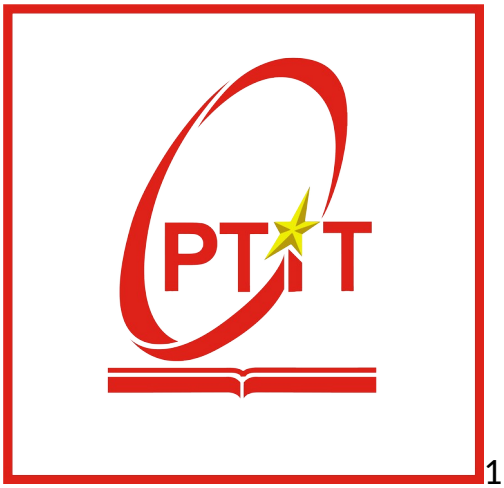
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG



1

Báo cáo hàng tuần

Môn học: Thực tập cơ sở

Giảng viên: Kim Ngọc Bách

Họ và tên: Nguyễn Hữu Phúc

Mã SV: B22DCAT224

Lớp: E22CQCN04-B

Báo cáo tuần 8

Tuần này, em nhận ra rằng là mỗi lần truy vấn dữ liệu trên MongoDB, em sẽ mất một nửa Gb Ram ở trên máy, cụ thể là hơn 7Gb Ram. Điều đó sẽ khiến cho em chạy thêm Kafka nữa thì có khả năng máy tính sẽ bị tràn Ram. Sau khi tìm hiểu thì em biết rằng ở MongoDB phiên bản mới nhất, cụ thể phiên bản em đang dùng là 8.0.6, mặc định MongoDB sẽ chiếm một nửa số Gb Ram trong máy. Vì vậy em đã xóa MongoDB trên docker và cài lại. Cùng với đó là học về kafka

I. Cài lại mongoDB trên docker

1. Giới hạn Ram Container

```
docker run -d \
  --name mongodb \
  -p 27017:27017 \
  -v mongo_data:/data/db \
  --memory="5g" \
  -e MONGO_INITDB_ROOT_USERNAME=nguyenphuc \
  -e MONGO_INITDB_ROOT_PASSWORD=123456 \
  mongo:8.0.6 \
  mongod --wiredTigerCacheSizeGB=4
```

2. Copy file summary.bson vào container:

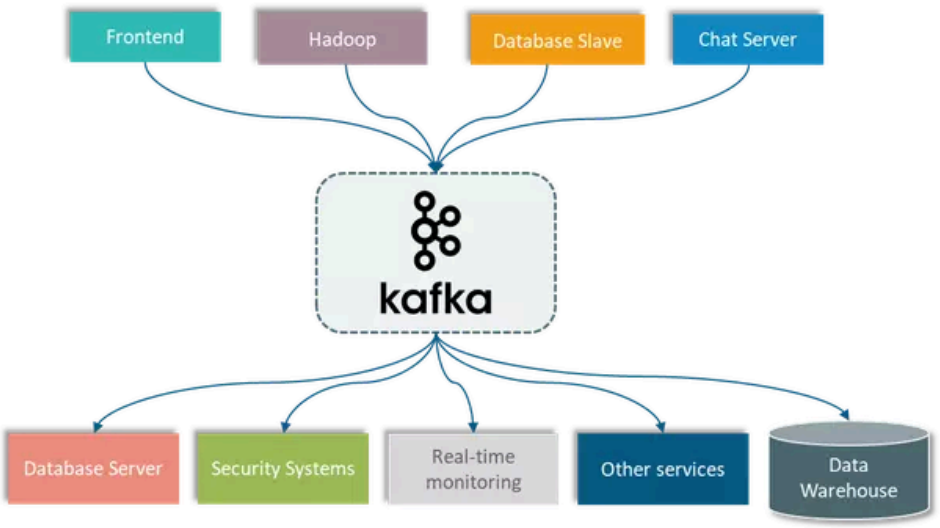
```
docker cp /home/nguyenphuc/Documents/GlamiraUserFlowInsightsProject/
glamira_ubl_oct2019_nov2019/dump/countly/summary.bson
mongodb:/tmp/summary.bson
```

3. Tải data vào MongoDB

```
docker exec mongodb mongorestore --db Glamira --collection summary
--username nguyenphuc --password 123456 --authenticationDatabase admin
/tmp/summary.bson
```

