Kafka Documentation

Kafka

Apache Kafka is an open-source distributed event streaming platform . Apache kafka is used for building real-time data pipelines and streaming applications. Kafka is designed to handle large volumes of data in a scalable and fault-tolerant manner, making it ideal for use cases such as real-time analytics, data ingestion, and event-driven architectures.

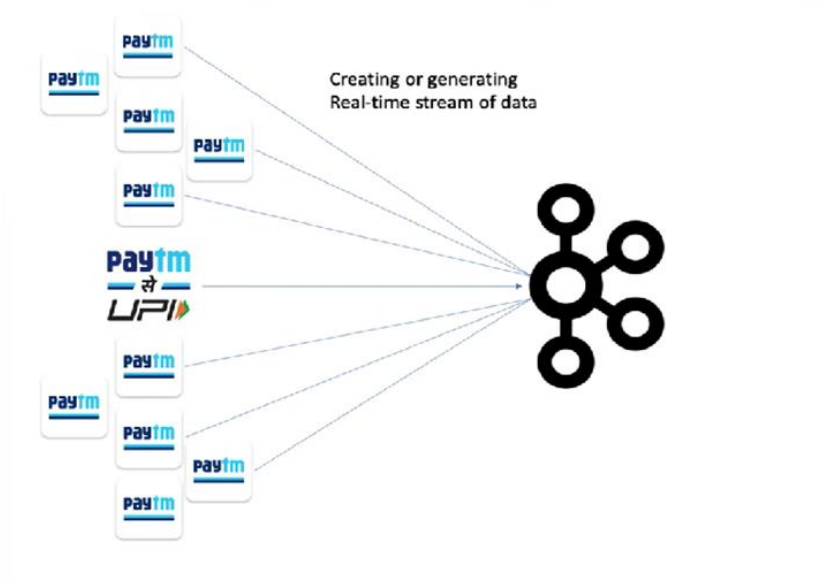
Event Streaming Platform includes two parts :-

- Creating Real-time Stream
- Processing Real-time Stream

Creating Real-time Stream

For example as we use paytm and do the payment process that event will go to the kafka server but I am not only the one to do the payment . Kafka server receives millions or billions of events in each minute or each second .

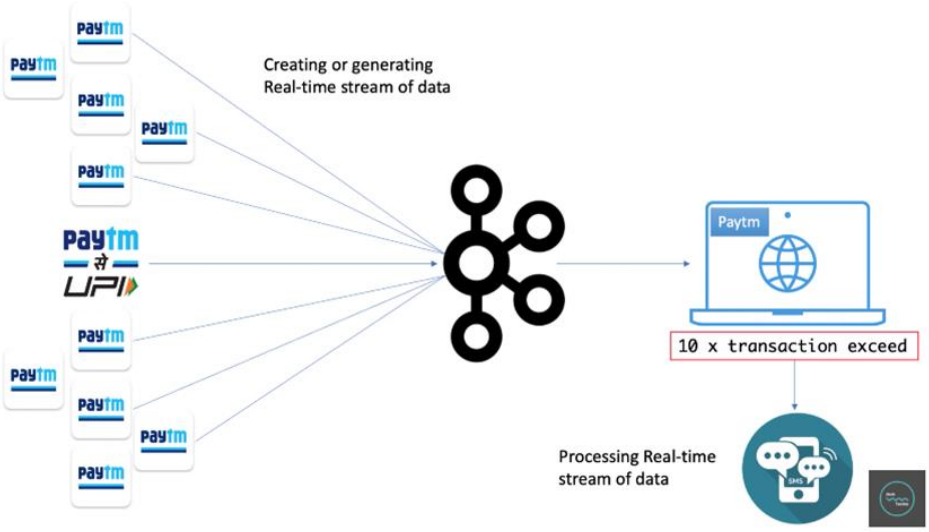
So sending the stream of continuous data from paytm to kafka server is called creating real time stream of data or generating real time stream of data.



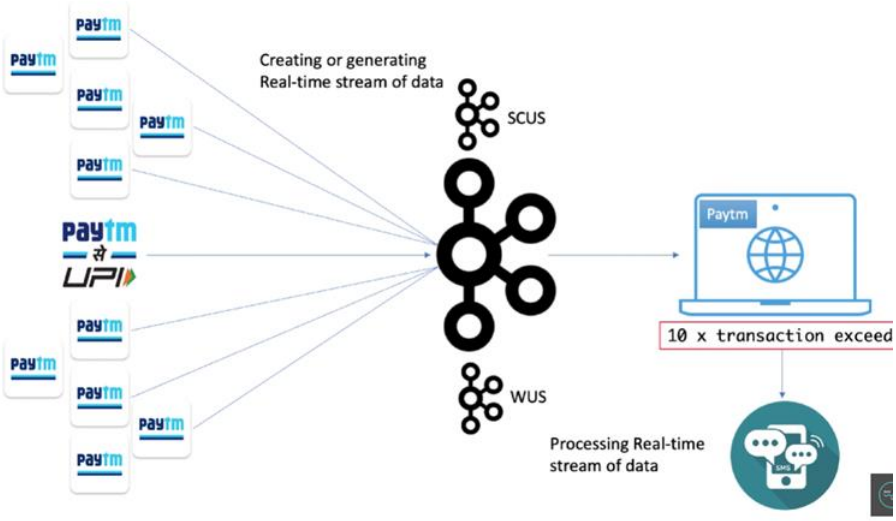
Processing Real-time Stream

Now suppose there is a limit of 10 transactions per day on paytm . A user can do only 10 transactions per day if it exceeds the limit then the client application wants to send the notification to the user . in such scenario my client application

needs to continuously face the data and needs to do the validatation to check the count of specific user in each and every second . so this continuous listening to the kafka messages and processes them is called Processing real time stream of data



As we know kafka is an distributed event streaming platform We can also distribute our kafka server . Following example has the 3 kafka server running in different regions to perform distributed event streaming .



Event streaming is applied to a wide variety of use cases across a large number of industries and organizations. For example:

- As a messaging system. For example Kafka can be used to process payments and financial transactions in real-time, such as in stock exchanges, banks, and insurance companies.

- Activity tracking. For example Kafka can be used to track and monitor cars, trucks, fleets, and shipments in real-time, such as for taxi services, in logistics and the automotive industry.
- To gather metrics data. For example Kafka can be used to continuously capture and analyze sensor data from IoT devices or other equipment, such as in factories and wind parks.
- For stream processing. For example use Kafka to collect and react to customer interactions and orders, such as in retail, the hotel and travel industry, and mobile applications.
- To decouple a system. For example, use Kafka to connect, store, and make available data produced by different divisions of a company.
- To integrate with other big data technologies such as Hadoop.

Where does kafka come from

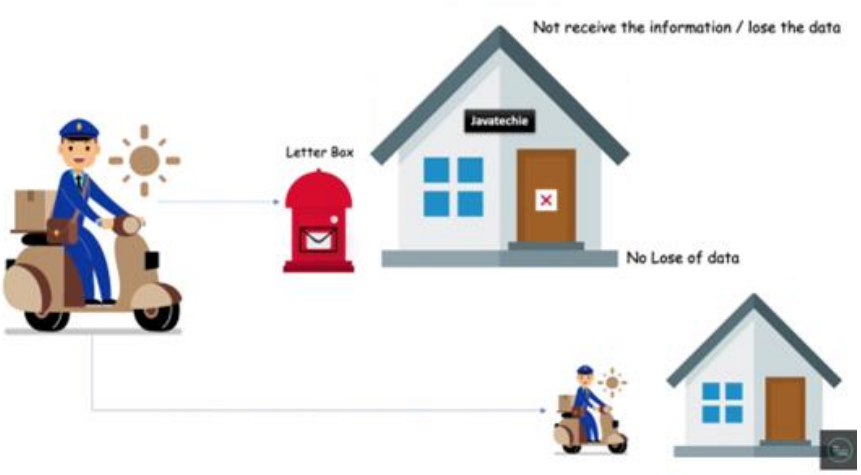
Kafka was originally developed at Linkedin , and was subsequently open sourced in early 2011 . Kafka server comes under the Apache software foundation .

Need Of Kafka

Lets understand this with an example .

