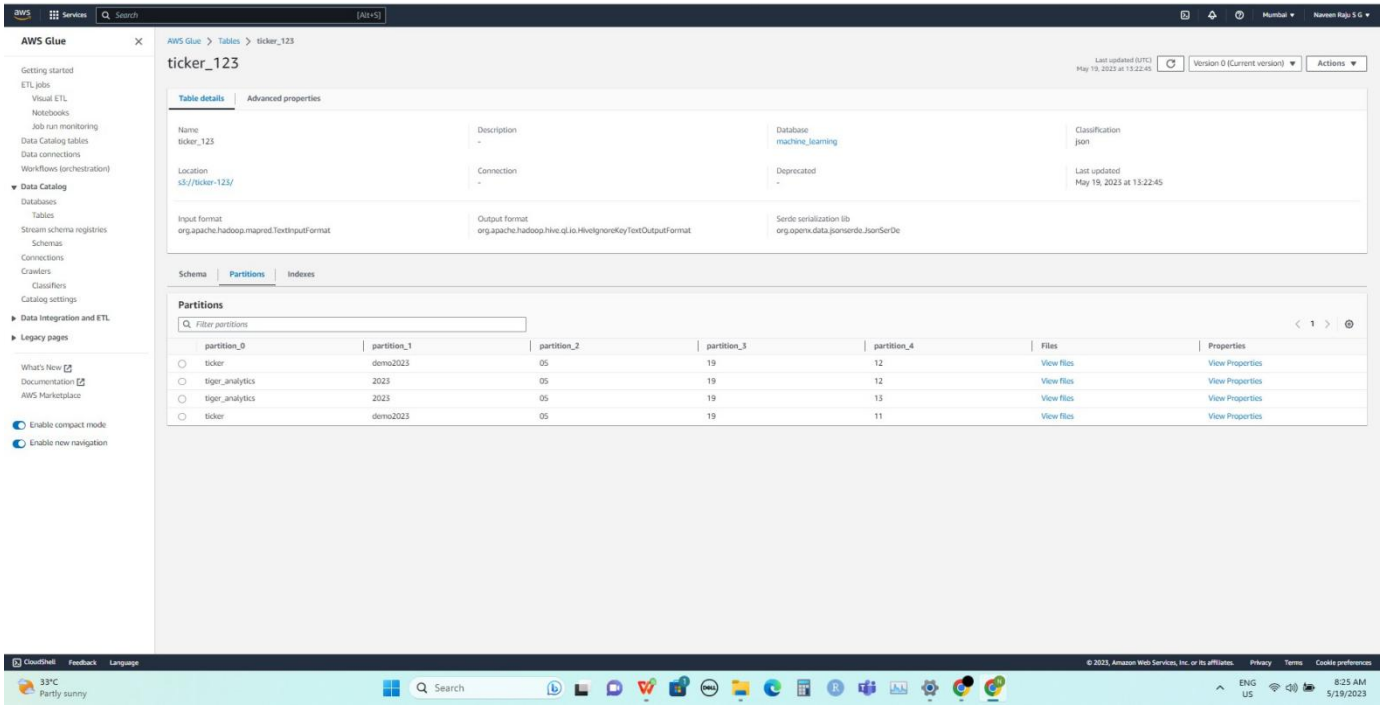


Click on the table “ticker_123” to see:

Schema

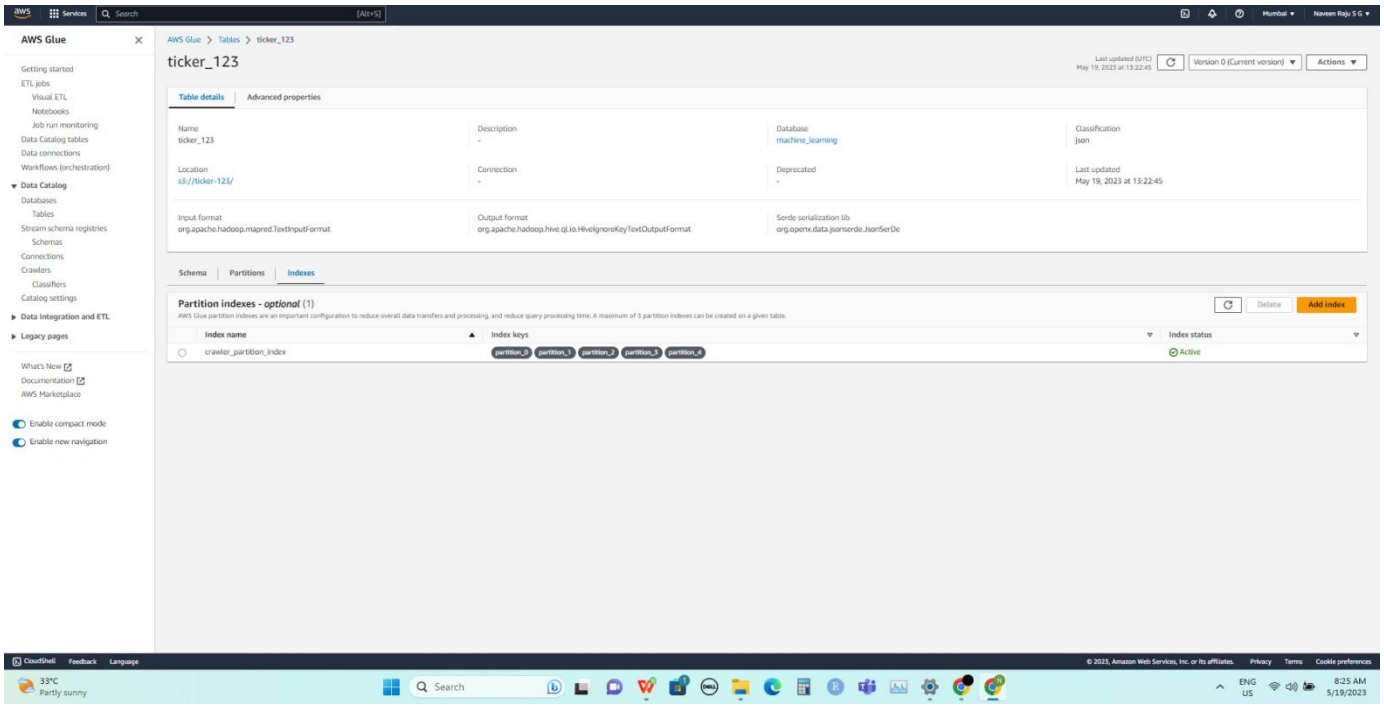


Partitions

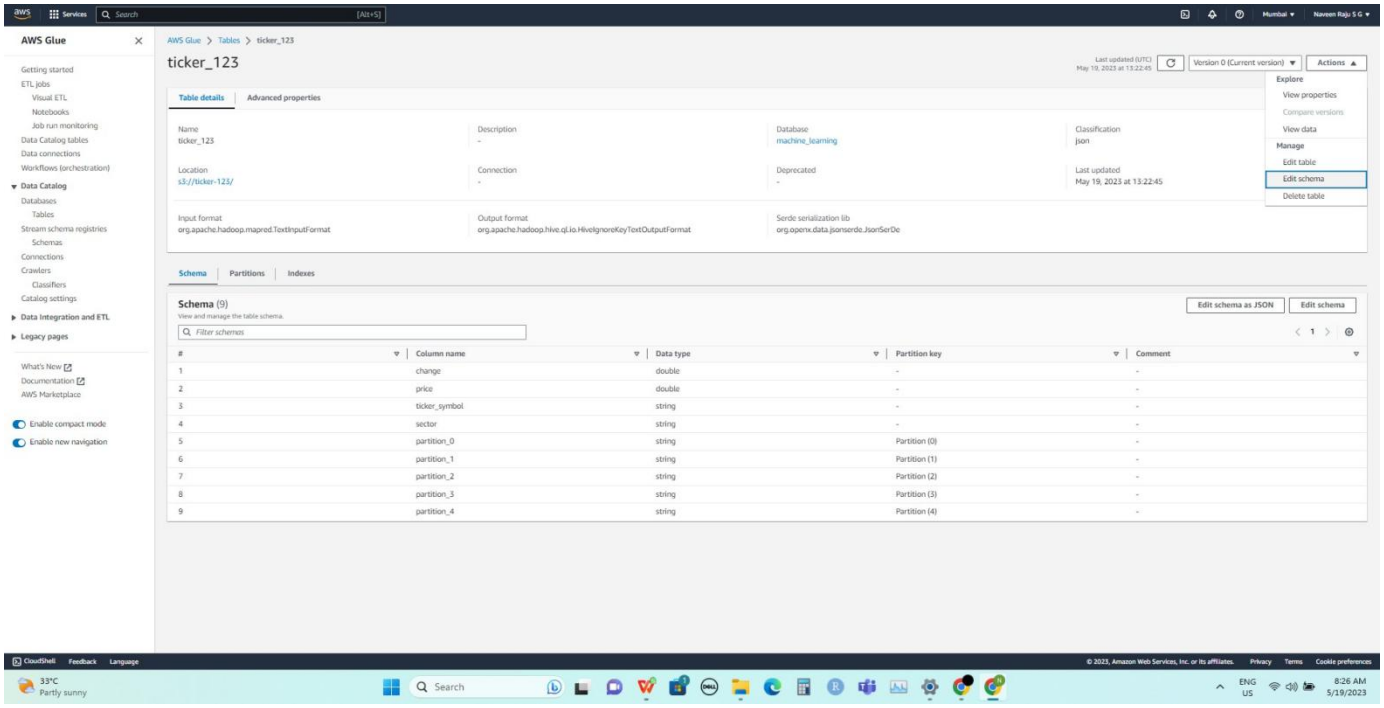


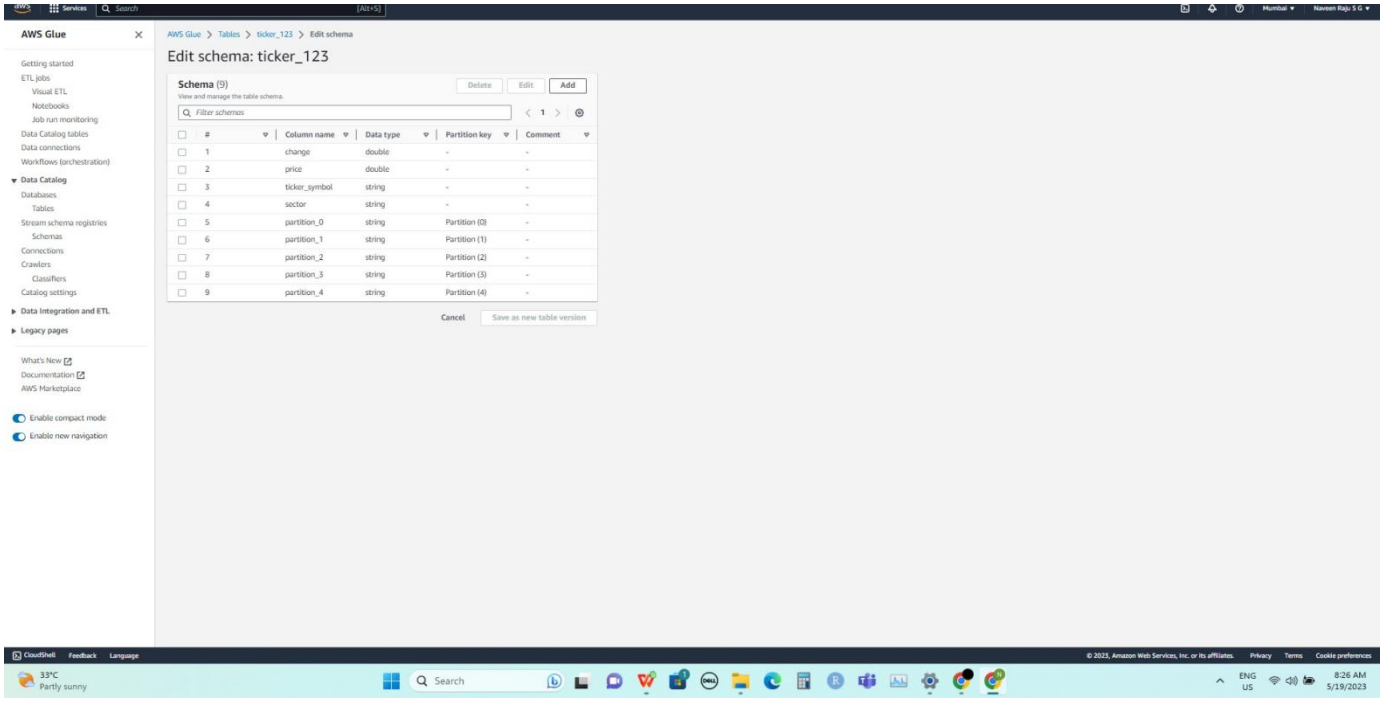
Partitions Indexes

AWS Glue partition indexes are an important configuration to reduce overall data transfer and processing, and reduce query processing time.

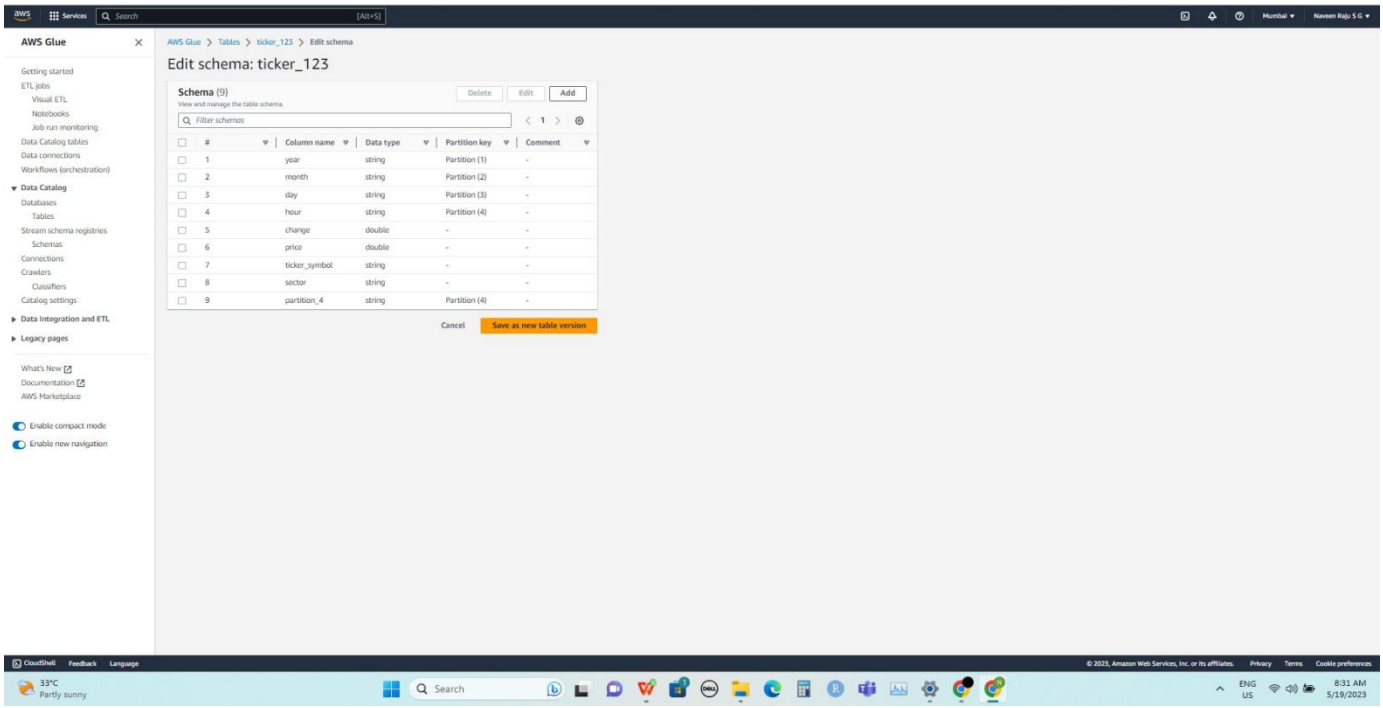


On the same window Navigate to Actions->Edit Schema





Edit names of all partitions as year, month, day, hour because this was the format the files were created in S3 buckets.



