TRƯỜNG ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN

QUẢN TRỊ DỮ LIỆU - CT467

Chương 6: NoSQL

Biên soạn:
Ths. Nguyễn Thị Kim Yến

Email: Ntkyen@ctu.edu.vn

NỘI DUNG:

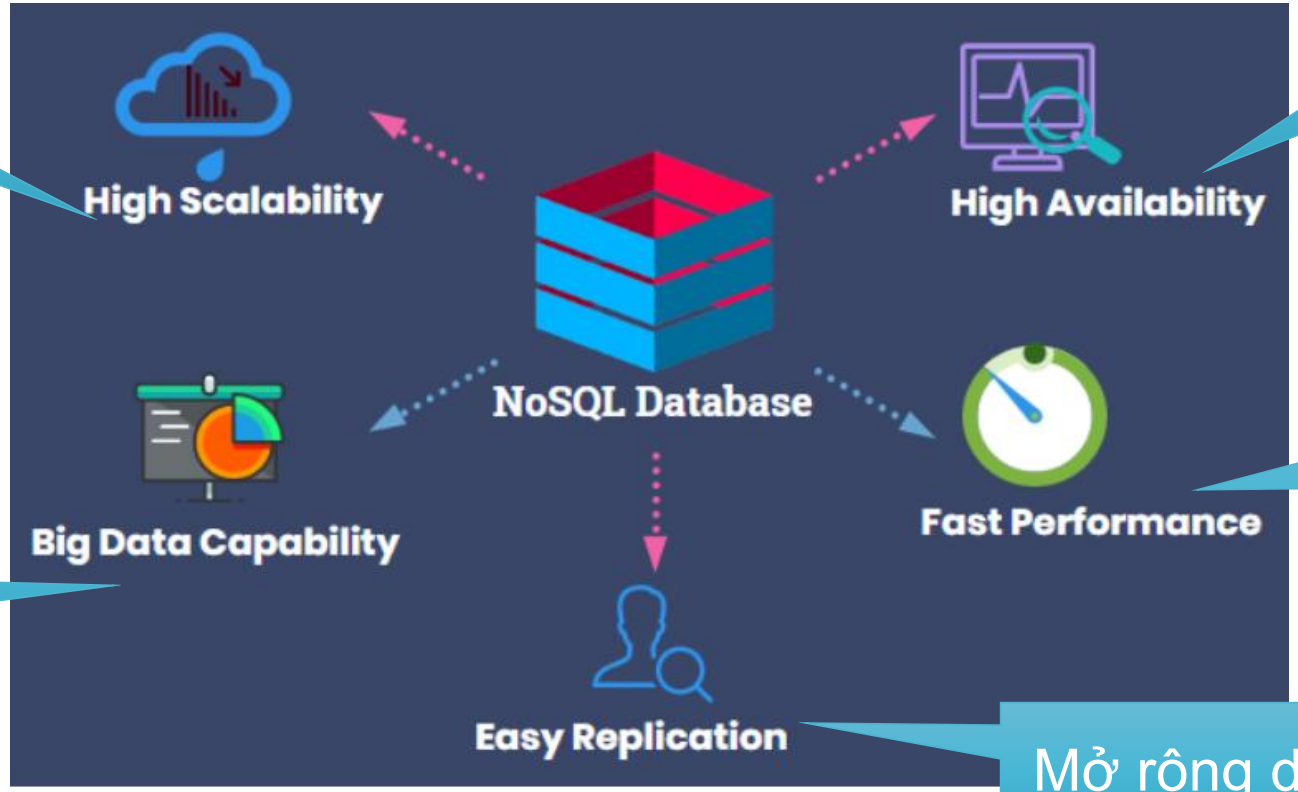
1. Giới thiệu
2. NoSQL là gì?
3. Làm việc với NoSQL

1. Giới thiệu

- Thời đại Web 2.0 bùng nổ, hàng tỷ nội dung được chia sẻ lên Web. Khối lượng dữ liệu và xử lý truy vấn trở nên rất lớn.
- Các hệ thống CSDL quan hệ truyền thống tỏ ra yếu kém trước khối lượng dữ liệu đồ sộ và lượng truy vấn khổng lồ. Nhược điểm:
  - Hệ thống thiên về xử lý tập trung, nhưng DL hiện được lưu trữ phân tán
  - Các phép toán kết nối (join) có chi phí cao
  - Khó mở rộng theo chiều ngang (horizontal scaling/scale out)
  - Chi phí cao: bản quyền, phần cứng, bảo trì,...

