

# 名詞解釋

## Kafka Producer & Kafka Consumer

Kafka 是一個分散式消息系統，它允許應用程序之間進行高效、可擴展的異步通信。在 Kafka 中，有兩個主要的元件，分別是生產者（Producer）和消費者（Consumer）。

### 1. Kafka 生產者（Producer）：

- 生產者是負責將數據消息發送到 Kafka 集群的應用程序或服務。
- 生產者將消息發布到 Kafka 主題（Topic），主題是消息的分類或主題。
- 生產者可以將消息發布到一個或多個主題，並可以指定消息的密鑰（Key）和分區（Partition）。
- 生產者將消息寫入 Kafka 集群後，它們不再關心消息的後續處理。生產者可以持續發送消息，而不必等待消費者處理。

### 2. Kafka 消費者（Consumer）：

- 消費者是負責從 Kafka 主題中讀取消息的應用程序或服務。
- 消費者可以訂閱一個或多個主題，以接收相關的消息。
- 消費者通常以消費者組（Consumer Group）的形式運行，每個消費者組可以有多个消費者實例，這些實例共同消費主題上的消息。
- Kafka 確保每條消息只被消費者組中的一個消費者實例處理，以實現消息的平衡分發。
- 消費者從指定的分區讀取消息，並可以在讀取消息後進行處理，例如數據處理、存儲或其他操作。

總結來說，Kafka 生產者用於將消息寫入 Kafka 主題，而 Kafka 消費者用於從主題中讀取消息，進行處理或儲存等操作。消費者通常以消費者組運行，以實現消息處理的分佈式和冗餘性，以確保高可用性和可擴展性。這使 Kafka 成為一個非常強大的消息系統，適用於大規模、高流量的數據流處理應用。

## Kafka Broker

Kafka Broker 是 Apache Kafka 集群中的核心元件之一，它是負責接收、存儲和分發消息的伺服器。以下是有關 Kafka Broker 的簡介：

1. 伺服器角色：Kafka Broker 是 Kafka 集群中運行 Kafka 伺服器軟體的實例。一個 Kafka 集群通常包含多個 Broker，它們共同協作以實現消息的分發和冗餘。

2. 消息存儲：Kafka Broker 負責存儲發送到 Kafka 集群的消息。這些消息存儲在 Broker 上的主題（Topics）中，每個主題都可以有多個分區（Partitions）。消息被持久性地保存，並根據配置的保留策略保留一段時間。

3. 分發消息：Kafka Broker 通過提供發送消息的接口來允許 Kafka 生產者將消息發佈到特定主題。同時，它也提供訂閱消息的接口，以供 Kafka 消費者讀取和處理消息。Broker 負責確保消息從生產者到消費者之間的有效分發。

4. 高可用性：Kafka 集群通常包含多個 Broker，這樣即使其中一個 Broker 發生故障，系統仍然可用。Kafka 通過分區（Partitions）的複製和領袖（Leader）和追隨者（Follower）的概念實現高可用性，確保即使在 Broker 失故障情況下，數據仍然可用。

5. 資料保留：Kafka Broker 配置了消息的保留策略，該策略確定消息存儲的時間長度。這允許根據需求保存數據，以滿足不同應用程序和法規的要求。

總之，Kafka Broker 是 Kafka 集群的核心組件，負責消息的接收、存儲和分發。它確保了高可用性、可擴展性和持久性，使 Kafka 成為一個流行的選擇，用於處理實時數據流、日誌記錄、事件驅動架構等各種應用。

## ZooKeeper

Apache ZooKeeper（通常簡稱ZooKeeper）是一個開源的分散式協調服務，用於協助分散式應用程序協調和管理數據。ZooKeeper 的主要目標是提供一個高可用性的環境，用於解決分散式系統中的一些共同問題，例如配置管理、分布式鎖、協調和分發信息等。以下是有關ZooKeeper的關鍵概念和功能的簡要說明：

