Real-Time Stock Market Data Extraction with Apache Kafka:
An End-to-End Data Engineering Project

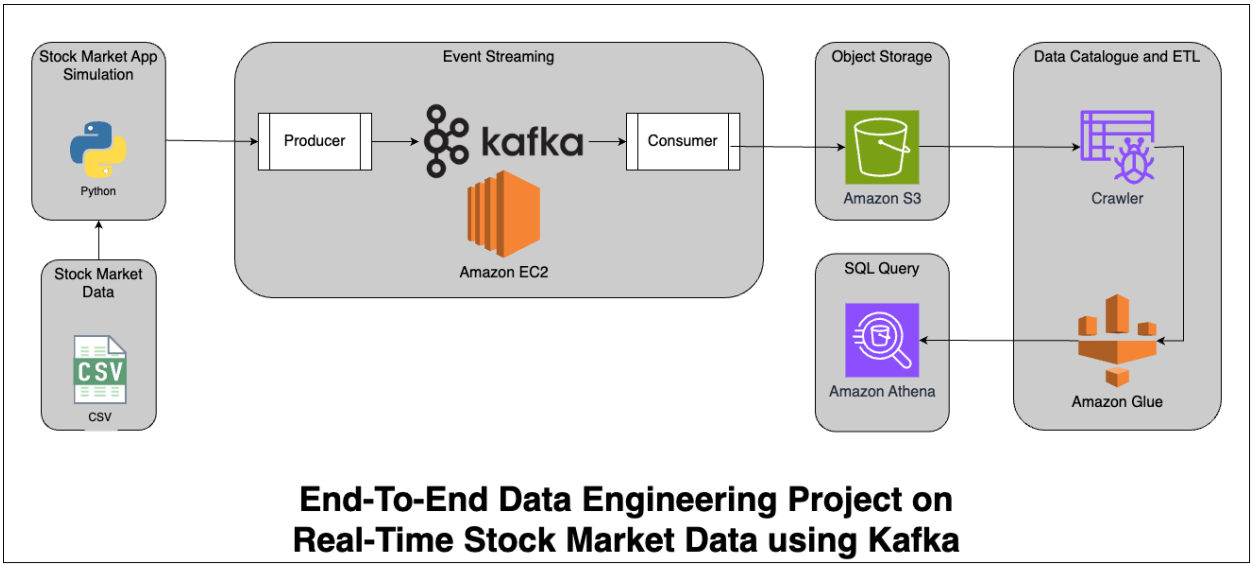
By Randy A. Velasco

I. Introduction:

In the ever-evolving landscape of financial markets, real-time data analysis plays a pivotal role in decision-making for investors and traders. The Real-Time Stock Market Data Analysis project is a comprehensive demonstration of end-to-end data engineering capabilities, showcasing the integration of cutting-edge technologies to process and analyze stock market data in real-time.

II. Project Overview:

This project leverages the power of Apache Kafka as a distributed streaming platform to ingest and process real-time stock market data. The seamless integration of Python for data manipulation and AWS services such as Glue, Athena, and SQL for data transformation and analysis enhances the project's scalability, efficiency, and accessibility.



