HỆ QUẢN TRỊ CSDL CASSANDRA

Biên soạn: Nguyễn Văn Lễ

Nội dung

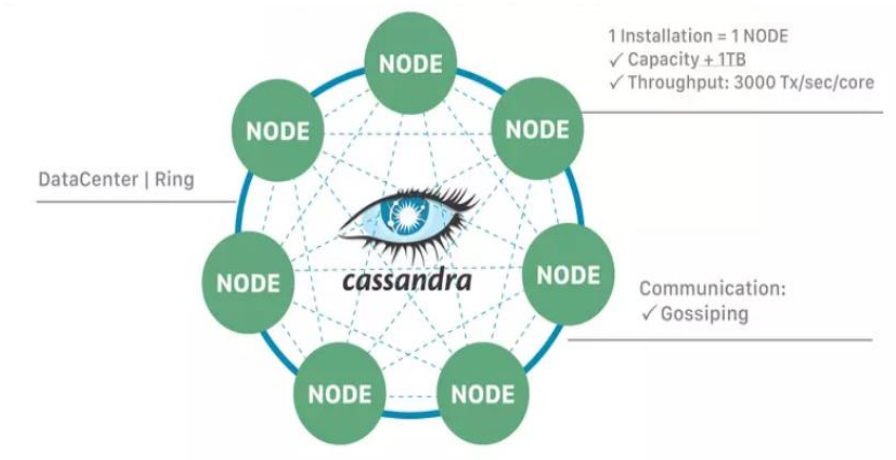
- Giới thiệu Cassandra
- Thiết kế mô hình dữ liệu
- Các thao tác cơ bản
- Truy vấn dữ liệu

Giới thiệu Cassandra

- Apache Cassandra (Cassandra) là một hệ cơ sở dữ liệu NoSQL phân tán, mã nguồn mở, sử dụng mô hình Column-Family, được thiết kế để xử lý lượng dữ liệu lớn trên nhiều máy chủ.
- Cassandra cung cấp hỗ trợ cho các clusters trên data centers.
- Cassandra được thiết kế để triển khai kết hợp các kỹ thuật lưu trữ và sao chép phân tán.

Các tổ chức sử dụng Cassandra

Giới thiệu Cassandra



Đặc điểm Cassandra

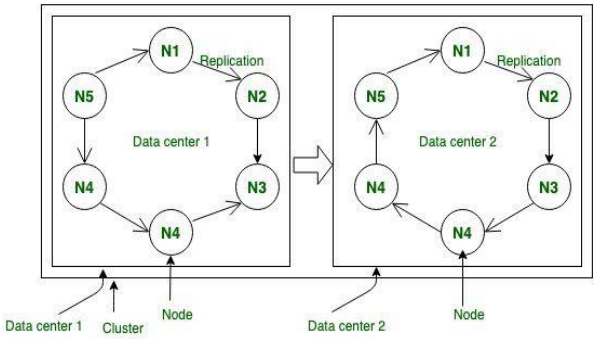
- Một số đặc tính nổi bật của Cassandra
- Scalability: Khả năng mở rộng cao
- Availability: Tính sẵn sàng cao
- Reliability: Độ tin cậy cao
- Distributed: Tính phân tán
- Masterless: Không sử dụng cơ chế Master-Slave mà các node đều đóng vai trò như nhau trong xử lý dữ liệu (P2P).
- Cassandra có khả năng mở rộng đàn hồi. Cluster có thể dễ dàng thu nhỏ hoặc mở rộng.

Đặc điểm Cassandra

- Cassandra là một CSDL NoSQL tương tự như MongoDB được phát triển bởi Facebook.
- Cassandra là hệ cơ sở dữ liệu phân tán, dữ liệu được lưu trữ trên nhiều node của nhiều máy khác nhau, theo cơ chế P2P.
- Hiệu năng xử lý của hệ thống cũng tăng theo số node.
- Ngôn ngữ phát triển Cassandra là Java.

Đặc điểm Cassandra

- Datacenter: Là một tập hợp các node. Mỗi node sẽ xử lý một phần của dữ liệu, tập các node xử lý đủ 100% dữ liệu gọi là một datacenter.
- Cluster: Là thành phần mà có thể chứa một hoặc nhiều datacenter.



Đặc điểm Cassandra

- Commitlog: Là cơ chế để xử lý khôi phục sự cố (crash-recovery mechanism). Mọi thao tác ghi dữ liệu (write operation) đều được ghi vào commitlog.
- Memtable: Là một kiến trúc dữ liệu lưu trên bộ nhớ (In-memory data), là nơi chứa thông tin dữ liệu được update trên memory mà chưa được update xuống đĩa.