1. Giới thiệu (tt)

- Có nhiều giải pháp được đưa ra, trong đó **NoSQL** là một giải pháp đem lại tính hiệu quả cao. Bởi những lý do sau:



Khả năng mở rộng cao

Tính khả dụng cao

Khả năng xử lý dữ liệu lớn

Hiệu suất nhanh

Mở rộng dễ dàng

Vậy NoSQL là gì?

2. NoSQL là gì?

- **NoSQL** là viết tắt của *"Not Only SQL"* hoặc *"Not SQL"*. Một thuật ngữ tốt hơn NoREL (Not RELational), hay thường gọi là NoSQL.
- NoSQL được sử dụng như một thuật ngữ chung cho tất cả các CSDL không tuân theo quy tắc của các CSDL quan hệ (RDBMS) phổ biến và thường liên quan đến việc xử lý và thao tác với một lượng dữ liệu lớn (Big data)
- NoSQL bỏ qua tính toàn vẹn của dữ liệu và transaction để đổi lấy hiệu suất nhanh và khả năng mở rộng (scalability)

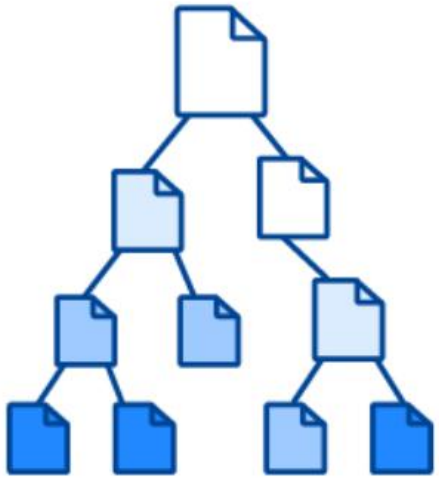
2.1 Một số thuật ngữ NoSQL liên quan

- **Non-relational:** không có ràng buộc dữ liệu
- **Distributed storage:** dữ liệu được lưu trữ phân tán (LAN, Internet…)
- **Eventual consistency:** tính nhất quán cuối, tức là đến cuối cùng vẫn đảm bảo tính nhất quán dữ liệu
- **Vertical scalable:** khả năng mở rộng về chiều dọc
- **Horizontal scalable:** khả năng mở rộng về chiều ngang
- **Distributed Data:** dữ liệu phân tán
- **Deployment Flexibility:** triển khai linh hoạt
- **Durability:** Lưu trữ tốt

2.2 Các loại CSDL NoSQL

Document

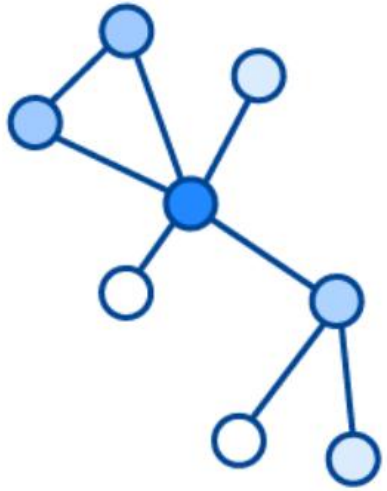
CouchDB, MongDB



DL thêm vào sẽ được lưu trữ dưới dạng cấu trúc JSON tự do hoặc "tài liệu"

