Contents

[Storage Architecture 2](#_Toc152607697)

[Topic 2](#_Toc152607698)

[Partition 2](#_Toc152607699)

[Replica 2](#_Toc152607700)

[Logs/Segments 3](#_Toc152607701)

[Offset 4](#_Toc152607702)

[Index 4](#_Toc152607703)

[Timestamp 4](#_Toc152607704)

[Cluster 4](#_Toc152607705)

[Zookeeper 4](#_Toc152607706)

[Controller 5](#_Toc152607707)

[Brokers 5](#_Toc152607708)

Unsorted Starts

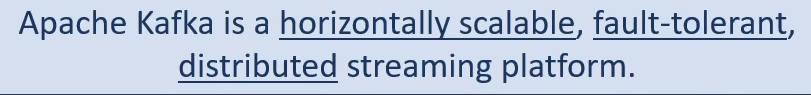
# Unsorted Starts

1. **Topic creation Command**:  
   %KAFKA\_HOME%\bin\windows\**kafka-topics.bat** --create --zookeeper localhost:2181 **--topic invoice** --partitions 5 --replication-factor 3 **--config segment.bytes**=1000000

Unsorted Ends

Definition

# Definition

1. 

##### Storage Architecture

###### Topic

1. Logical name. All the msgs can not be put in a single topic so topic is divided into partitions where each partition is physical Dir.

###### Partition

1. Topic is divided into partitions as all the msgs can not be put into a single topic. Partition is a physical dir.
2. Diagram

   Description automatically generated
3. Partitions are equally distributed over all the brokers.   
   If Partition# > Broker# then some brokers will get more partitions than the other brokers.
4. For each partition and replica, a partition and a replica directory are created under log Dir (**log.dirs**=../tmp/kafka-logs-0)
5. Partitions starts from zero.

###### Replica

1. Number of copies for each partition.
2. **Command**:  
   %KAFKA\_HOME%\bin\windows\**kafka-topics.bat** --create --zookeeper localhost:2181 **--topic invoice** --partitions 5 **--replication-factor 3** --config segment.bytes=1000000  
   Graphical user interface, text

   Description automatically generated
3. Replicas can be classified into two categories.  
   Graphical user interface, text, application

   Description automatically generated

###### Logs/Segments

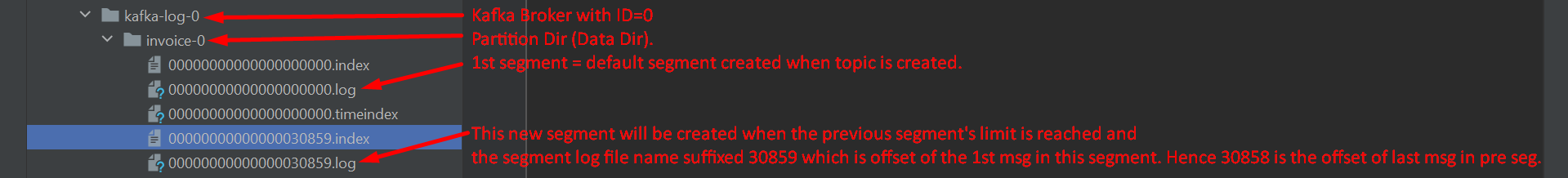
1. The msgs are stored in the log file in the partition directory (Data Directory).
2. However, instead of storing all the msgs into a single log file, Kafka generates several smaller log files known as **segments**.  
   Graphical user interface, text

   Description automatically generated
3. Each segment has a size and when reached, a new segment is created and next msgs are stored there.   
   **The default limit** is 1 GB or a week of data.   
   We can change it when creating a topic like:  
   A screen shot of a computer

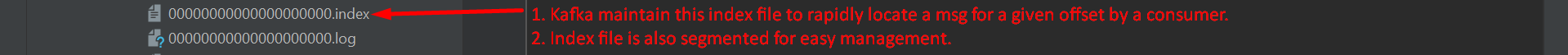
   Description automatically generated

###### Offset

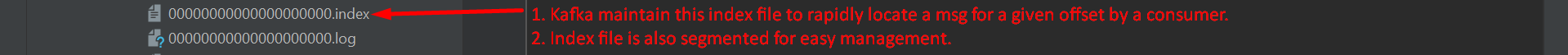
1. A screenshot of a computer

   Description automatically generated
2. Each Kafka Msg within a single partition is uniquely identified by 64-bit (4 Bytes = Java Int) integer offset.
3. For easily identification, the segment file name will be suffixed with the offset of the 1st msg in the segment.
4. 

###### Index

1. To help Broker to locate msgs rapidly for a given offset, Kafka maintains an index of offset.
2. 
3. Index files are also segmented like log and timestamp files segmented.   
   They are put also under partition directory along with log and timestamp files.

###### Timestamp

1. To help Broker to locate msgs based on timestamp, Kafka maintains an timestamp files.
2. 
3. Timestamp files are also segmented like log and index files segmented.   
   They are put also under partition directory along with log and index files.  
   A screenshot of a computer

   Description automatically generated

##### Cluster

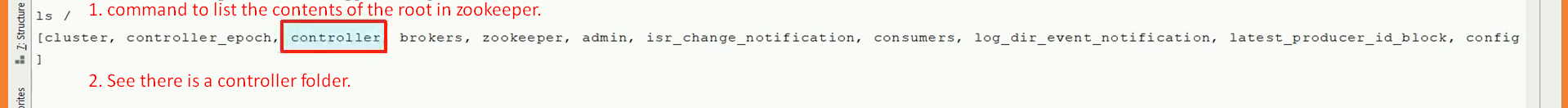
1. A set of Brokers working together to share workload.
2. Cluster Membership (Who joined and left) and Administrative (When one dies, who will take care of responsibility). Both by Master node.

##### Zookeeper

1. To manage list of active brokers.
2. server.properties =>   
   Each broker connects with Zookeeper and create Ephemeral Node called ZNode under zookeeper brokers/ids directory.
3. The following commands will list all info about brokers, cluster, consumers, controller etc.Text

   Description automatically generated

##### Controller

1. To perform administrative task such as when one dies, who will take care.
2. One of the brokers from the cluster is elected as controller.
3. **Selection**: The 1st broker connecting to the Zookeeper becomes controller by creating Ephemeral Controller Node in the zookeeper whereas other get “Node already exists”.  
   

##### Brokers

1. **Commands**:
   1. %KAFKA\_HOME%\bin\windows\**kafka-server-start.bat** %KAFKA\_HOME%\etc\kafka\**server-0.properties**
   2. **server-0.properties**
      1. # A comma separated list of directories under which to store log files

**log.dirs**=../tmp/kafka-logs-0 🡸 Path is relative to the working Dir where it will be run.

1. This directory will maintain the state of this broker/server.
2. It will contain:
   1. Log/Data Directory which in turn will contain log files having msgs.
      1. # The id of the broker. This must be set to a unique integer for each broker.

**broker.id**=0

* + 1. # The address the socket server listens on

**listeners**=PLAINTEXT://:9092

1. Partition Broker Assignment:  
   A picture containing calendar

   Description automatically generated
2. Graphical user interface

   Description automatically generatedd

###### Leader

1. Since replica can lag behind so leader has to maintain a list of ISR (In-Sync-Replica) in ZookeeperA screenshot of a computer

   Description automatically generated
2. A diagram of a diagram

   Description automatically generatedDiagram

   Description automatically generated
3. Two cases when it is removed from ISR list.
   1. When a follower has not made a request for the last 10 seconds.
   2. When a follower made a request, but it has not caught up to the most recent message in 10 seconds.