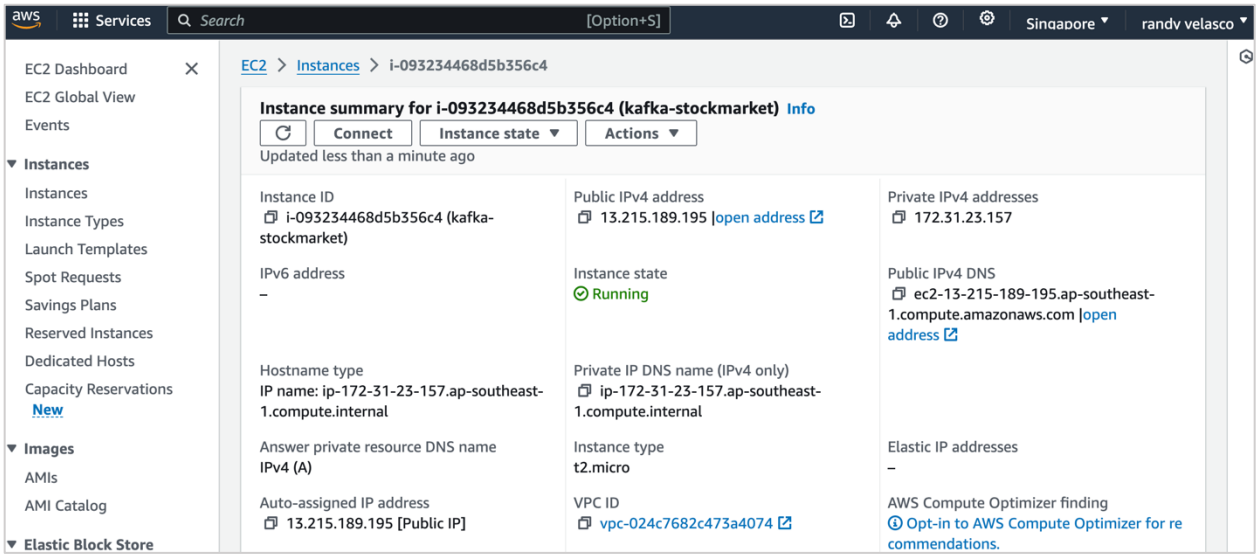
The workflow commences with the generation of a stock market data stream to simulate API behavior. This process utilizes a CSV file containing stock market data. Subsequently, a Python script is executed every second to randomly capture a single data row from the CSV. The captured data is then transmitted to Apache Kafka through a broker and producer, ultimately being published to a specific topic.

The data is consumed by a consumer and subsequently stored in JSON format on Amazon S3. Amazon Glue undertakes data cataloging tasks using a crawler, organizing and cataloging the data before saving it in a database. Amazon Athena is then employed to execute SQL queries, ensuring the accuracy of the entire workflow process.

III. Actual Steps Taken:

1. Configured Apache Kafka in Amazon EC2:

   a) Launched Amazon EC2 Instance:



   b) Connect to Amazon EC2 instance via SSH, download and install Apache Kafka and Java Runtime Environment

   Apache Kafka is a distributed event streaming platform used for building real-time data pipelines and streaming applications. It will be used to capture streaming data generated by Python simulating a stock market API source. Kafka captures streaming data through producers that publish records to the "KSMP" topic. These records are then consumed by the consumer and saved in Amazon S3.

   Java plays a central role in Apache Kafka, as Kafka is implemented in Java and is designed to run on the Java Virtual Machine (JVM).

   Code:

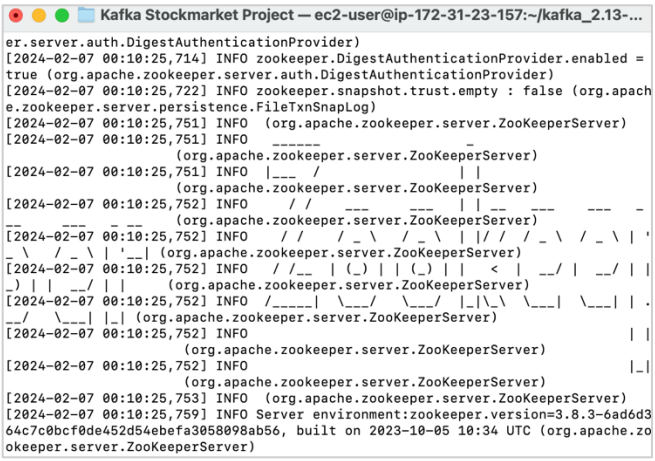
```
> ssh -i "kafka_stockmarket.pem" ec2-user@ec2-13-215-189-195.ap-southeast-1.compute.amazonaws.com
> wget https://downloads.apache.org/kafka/3.6.1/kafka_2.13-3.6.1.tgz
> tar -xvf kafka_2.13-3.6.1.tgz
> sudo yum install java-1.8.0-openjdk
```

c) Run Apache Zookeeper via SSH

The primary function of ZooKeeper in Kafka is to manage and coordinate the distributed nature of Kafka brokers and to maintain metadata and configuration information.