Kiểu dữ liệu trong Cassandra

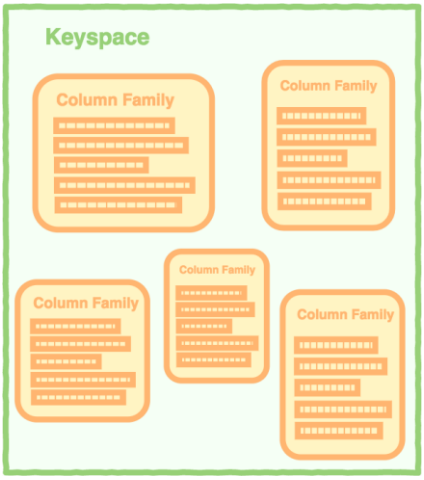
Kiểu dữ liệu	Mô tả
ascii	Biểu diễn cho một chuỗi ký tự ASCII
bigint	Đại diện cho số nguyên có dấu dài 64-bit
blob	Lưu trữ các byte tùy ý
boolean	Giá trị true hoặc false
counter	Đại diện một số nguyên dài 64-bit, giá trị của cột này chỉ có hai hoạt động trên cột này, tăng và giảm.
date	Lưu trữ giá trị ngày, không có giờ
decimal	Giá trị thập phân
double	Lưu trữ một giá trị dấu chấm động dài 64-bit.
float	Lưu trữ một giá trị dấu chấm động dài 32-bit.
inet	Biểu diễn cho một chuỗi địa chỉ IP trong định dạng của IPv4 hoặc IPv6.
int	Lưu trữ số nguyên có dấu dài 32-bit
smallint	Biểu diễn số nguyên16-bit
text	Biểu diễn chuỗi UTF8
time	Chuỗi thời gian có dạng 01:02:03.123
timestamp	Lưu trữ dữ liệu ngày và giờ với độ chính xác mili giây
timeuuid	Lưu trữ chuỗi UUID phiên bản 1
tinyint	Số nguyên 1 byte 8 bit
uuid	Lưu trữ chuỗi UUID chuẩn
varchar	Lưu trữ chuỗi UTF8 tương tự kiểu Text
varint	Số nguyên với độ chính xác tùy ý

Một số khái niệm

- Keyspace
- Table
- Primary key
- Partition key
- Clustering key

Keyspace

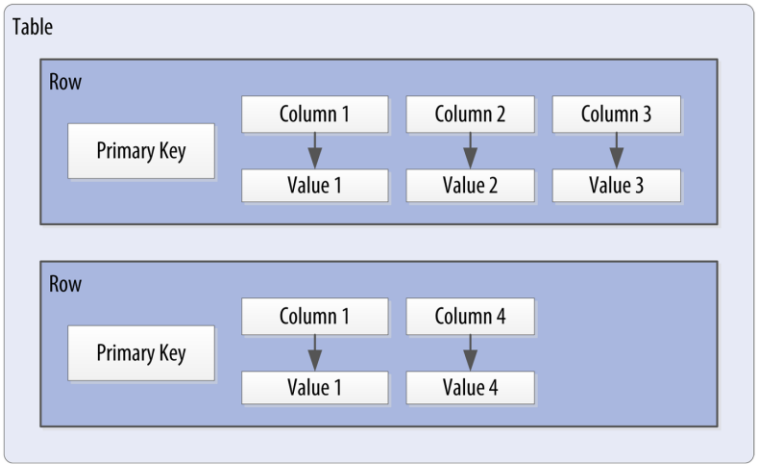
- Là đơn vị chính để tổ chức dữ liệu trong Cassandra tương đương với database trong hệ quản trị cơ sở dữ liệu quan hệ.
- Mỗi Keyspace chứa các bảng gọi là Column Family.



Table

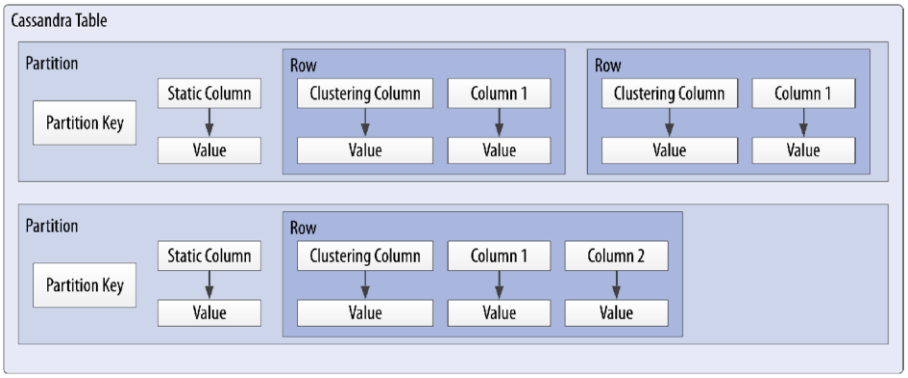
- Bảng là đối tượng chứa tập hợp các dòng có thứ tự.
- Mỗi dòng gồm tập hợp các cột được sắp thứ tự và được xác định bởi khóa phân vùng (Partition key).
- Mỗi dòng được gán cho một node trong cụm Cassandra.

