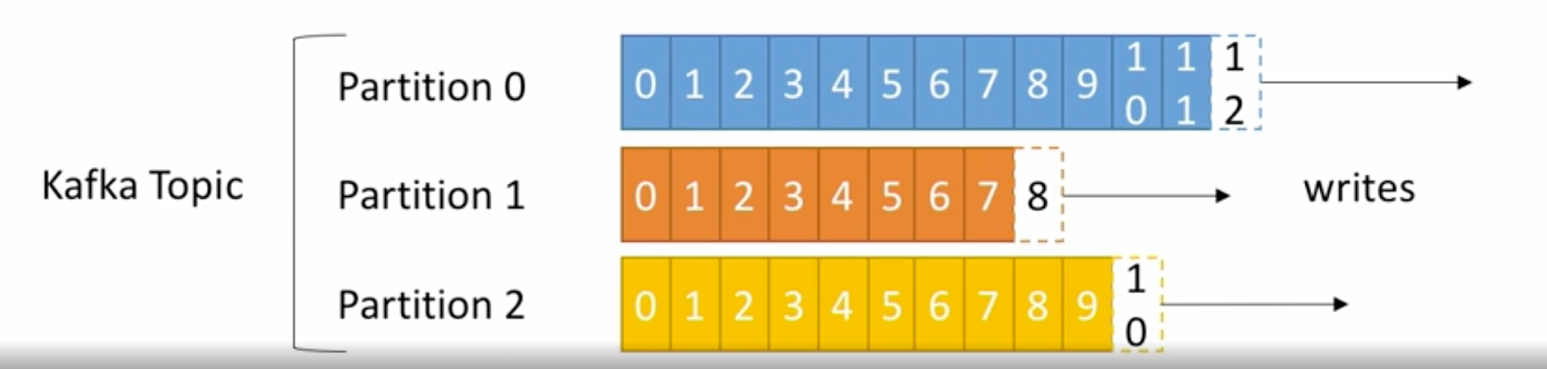
***KAFKA Notes***

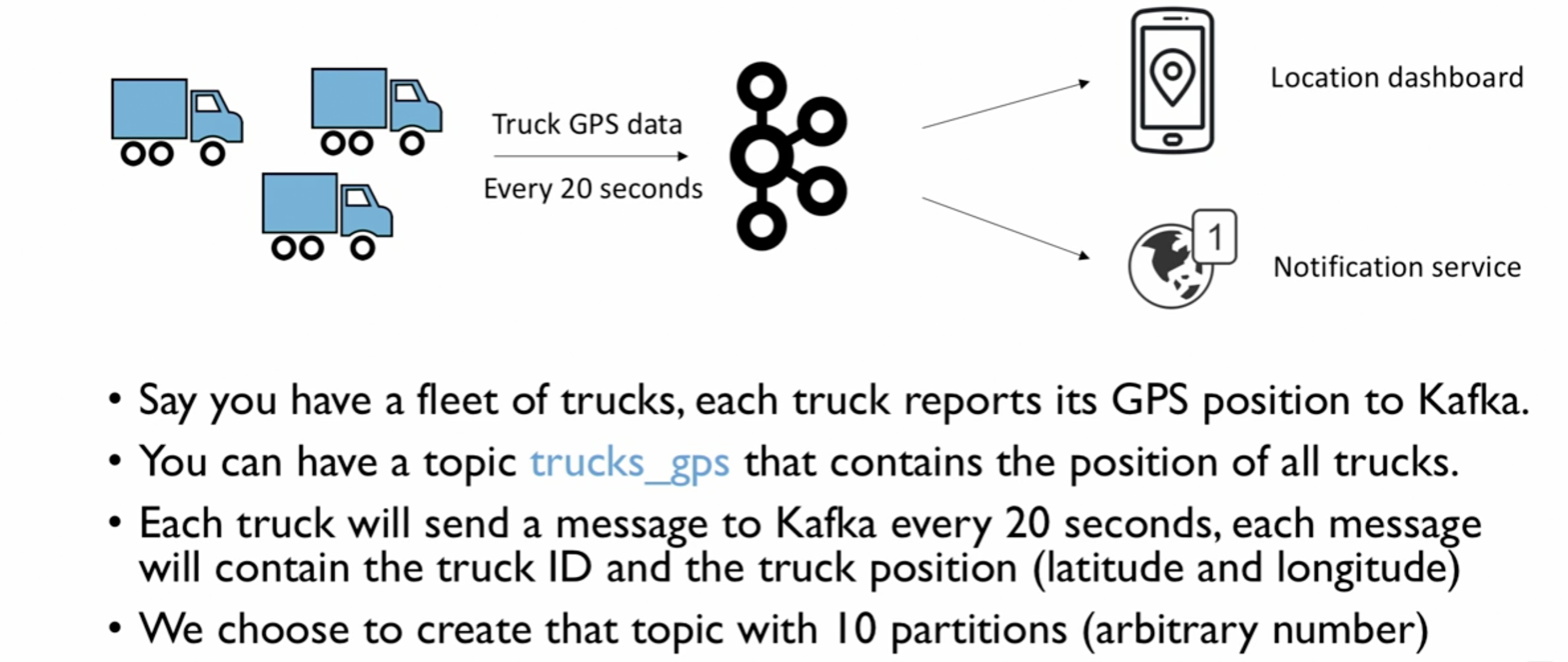
* **Topics**:
  + Topic is a particular stream of data in Kafka.
  + Similar to table in a relational database.
  + In Kafka, we can have as many topics as we want; just like there is no limitation on number of tables in a relational database.
  + A topic is uniquely identified by a name.
* **Partition**:
  + The Kafka topics are split into partitions.
  + Each partition is ordered.
  + Each message within each partition gets an incremental id called **offset**.
  + The number of partitions that you want in a topic is specified while creating a topic. The number of partitions can also change after the topic is created.
  + Each partition can have different number of messages.