Graph

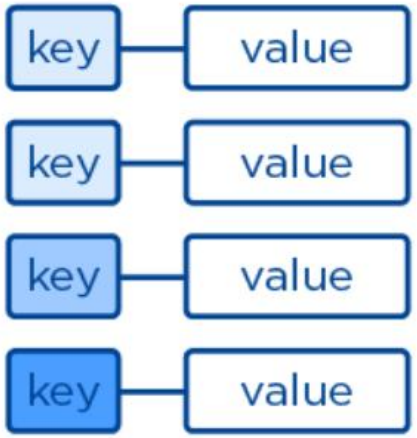
Neo4j, Hyper Graph DB



DL được biểu diễn dưới dạng mạng hoặc đồ thị. Mỗi node là 1 đoạn DL dạng tự do

Key-Value

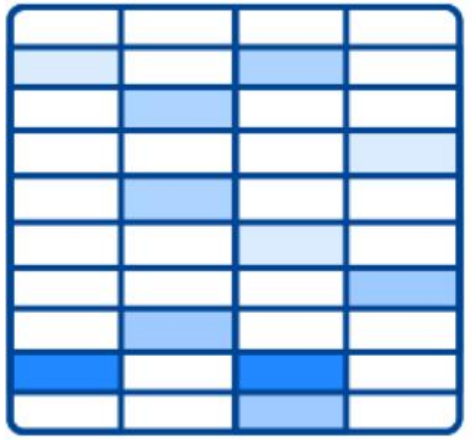
Redis, Riak



DL có thể lưu giá trị từ đơn giản đến các cấu trúc phức tạp, sử dụng phím được trong CSDL

Wide-column

HBase, Cassandra



DL bảng với hàng và cột thay đổi. Bất kỳ cột nào có thể được nhóm hoặc tổng hợp khi cần thiết

Document-based model

mongoDB.

CouchDB
relax

```
{
  "_id": 1,
  "first_name": "Tom",
  "email": "tom@example.com",
  "cell": "765-555-5555",
  "likes": [
    "fashion",
    "spas",
    "shopping"
  ]
}
```

Một document mô tả thông tin của một người

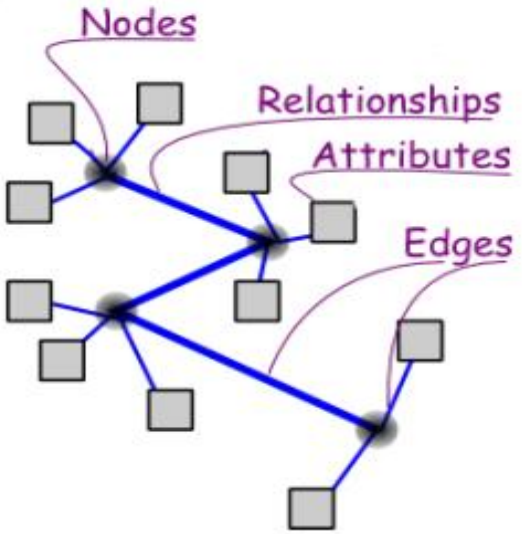
```
[
  {
    "_id": 1,
    "first_name": "Tom",
    "email": "tom@example.com",
    "cell": "765-555-5555",
    "likes": [
      "fashion", "spas", "shopping"
    ]
  },
  {
    "_id": 2,
    "first_name": "Donna",
    "email": "donna@example.com",
    "spouse": "Joe",
    "likes": [
      "spas", "shopping", "live tweeting"
    ],
    "businesses": [
      {
        "name": "Castle Realty",
        "status": "Thriving",
        "date_founded": {
          "$date": "2013-11-21T04:00:00Z"
        }
      }
    ]
  }
]
```