Code:

> bin/zookeeper-server-start.sh config/zookeeper.properties

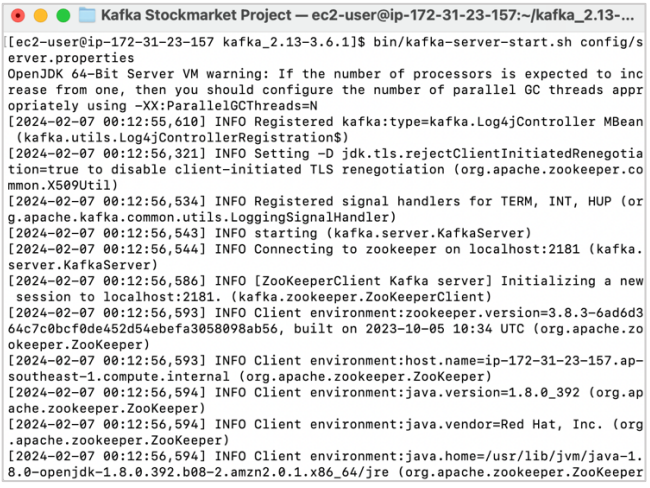


d) Run Apache Kafka Server

Apache Kafka server functions as a distributed event streaming platform. It manages the ingestion, storage, and real-time streaming of data. Kafka enables producers to publish records to topics, and consumers subscribe to these topics, allowing scalable, fault-tolerant data streaming across multiple nodes. The server maintains a distributed commit log, ensuring durability and sequential storage.

Code:

> bin/kafka-server-start.sh config/server.properties

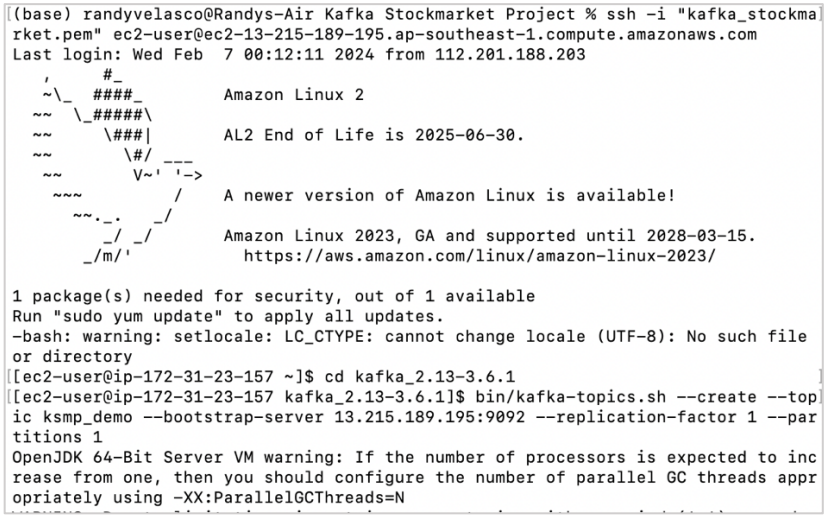


e) Create Kafka topic and run Kafka Producer

In Kafka, a topic is a category or feed name to which records (messages) are published by producers and from which records are consumed by consumers. Topics serve as a way to organize and categorize the messages within the Kafka messaging system. Multiple producers can publish records to a topic, and multiple consumers can subscribe to and consume records from the same topic. Topics help in organizing and managing the flow of data within the Kafka cluster. For this project, the topic is named "KSMP".