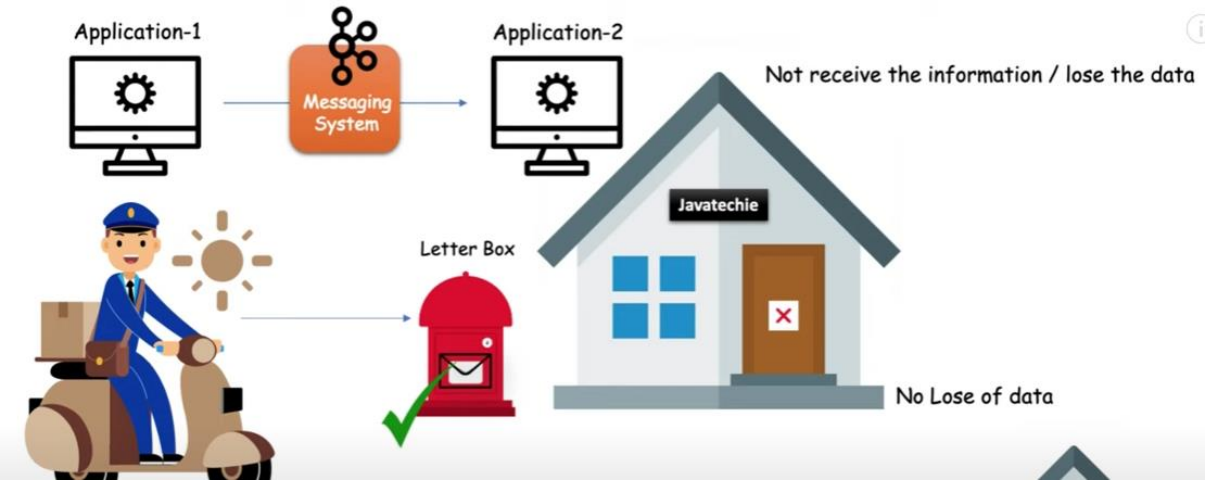
Lets assume that there is a parcel or a letter of my name with a postman . he comes to my house to handover to me but i was not available . seeing that the postman returned and tried the next day again . but i was not available for 2-3 days . now the postman might have forgot about the parcel or might have returned to the company . But the important thing is that i have lose of data or i haven't received the data . To overcome this I have set the letter box outside of my house . whenever i am not available at home the postman can put the letter into the letter box . this will basically help to avoid the lose of data .

Here the letter box acts as a middle man between the postman and me.



Lets connect this with an technical example .

Why we need kafka



- Suppose there are two application 1 and 2
- Application 1 wants to send the data to application 2
- but if the a2 is not available to receive the data so we will loss the data
- To overcome this communication failure we need to install the messaging system ie kafka
- So if a2 is not available it can collect the data from kafka whenever comes online

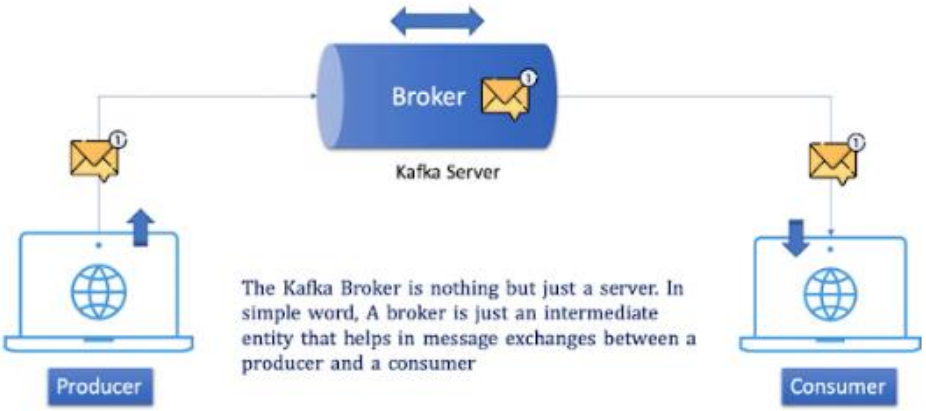
Kafka Architecture and Components

Components

- Producer
- Consumer
- Broker
- Cluster
- Topic
- Partitions
- Offset
- Consumer Groups
- Zookeeper

Producer

Producer is the source of data who will publish the message or events . The consumer acts as an receiver and it is responsible for receiving and consuming any messages . but they won't directly communicate with each other . To process the messages there must be an middle man between them i.e the kafka server .

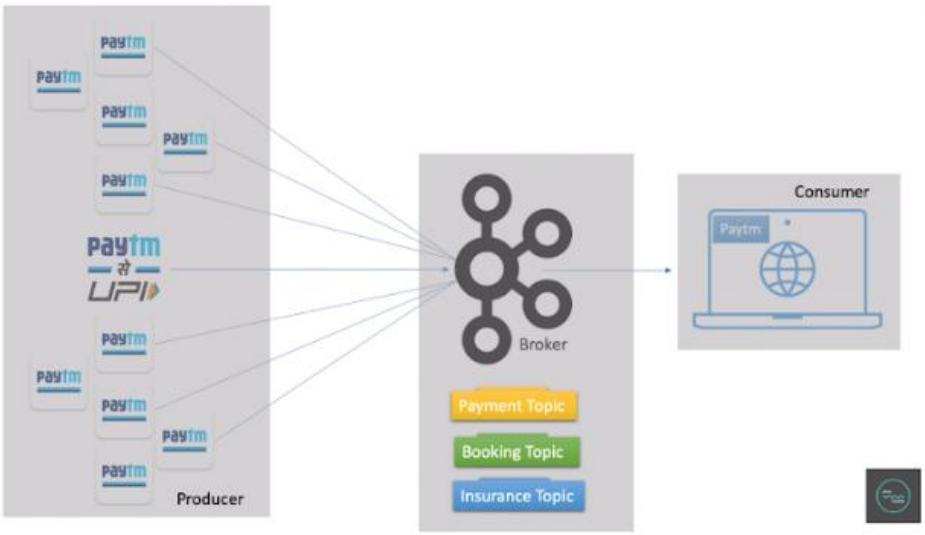


Cluster

A cluster is an common terminology in the distributed computing system , it is nothing but just a group of computers or servers that are working for a common purpose . since kafka can have a multiple server or broker in a single kafka cluster.

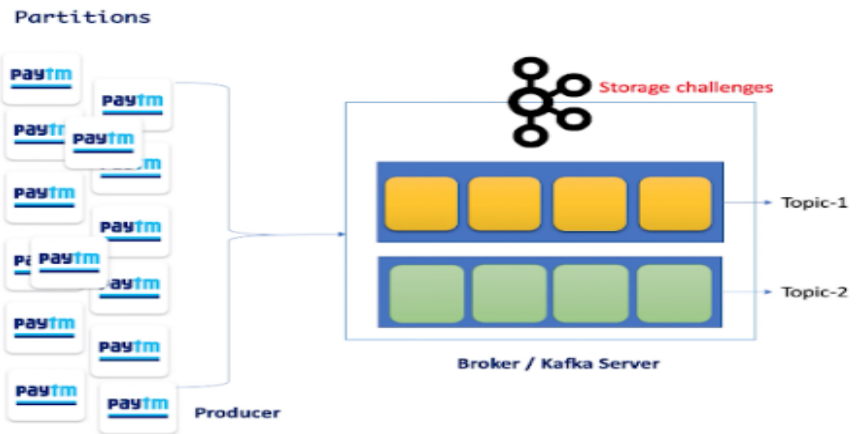
Topic

The topic basically segregate different topics into different sections like the payment related chat into payment topic and booking related chat into booking topic . It specifies the category of the message . listeners can then just respond to the messages that belong to the topics they are listening to.



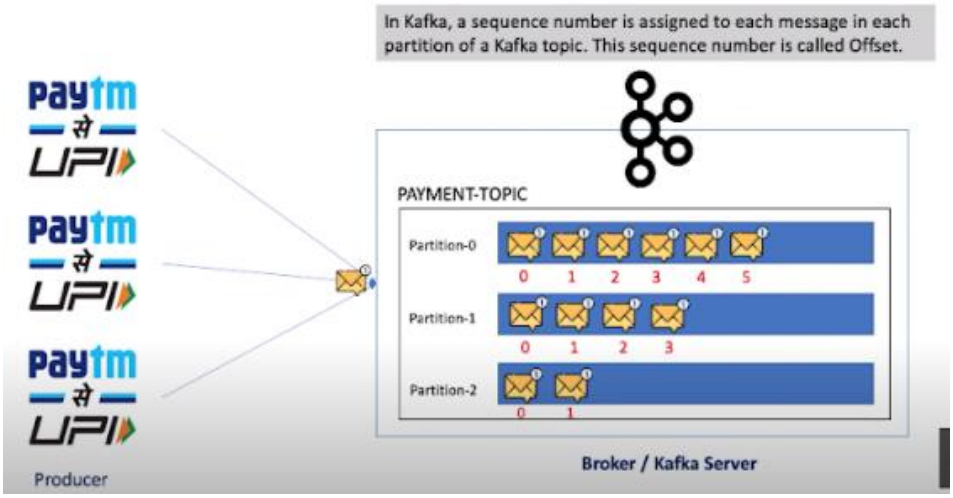
Partitions

Suppose there is a millions and billions of data transferred to the kafka . it is difficult to handle such amount of data therefore we have to break the kafka topic in multiple parts and distribute those parts into different machines . This concept is called topic partitioning and each part is called partition in kafka terminology.



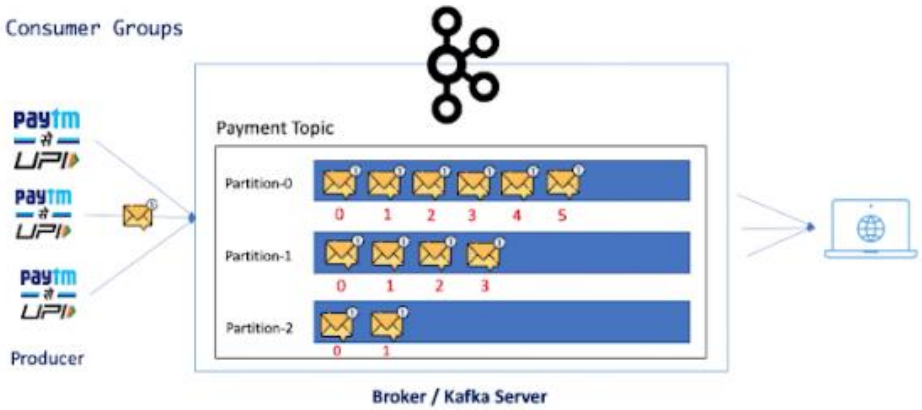
Offset

In kafka , a sequence number is assigned to each message in each partition of a kafka topic. The sequence number is called offset.

