Apache Kafka Interview Questions

(Q.1) What is Apache Kafka?

Ans. Apache Kafka is a publish-subscribe open source message broker application. This messaging application was coded in “Scala”. Basically, this project was started by the Apache software. Kafka’s design pattern is mainly based on the transactional logs design. For detailed understanding of Kafka, go through.

(Q.2) Enlist the several components in Kafka.

Ans. The most important elements of Kafka are:

- Topic – Kafka Topic is a bunch or a collection of messages.
- Producer – In Kafka, Producers issue communications as well as publish messages to a Kafka topic.
- Consumer – Kafka Consumers subscribes to a topic(s) and also reads and processes messages from the topic(s).
- Brokers –While it comes to manage storage of messages in the topic(s) we use Kafka Brokers. For detailed understanding of Kafka components, go through,

(Q.3) Explain the role of the offset.

Ans. There is a sequential ID number given to the messages in the partitions what we call an offset. So, to identify each message in the partition uniquely, we use these offsets.

(Q.4) What is a Consumer Group?

Ans. The concept of Consumer Groups is exclusive to Apache Kafka. Basically, every Kafka consumer group consists of one or more consumers that jointly consume a set of subscribed topics.

(Q.5) What is the role of the ZooKeeper in Kafka?

Ans. Apache Kafka is a distributed system built to use Zookeeper. Zookeeper's main role here is to build coordination between different nodes in a cluster. However, we also use Zookeeper to recover from previously committed offset if any node fails because it works as a periodic commit offset.

(Q.6) Is it possible to use Kafka without ZooKeeper?

Ans. It is impossible to bypass Zookeeper and connect directly to the Kafka server, so the answer is no. If somehow, ZooKeeper is down, then it is impossible to service any client request.

(Q.7) What do you know about Partition in Kafka?

Ans. In every Kafka broker, there are few partitions available. And, here each partition in Kafka can be either a leader or a replica of a topic.

(Q.8) Why is Kafka technology significant to use?

Ans. There are some advantages of Kafka, which makes it significant to use:

- High-throughput : We do not need any large hardware in Kafka, because it is capable of handling high-velocity and high-volume data. Moreover, it can also support message throughput of thousands of messages per second.
- Low Latency : Kafka can easily handle these messages with the very low latency of the range of milliseconds, demanded by most of the new use cases.
- Fault-Tolerant : Kafka is resistant to node/machine failure within a cluster.
- Durability : As Kafka supports message replication, messages are never lost. It is one of the reasons behind durability.
- Scalability : Kafka can be scaled-out, without incurring any downtime on the fly by adding additional nodes.

(Q.9) What are the main APIs of Kafka?

Ans. Apache Kafka has 4 main APIs:

- Producer API
- Consumer API
- Streams API
- Connector API

(Q.10) What are consumers or users?

Ans. Mainly, Kafka Consumer subscribes to a topic(s), and also reads and processes messages from the topic(s). Moreover, with a consumer group name, Consumers label themselves. In other words, within each subscribing consumer group, each record published to a topic is delivered to one consumer instance. Make sure it is possible that Consumer instances can be in separate processes or on separate machines.

Q.11 Explain the concept of Leader and Follower.

Ans. In every partition of Kafka, there is one server which acts as the Leader, and none or more servers play the role as Followers.

Q.12 What ensures load balancing of the server in Kafka?

Ans. The main role of the Leader is to perform the task of all read and write requests for the partition, whereas Followers passively replicate the leader. Hence, at the time of Leader failing, one of the Followers took over the role of the Leader. Basically, this entire process ensures load balancing of the servers.

Q.13 What roles do Replicas and the ISR play?

Ans. Basically, a list of nodes that replicate the log is Replicas. Especially, for a particular partition. However, they are irrespective of whether they play the role of the Leader. In addition, ISR refers to In-Sync Replicas. On defining ISR, it is a set of message replicas that are synced to the leaders.

Q.14 Why are Replications critical in Kafka?

Ans. Because of Replication, we can be sure that published messages are not lost and can be consumed in the event of any machine error, program error or frequent software upgrades.

Q.15 If a Replica stays out of the ISR for a long time, what does it signify?

Ans. Simply, it implies that the Follower cannot fetch data as fast as data accumulated by the Leader.

Q.16 What is the process for starting a Kafka server?

Ans. It is the very important step to initialize the ZooKeeper server because Kafka uses ZooKeeper.So, the process for starting a Kafka server is: In order to start the ZooKeeper server: `> bin/zookeeper-server-start.sh config/zookeeper.properties` Next, to start the Kafka server: `> bin/kafka-server-start.sh config/server.properties`

Q.17 In the Producer, when does QueueFullException occur?

Ans. whenever the Kafka Producer attempts to send messages at a pace that the Broker cannot handle at that time QueueFullException typically occurs. However, to collaboratively handle the increased load, users will need to add enough brokers(servers, nodes), since the Producer doesn't block.

