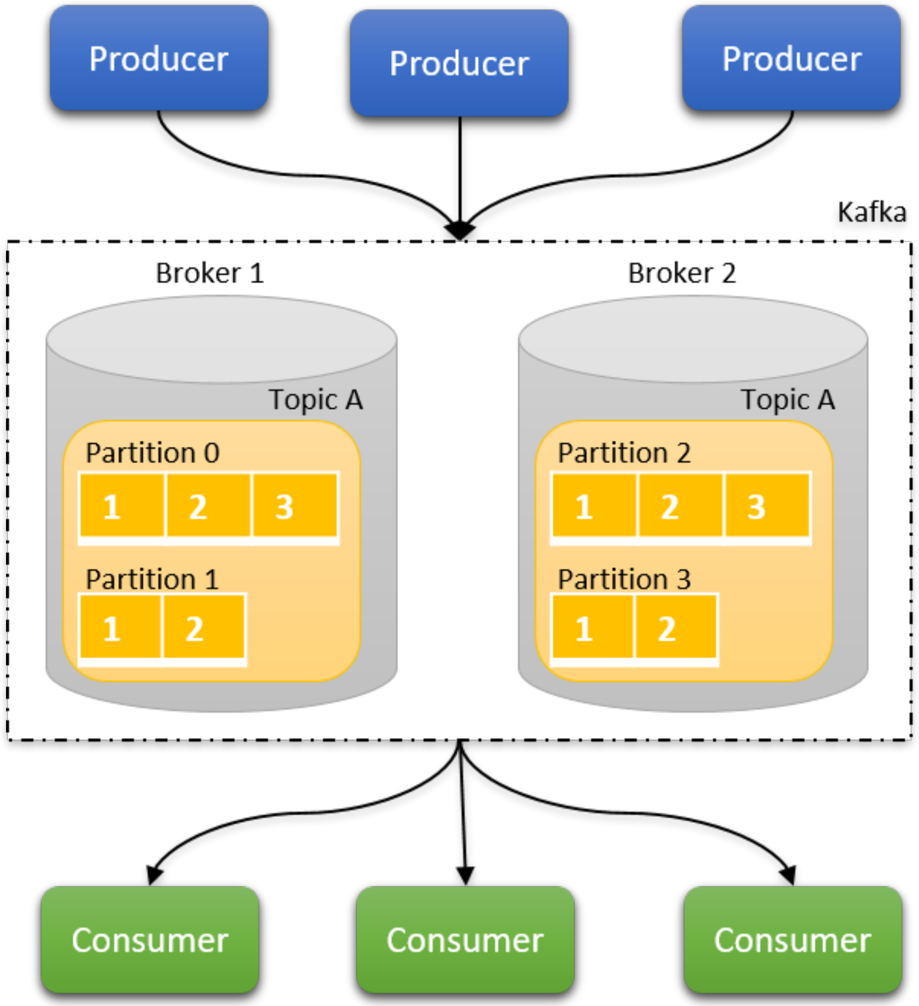
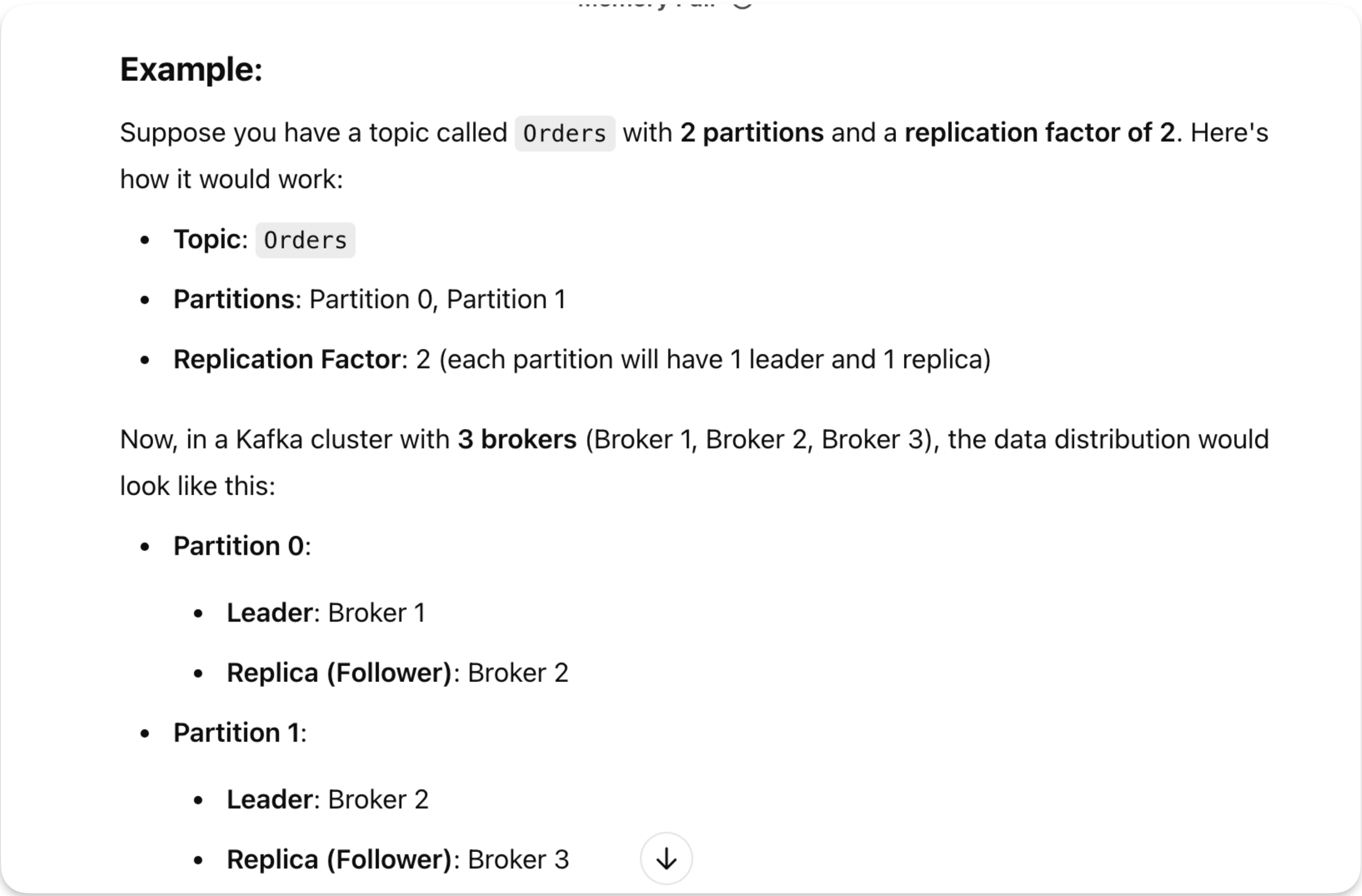