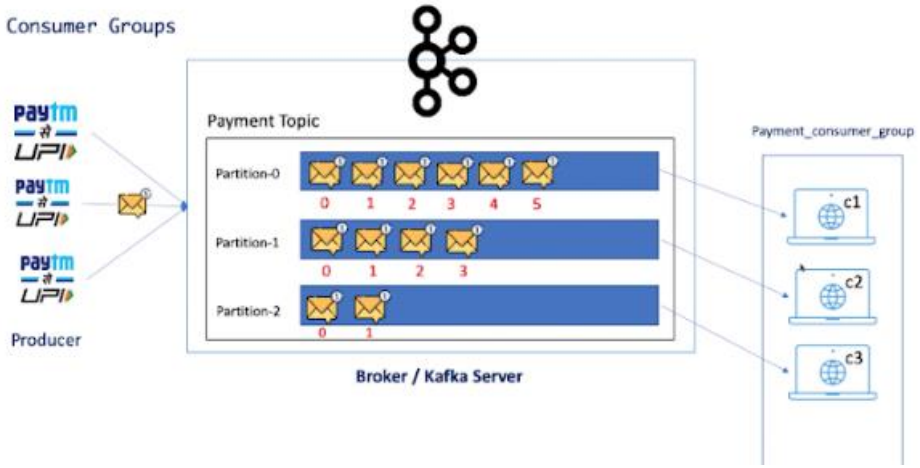


Consumer Groups

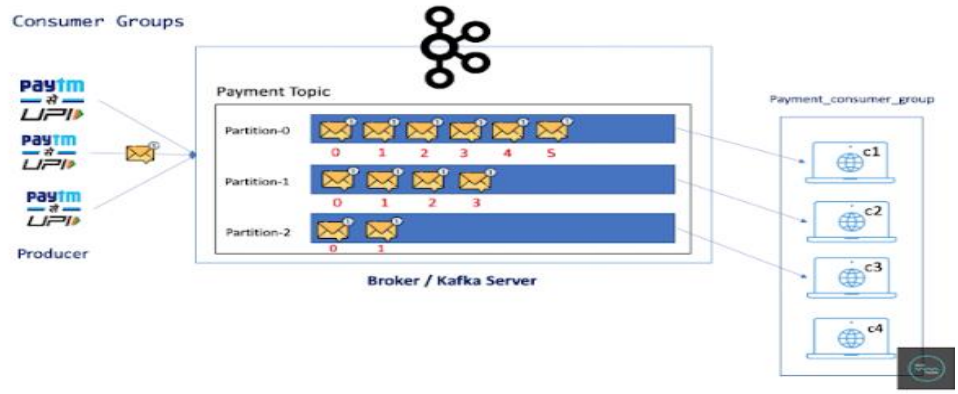
This is the single consumer reading from each and every partition which will definitely leads the performance because there is no concurrency . in this the single consumer is playing with all the three partitions which is not giving the throughput . to over come this the group of consumer is used .



To overcome this situation just share the workload by defining n number of consumer instance and grouping all of them by specifying the group name which is called as consumer groups .

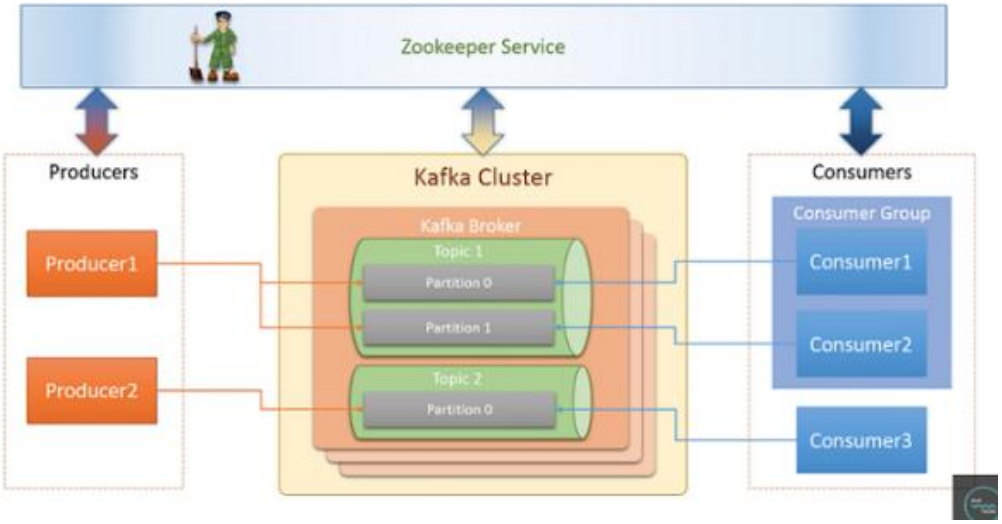


If there is more number of consumer compared to partition than that consumer will sit ideal as there is no partition left for him . for example



Zookeeper

Zookeeper is a prerequisite for kafka . kafka is a distributed system and it uses zookeeper for coordination and to track the status of kafka cluster nodes . ot also keeps track of kafka topics , partitions , offsets , etc.



Source and Sink Connectors:

- Kafka Connect is a framework for connecting Kafka with external systems, such as databases, key-value stores, search indexes, and file systems. It provides a scalable and reliable way to stream data between Kafka and these systems using connectors. There are two types of connectors:

1. **Source Connectors**: These connectors pull data from external systems into Kafka topics.
2. **Sink Connectors**: These connectors push data from Kafka topics to external systems.

Source Connectors

Source connectors are used to import data from external systems into Kafka. They capture data changes from various sources and stream them into Kafka topics.

Example: JDBC Source Connector

A JDBC source connector can pull data from a relational database into Kafka.

Sink Connectors

Sink connectors are used to export data from Kafka topics to external systems. They consume data from Kafka topics and write it to target systems.

Example: JDBC Sink Connector

A JDBC sink connector can push data from Kafka to a relational database.

Confluent kafka

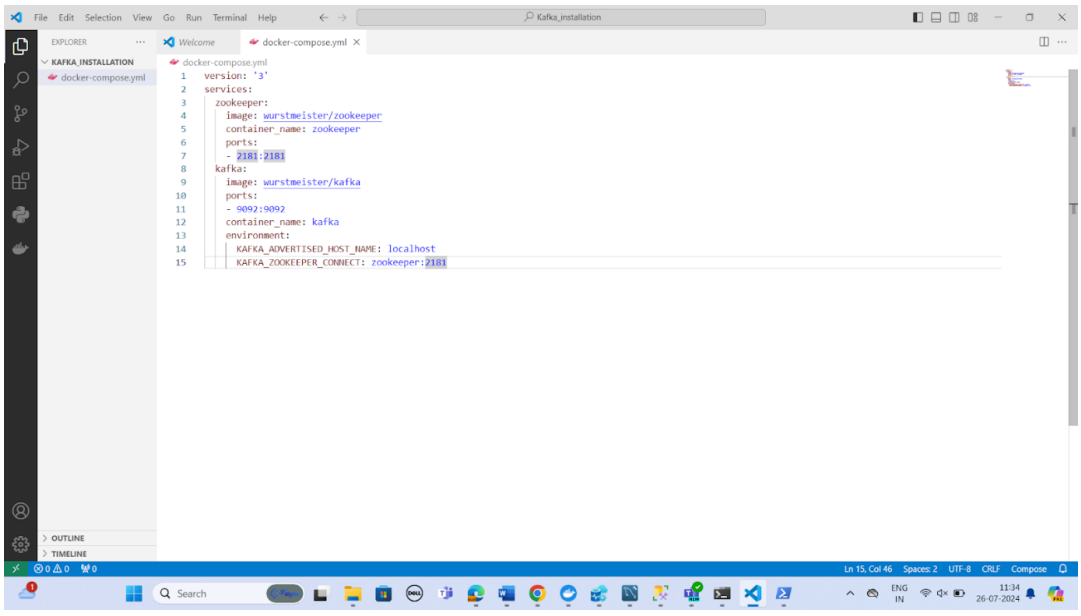
Confluent Kafka, often referred to simply as Confluent, is a comprehensive platform for building event-driven architectures and real-time data pipelines using Apache Kafka. Confluent enhances the open-source Apache Kafka with additional features, tools, and services to make it easier to deploy, manage, and utilize Kafka in a production environment.

