Họ và tên: Đào Vũ Bảo Quỳnh

Lớp: 12_ĐH_CNTT5

Mô phỏng truy vấn phân tán

Hệ quản trị NoSQL: Apache Cassandra

1. Bài toán mô phỏng: Hệ thống theo dõi lịch sử mượn - trả sách điện tử (E-book) của một nền tảng thư viện số

- Đặc điểm dữ liệu:

+ Một người dùng có thể mượn nhiều sách

+ Một cuốn sách được đã mượn bởi nhiều người

+ Truy vấn thường xuyên theo: Người dùng và khoảng thời gian

+ Dữ liệu phát sinh liên tục, yêu cầu mở rộng cao

2. Thiết kế bảng dữ liệu:

```
CREATE TABLE borrow_history_by_user (
    user_id text,
    ebook_id text,
    borrow_time timestamp,
    return_time timestamp,
    device_type text,
    PRIMARY KEY ((user_id), borrow_time, ebook_id)
);
```

user_id: Partition Key

borrow_time, ebook_id: Clustering Key

Mỗi partition lưu toàn bộ lịch sử mượn sách của một người dùng

Dữ liệu được sắp xếp theo thời gian mượn

3. Cấu hình cụm Cassandra mô phỏng

VD: Hệ thống triển khai 5 node

Node N1: Token 0 - 20%

Node N2: Token 20 - 40%

Node N3: Token 40 - 60%

Node N4: Token 60 - 80%

Node N5: Token 80 - 100%

Replication Factor = 3

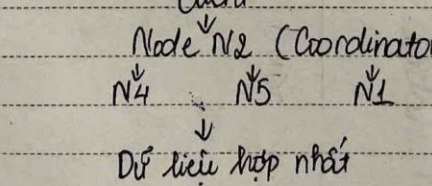
Mỗi bản ghi tồn tại trên 3 node khác nhau

KI.ONG

4. Mô phỏng lưu trữ dữ liệu phân tán
- Dữ liệu phát sinh: Người dùng U001 mượn E-book EB102 lúc 2025-10-01 20:30
- Quá trình:
+ Cassandra hash user_id = U001
+ Giá trị hash thuộc vùng token của Node N4
+ Dữ liệu được ghi đồng thời lên: Node N1, Node N4 (Primary replica), Node N5
U001 -> N4 -> N5 -> N1
Nếu 1 node bất kỳ trong 3 node trên gặp sự cố, dữ liệu vẫn được đảm bảo an toàn.
5. Mô phỏng truy vấn phân tán
- Truy vấn: Lấy lịch sử mượn sách của người dùng U001 trong tháng 01/2026

```
SELECT ebook_id, borrow_time, return_time
FROM borrow_history_by_user
WHERE user_id = 'U001'
AND ~~brow~~ borrow_time >= '2026-01-01'
AND borrow_time <= '2026-01-31';
```

- Cách Cassandra xử lý truy vấn:
+ Client gửi truy vấn đến Node N2 (Node bất kỳ)
+ Node N2 trở thành Coordinator
+ Node N2 xác định Partition U001 nằm trên N4, N5, N1
+ Node N2 gửi truy vấn song song tới 3 node replica
+ So sánh timestamp (Read Repair nếu cần)
+ Tổng hợp kết quả => trả về client



KI.ONG

6. Mô phỏng tình huống lỗi
- TH: Node N4 bị ngắt kết nối
- Hệ thống phản ứng:
+ Coordinator (N2) vẫn nhận dữ liệu từ Node N5, Node N1.
+ Không gián đoạn truy vấn
+ Khi N4 hoạt động lại → Read Repair tự động đồng bộ dữ liệu
7. Đánh giá mô phỏng.
- Ưu điểm:
+ Truy vấn nhanh theo người dùng
+ Phân tán dữ liệu đều
+ Không cần JOIN
+ Phù hợp dữ liệu tăng liên tục
- Nhược điểm:
+ Không truy vấn tốt theo ebook_id
+ Phải thiết kế bảng theo truy vấn trước
8. Kết luận
- Mô phỏng trên cho thấy Apache Cassandra xử lý truy vấn phân tán hiệu quả trong hệ thống thư viện số. Nhờ kiến trúc peer-to-peer, replication và coordinator node, hệ thống vẫn hoạt động ổn định ngay cả khi một node gặp sự cố.
9. Kiến trúc của Apache Cassandra
- Có kiến trúc phân tán ngang hàng (Peer-to-peer Distributed Architecture), không có master, đây là điểm cốt lõi giúp Cassandra mở rộng và chịu lỗi rất tốt.

Client
Bất kỳ Node nào (Coordinator)
Các Node khác trong Cluster (Replica Node)

- Cluster: Tập hợp nhiều Node Cassandra
- Node: Một máy chủ Cassandra độc lập
- Không có Master - Slave
- Tất cả node đều bình đẳng

KI.ONG

- Thành phần chính trong kiến trúc Cassandra:
+ Node: Mỗi node lưu trữ dữ liệu, nhận request từ client, có thể làm Coordinator. Không node nào làm trung tâm
+ Coordinator Node: Là node nhận truy vấn từ Client, chỉ đóng vai trò điều phối, không phải master. Nhiệm vụ xác định node chứa dữ liệu, gửi truy vấn đến các replica, tổng hợp kết quả trả về client. Thay đổi linh hoạt không cố định.
+ Partitioning (Phân vùng dữ liệu): Cassandra sử dụng Consistent Hashing. Partiton Key → Hash → token. Token quyết định dữ liệu nằm trên node nào.
+ Replication (Nhân bản dữ liệu): Mỗi dữ liệu được lưu trên nhiều node. Cấu hình bằng Replication Factor (RF)
VD: RF = 3 → Mỗi bản ghi lưu trên 3 node
+ Gossip Protocol: Các node tự động trao đổi trạng thái phát hiện node lỗi. Không cần server quản lý trung tâm
+ Cơ chế ghi dữ liệu (White Path): Client gửi request → Coordinator nhận request → Commit Log (Đảm bảo không mất dữ liệu) → MemTable (Bộ nhớ) → Khi đầy flush xuống SSTable.
+ Cơ chế đọc dữ liệu (Read Path): Coordinator gửi request đến các p replica → Đọc từ memtable, sstable → So sánh Timestamp → Trả dữ liệu mới nhất (Nếu lệch dữ liệu có Read Repair phát hiện).
- Kiến trúc Cassandra:
Cluster
↳ Keyspace
↳ Table
↳ Partition
↳ Row
↳ Column.
10. Tại sao Apache Cassandra được gọi là NoSQL nhưng dùng SQL?
- Vì Apache Cassandra có kiến trúc phân tán ngang hàng (peer-to-peer), không có node trung tâm. Mỗi node đều có vai

trò như nhau và dữ liệu được phân vùng bằng cơ chế consistent hashing, đồng thời được nhân bản trên nhiều node để ~~ban~~ đảm bảo tính sẵn sàng và chịu lỗi.

- Vì không tuân theo mô hình CSDL quan hệ: không hỗ trợ JOIN, không có ACID đầy đủ và không dùng khoá ngoại nên Cassandra được xếp vào NoSQL.

- Cassandra sử dụng CQL (Cassandra Query Language) - một ngôn ngữ có cú pháp giống SQL giúp người dùng dễ tiếp cận. Tuy nhiên Cassandra chỉ giống SQL về hình thức, còn bản chất truy vấn và kiến trúc vẫn là NoSQL.

KI.ONG