**Points to remember:**

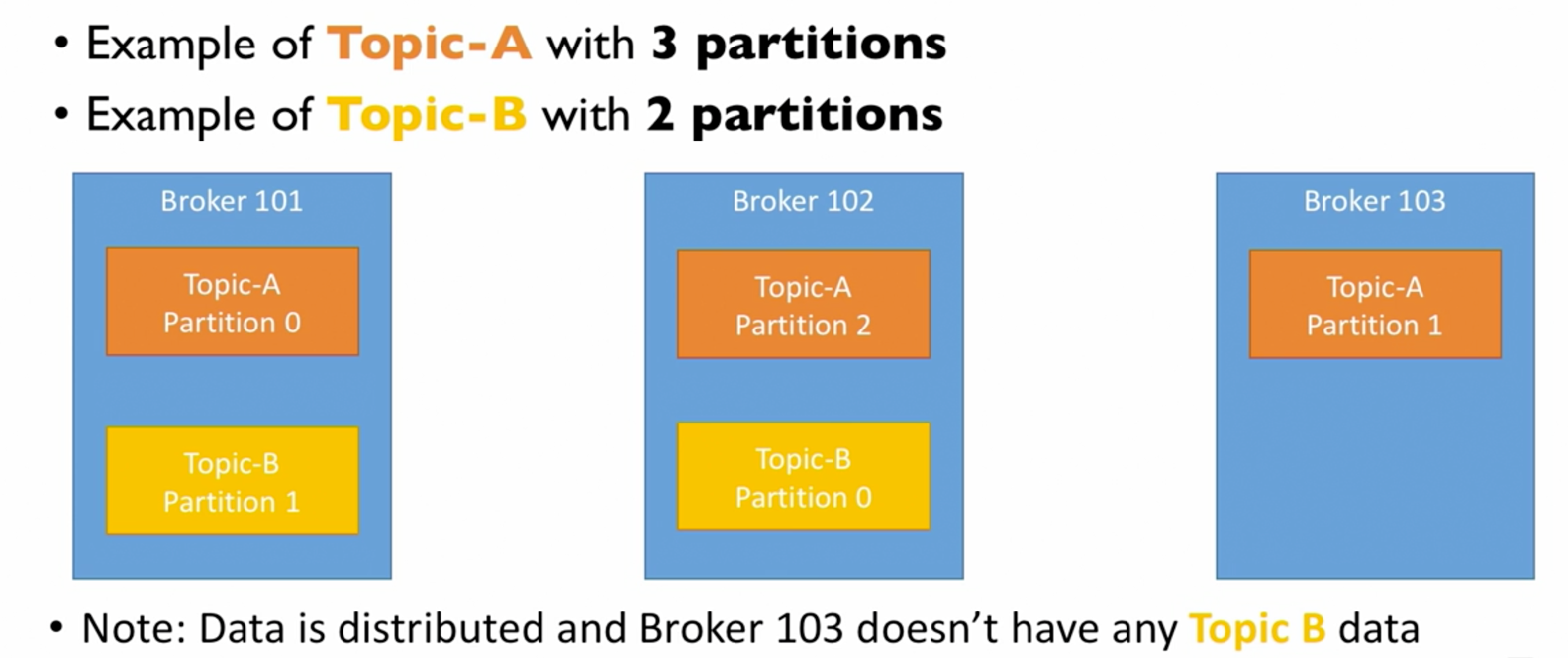
* **Offset** has a meaning only for a specific partition.
* Orderis guaranteed only within a partition; not across partitions.
* Data in kafka is kept only for a limited time; defaults to 1 week.
* Offsets keep incrementing, never decrease; even when the older messages are dropped.
* Once the data is written into a partition, it cannot be changed; immutability.
* Data is assigned randomly to a partition unless a key is provided.