Difference Between Confluent Kafka and Apache Kafka

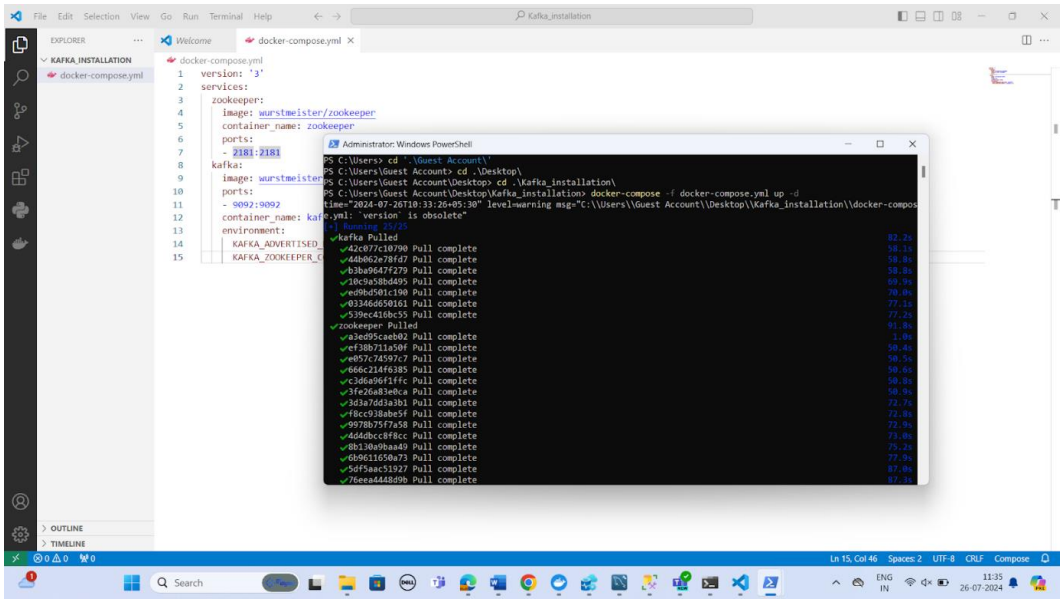
Confluent Kafka falls under the data processing category in the big data. On the other hand, Apache Kafka falls under the data operations category as it is a message queuing system.

Connection Of Kafka with MYSQL

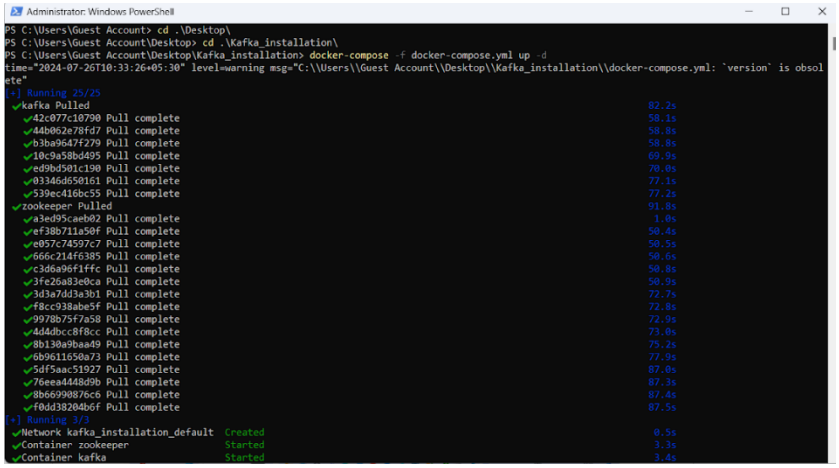
Step 1:- Write the compose.yml file to create a container and pull the Kafka Image



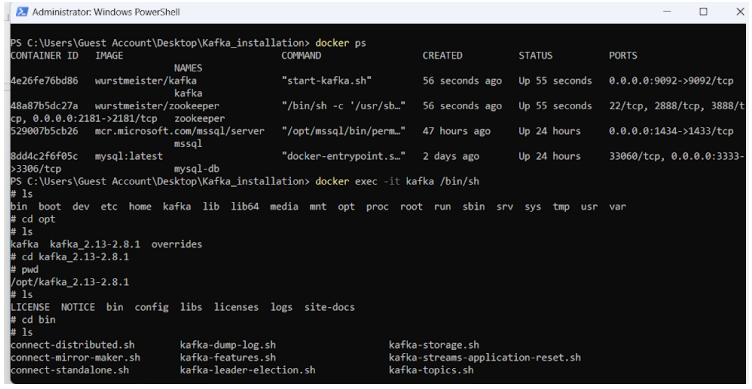
Step 2:- Kafka Image pulled successfully



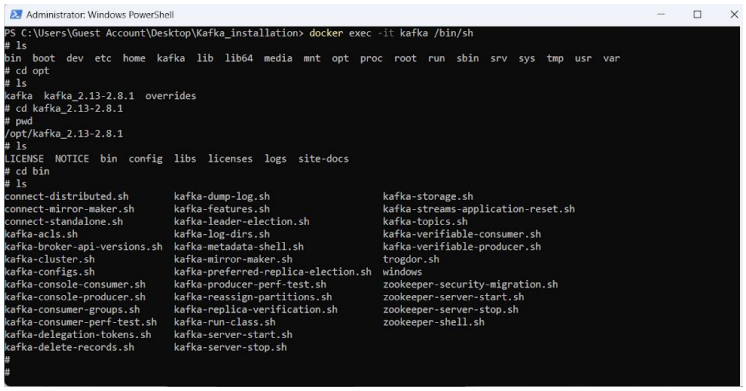
Step 3:- Container and zookeeper created



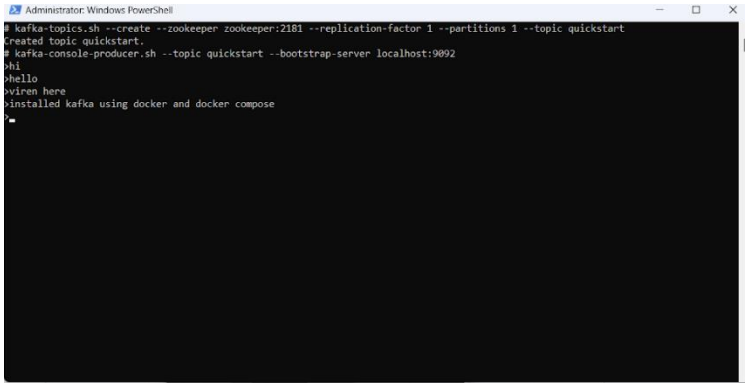
Step 4 :- Checked that the container is created using Docker ps



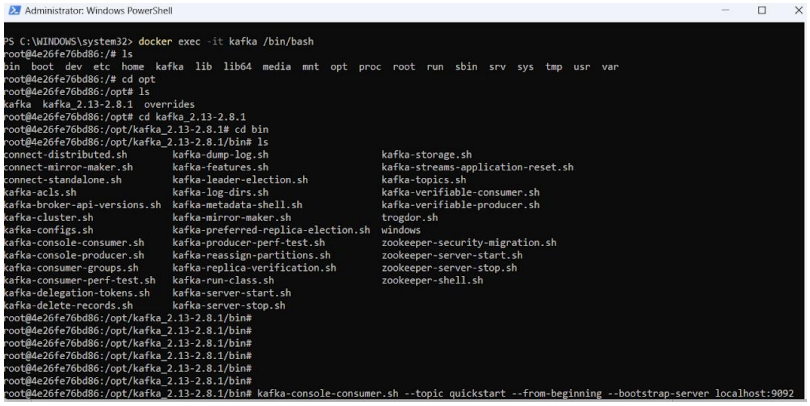
Step 5 :- Kafka bin includes all the syntax of commands like for producer , consumer , topic etc



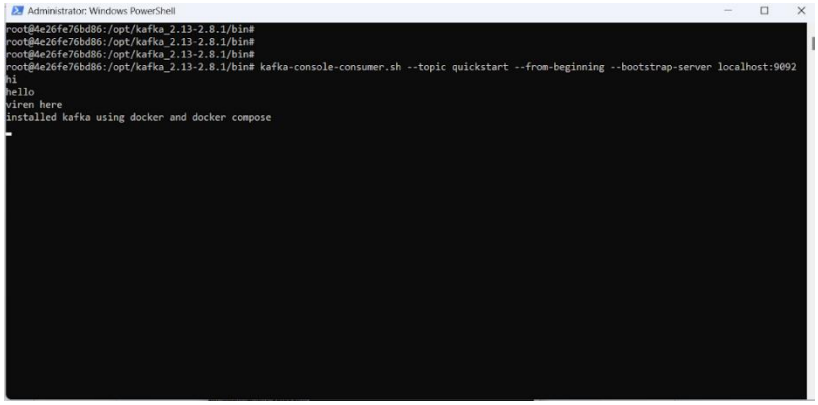
Step 6 :- Created a topic and inserted a data using producer command