Q.18 Explain the role of the Kafka Producer API.

Ans. An API which permits an application to publish a stream of records to one or more Kafka topics is what we call Producer API.

Q.19 What is the main difference between Kafka and Flume?

Ans. The main difference between Kafka and Flume are: Types of tool

- Apache Kafka– As Kafka is a general-purpose tool for both multiple producers and consumers.
- Apache Flume– Whereas, Flume is considered as a special-purpose tool for specific applications.

Replication feature

- Apache Kafka– Kafka can replicate the events.
- Apache Flume- whereas, Flume does not replicate the events.

Q.20 Is Apache Kafka a distributed streaming platform? If yes, what can you do with it?

Ans. Undoubtedly, Kafka is a streaming platform. It can help: To push records easily. Also, can store a lot of records without giving any storage problems Moreover, it can process the records as they come in

Q. 21 What can you do with Kafka?

Ans. It can perform in several ways, such as:

- In order to transmit data between two systems, we can build a real-time stream of data pipelines with it.
- Also, we can build a real-time streaming platform with Kafka, that can actually react to the data.

Q.22 What is the purpose of the retention period in the Kafka cluster?

Ans. However, the retention period retains all the published records within the Kafka cluster. It doesn't check whether they have been consumed or not. Moreover, the records can be discarded by using a configuration setting for the retention period. And, it results as it can free up some space.

Q.23 Explain the maximum size of a message that can be received by the Kafka?

Ans. The maximum size of a message that can be received by the Kafka is approx. 1000000 bytes.

Q.24 What are the types of traditional method of message transfer?

Ans. Basically, there are two methods of the traditional message transfer method, such as:

- Queuing: It is a method in which a pool of consumers may read a message from the server and each message goes to one of them.
- Publish-Subscribe: Whereas in Publish-Subscribe, messages are broadcasted to all consumers.

Q.25 What does ISR stand for in the Kafka environment?

Ans. ISR refers to In sync replicas. These are generally classified as a set of message replicas which are synced to be leaders.

Q.26 What is Geo-Replication in Kafka?

Ans. For our cluster, Kafka MirrorMaker offers geo-replication. Basically, messages are replicated across multiple data centres or cloud regions, with MirrorMaker. So, it can be used in active/passive scenarios for backup and recovery; or also to place data closer to our users, or support data locality requirements.

Q.27 Explain Multi-tenancy?

Ans. We can easily deploy Kafka as a multi-tenant solution. However, by configuring which topics can produce or consume data, Multi-tenancy is enabled. Also, it provides operations support for quotas.

Q.28 What is the role of Consumer API?

Ans. An API which permits an application to subscribe to one or more topics and also to process the stream of records produced to them is what we call Consumer API.

Q.29 Explain the role of Streams API?

Ans. An API which permits an application to act as a stream processor, and also consuming an input stream from one or more topics and producing an output stream to one or more output topics, moreover, transforming the input streams to output streams effectively, is what we call Streams API.

Q.30 What is the role of Connector API?

Ans. An API which permits to run as well as build the reusable producers or consumers which connect Kafka topics to existing applications or data systems is what we call the Connector API.

Q.31 Explain Producer?

Ans. The main role of Producers is to publish data to the topics of their choice. Basically, its duty is to select the record to assign to partition within the topic.

Q.32 Compare: RabbitMQ vs Apache Kafka

Ans. One of Apache Kafka's alternatives is RabbitMQ. So, let's compare both:

- Features
    - Apache Kafka– Kafka is distributed, durable and highly available, here the data is shared as well as replicated.
    - RabbitMQ– There are no such features in RabbitMQ.
- Performance rate
    - Apache Kafka– To the tune of 100,000 messages/second.
    - RabbitMQ- In case of RabbitMQ, the performance rate is around 20,000 messages/second.

Q.33 Compare: Traditional queuing systems vs Apache Kafka

Ans. Let's compare Traditional queuing systems vs Apache Kafka feature-wise:
Messages Retaining

- Traditional queuing systems– It deletes the messages just after processing completion typically from the end of the queue.
- Apache Kafka– But in Kafka, messages persist even after being processed. That implies messages in Kafka don't get removed as consumers receive them. Logic-based processing
- Traditional queuing systems–Traditional queuing systems don't permit processing logic based on similar messages or events.
- Apache Kafka– Kafka permits to process logic based on similar messages or events.

Q.34 Why Should we use Apache Kafka Cluster?

Ans. In order to overcome the challenges of collecting the large volume of data, and analyzing the collected data we need a messaging system. Hence Apache Kafka came into the story. Its benefits are:

- It is possible to track web activities just by storing/sending the events for real-time processes.
- Through this, we can Alert as well as report the operational metrics.
- Also, we can transform data into the standard format. *Moreover, it allows continuous processing of streaming data to the topics.
- Due to its wide use, it is ruling over some of the most popular applications like ActiveMQ, RabbitMQ, AWS etc.