Cấu trúc bảng



Cấu trúc bảng (tt)

- Static column là cột tĩnh sẽ có cùng giá trị cho tất cả các dòng



Ví dụ table

year	name	id	runtime
2015	Insurgent	1	119
2014	Interstellar	2	98
2014	Mockingjay	3	122

Các dạng khóa

- PRIMARY KEY là khóa chính của bảng bao gồm các khóa:
  - Partition key (khóa phân vùng) chịu trách nhiệm phân bổ dữ liệu qua các node (server).
  - Clustering key (khóa phân cụm) chịu trách nhiệm sắp xếp dữ liệu theo Partition key.
- Composite/Compound key là thuật ngữ chỉ key được tạo nên bởi nhiều hơn một field.

Ví dụ khóa

Primary key: mapn,mahg

Partition key: mapn

Cluster key: mahg

mapn	mahg	dongia	soluong
pn02	h001	3000	20
pn01	h001	3000	20
pn01	h002	2000	10
pn01	h003	2000	10

Một số thao tác cơ bản

- Tạo Keyspace
- Tạo bảng
- Cập nhật dữ liệu
- Truy vấn dữ liệu

Tạo Keyspace

- Cấu trúc:

```
CREATE KEYSPACE keyspace_name
WITH replication = {
'class': 'SimpleStrategy',
'replication_factor': replication_factor
};
```

– Class: *SimpleStrategy* sử dụng cho môi trường phát triển hoặc môi trường đơn giản,

NetworkTopologyStrategy thích hợp hơn cho môi trường dữ liệu lớn với cấu hình nhiều nút.

– Replication_factor: Số lượng bản sao dữ liệu cần lưu.

Ví dụ tạo Keyspace

- Tạo keyspace có tên qlsv với class SimpleStrategy và replication factor là 3:

```
CREATE KEYSPACE qlsv
WITH replication = {
  'class': 'SimpleStrategy',
  'replication_factor': 3
};
```


Tạo bảng

Cấu trúc:

```
CREATE TABLE table_name
(
    column1 datatype1,
    column2 datatype2,
    ...
    PRIMARY
    KEY(partition_key,clustering_column1,
    clustering_column2,…)
)   [WITH   CLUSTERING   ORDER   BY(column_i
ASC/DESC)];
```

Ví dụ tạo bảng

Cấu trúc:

```
CREATE TABLE nhaphang
(
  mapn text,
  mahang text,
  dongia float,
  soluong int,
  primary key(mapn,mahang)
) WITH CLUSTERING ORDER BY(mahang DESC);
```


Nhập dữ liệu

Cấu trúc:

```
INSERT INTO <table_name>(<column1>,<column2>,
<column3>,...)
VALUES(<value1>, <value2>, <value3>, ...);
```

Ví dụ:

```
INSERT INTO nhaphang(mapn,mahang,dongia,soluong)
VALUES ('pn001','h001',3000,20)
```

Nhập nhiều dòng dữ liệu

Sử dụng BEGIN BATCH…APPLY BATCH

Ví dụ:

```
BEGIN BATCH
  INSERT INTO nhaphang(mapn,mahang,dongia,soluong)
  VALUES('pn001','h002',4000,10);
 INSERT INTO nhaphang(mapn,mahang,dongia,soluong)
  VALUES('pn001','h003',5000,15);
APPLY BATCH
```


Xóa dữ liệu

Cấu trúc:

```
DELETE FROM <table_name>
WHERE<primary_key_column>=<primary_key_value>;
```

Ví dụ:

```
DELETE FROM nhaphang WHERE mapn='pn001' and
mahang='h002'
```

Sửa dữ liệu

Cấu trúc:

```
UPDATE<table_name>
SET <column1>=<value1>,<column2>=<value2>, ...
WHERE <primary_key_column> = <value>;
```

Ví dụ:

```
UPDATE nhaphang
SET dongia=3500,soluong=5
WHERE mapn='pn001' and mahang='h001'
```


Truy vấn dữ liệu

Cú pháp:

```
SELECT *|<column1>, <column2>, ...
FROM <table_name>
WHERE <primary_key_column> = <value>;
```

Ví dụ:

```
SELECT * FROM nhaphang
WHERE mapn='pn001' AND mahang='h002';
```

Truy vấn dữ liệu (tt)

Cú pháp:

Trường hợp điều kiện truy vấn trên các thuộc tính ko phải khóa phân vùng thì sử dụng ALLOW FILTERING