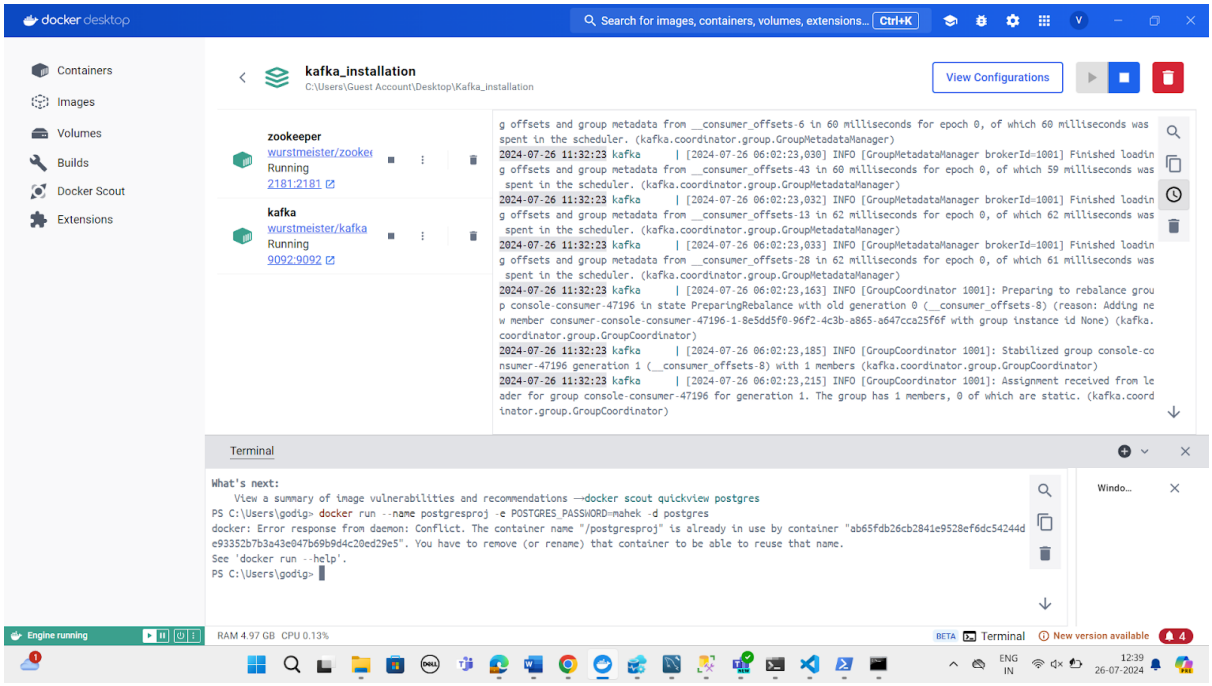


Step 7:- using the consumer command consumed all the data that was produced

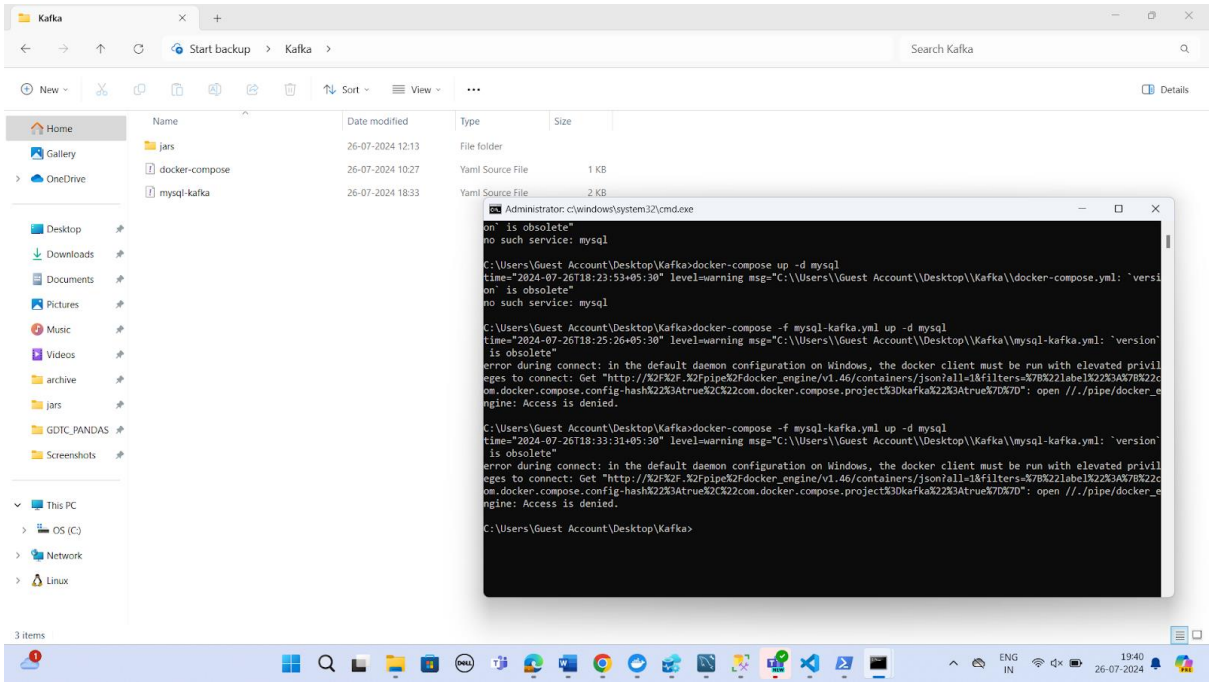


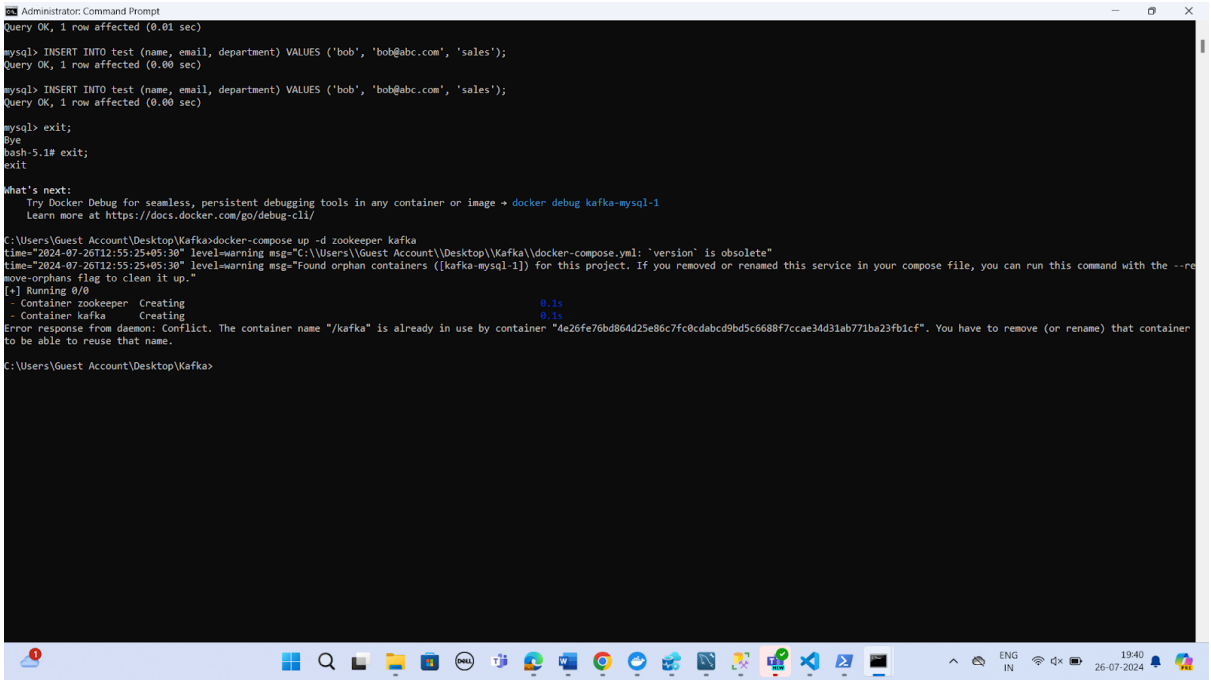
Step 8:- All the data that was produced was consumed by a consumer here