7) <u>Configure, create and run AWS Glue studio ETL job</u>

Navigate to “Data Integration and ETL“ -> AWS Glue Studio -> ETL Jobs

Now lets convert the JSON format table created in last step to Parquet format table.

Parquet File - is a columnar storage file format widely used in the big data ecosystem.

Three Main Key feature of Parquet File:

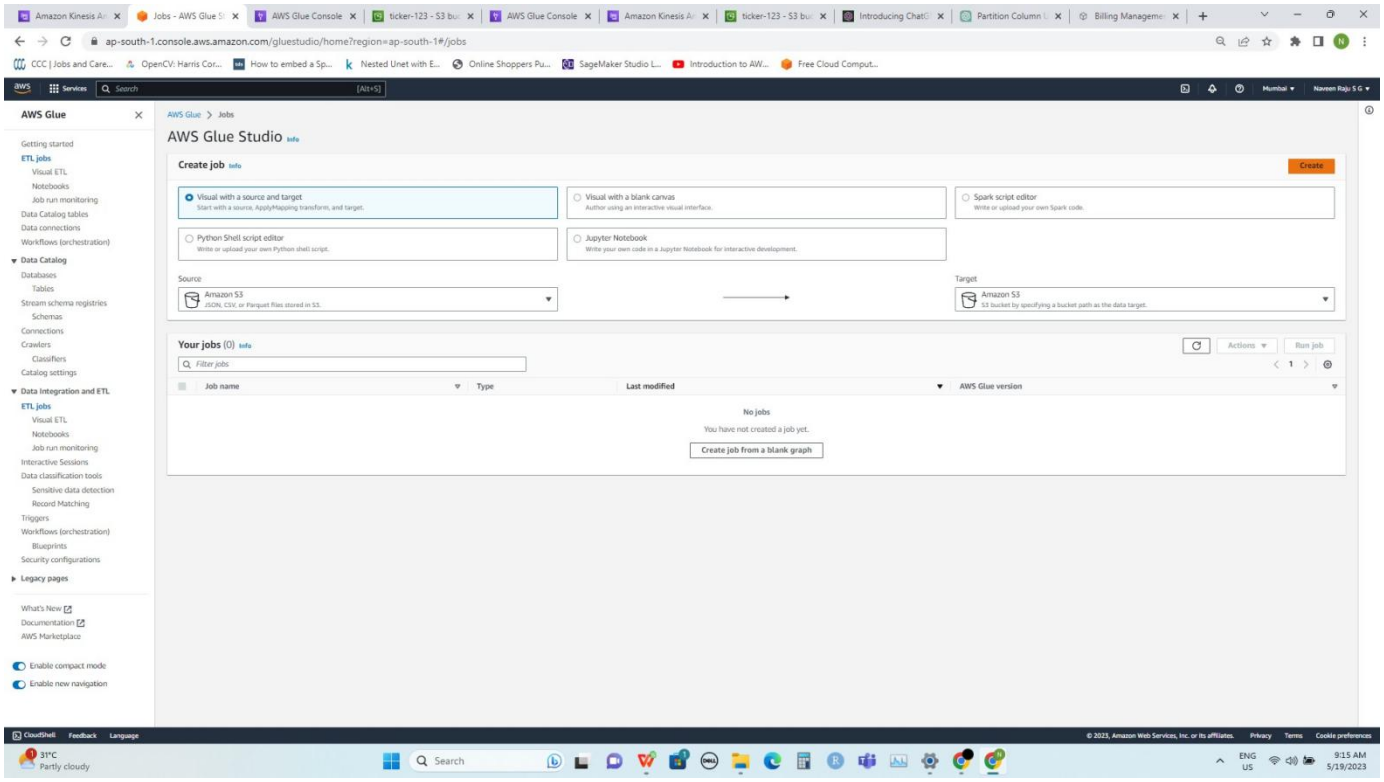
Columnar Storage - Parquet stores data in a columnar format rather than the traditional row-based format. This means that the values of each column are stored together, which offers several advantages for analytical processing.

Performance - Due to its columnar nature, Parquet is highly optimized for analytical queries and can significantly speed up query performance.

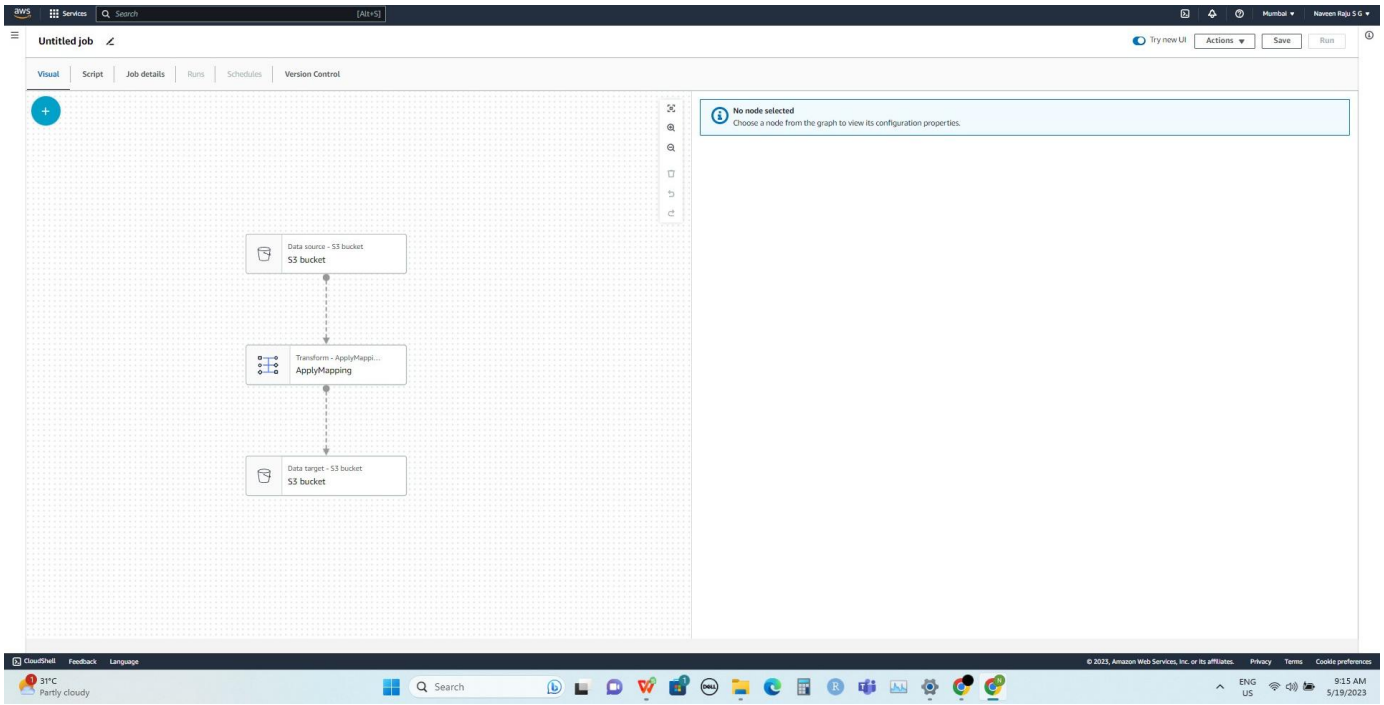
Schema Evolution: Parquet supports schema evolution, meaning you can add new columns to your data over time without breaking compatibility with existing data. This makes it easier to evolve and manage data schemas in a flexible manner.

Create Job config: Visual with Source and Target

Click on create

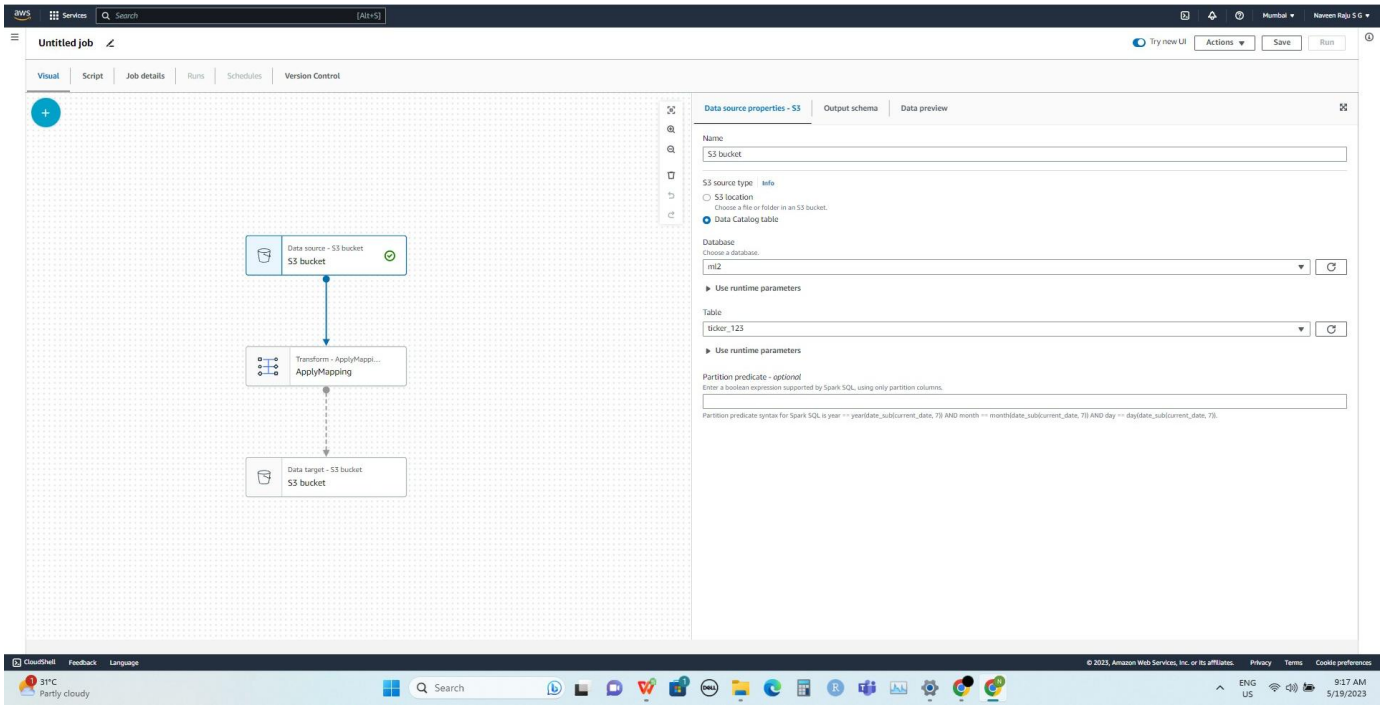


Configure 3 parts of the flow chart : Data Source - S3 bucket, Transform - ApplyMapping, Data target - S3 bucket and also configure Job details



I) Configure Data Source-S3 bucket

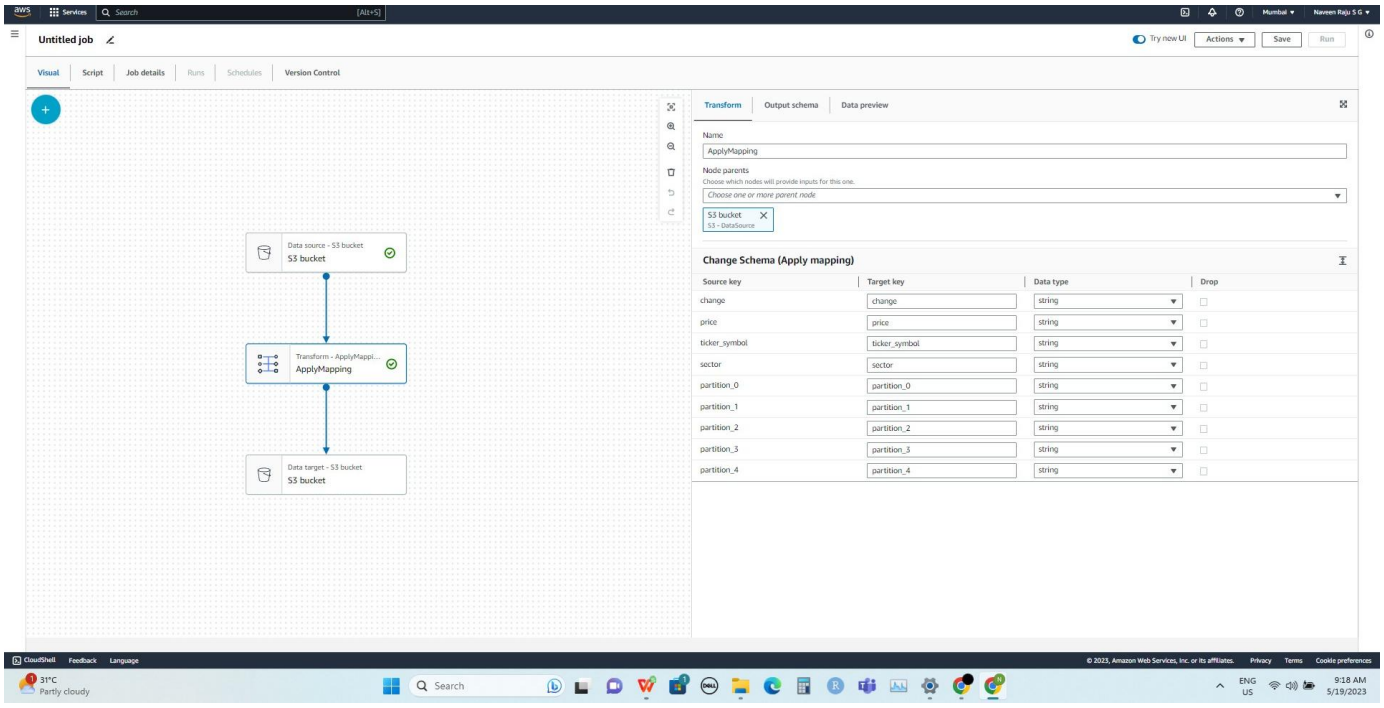
Name : S3 bucket
S3 Source type : Data Catalog table
Database : machine_learning
Table : ticker_123