Code:

> bin/kafka-topics.sh --create --topic ksmp --bootstrap-server 13.215.189.195:9092 --replication-factor 1 --partitions 1
> bin/kafka-console-producer.sh --topic ksmp --bootstrap-server 13.215.189.195:9092



f) Run Kafka Consumer

A Kafka consumer is responsible for subscribing to Kafka topics, retrieving and processing records/messages produced by Kafka producers. For this project, the consumer retrieves data from the "ksmp" topic and save it to Amazon S3.

Code:

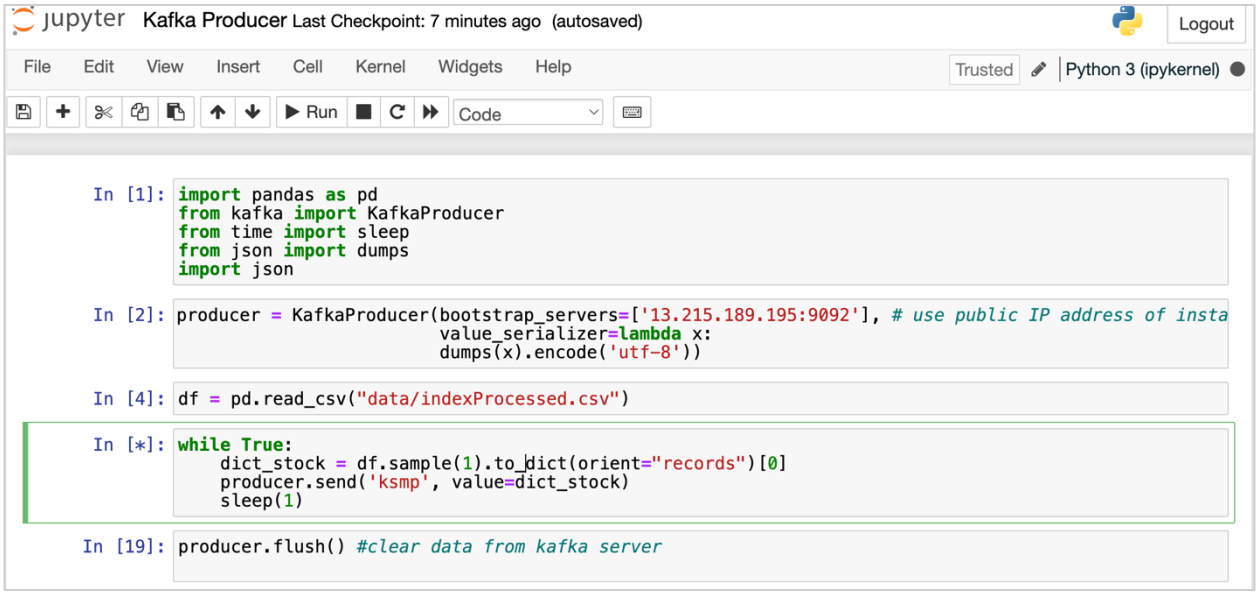
> bin/kafka-console-consumer.sh --topic ksmp --bootstrap-server 13.215.189.195:9092

```
(base) randyvelasco@Randys-Air Kafka Stockmarket Project % ssh -i "kafka_stockma
rket.pem" ec2-user@ec2-13-215-189-195.ap-southeast-1.compute.amazonaws.com
Last login: Wed Feb  7 00:14:08 2024 from 112.201.188.203
   ,     #_
   ~\_  ####_        Amazon Linux 2
  ~~  \_#####\
  ~~     \###|       AL2 End of Life is 2025-06-30.
  ~~       \#/ ___
   ~~       V~' '->
    ~~~         /    A newer version of Amazon Linux is available!
      ~~._.   _/
         _/ _/       Amazon Linux 2023, GA and supported until 2028-03-15.
       _/m/'           https://aws.amazon.com/linux/amazon-linux-2023/

1 package(s) needed for security, out of 1 available
Run "sudo yum update" to apply all updates.
-bash: warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file
or directory
[ec2-user@ip-172-31-23-157 ~]$ cd kafka_2.13-3.6.1
[ec2-user@ip-172-31-23-157 kafka_2.13-3.6.1]$ bin/kafka-console-consumer.sh --to
pic ksmp --bootstrap-server 13.215.189.195:9092
OpenJDK 64-Bit Server VM warning: If the number of processors is expected to inc
rease from one, then you should configure the number of parallel GC threads appr
opriately using -XX:ParallelGCThreads=N
dd
```