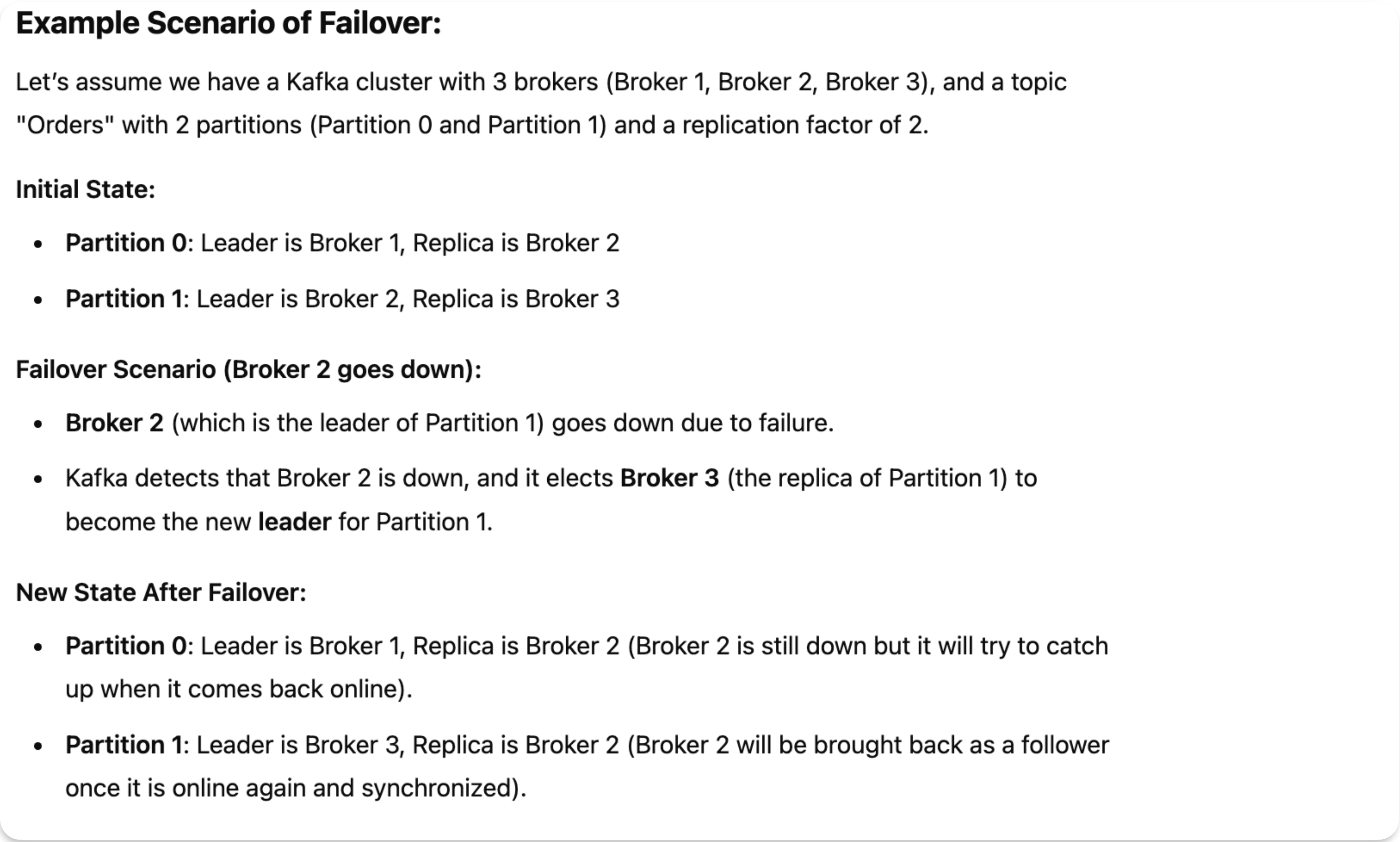
Example:

Suppose you have a topic called `Orders` with **2 partitions** and a **replication factor of 2**. Here's how it would work:

- **Topic**: `Orders`
- **Partitions**: Partition 0, Partition 1
- **Replication Factor**: 2 (each partition will have 1 leader and 1 replica)

Now, in a Kafka cluster with **3 brokers** (Broker 1, Broker 2, Broker 3), the data distribution would look like this:

- **Partition 0**:
  - **Leader**: Broker 1
  - **Replica (Follower)**: Broker 2
- **Partition 1**:
  - **Leader**: Broker 2
  - **Replica (Follower)**: Broker 3



Example Scenario of Failover:

Let's assume we have a Kafka cluster with 3 brokers (Broker 1, Broker 2, Broker 3), and a topic "Orders" with 2 partitions (Partition 0 and Partition 1) and a replication factor of 2.

Initial State:

- **Partition 0**: Leader is Broker 1, Replica is Broker 2
- **Partition 1**: Leader is Broker 2, Replica is Broker 3

Failover Scenario (Broker 2 goes down):

- **Broker 2** (which is the leader of Partition 1) goes down due to failure.
- Kafka detects that Broker 2 is down, and it elects **Broker 3** (the replica of Partition 1) to become the new **leader** for Partition 1.

New State After Failover:

- **Partition 0**: Leader is Broker 1, Replica is Broker 2 (Broker 2 is still down but it will try to catch up when it comes back online).
- **Partition 1**: Leader is Broker 3, Replica is Broker 2 (Broker 2 will be brought back as a follower once it is online again and synchronized).