II) Configure Transform-Apply Mapping

Name : Apply Mapping

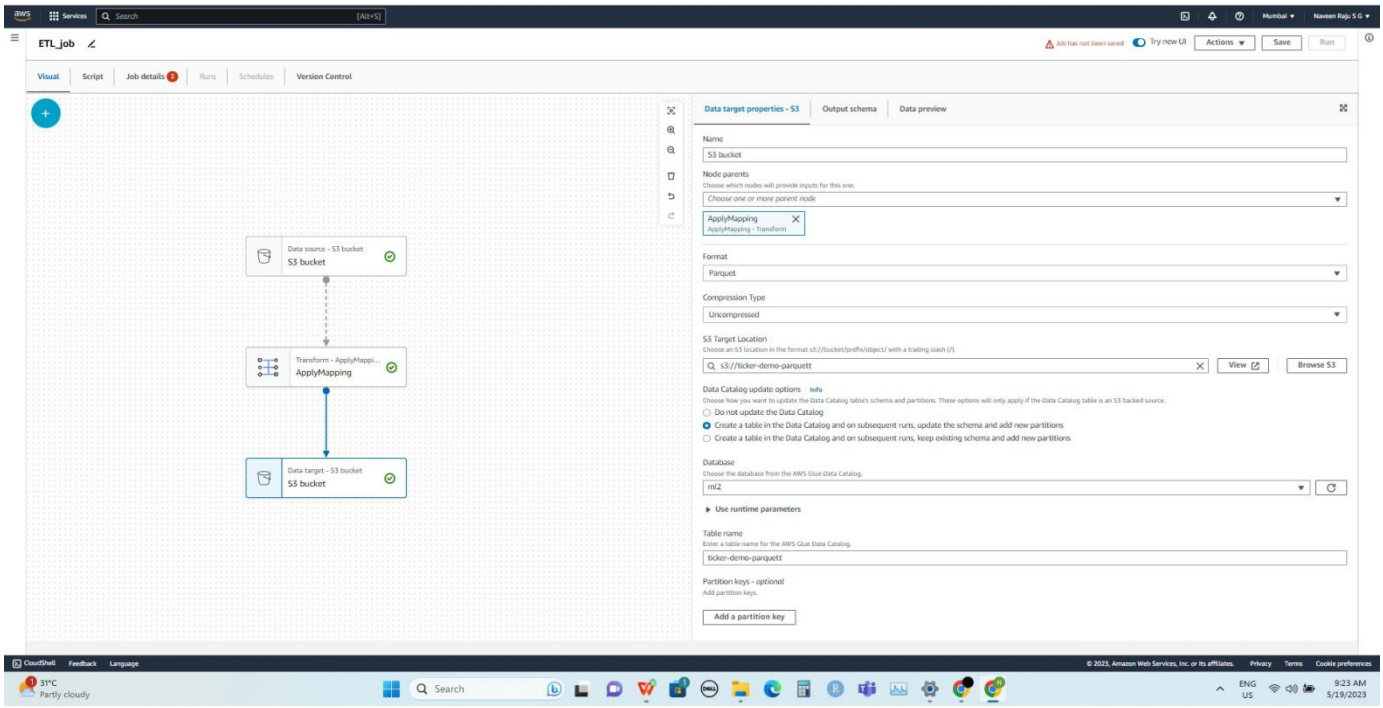
Node parents (which node will provide inputs for this) : S3 bucket



III) Configure Data target - s3 bucket

Name : S3 bucket
Node parents (Node which will provide input to this one) : ApplyMapping
Format : Parquet
Compression Type : Uncompressed
S3 Target location : s3://ticker-demo-parquett (Create this s3 bucket before giving this location)

Data Catalog update options : Create a table in the Data Catalog and on subsequent runs, update the schema and add new partitions
Database : machine_learning
Table name : ticker-demo-parquett



IV) Configure Job details

Create IAM role:

Navigate to Identity and Access Management(IAM) page and create a new role

Step 1 (Select trusted entity):
Trusted entity type : AWS service
Use case : Glue (Allows Glue service to call AWS services on our behalf)

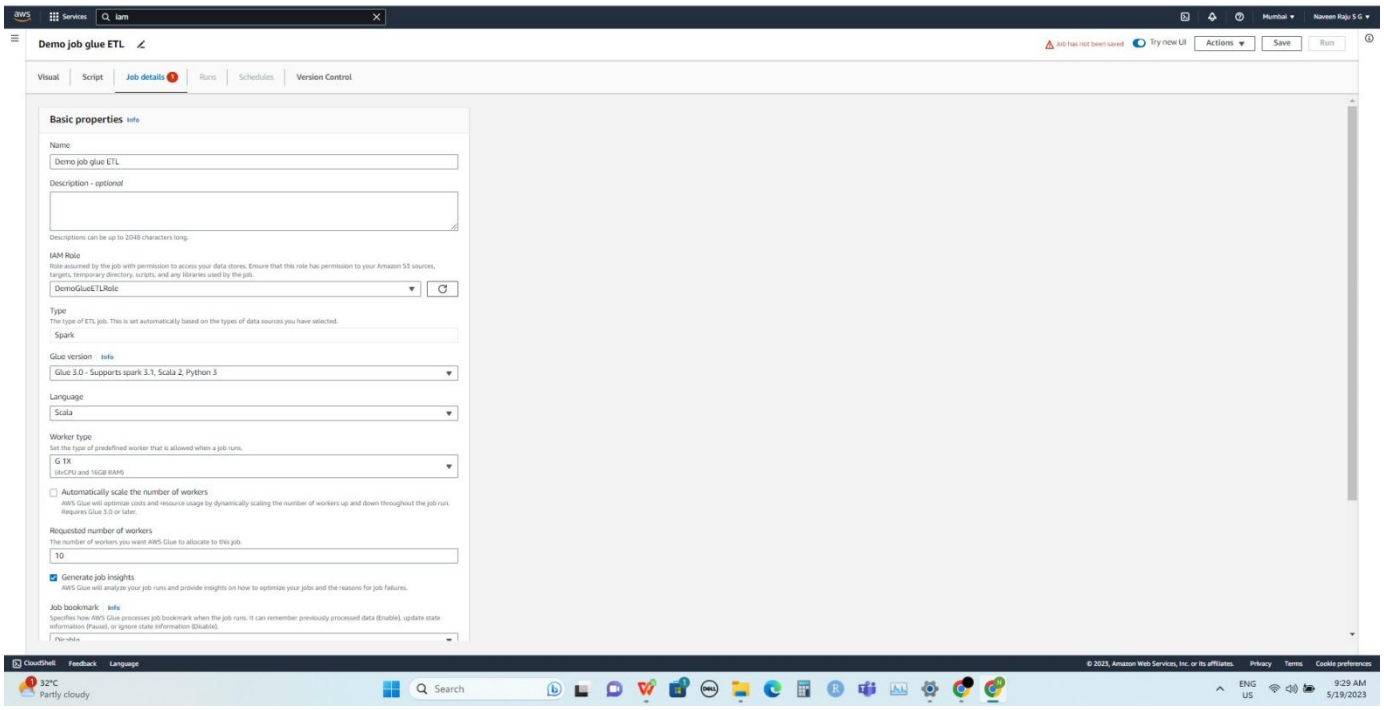
Step 2 (Add permissions):
AmazonS3FullAccess
AWSGlueServiceRole

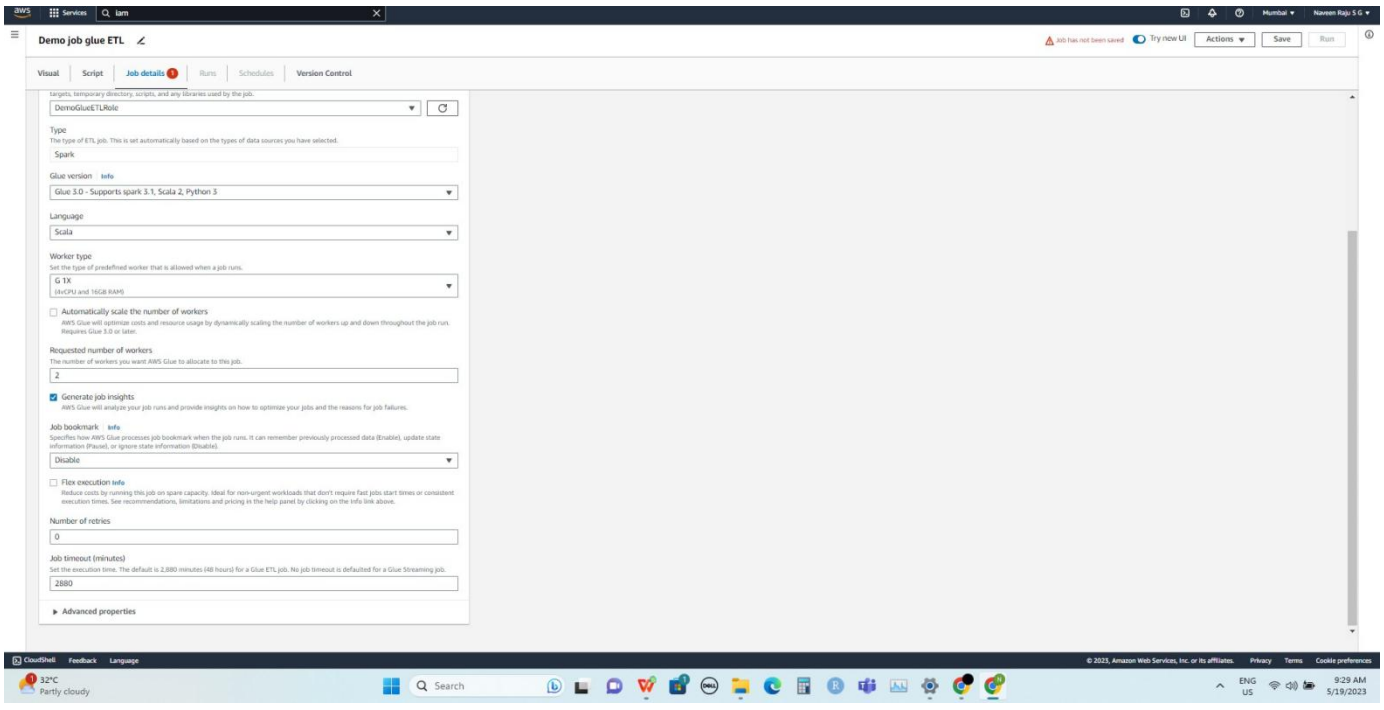
Name of the role : DemoGlueETLRole

Click on create role.

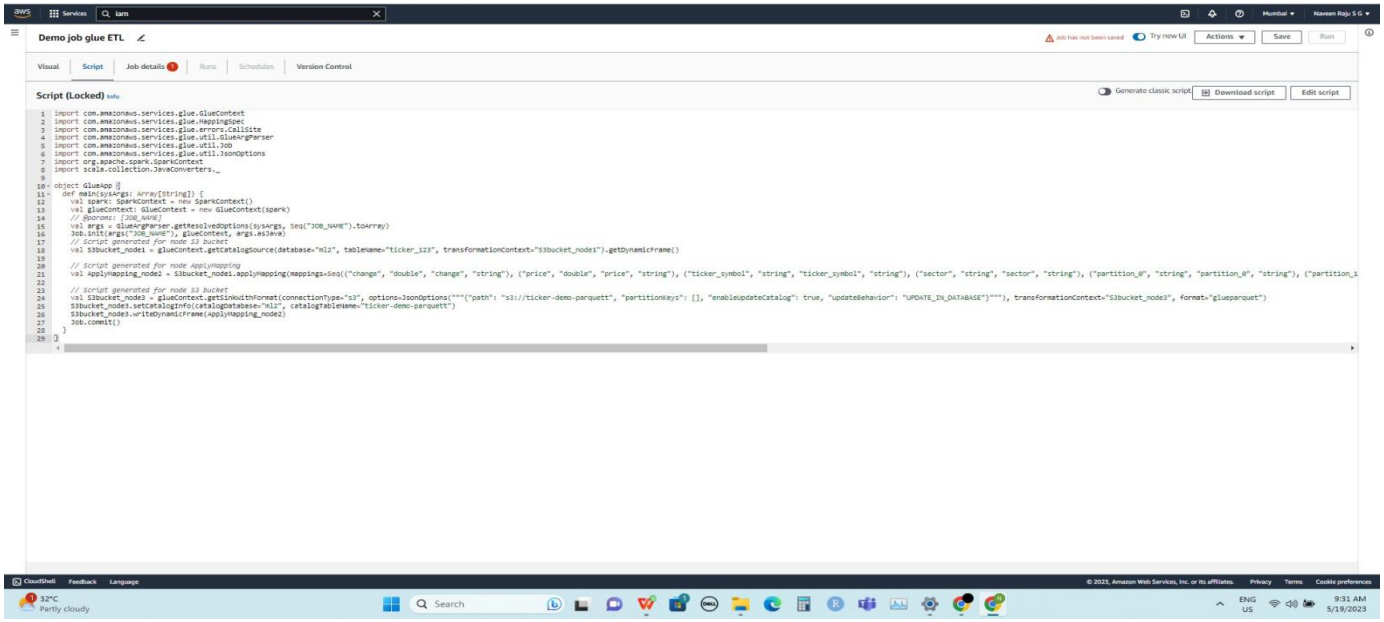
Job Details config:

Name : Demo job glue ETL
IAM Role : DemoGlueETLRole
Type (The type of ETL job. This is set automatically based on the types of data sources you have selected) : Spark
Glue version : Glue 3.0 - Supports spark 3.1,Scala 2, Python 3
Language : Scala
Worker type(Predefined worker that is allowed when a job runs) : G1X (4VCPU and 16GB RAM)
Requested number of workers (The number of workers we want AWS Glue to allocate to this job) : 2
Job timeout (Max execution time is default 2880 minutes for a Glue ETL job) : 2880