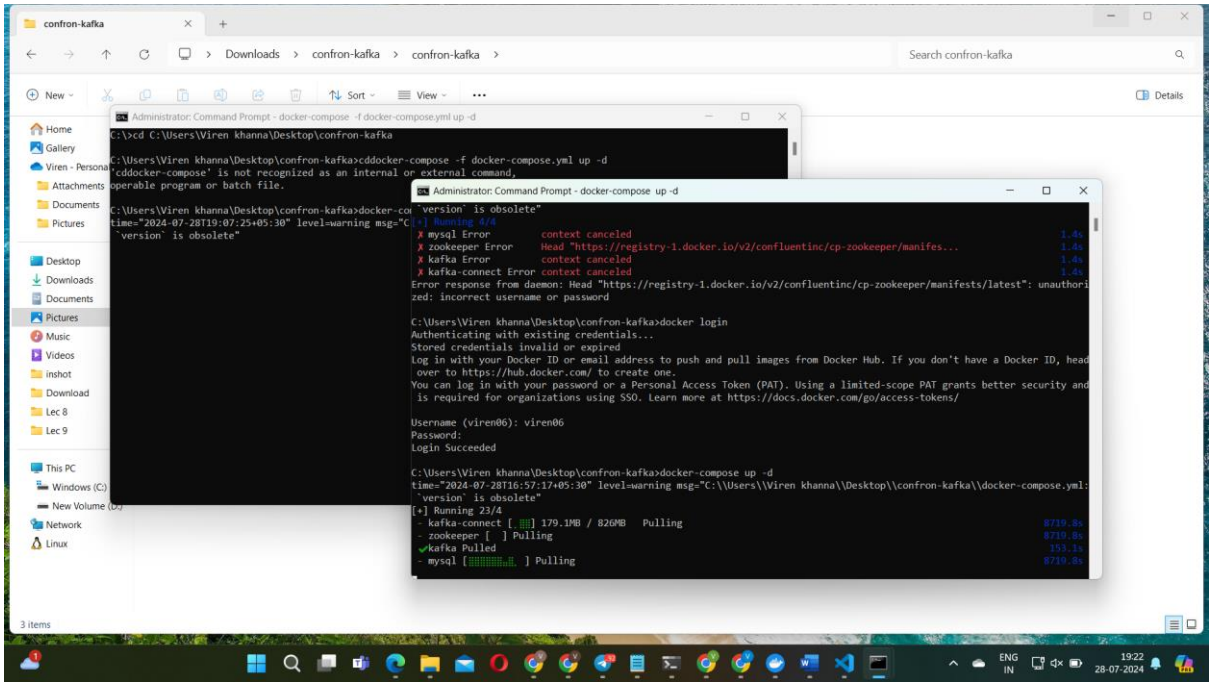




These were the errors while connecting Confluent kafka with mysql

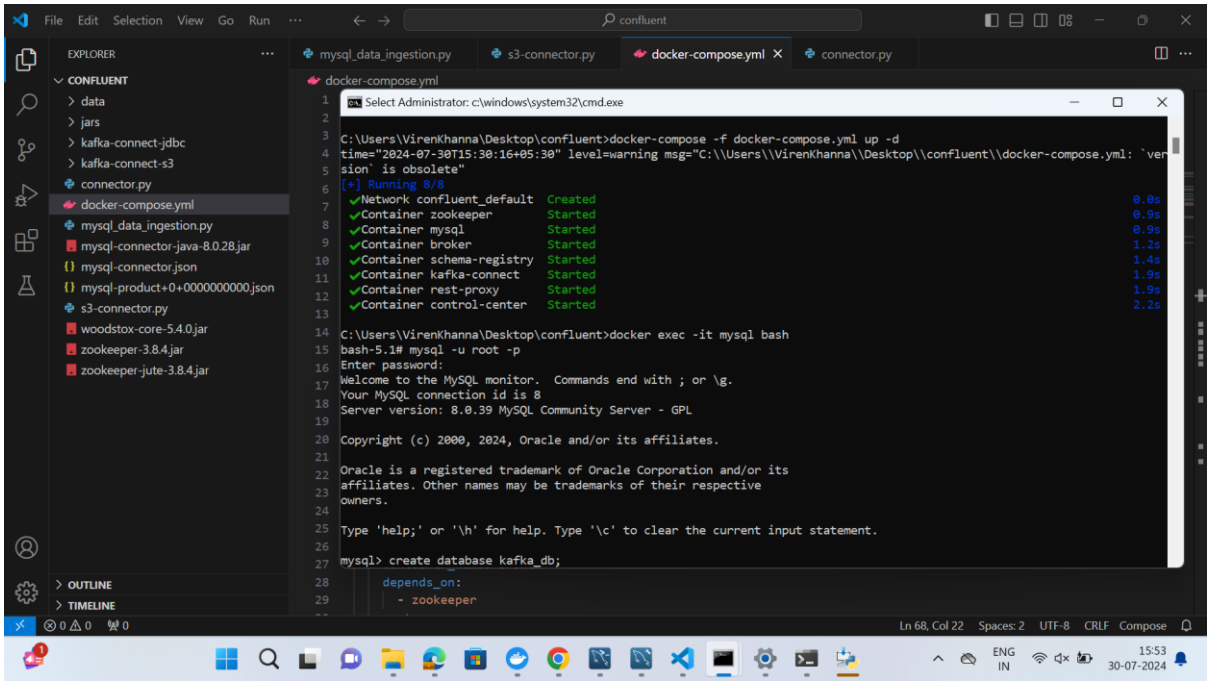




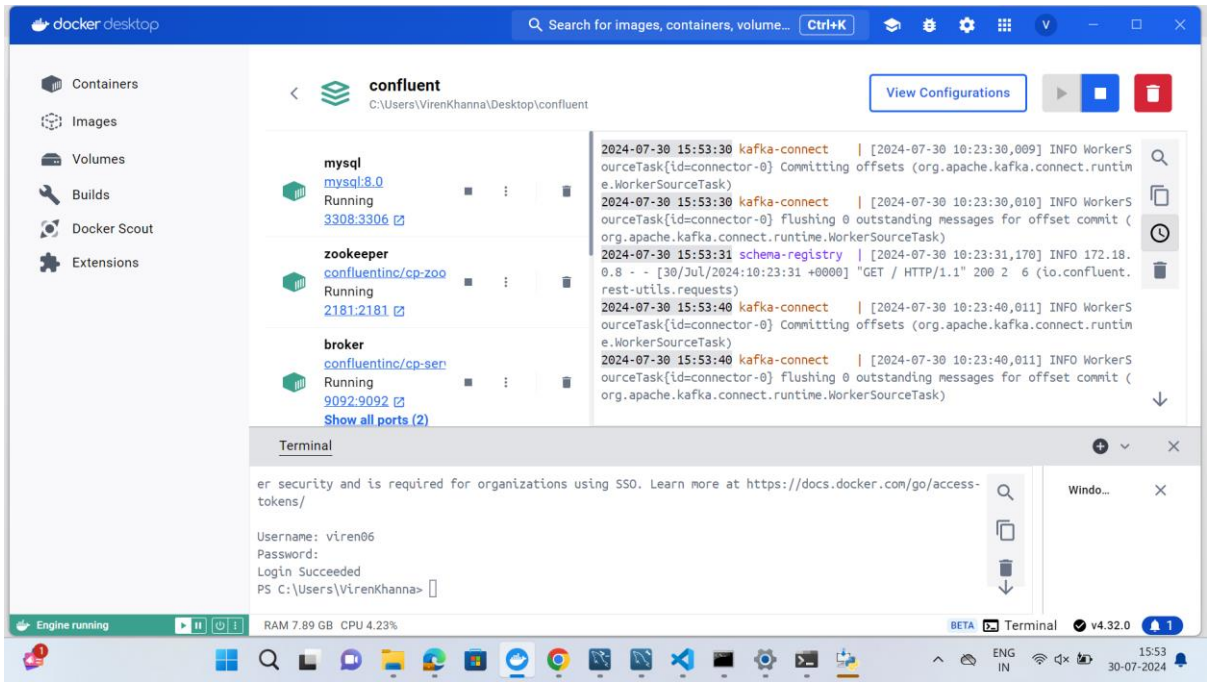


Connection of Confluent Kafka With MYSQL

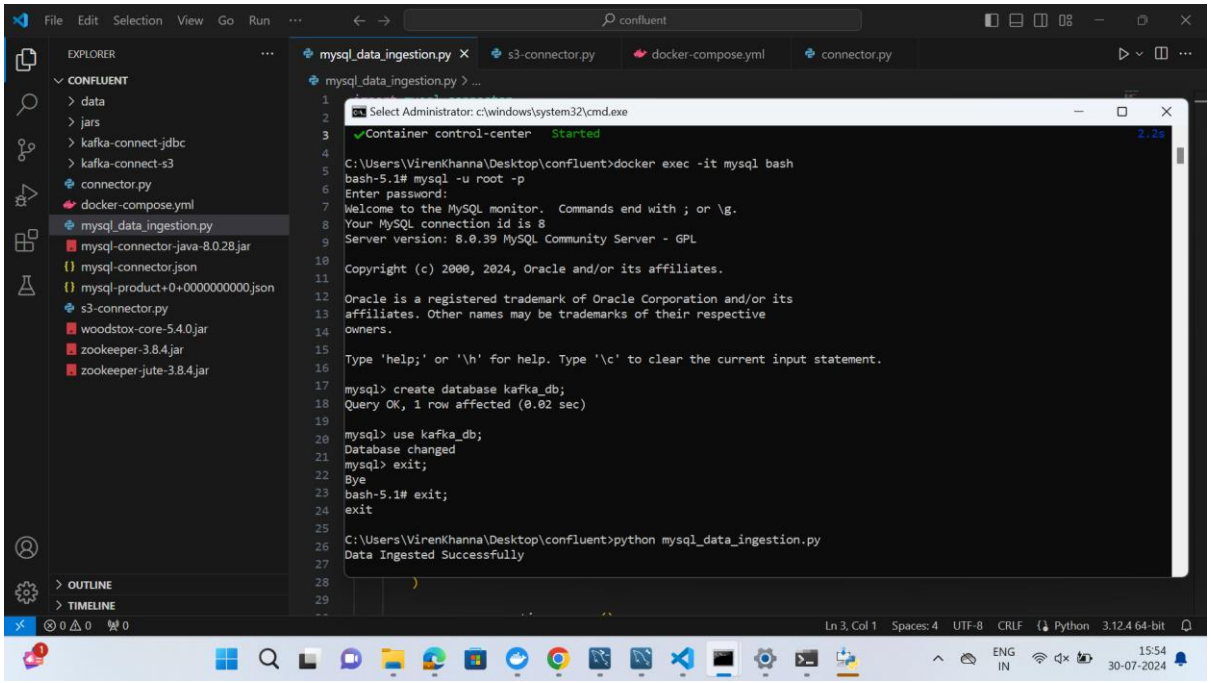
Step 1 :- Run docker compose file and start all the services like mysql , kafka etc