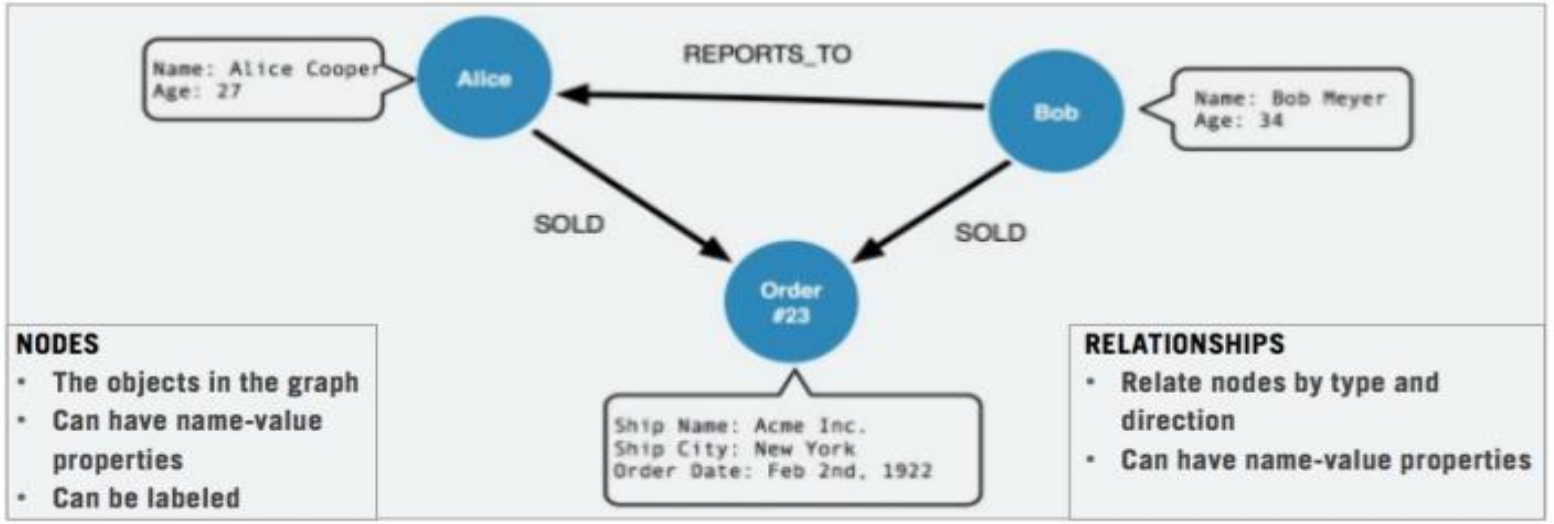
Một collection bao gồm 2 document

Dữ liệu được lưu trữ như là một tập hợp (collection) của các tài liệu (document) có cấu trúc tương tự nhau