Demo job glue ETL
Visual
Script
Job details
Runs
Schedules
Version Control
Basic properties
Name
Demo job glue ETL
Description - optional
IAM Role
DemoGlueETLRole
Type
Spark
Glue version
Glue 3.0 - Supports spark 3.1, Scala 2, Python 3
Language
Scala
Worker type
G 1X
Automatically scale the number of workers
Requested number of workers
10
Generate job insights
Job bookmark


Demo job glue ETL
Visual
Script
Job details
Runs
Schedules
Version Control
DemoGlueETLRole
Type
Spark
Glue version
Glue 3.0 - Supports spark 3.1, Scala 2, Python 3
Language
Scala
Worker type
G 1X
Automatically scale the number of workers
Requested number of workers
2
Generate job insights
Job bookmark
Disable
Flex execution
Number of retries
0
Job timeout (minutes)
2880
Advanced properties

View the script generated:



Continue editing JobConfig ->Advance Properties:

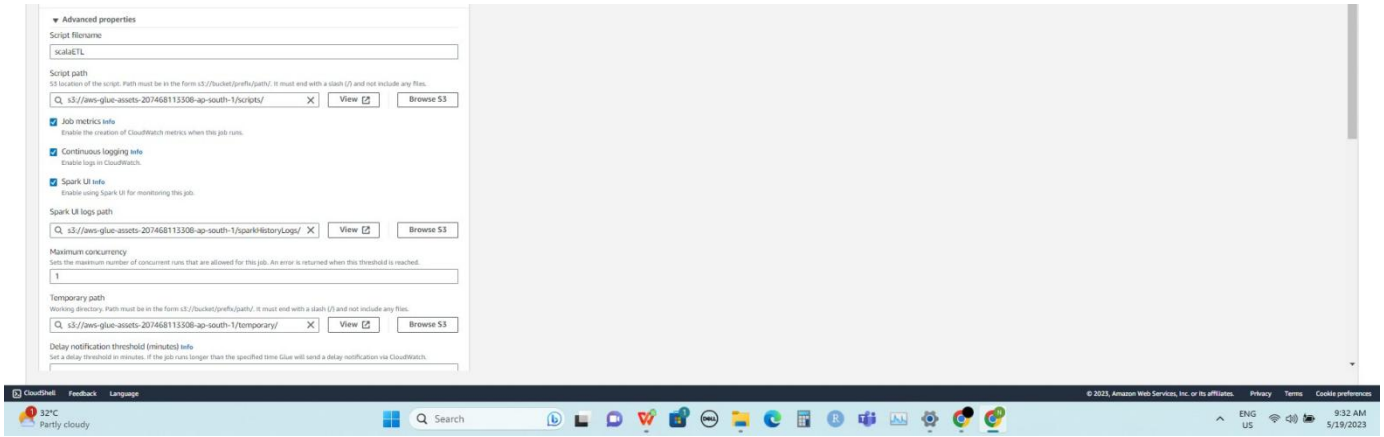
Script filename : scalaETL
Script Path (S3 path) : keep the default path which would have auto generated

Select the check box of the following:
Job metrics (Enable the creation of Job Metrics when the job runs)
Continuous logging (Enable logs in Cloud watch)
Spark UI (Enable using Spark UI for monitoring this job)

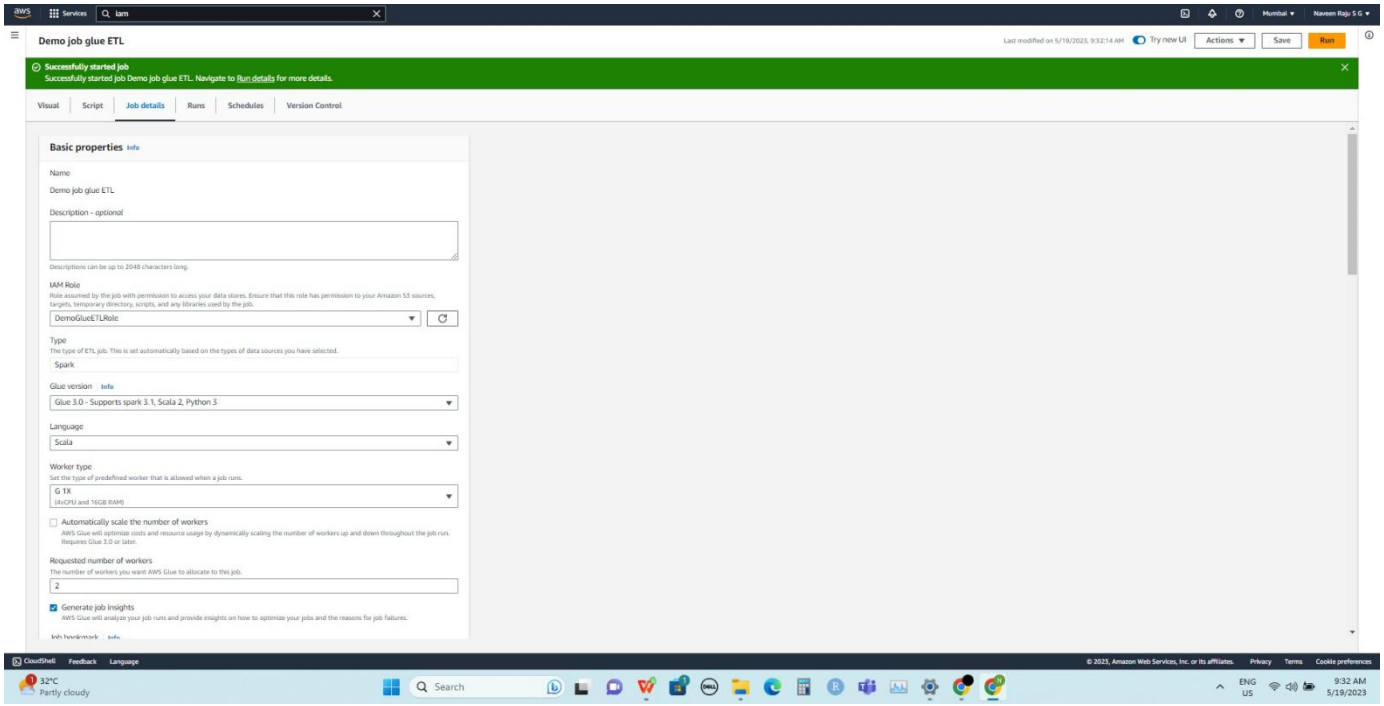
Spark UI logs path : keep the default path which would have auto generated

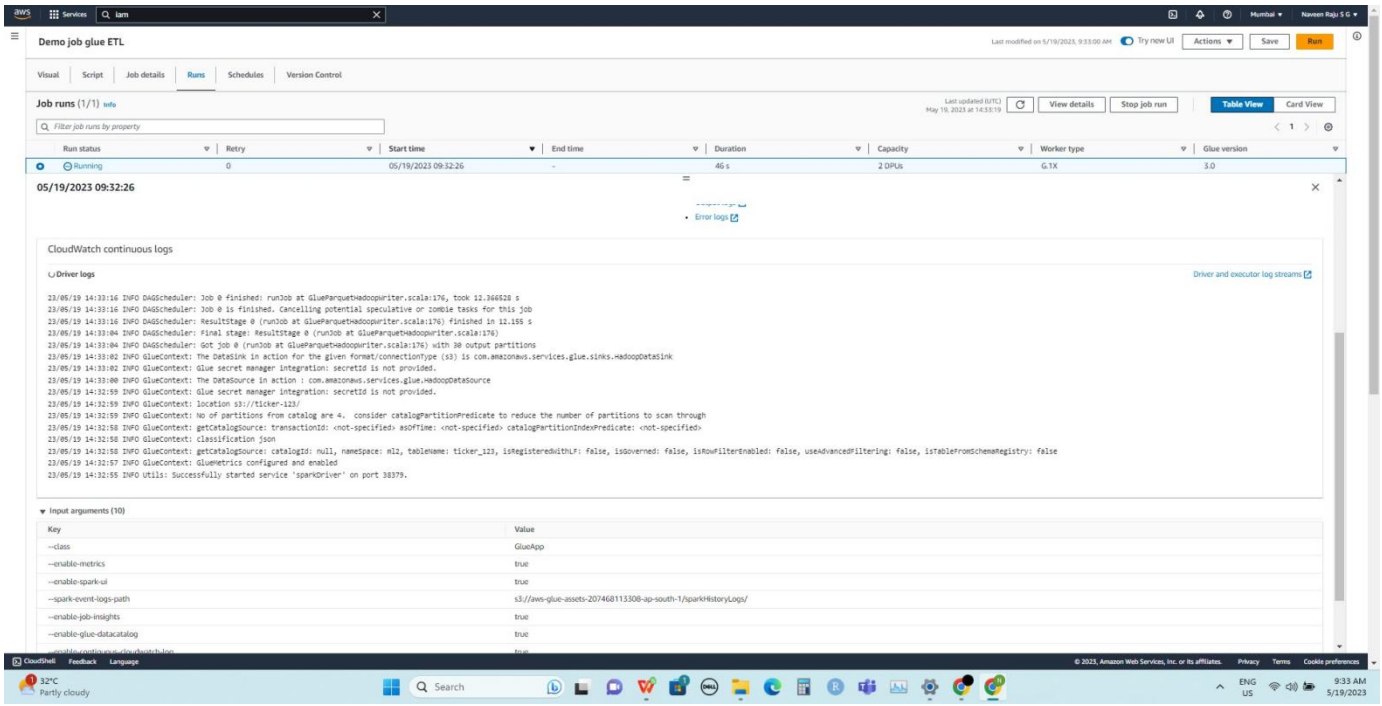
Maximum concurrency (Maximum concurrent runs for this jobs that are allowed) : 1

Temporary path(working directory) : keep the default path which would have auto generated
Security Configuration (encryption) : None

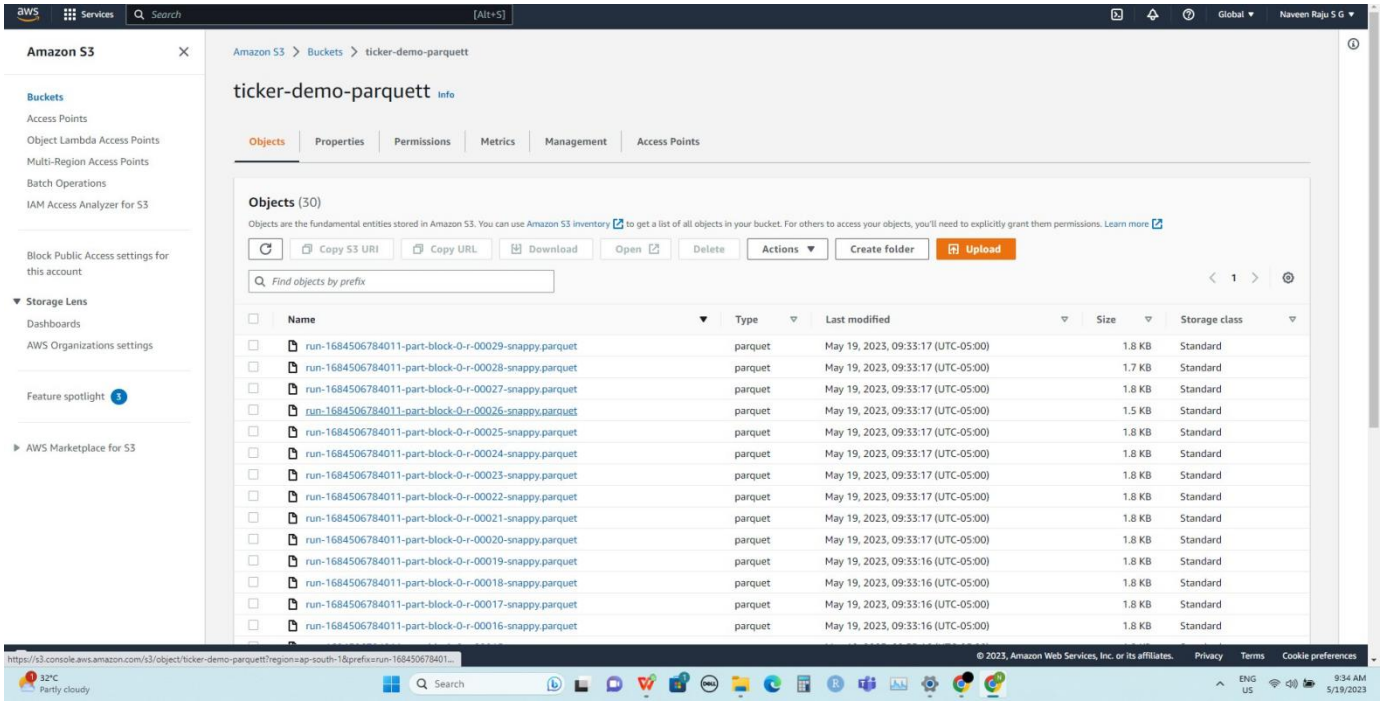


Save the config and Run the job:

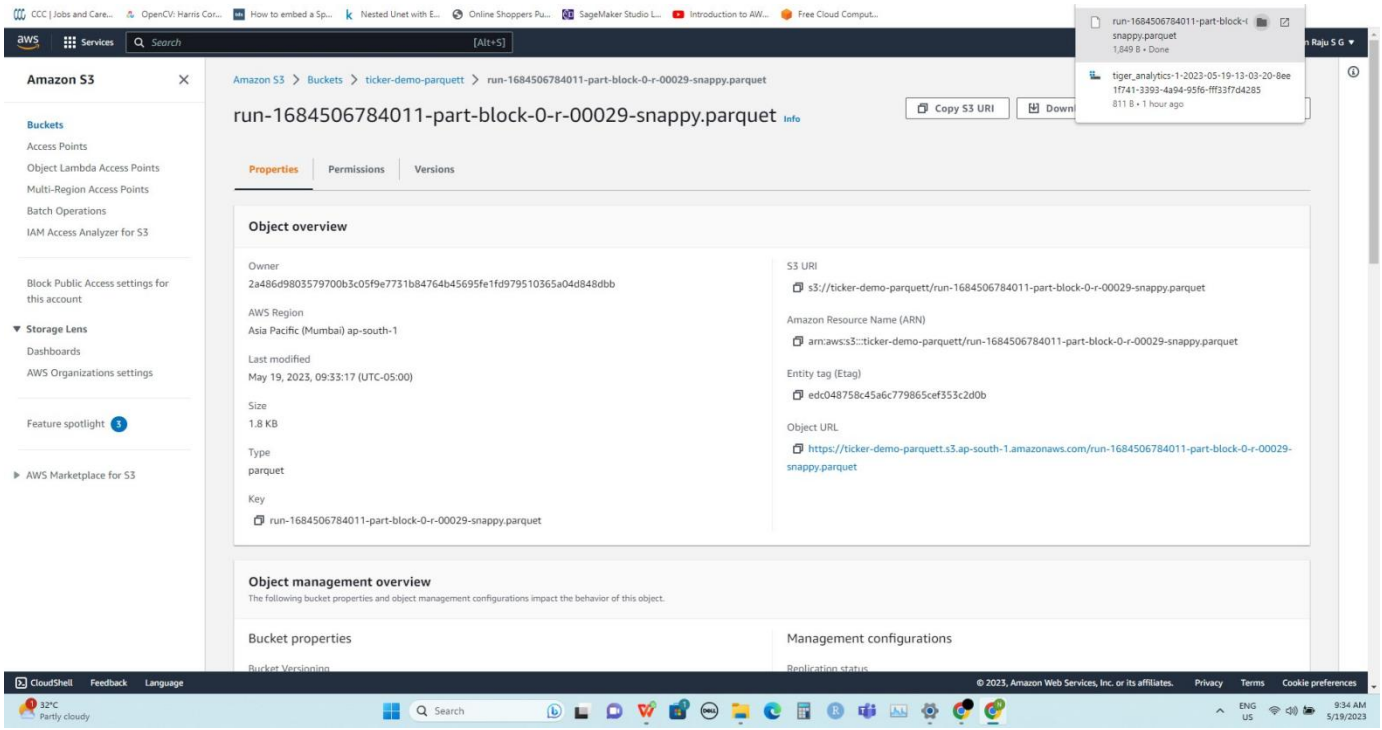




V)View results in S3 buckets



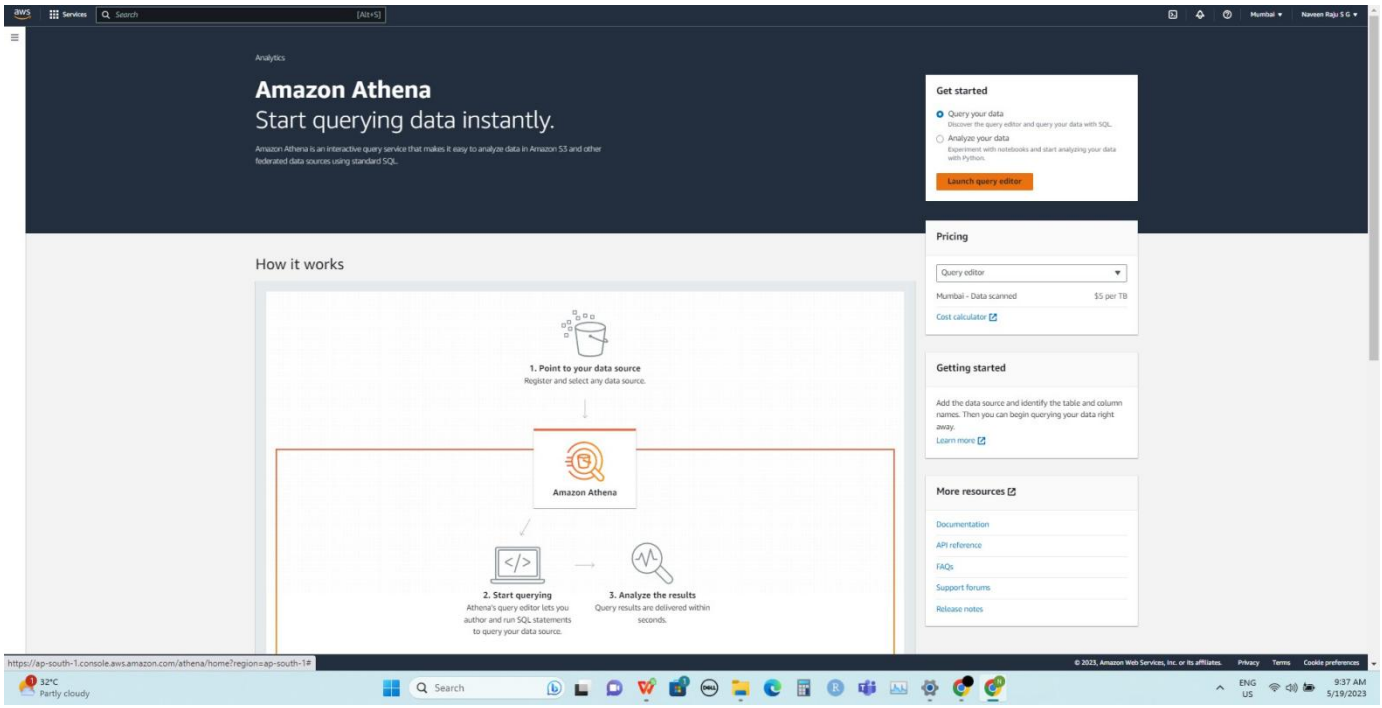
Download the files to view



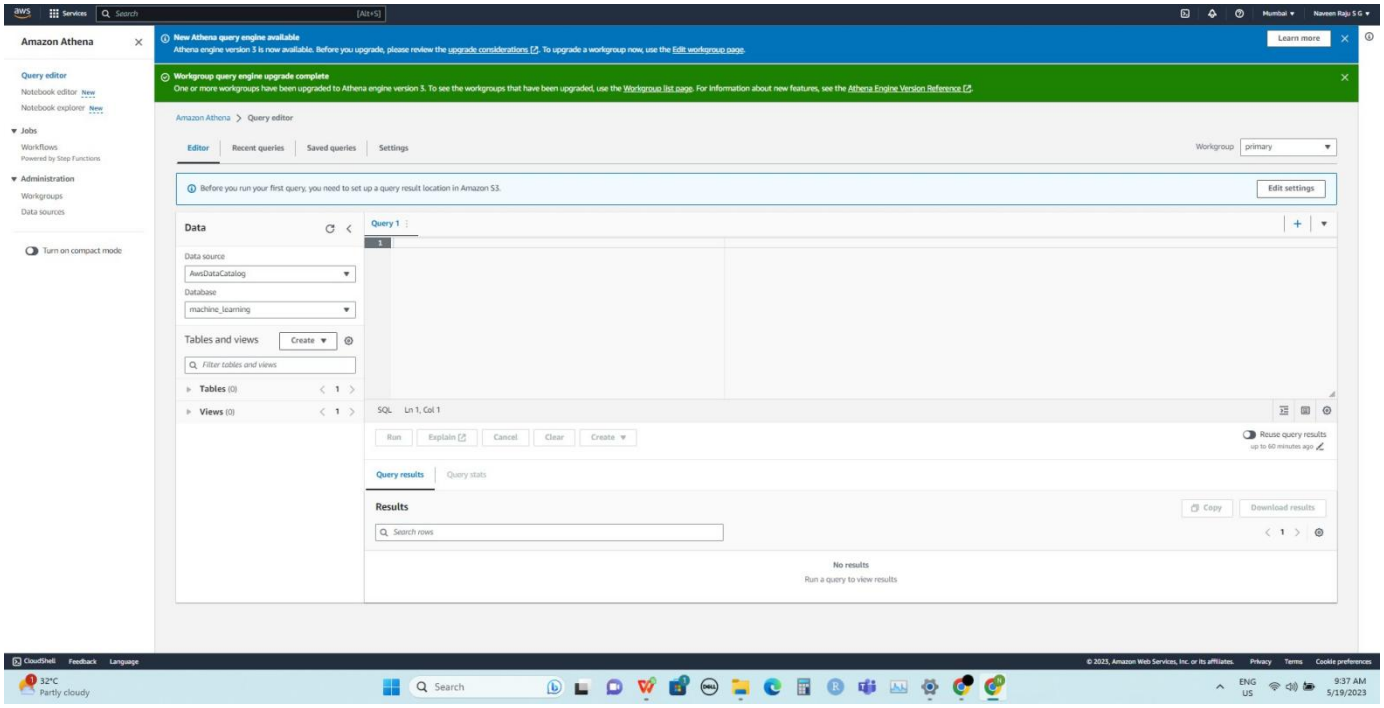
8) Configure, create and run AWS Athena interactive query service

I) Launch query editor and edit settings

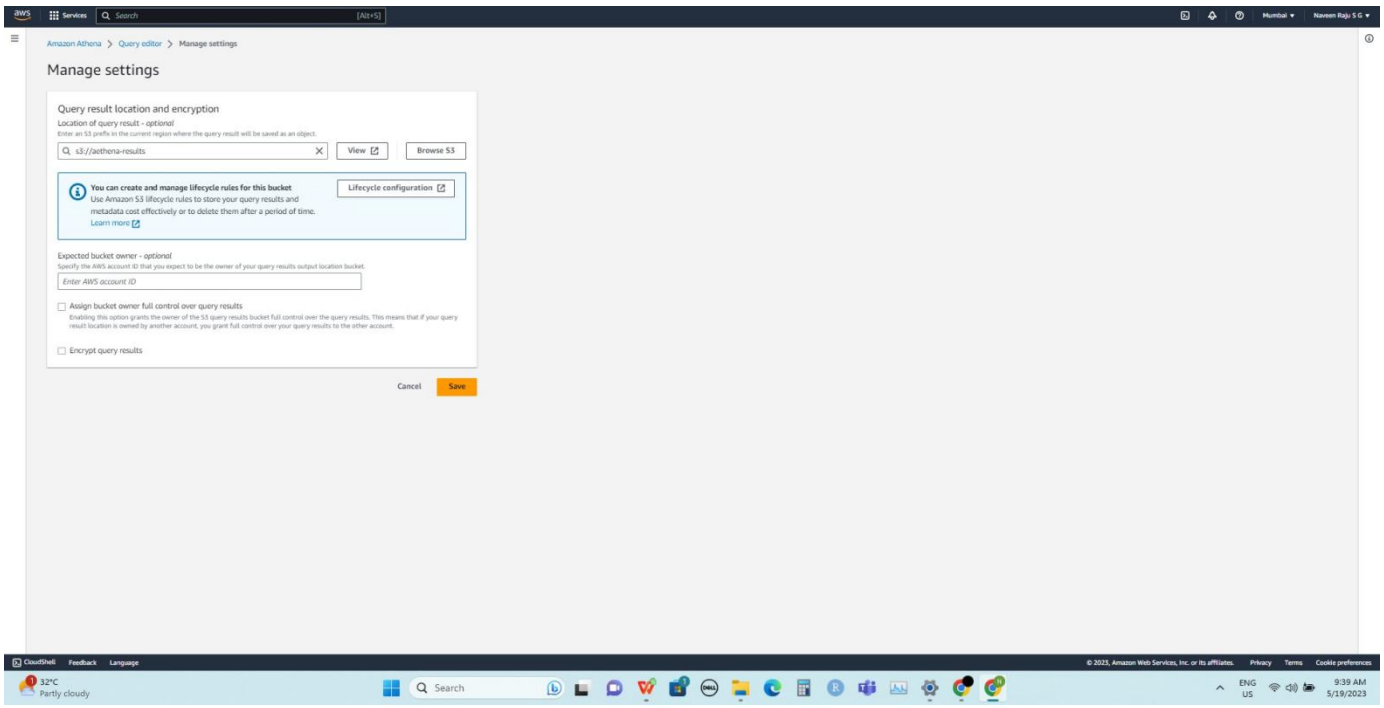
Let us query the data created by AWS Glue ETL jobs in previous step



Before we run our first query we need to first set up a query result location in S3.

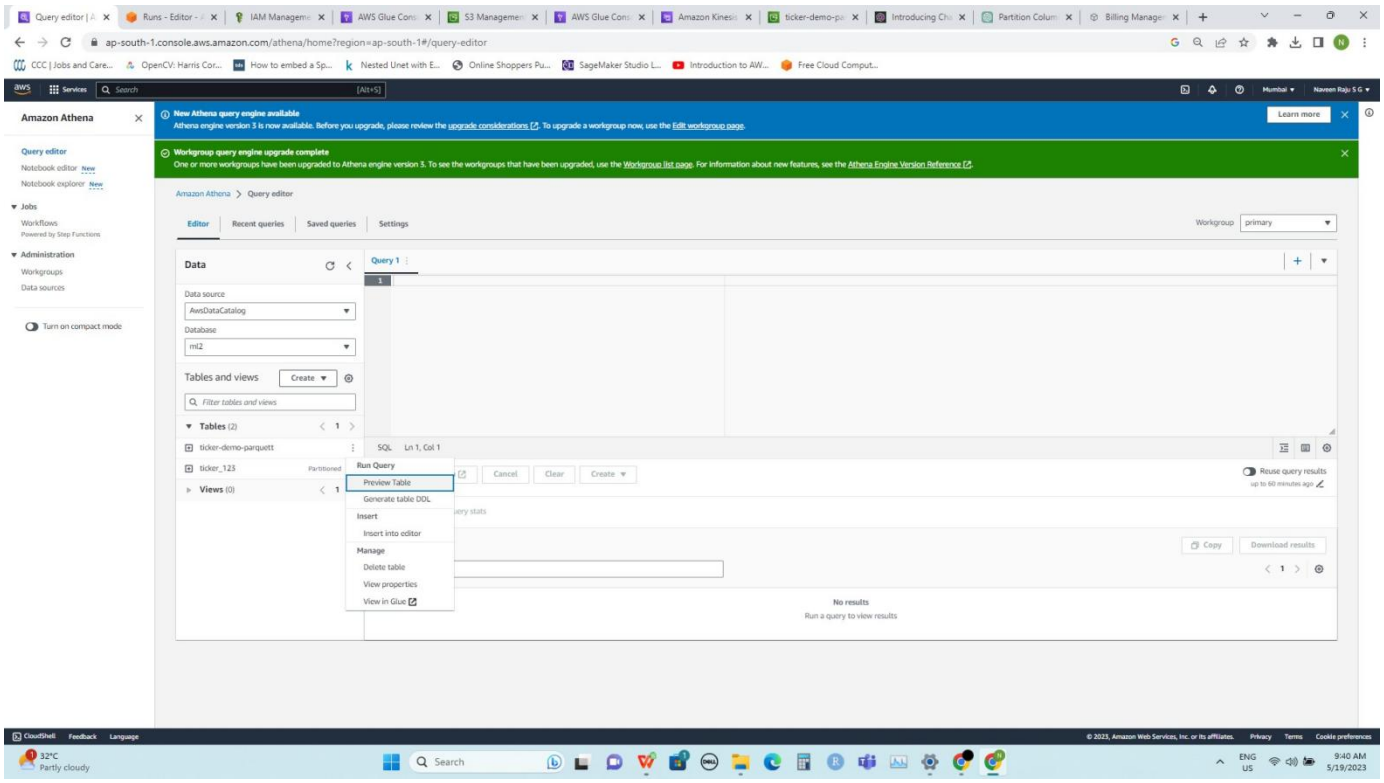


Create a S3 bucket named “aethena_results”