Created containers of mysql , zookeeper , broker

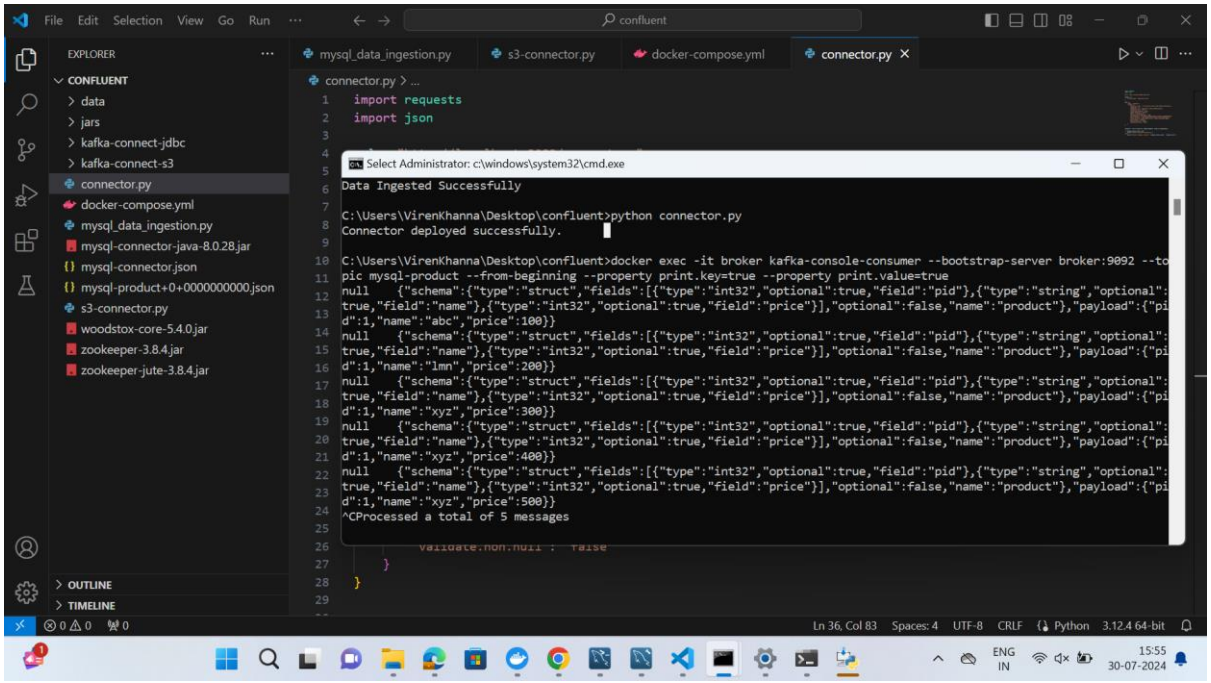


Step 2 :- Created database of name kafka_db and inserted random 5 data into it

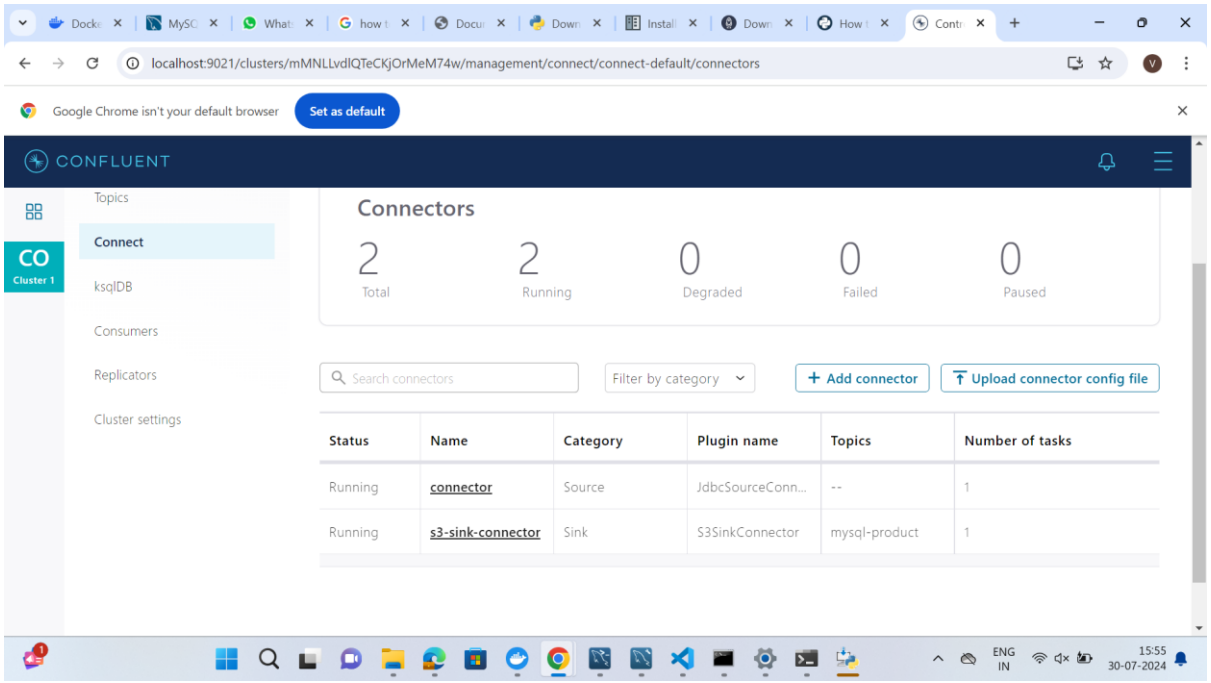


Step 3 :- ran the connector file which will connect to the localhost:9021

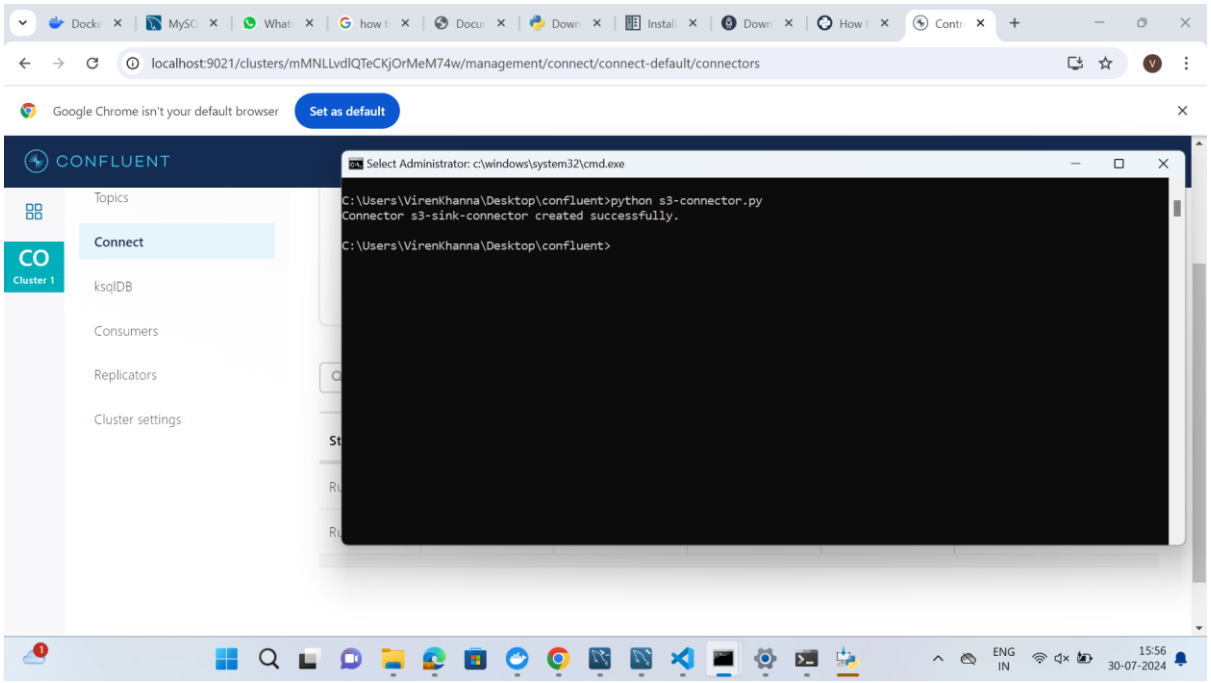
Step 4 :- checked wether the data is stored