Ví dụ:

```
SELECT * FROM nhaphang
WHERE soluong > 5 ALLOW FILTERING;
```


Mô hình dữ liệu trong Cassandra

- No join – không có phép nối
- No referential integrity – không có tính toàn vẹn tham chiếu
- Denormalization - không chuẩn hóa
- Query-first design – thiết kế dữ liệu bằng cách mô hình hóa các truy vấn phổ biến sau đó tạo các bảng hỗ trợ sau
- Designing for optimal storage – thiết kế tối ưu lưu trữ
- Sorting is a design decision – các thuộc tính sắp xếp được quyết định khi thiết kế

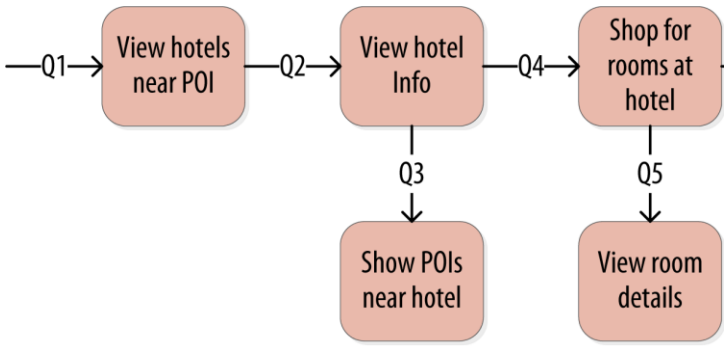
Thiết kế mô hình dữ liệu

Ví dụ:
Cần xây dựng cơ sở dữ liệu Cassandra lưu trữ thông tin về các khách sạn, khách lưu trú tại khách sạn, danh sách các phòng trong mỗi khách sạn, giá cả và tình trạng phòng trống, danh sách các địa điểm quan tâm gần khách sạn để khách có thể ghé thăm trong thời gian lưu trú.

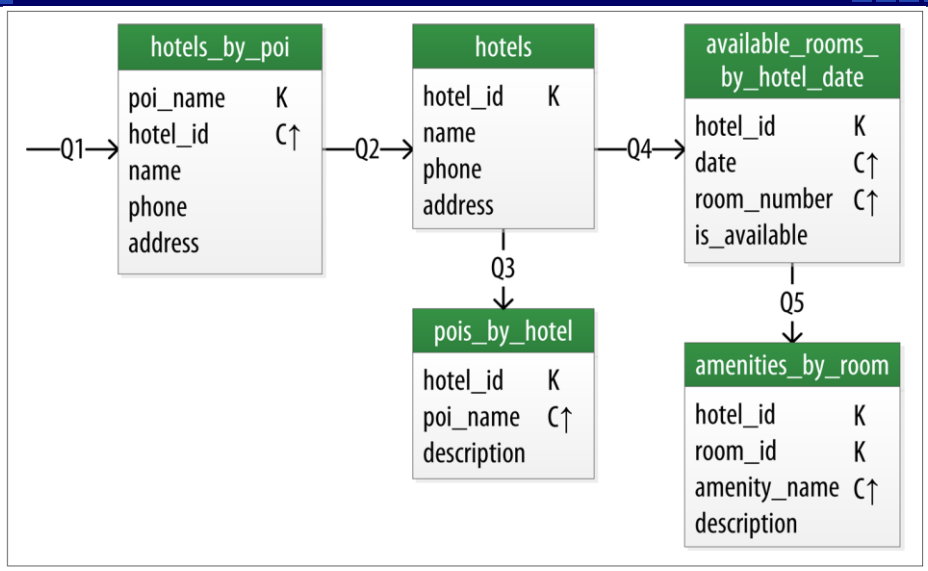
Xác định các truy vấn của ứng dụng

- Q1: Tìm khách sạn gần một điểm tham quan nhất định.
- Q2: Tìm thông tin về một khách sạn nhất định, tên và vị trí của khách sạn đó.
- Q3: Tìm điểm tham quan gần một khách sạn nhất định.
- Q4: Tìm phòng trống trong một khoảng thời gian nhất định.
- Q5: Tìm giá và tiện nghi cho một phòng.

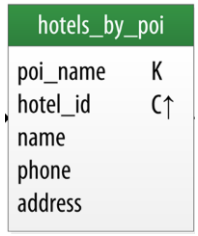
Luồn dữ liệu của ứng dụng



Mô hình dữ liệu



Tạo bảng dựa vào mô hình




```
CREATE TABLE hotels_by_poi (
poi_name text,
hotel_id text,
address text,
name text,
phone text,
PRIMARY KEY (poi_name, hotel_id))
```

```
SELECT poi_name,hotel_id,name,phone,address
FROM hotels_by_poi
WHERE poi_name = 'Lake Houron'
ORDER BY hotel_id;
```


Bài tập

- Viết lệnh tạo các bảng còn lại và viết truy vấn tương ứng.