II) Config editor

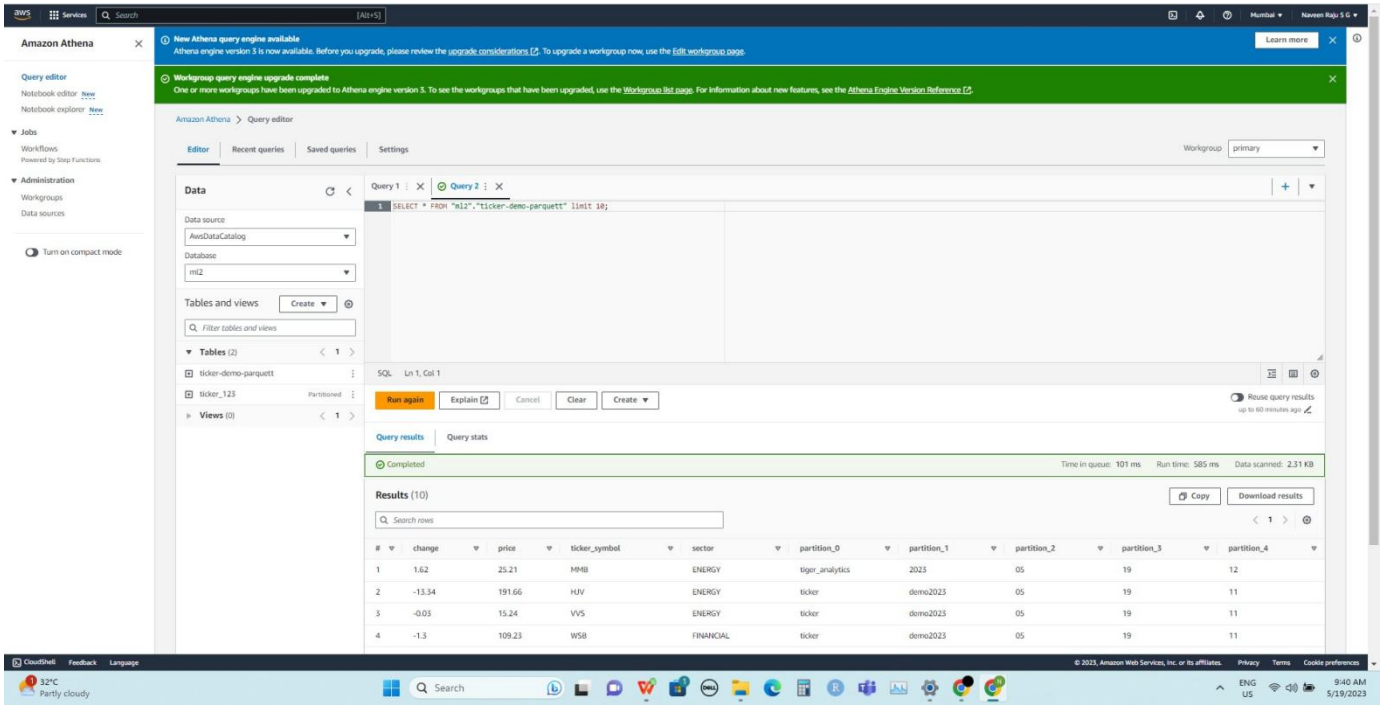
Data source : AWSDataCatalog
Database : machine_learning



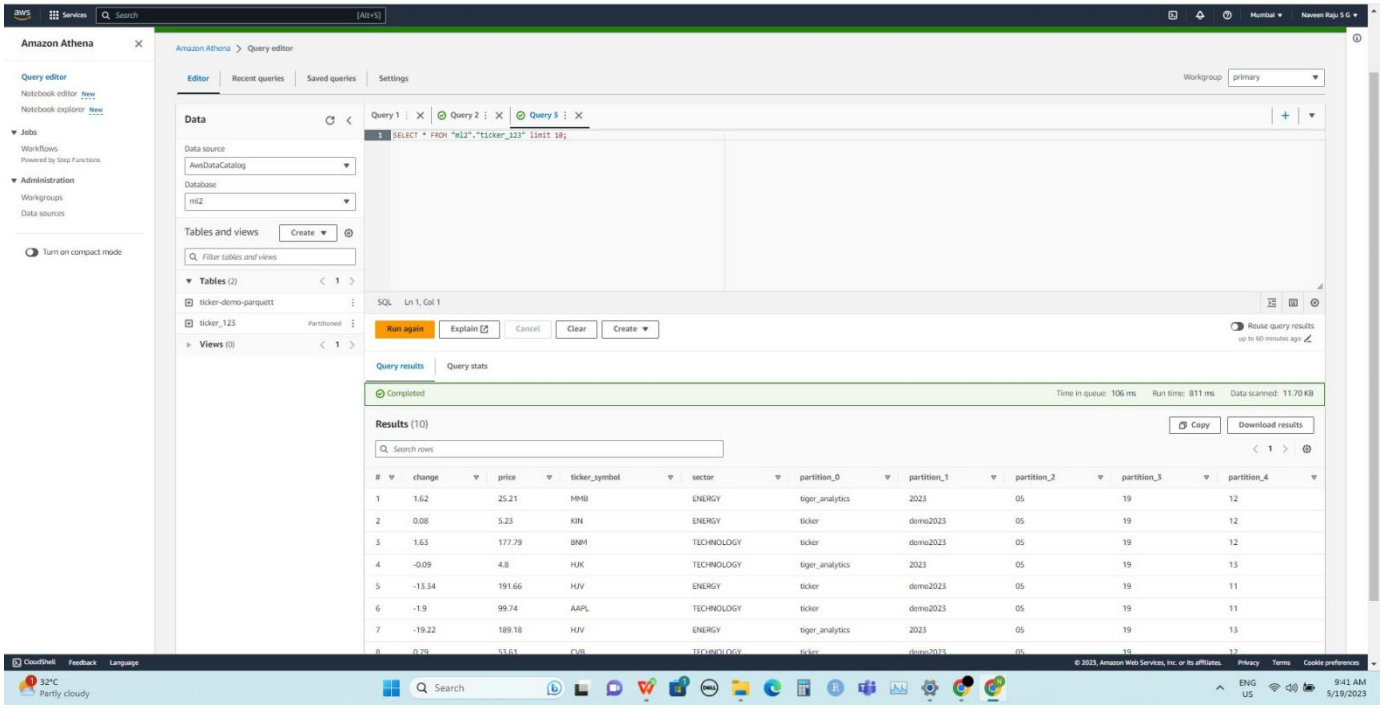
III) Write queries

Run the query by writing it in query editor:

SELECT * FROM “ml2”.”ticker-demo-parquett” limit 10; (Show first 10 records)

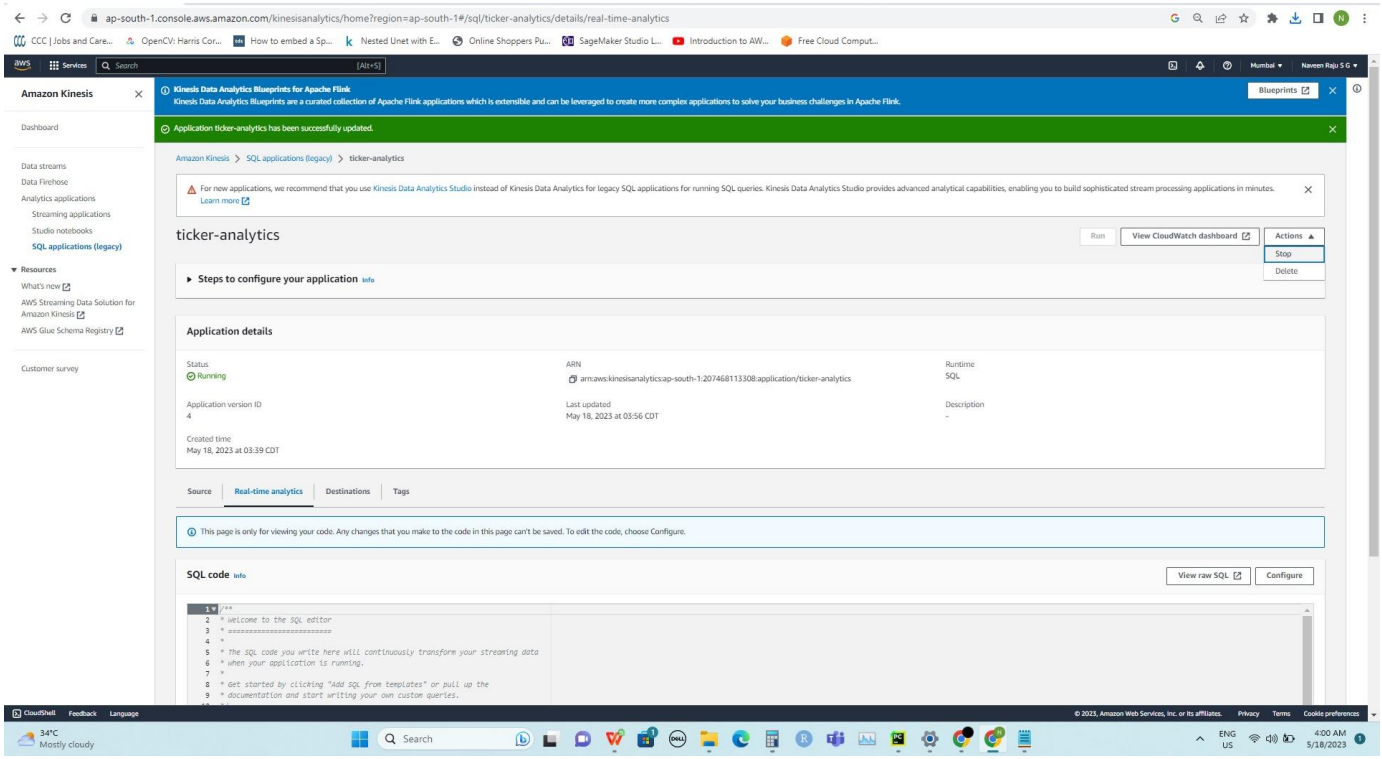


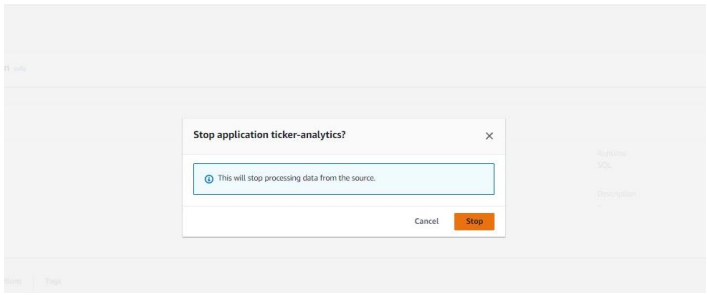
SELECT * FROM “ml2”.”ticker_123” limit 10; (Show first 10 records)



9) Deleting resources:

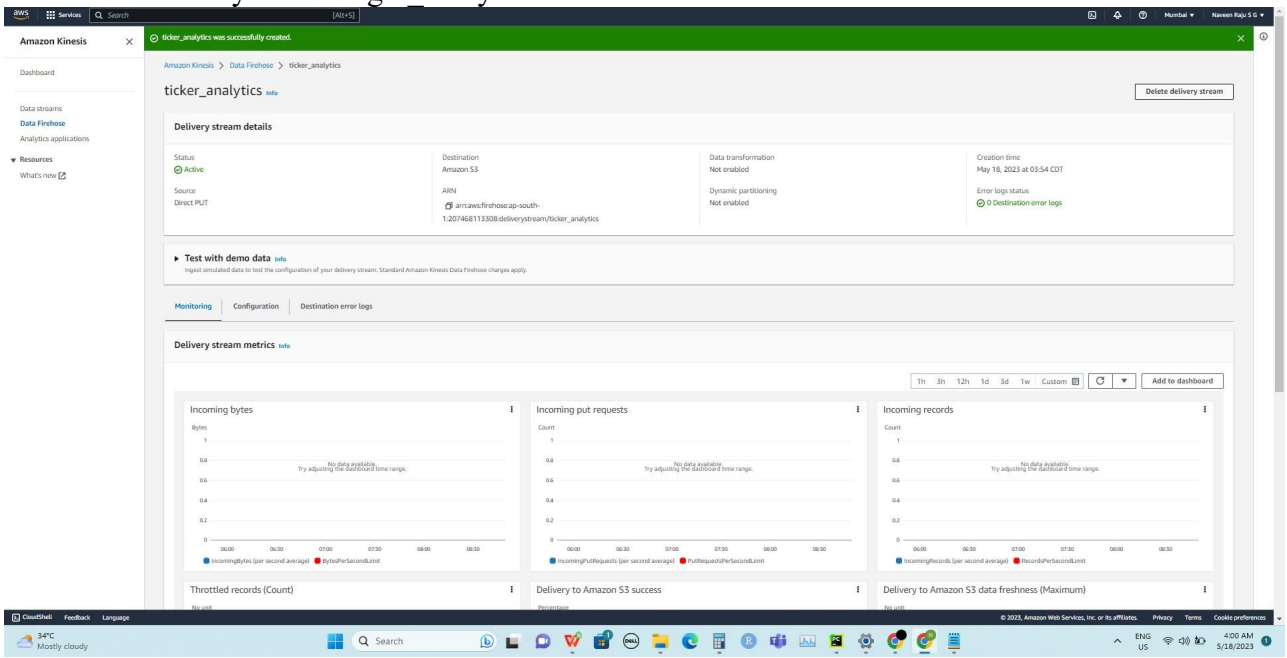
A) Stop the application created

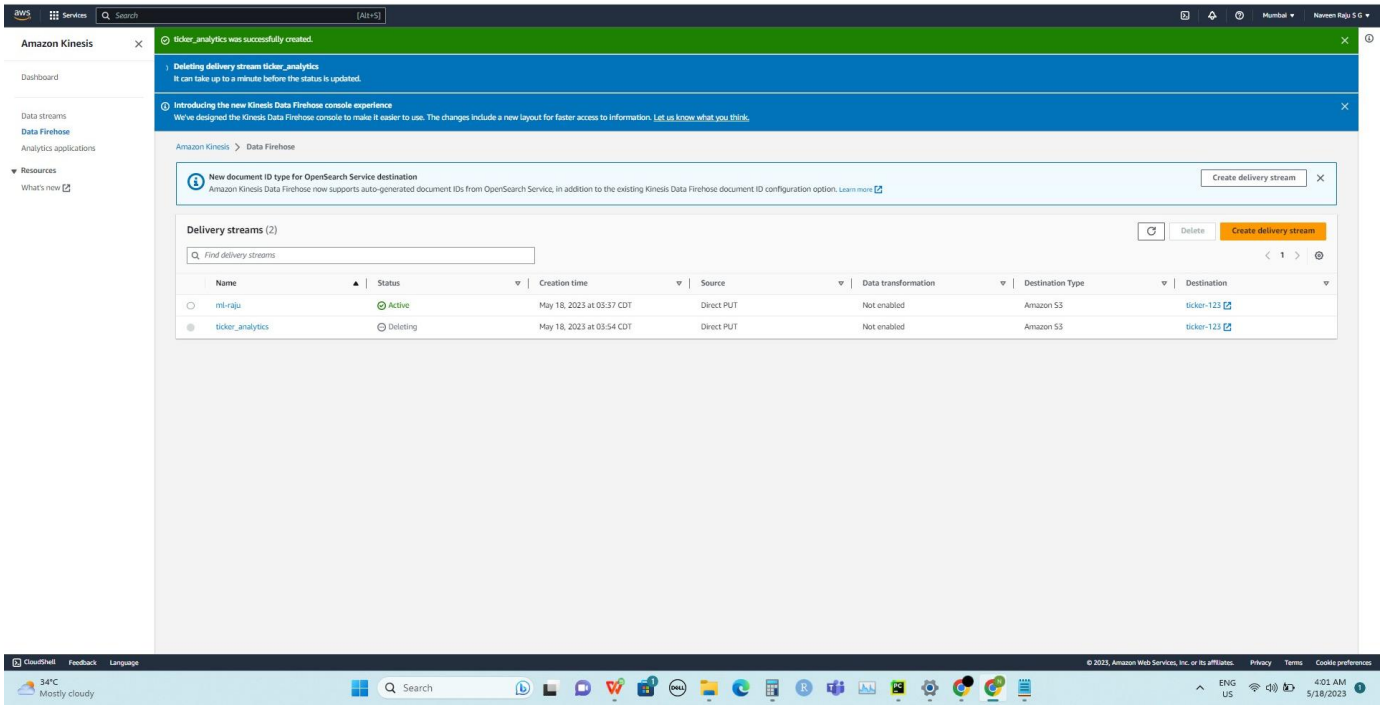




B) Delete delivery stream (stream that delivers results to S3 buckets)