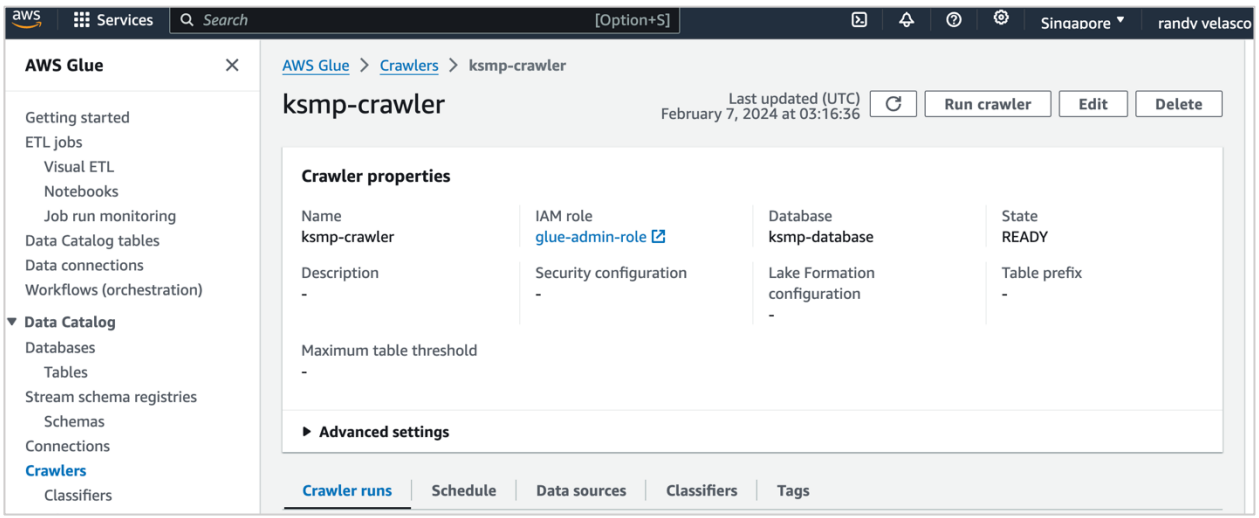
2. Execute Python script to simulate streaming stock market data from a CSV file.

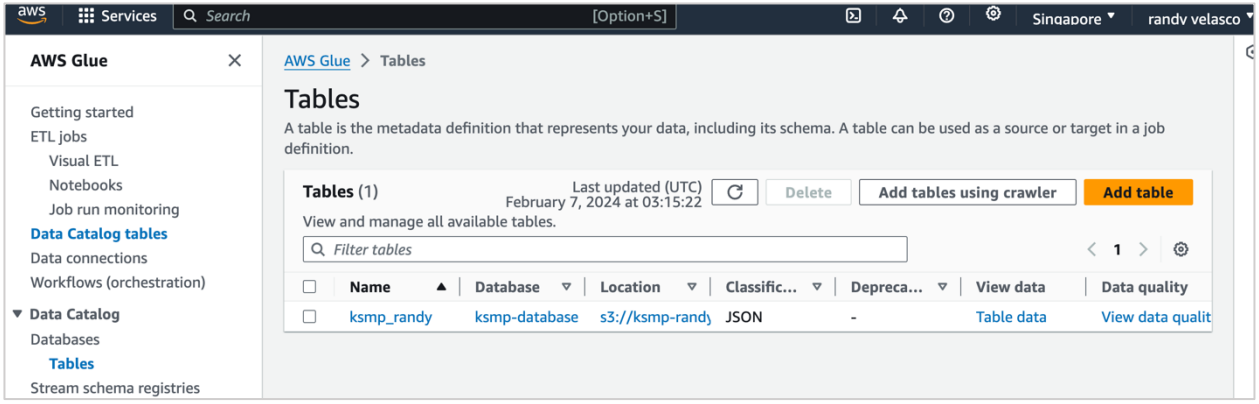
   Below is a Python script to simulate what a stock market API does. A dataframe is created from a CSV file that contain stock market data. Every second, a sample row is sent to Apache Kafka server via the Kafka producer.