1. **分布式系統協調**：ZooKeeper旨在為分散式應用程序提供一個共享的協調和控制基礎設施。它可以幫助不同部分的應用程序在分散式環境中協同工作。
2. **分布式配置管理**：ZooKeeper可以用來存儲和管理分散式系統的配置信息，例如主機名、端口、連接字符串等。這使得系統的配置變更變得容易，並且能夠在運行時進行動態更新。
3. **分布式鎖**：ZooKeeper提供了一種機制來實現分布式鎖，以協調多個進程之間的操作順序，防止競爭條件，確保操作的原子性。
4. **分布式協議**：ZooKeeper可以用於實現分布式協議，確保多個節點之間的相對順序，以協調事件的發生。
5. **數據一致性**：ZooKeeper保證了分散式環境中的數據一致性，這對於需要強一致性的應用程序非常重要。
6. **高可用性**：ZooKeeper自身的架構具有高可用性，並且能夠容忍部分節點的故障。它使用選主（Leader）和追隨者（Follower）的模型，確保即使在部分節點故障的情況下，系統仍能正常運行。
7. **輕量和快速**：ZooKeeper是一個輕量級的服務，並且可以快速執行。它設計用於高吞吐量和低延遲的操作。
8. **開發者友好**：ZooKeeper提供了多種編程語言的客戶端庫，使開發者可以方便地與ZooKeeper集成。

ZooKeeper通常用於協調分散式系統的操作，例如Apache Kafka、Apache HBase、分布式數據庫等。它提供了一個可靠的基礎設

施，以解決分散式系統中的一些關鍵問題，並確保各種應用程序能夠協同工作並達到高可用性和一致性。

## Kafka Connect

Kafka Connect是用來連接Kafka和外部資料系統的工具，它提供了資料匯入和匯出的能力。以下是Kafka Connect的主要功能：

1. **資料匯入：** Kafka Connect 可用於從外部資料來源（如資料庫、日誌檔案、訊息佇列等）將資料匯入到 Kafka 主題。它提供了各種連接器，方便輕鬆將不同資料來源的資料傳輸到 Kafka。
2. **資料匯出：** Kafka Connect 也可用於將 Kafka 主題中的資料匯出至外部資料系統。這使得您可以將 Kafka 作為資料整合的中間層，將資料傳遞到多個目的地。
3. **連接器管理：** Kafka Connect提供了可設定的連接器，用於定義資料來源和目標之間的資料流。這些連接器可以配置、管理和監視，以確保資料的可靠傳輸。
4. **多元化和可擴展性：** Kafka Connect是多元化的，可以在多個工作節點上運行，以實現高吞吐量和可擴展性。

在Kafka結構中，SchemaRegistry和KafkaConnect通常一起使用，以確保資料的有效傳輸系統和一致性。KafkaConnect用於連接不同的資料來源和目標，而SchemaRegistry用於管理資料模式，以確保資料的一致這兩個元件協同工作，有助於建立強大的資料流處理應用程式。

## Schema Registry

SchemaRegistry是Kafka生態系中的關鍵元件，用於管理Avro、JSONSchema等訊息序列化及反序列化的模式。以下是SchemaRegistry的主要功能：

1. **模式管理：** SchemaRegistry用於儲存和管理不同訊息主題的資料模式。它允許Kafka生產者和消費者使用統一的資料模式，確保資料的一致性和可互通性。
2. **模式註冊：** 生產者在將訊息傳送到Kafka時，將訊息的資料模式註冊到Schema註冊表。這使得消費者可以取得訊息的模式，以便正確解析和反序列化資料。
3. **模式演進：** 當資料模式需要變更時，模式註冊表允許進行模式演進，以確保新的和舊的訊息都能夠被處理。這有助於系統的升級和維護。
4. **資料驗證：** SchemaRegistry可以驗證傳送到Kafka主題的訊息是否與註冊的資料模式相容，從而保證資料的一致性和一致性。

🕒 修訂版本 #3

★ 由 treeman 建立於 20 🕒🕒🕒 2023 15:51:13

✍ 由 treeman 更新於 26 🕒@🕒🕒 2024 18:15:31