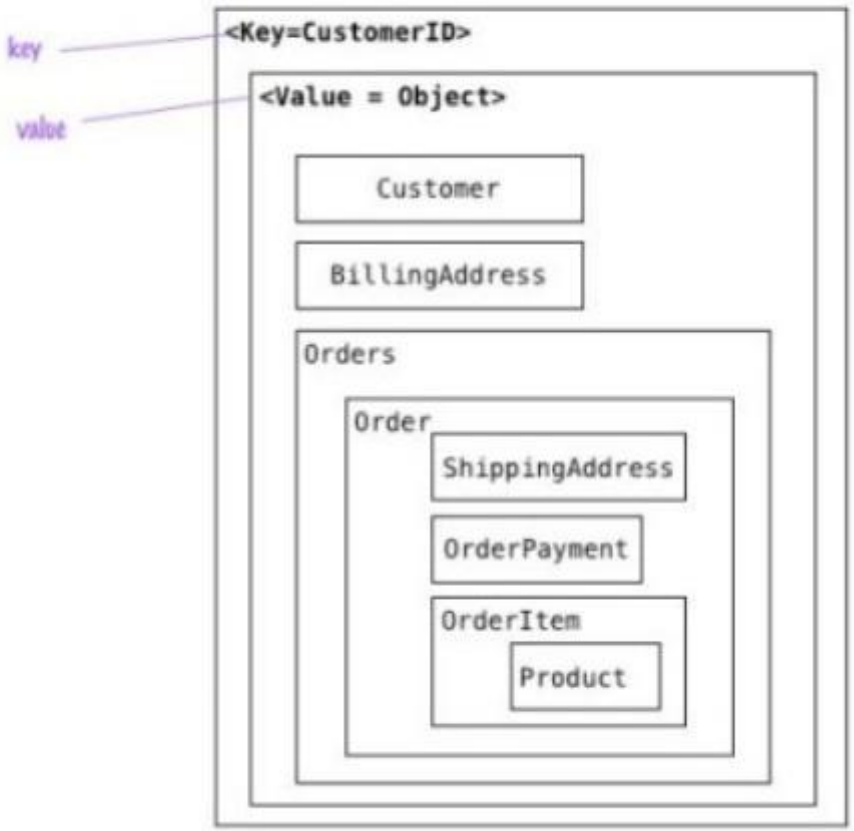
Graph-based model

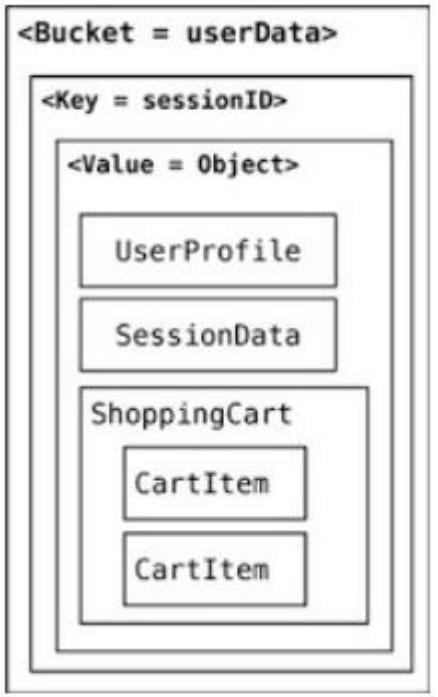
- Dựa trên lý thuyết đồ thị
- Scale theo chiều dọc, không phân cụm
- Có thể áp dụng các giải thuật của lý thuyết đồ thị
- Hỗ trợ giao dịch và 4 tính chất ACID cho các giao dịch





Key-value model





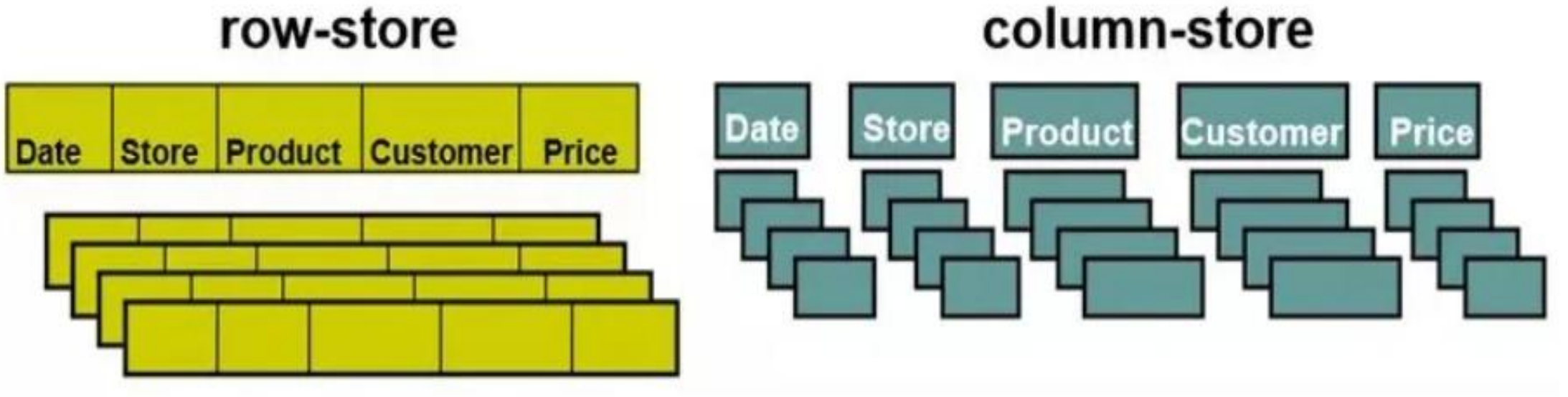
redis riak

amazon DynamoDB

Key	***Value***
Tom Cruise	Birthdate: July 3, 1962 Occupation: Actor Residence: Florida, USA
Roger Federer	Birthdate: Aug 8, 1981 Occupation: Tennis Player Residence: Bottmingen, Switzerland
Bill Gates	Birthdate: Oct 28, 1955 Occupation: Entrepreneur, Philanthropist Residence: Medina, WA, USA
...	...