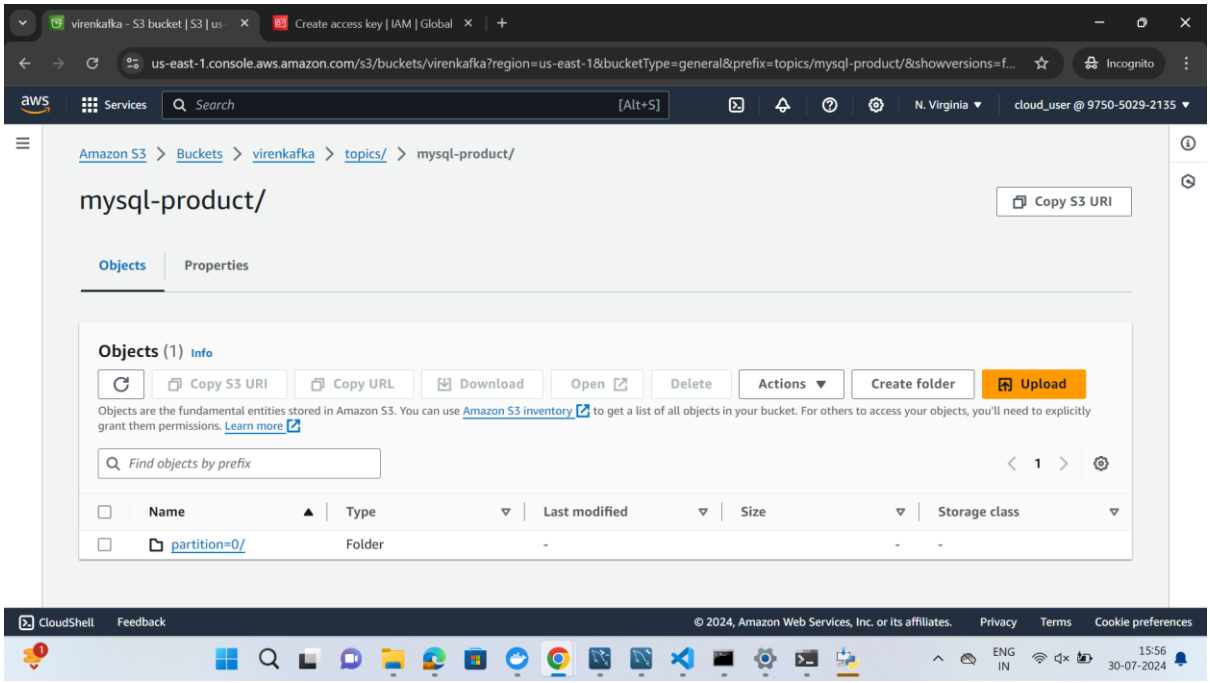
This is the localhost:9021



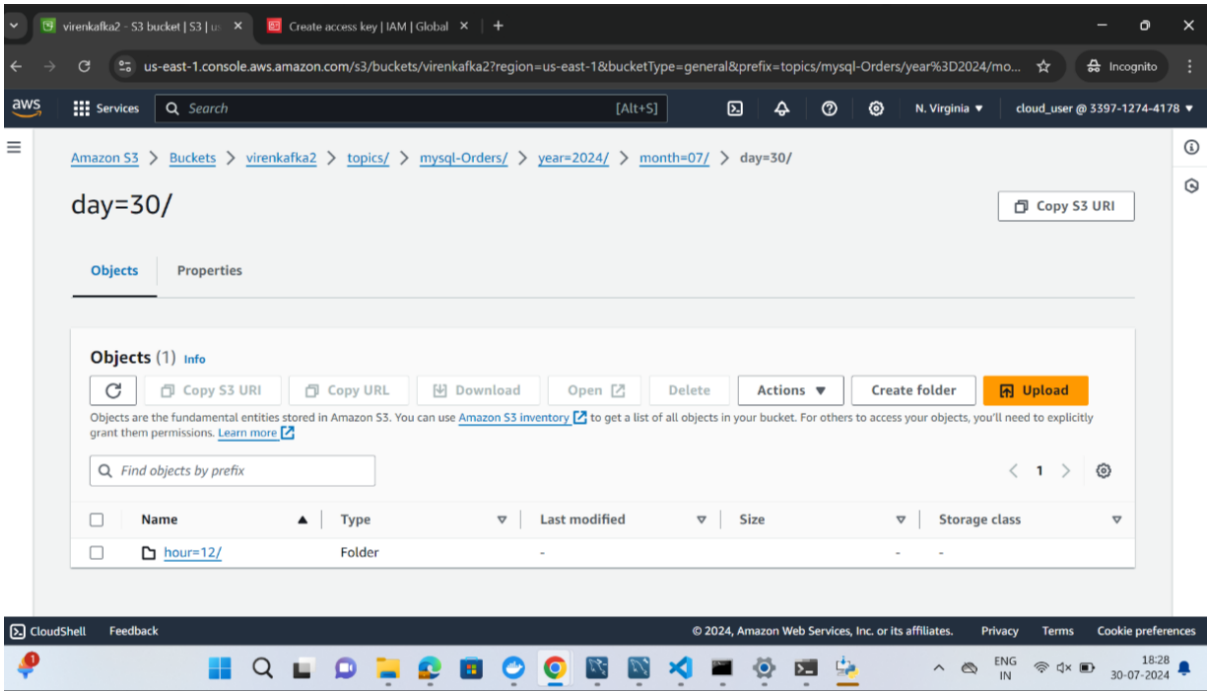
Step 5 :- ran S3 connector file which will connect with s3 bucket and will dump the data into bucket



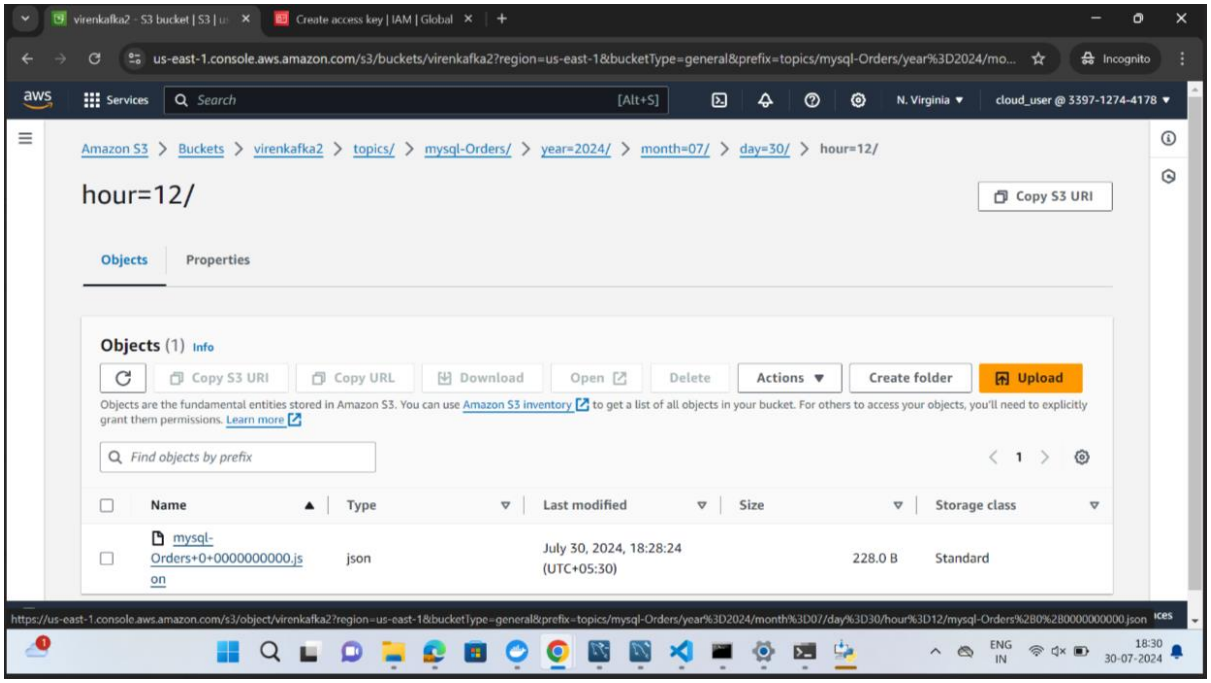
Step 6 :- Consumer dumped all the data into my s3 bucket



Step 7 :- partition the data by month year day and hour



The data in S3 is partitioned


virenkafka2 - S3 bucket | S3 | u
Create access key | IAM | Global
us-east-1.console.aws.amazon.com/s3/buckets/virenkafka2?region=us-east-1&bucketType=general&prefix=topics/mysql-Orders/year%3D2024/mo...
Incognito
aws
Services
Search
[Alt+S]
N. Virginia
cloud_user @ 3397-1274-4178
Amazon S3 > Buckets > virenkafka2 > topics/ > mysql-Orders/ > year=2024/ > month=07/ > day=30/ > hour=12/
hour=12/
Copy S3 URI
Objects
Properties
Objects (1) Info
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
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
Find objects by prefix
1
Name
Type
Last modified
Size
Storage class
mysql-Orders+0+0000000000.json
json
July 30, 2024, 18:28:24 (UTC+05:30)
228.0 B
Standard
https://us-east-1.console.aws.amazon.com/s3/object/virenkafka2?region=us-east-1&bucketType=general&prefix=topics/mysql-Orders/year%3D2024/month%3D07/day%3D30/hour%3D12/mysql-Orders%2B0%2B0000000000.json
ENG
IN
18:30
30-07-2024