3. Perform data transformation using Apache Glue

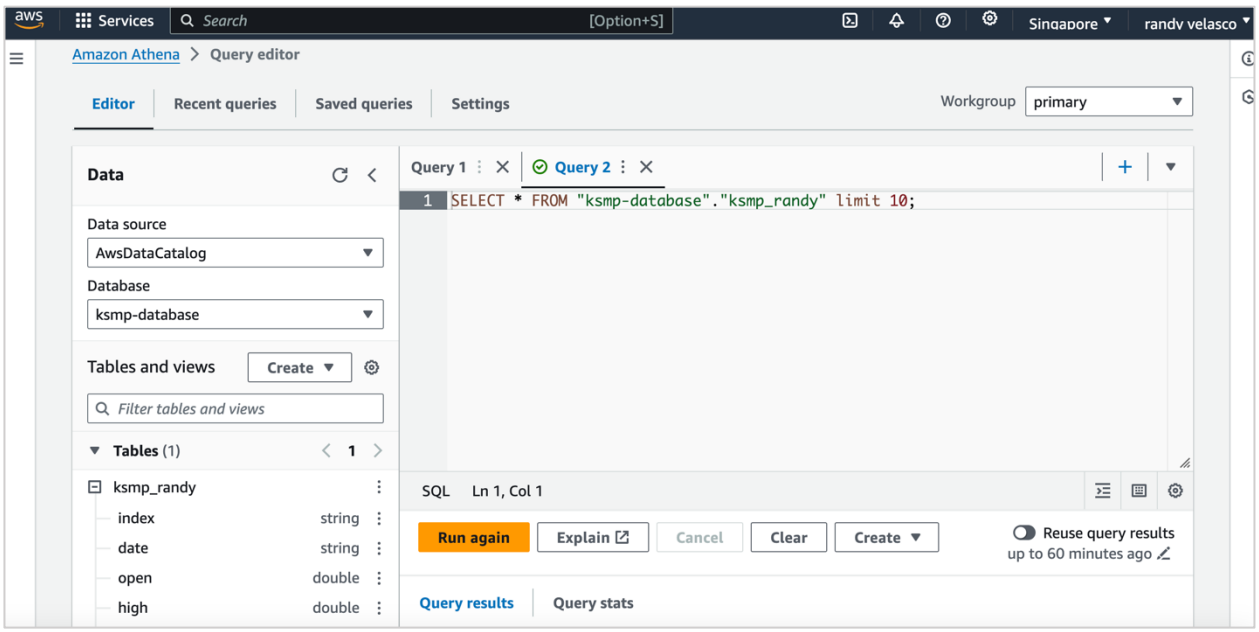
   Transform raw data into a structured and queryable format using Apache Glue. Use a crawler to automatically scan and identify the schema and structure of data stored in Amazon S3. Amazon Glue automatically creates metadata tables in the Glue Data Catalog based on the schema it infers or detects from the source.