II. Tìm hiểu về cấu trúc kafka

1. **Kafka là gì**

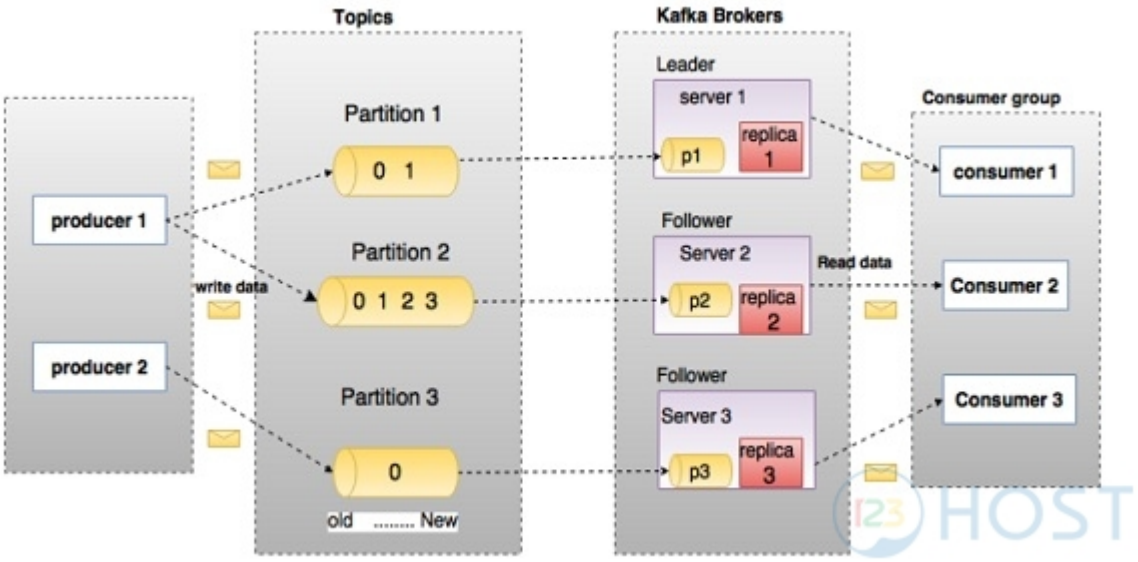


Kafka là một nền tảng streaming dữ liệu phân tán, xử lí lượng lớn dữ liệu theo thời gian thực. Nó hoạt động như một hệ thống message queue hoặc distributed log, cho phép ứng dụng gửi, lưu trữ và xử lí dữ liệu một cách hiệu quả.

Lợi ích của Kafka:

- Reliability: Kafka được phân tán (distribute), phân chia (partition), đồng bộ (replicate) và chịu lỗi (fault tolerance).
- Scalability: Kafka có thể mở rộng dễ dàng mà không bị downtime.
- Durability: Thông điệp được lưu trữ trên disk trong thời gian dài.
- Performance: Kafka có thông lượng cao trong cả publish-subscribe thông điệp. Nó duy trì được độ ổn định kể cả khi lưu trữ đến hàng TeraByte thông điệp.