**Example of topic:**



* **Kafka Cluster:**
  + Kafka is usually clustered (distributed system).
  + Kafka cluster is composed of multiple servers called **brokers.**
  + Each broker is identified by an ID (integer type).
  + Each broker contains certain topic partitions; i.e. each broker will have some data, but not all of the data for a topic.
  + After connecting to any broker (called a **bootstrap broker**), you will be connected to an entire cluster.
  + Good number to start with brokers is 3, but in production systems, it may be up to hundreds, depending on the need and complexity of the system.

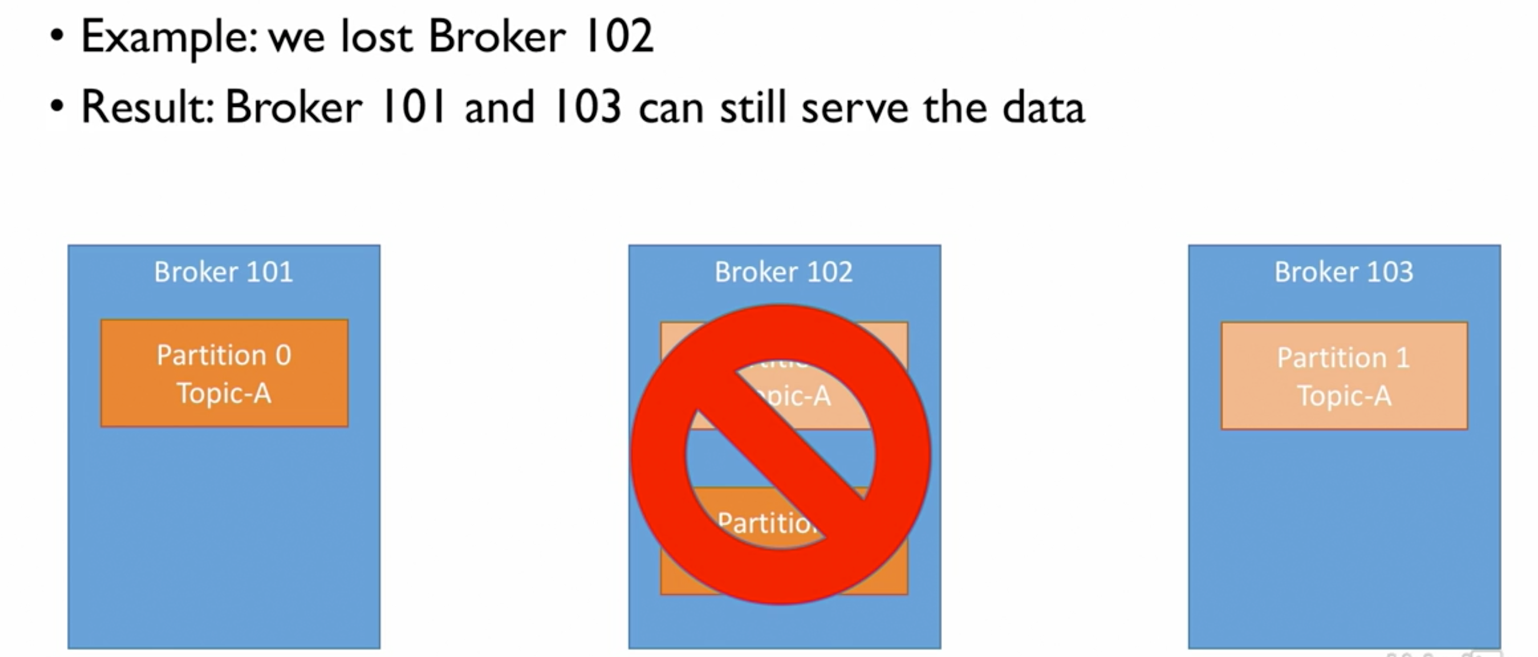
**Broker and Partitions:**

****

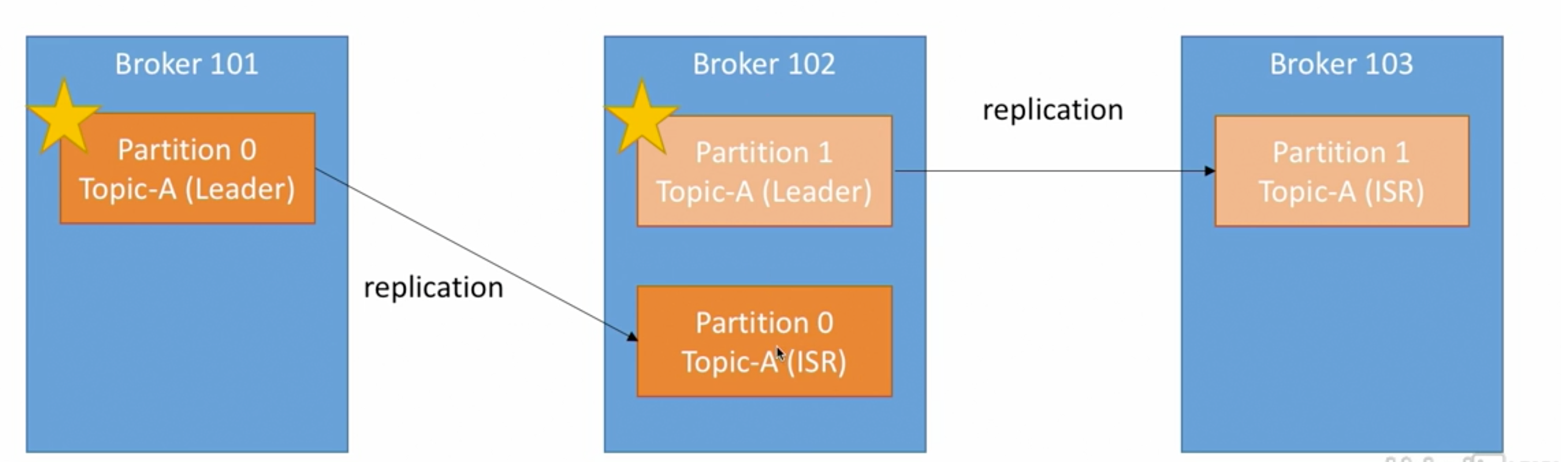
* **Replication Factor:**
  + The integral number by which the partition data is replicated on brokers is called a replication factor.
  + Since kafka is a distributed system, we need to make sure that the data integrity is maintained even if some of the brokers are down. Replication helps to achieve this.
  + The general strategy of replication is that the partition data on a broker is replicated to another broker. So when one of the brokers is down, the other broker can serve the same data.
  + The replication factor should always be > 1 (ideally between 2 & 3)
  + The replication factor has to be decided while creating a topic.
  + Replication factor of N means that there will be a total of N copies of each partition in the topic.

**For example:**

**Topic A has 2 partition and the replication factor is also 2**

****

* **Concept of leader for a partition:**
  + At any point, only one broker can be a leader for a given partition.
  + And only that leader can send and receive the data for a partition.
  + The other brokers only synchronize the data.
  + So, each partition has One leader and multiple **in-sync Replica (ISR)**



* **Producers:**