Q.35 Explain the term "Log Anatomy".

Ans. We view logs as the partitions. Basically, a data source writes messages to the log. One of the advantages is, at any time one or more consumers read from the log they select.

Q.36 What is a Data Log in Kafka?

Ans. As we know, messages are retained for a considerable amount of time in Kafka. Moreover, there is flexibility for consumers that they can read as per their convenience. Although, there is a possible case that if Kafka is configured to keep messages for 24 hours and possibly that time the consumer is down for a time greater than 24 hours, then the consumer may lose those messages. However, still, we can read those messages from the last known offset, but only at a condition that the downtime on part of the consumer is just 60 minutes. Moreover, on what consumers are reading from a topic Kafka doesn't keep state.

Q.37 Explain how to Tune Kafka for Optimal Performance.

Ans. So, ways to tune Apache Kafka it is to tune its several components:

- Tuning Kafka Producers
- Kafka Brokers Tuning
- Tuning Kafka Consumers

Q.38 State Disadvantages of Apache Kafka.

Ans. Limitations of Kafka are:

- No Complete Set of Monitoring Tools
- Issues with Message Tweaking
- Not support wildcard topic selection
- Lack of Pace

Q.39 Enlist all Apache Kafka Operations.

Ans. Apache Kafka Operations are:

- Addition and Deletion of Kafka Topics
- How to modify the Kafka Topics
- Distinguished Turnoff
- Mirroring Data between Kafka Clusters
- Finding the position of the Consumer
- Expanding Your Kafka Cluster
- Migration of Data Automatically
- Retiring Servers
- Data Centers

Q.40 Explain Apache Kafka Use Cases?

Ans. Apache Kafka has so many use cases, such as:

- Kafka Metrics It is possible to use Kafka for operational monitoring data. Also, to produce centralized feeds of operational data, it involves aggregating statistics from distributed applications.
- Kafka Log Aggregation Moreover, to gather logs from multiple services across an organization.
- Stream Processing While stream processing, Kafka's strong durability is very useful.

Q.41 Some of the most notable applications of Kafka.

Ans. Some of the real-time applications are:

- Netflix
- Mozilla
- Oracle

Q.42 Features of Kafka Stream.

Ans. Some best features of Kafka Stream are

- Kafka Streams are highly scalable and fault-tolerant.
- Kafka deploys to containers, VMs, bare metal, cloud.
- We can say, Kafka streams are equally viable for small, medium, & large use cases.
- Also, it is fully in integration with Kafka security.
- Write standard Java applications.
- Exactly-once processing semantics.
- Moreover, there is no need for a separate processing cluster.

Q.43 What do you mean by Stream Processing in Kafka?

Ans. The type of processing of data continuously, real-time, concurrently, and in a record-by-record fashion is what we call Kafka Stream processing.

Q.44 What are the types of System tools?

Ans. There are three types of System tools:

- Kafka Migration Tool It helps to migrate a broker from one version to another.
- Mirror Maker Mirror Maker tool helps to offer a mirror of one Kafka cluster to another.
- Consumer Offset Checker For the specified set of Topics as well as Consumer Group, it shows Topic, Partitions, Owner.

Q.45 What are Replication Tools and their types?

Ans. For the purpose of stronger durability and higher availability, a replication tool is available here. Its types are −

- Create Topic Tool
- List Topic Tool
- Add Partition Tool

Q.46 What is the Importance of Java in Apache Kafka?

Ans. For the need of the high processing rates that come standard on Kafka, we can use the Java language. Moreover, for Kafka consumer clients also, Java offers good community support. So, we can say it is the right choice to implement Kafka in Java.

Q.47 State one best feature of Kafka.

Ans. The best feature of Kafka is “Variety of Use Cases”. It means Kafka is able to manage the variety of use cases which are very common for a Data Lake. For Example log aggregation, web activity tracking, and so on.

Q.48 Explain the term “Topic Replication Factor”.

Ans. It is very important to factor in topic replication while designing a Kafka system. Hence, if in any case, a broker goes down its topics’ replicas from another broker can solve the crisis.

Q.49 Explain some Kafka Streams real-time Use Cases.

Ans. So, the use cases are:

- The New York Times: This company uses it to store and distribute, in real-time, published content to the various applications and systems that make it available to the readers. Basically, it uses Apache Kafka and the Kafka Streams both.
- Zalando: As an ESB (Enterprise Service Bus) as the leading online fashion retailer in Europe Zalando uses Kafka.
- LINE: Basically, to communicate to one another LINE application uses Apache Kafka as a central data hub for their services.

Q.50 What are Guarantees provided by Kafka?

Ans. They are:

- The order will be the same for both the Messages sent by a producer to a particular topic partition. That
- Moreover, the consumer instance sees records in the order in which they are stored in the log.
- Also, we can tolerate up to N-1 server failures, even without losing any records committed to the log.