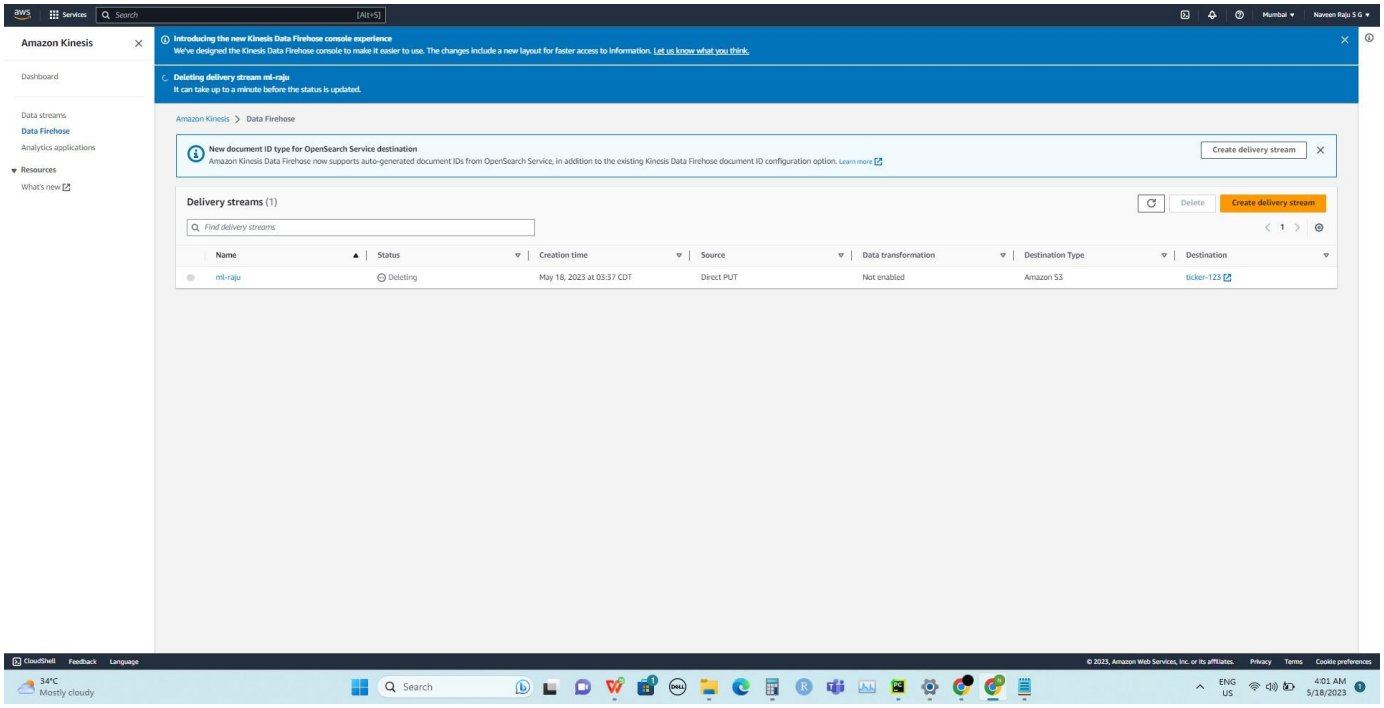
Delete the delivery stream tiger_analytics





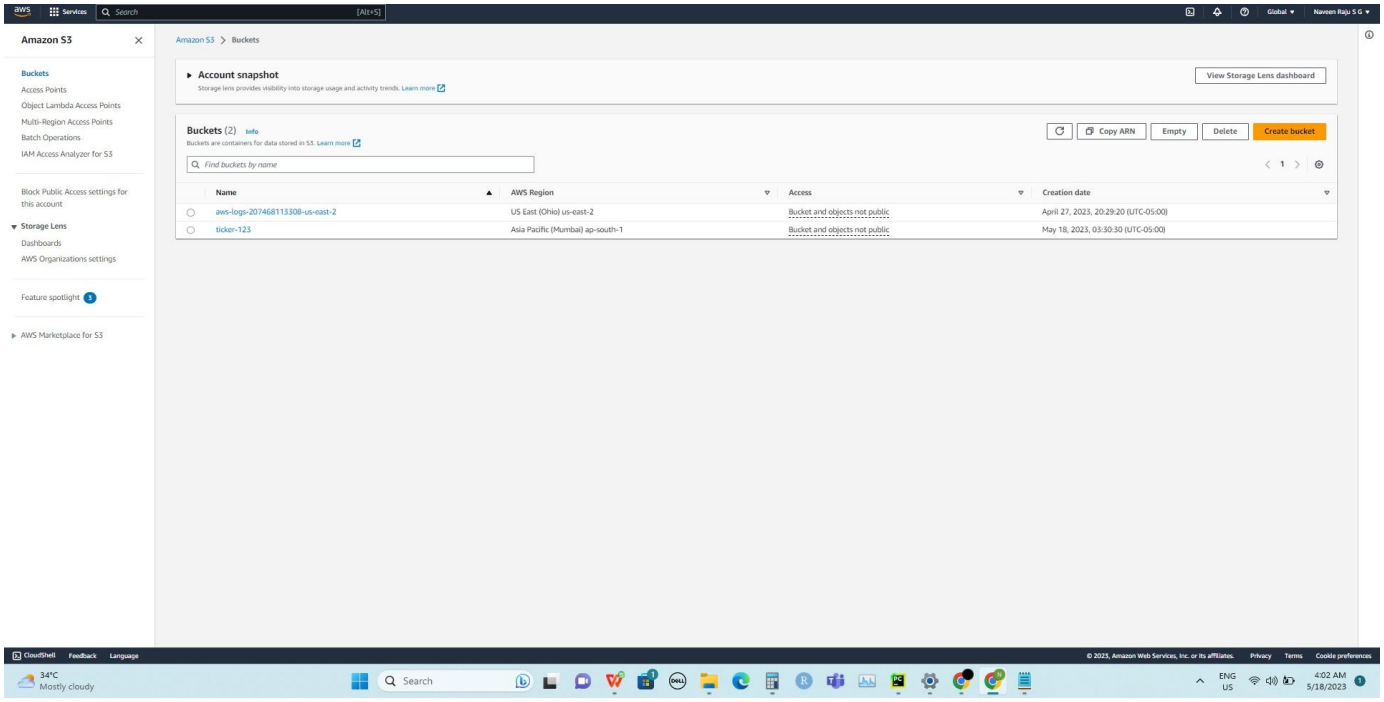
C) Delete delivery stream (ingest data from producers that are configured to send data to Kinesis Data FireHose)

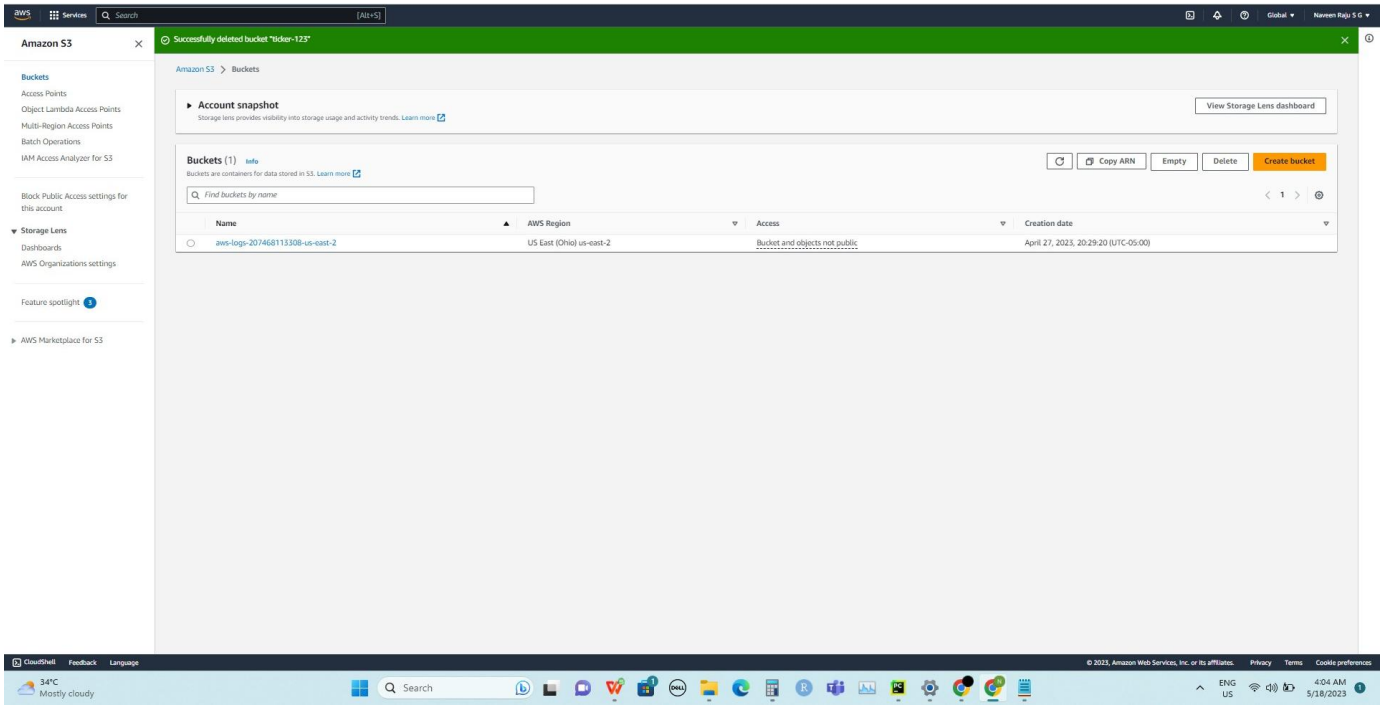
Delete the delivery stream ml_raju:



D) Delete S3 buckets

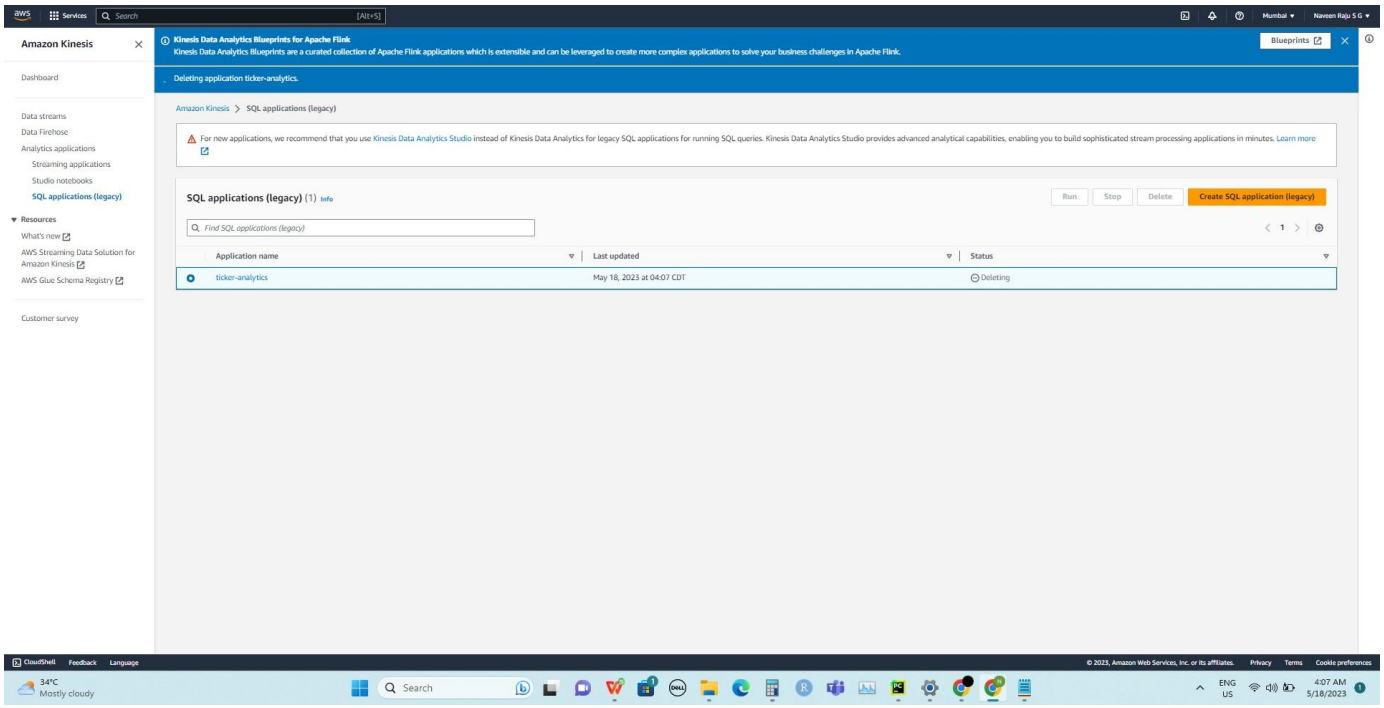
Delete all S3 bucket ticker-123

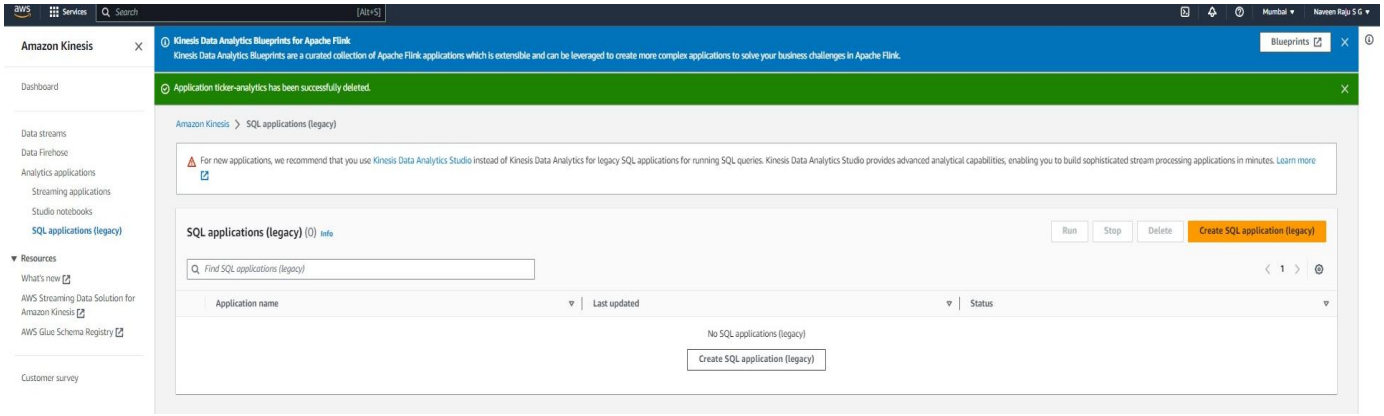




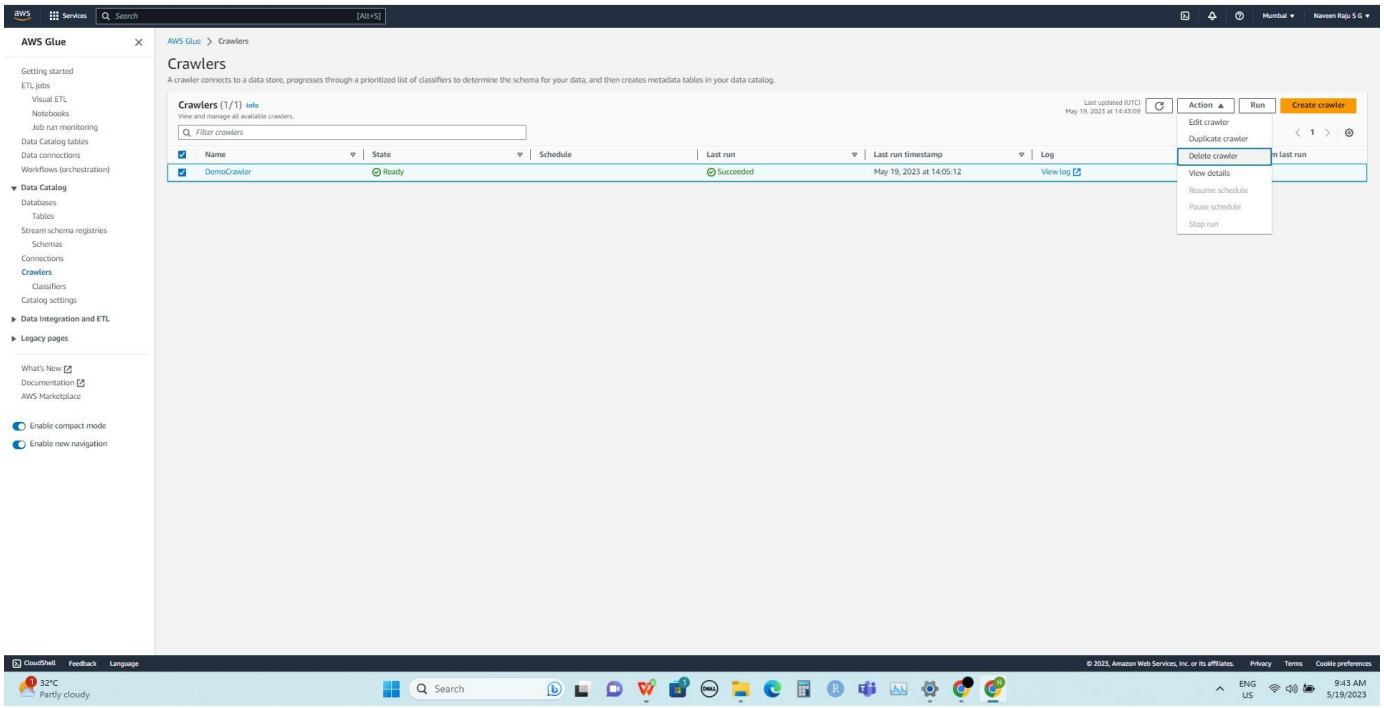
E) Delete SQL application (Analytics applications)

Delete tiger-analytics application:

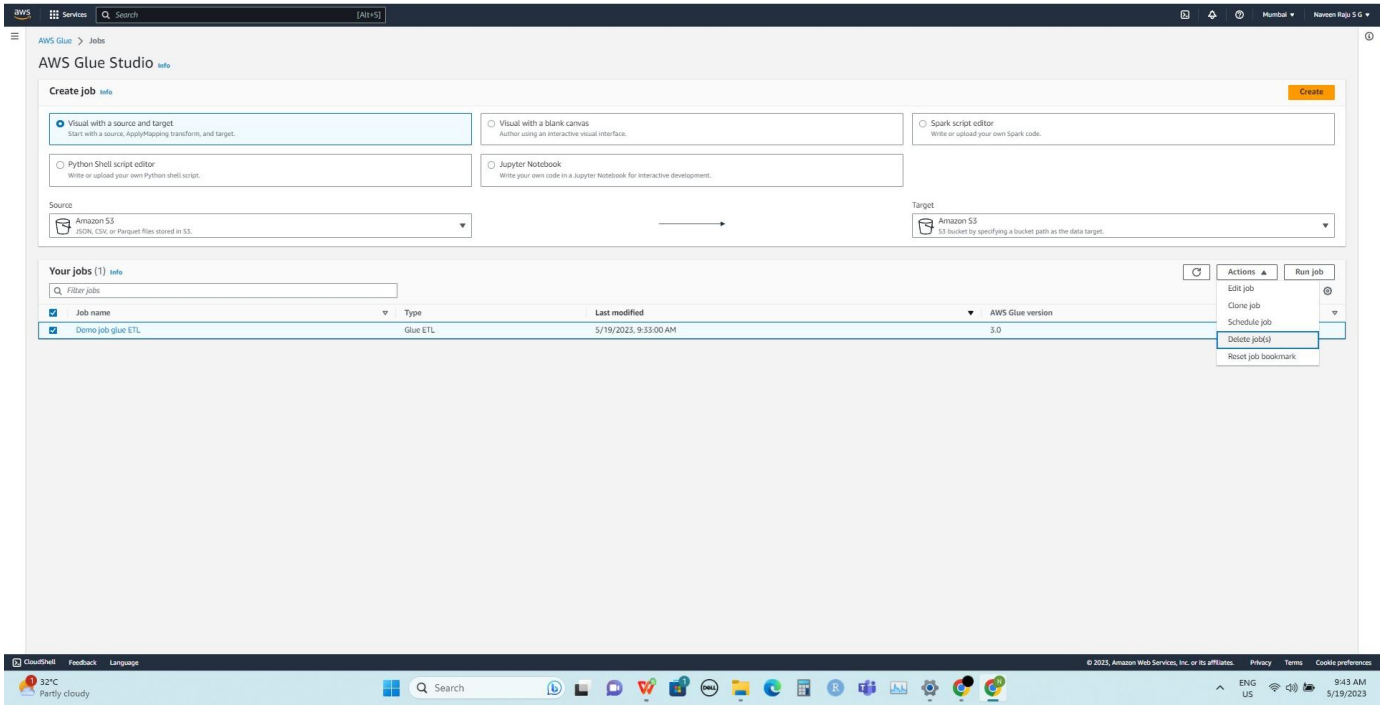




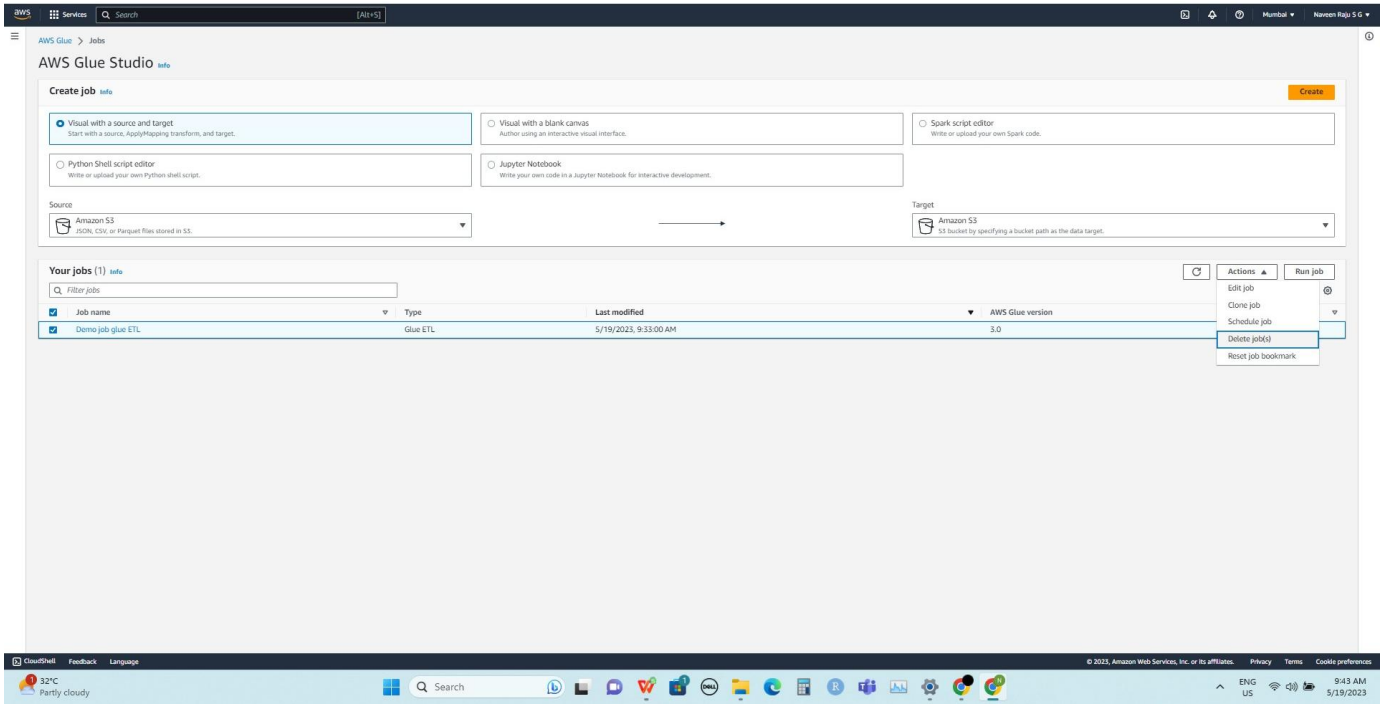
F) Delete Crawler



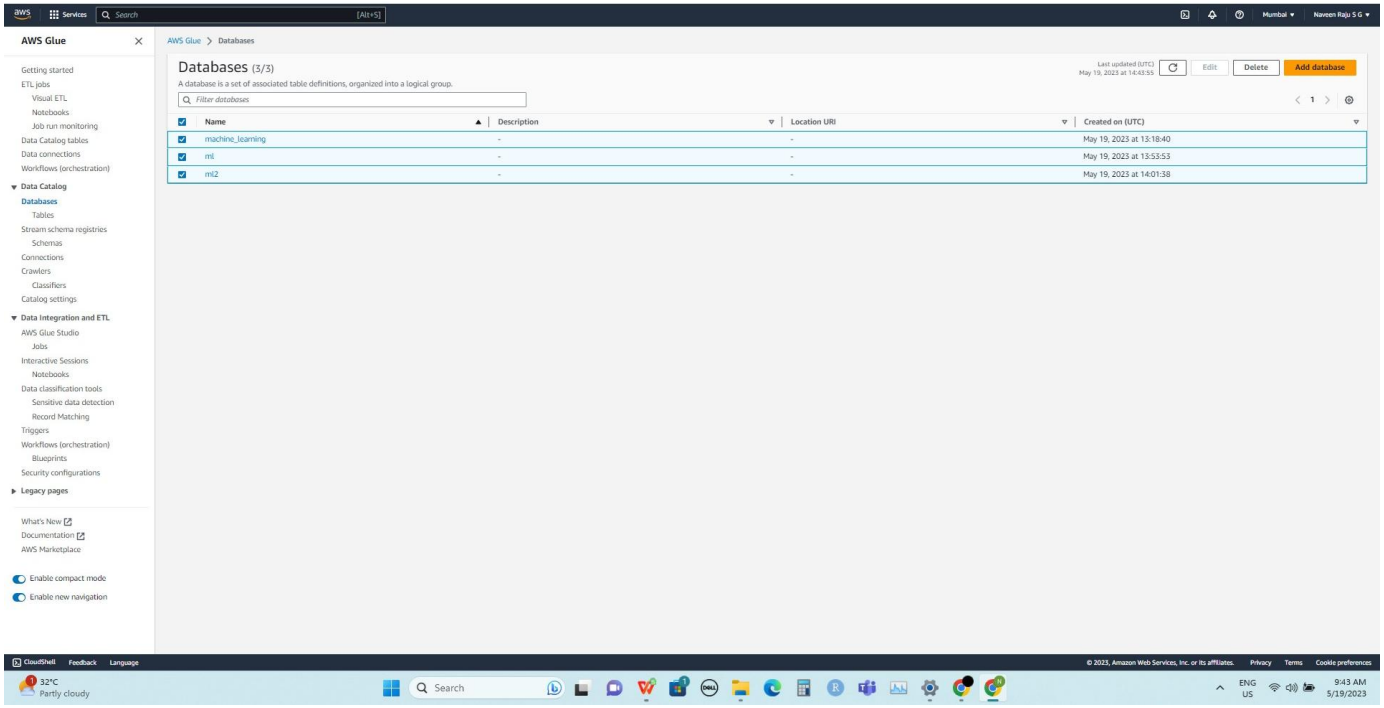
G) Delete AWS Glue Studio Job



H) Delete AWS Glue Databases



I)Delete AWS Glue Databases



J) Delete all S3 buckets that were created