2. Cấu trúc của Apache Kafka

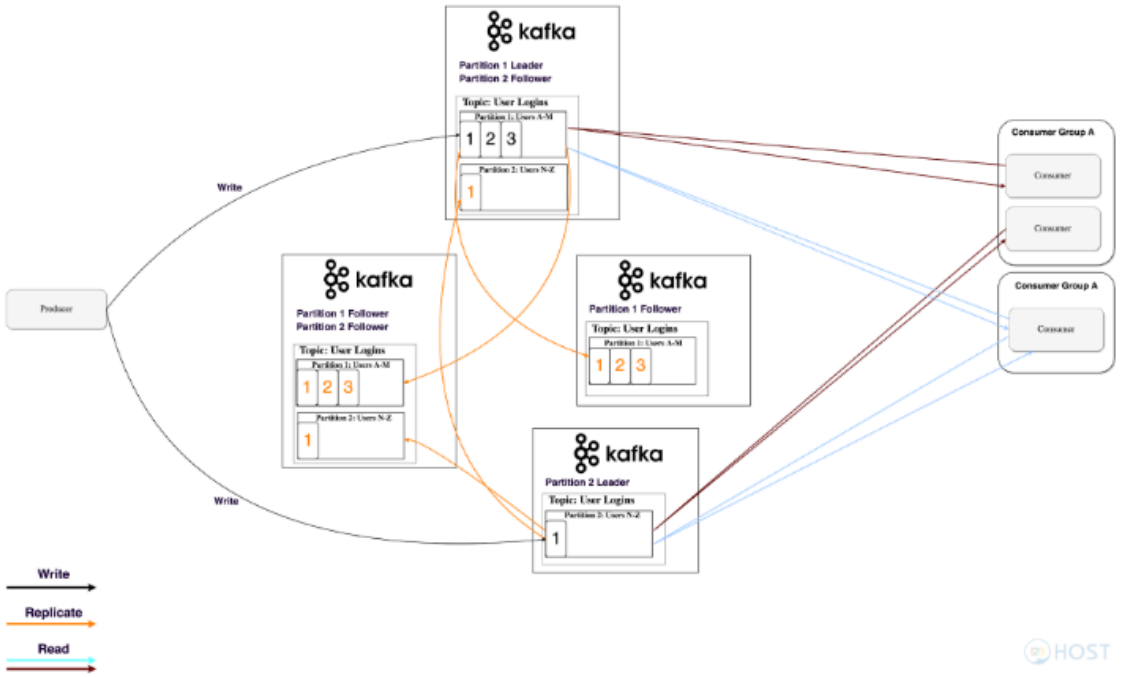


Các thuật ngữ:

- Topic
  - Kafka gom các thông điệp cùng loại lại thành một danh mục gọi là topic.
- Partition:
  - Mỗi topic được chia thành nhiều partition để phân tán dữ liệu và tăng khả năng xử lí song song. Partition là đơn vị lưu trữ vật lý của topic trên các broker
  - Ý nghĩa: Partition giúp Kafka mở rộng quy mô bằng cách phân phối dữ liệu trên nhiều brokers và cho phép nhiều consumer đọc dữ liệu song song
- Partition offset:
  - Mỗi thông điệp được lưu trên partition sẽ được gán với một giá trị theo thứ tự tăng dần gọi là offset

    - Giống như index của mảng
- Replicas of Partition
    - Là bản sao của partition, được lưu trữ trên các broker khác nhau. Chỉ được dùng để tránh mất dữ liệu, không dùng để đọc bản ghi
- Broker
    - Là một Kafka server
    - Nếu ra chạy đồng thời nhiều broker, thì ta sẽ gọi đó là một Kafka cluster
    - Vai trò:
        - Lưu trữ partition của các topic
        - Xử lí message từ producers và consumers
        - Phối hợp với các broker khác thông qua Zookeeper
    - Ví dụ: Nếu cluster có 3 broker:
        - Broker 1 lưu partition 1 và 2
        - Broker 2 lưu partition 3
        - Broker 3 lưu replica của par 1,2,3
- Producers:
    - Ứng dụng truyền messages đến cho Kaka Server (Broker, node) thông qua các Topic
    - Producer truyền messages đến Broker thì Broker sẽ đưa messages đó đến cuối hàng đợi của một partition
    - Producer có thể chọn gửi cho partition mong muốn
- Consumers
    - Ứng dụng đọc messages từ Broker thông qua các Topic
    - Có thể đọc từ 1 hay nhiều topic
    - Theo dõi offset để biết vị trí đọc hiện tại
    - Nhiều consumer có thể cùng đọc từ một topic (consumer group)
- Leader
    - Là broker chịu trách nhiệm chính cho việc đọc/ghi dữ liệu của một partition
- Follower
    - Là các replica của partition, lưu trữ bản sao và đồng bộ với leader
    - Nếu leader bị lỗi, Zookeeper sẽ chọn một follower để trở thành leader mới
- Zookeeper: được dùng để quản lí và điều phối các broker trong cluster
    - Lưu trữ metadata
    - Phát hiện lỗi broker và thông báo cho producer/consumer
    - Thực hiện leader election khi leader của một partition bị lỗi

3. Cách thức hoạt động



a. Producer gửi messages
- Gửi đến một topic trong Kafka, topic này được chia thành nhiều partition để dễ xử lí
- Mỗi partition có một leader (broker chịu trách nhiệm chính) và các follower (các broker khác chứa bản sao của partition)

b. Partition và thứ tự messages
- Mỗi partition giống như một hàng đợi, lưu trữ messages theo thứ tự đến (dựa trên offset)
- Kafka đảm bảo rằng các messages trong cùng một partition được xử lí theo đúng thứ tự

c. Consumer đọc messages
- Đọc messages từ topic để xử lí, đọc những gì nó muốn
- Theo dõi offset để biết mình đọc đến đâu trong partition
- Kafka lưu trữ messages trong thời gian dài. Consumer có thể đọc lại dữ liệu cũ nếu cần

d. Consumer Group và vấn đề trùng lặp/thứ tự
- Một consumer có thể đọc nhiều partition
- Nhưng một partition chỉ đọc được bởi một consumer duy nhất thuộc một group

e. Vai trò của Zookeeper trong việc tìm Leader (quản lí thông tin)
- Lưu trữ metadata của cluster
- Khi producer muốn gửi messages, nó hỏi một broker bất kì
- Broker này đã giao tiếp với Zookeeper trước đó, nên nó biết broker nào là leader của partition cần gửi
- Broker trả lời producer, và producer gửi messages đến đúng leader