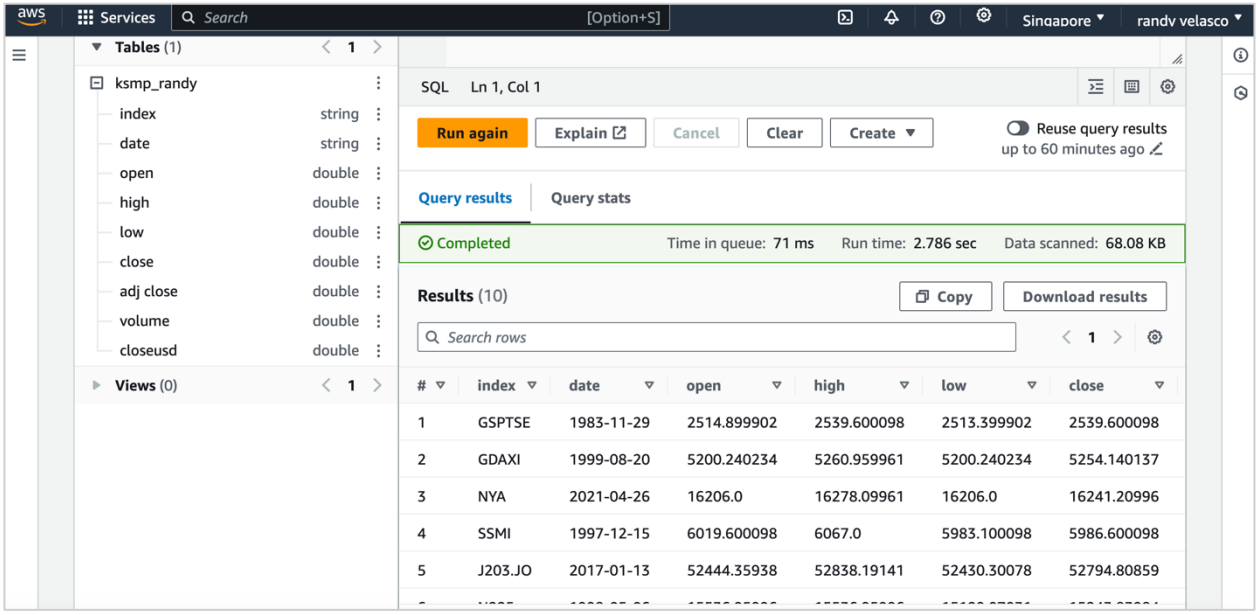




4. Perform SQL query using Amazon Athena

Use Amazon Athena to query tables and data cataloged by AWS Glue in Amazon S3 using standard SQL. This is to ensure the accuracy of the entire workflow process.





IV. Skills Showcased

- Apache Kafka: Showcase proficiency in setting up and configuring Kafka for real-time data streaming, ensuring reliable and scalable data ingestion.
- Python: Demonstrate skills in data manipulation and preprocessing using Python, ensuring the seamless integration of real-time data into the Kafka pipeline.
- Amazon Web Services (AWS):
    - AWS Glue: Illustrate expertise in automated ETL (Extract, Transform, Load) processes, facilitating the transformation of raw data into a structured and quarriable format.

    - Amazon Athena: Showcase the ability to use serverless SQL queries for interactive analysis of data stored in Amazon S3, providing a cost-effective and efficient solution for ad-hoc querying.
- SQL: Demonstrate proficiency in writing SQL queries to perform complex data analysis tasks, showcasing the ability to derive valuable insights from real-time stock market data.

V. Conclusion:

The Real-Time Stock Market Data Extraction project provides a hands-on demonstration of end-to-end data engineering skills, combining the power of Apache Kafka, Python, and AWS services to enable real-time data analysis in the dynamic and fast-paced realm of stock markets. This project not only highlights technical proficiency but also emphasizes the ability to derive meaningful insights from real-time data for informed decision-making in financial markets.

---------- (E N D) ----------