Column-based model

- Dữ liệu được lưu trữ trong database dưới dạng các cột.
- Mỗi cột được xử lý riêng biệt.
- Giá trị của CSDL cột đơn được lưu trữ liền kề.



Ưu, nhược điểm các loại CSDL NoSQL

CSDL NoSQL	Ưu điểm	Nhược điểm
Document	Dùng khi DL nguồn không được mô tả đầy đủ	Không có cú pháp chuẩn cho câu truy vấn dữ liệu
Grap	Ứng dụng các thuật toán trên đồ thị như Đường đi ngắn nhất, liên thông,...	Phải duyệt nội bộ đồ thị, để trả lời lại các truy vấn. Không dễ để phân tán
Key-Value	Tìm kiếm rất nhanh	Lưu DL không theo dạng (schema) nhất định
Wide-column	Tìm kiếm nhanh, Phân tán DL tốt	Hỗ trợ được với rất ít phần mềm

2.3 Khi nào sử dụng NoSQL?

- NoSQL phù hợp với các trường hợp sau:
  - Các hệ thống cần phát triển nhanh, linh hoạt
  - Dữ liệu được lưu trữ có cấu trúc và bán cấu trúc
  - Khối lượng dữ liệu khổng lồ
  - Có yêu cầu về mở rộng theo chiều ngang
  - Các hệ thống theo mô hình micro-services hoặc streeaming realtime

2.4 So sánh SQL và NoSQL

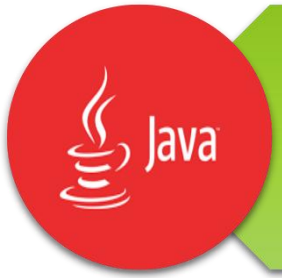
Tiêu chí SS	SQL	NoSQL
Mô hình phát triển	- Kết hợp mã nguồn mở + mã nguồn đóng	- Mã nguồn mở
Phần cứng	- Đòi hỏi phần cứng cao	- Không đòi hỏi quá cao về phần cứng
Mở rộng DL	- Khi muốn bổ sung thêm cột cho 1 bảng phải khai báo trước	- Không cần khai báo trước
Schemas	- Cố định	- Uyển chuyển
Joins	- Thường xuyên yêu cầu	- Không thường xuyên yêu cầu
Thao tác DL	- Sử dụng các câu lệnh SELECT, INSERT, UPDATE, DELETE	- Thông qua thư viện lập trình hướng đối tượng
Tốc độ read/write	- Đảm bảo tính ràng buộc DL giữa các bảng - Bảo toàn tính nhất quán về DL ở các server với nhau	- Bỏ qua các cơ chế ràng buộc của các bảng - Có tính nhất quán cuối (DL được lưu trong RAM, sau đó đẩy xuống HDD)

2.5 So sánh thuật ngữ CSDL SQL và NoSQL

SQL	MongoDB	DynamoDB	Cassandra	Couchbase
Bảng	Bộ sưu tập	Bảng	Bảng	Bộ chứa dữ liệu
Hàng	Tài liệu	Mục	Hàng	Tài liệu
Cột	Trường	Thuộc tính	Cột	Trường
Khóa chính	Id đối tượng	Khóa chính	Khóa chính	ID văn bản
Chỉ mục	Chỉ mục	Chỉ mục thứ cấp	Chỉ mục	Chỉ mục
Chế độ xem	Chế độ xem	Chỉ mục thứ cấp toàn cục	Chế độ xem cụ thể hóa	Chế độ xem
Bảng hoặc đối tượng lồng nhau	Văn bản nhúng	Bản đồ	Bản đồ	Bản đồ
Mảng	Mảng	Danh sách	Danh sách	Danh sách

3. Làm việc với NoSQL

- NoSQL với các ngôn ngữ lập trình:



1.NoSQL với Java

2.NoSQL với Python

3.NoSQL với PHP

3.1 NoSQL với Java

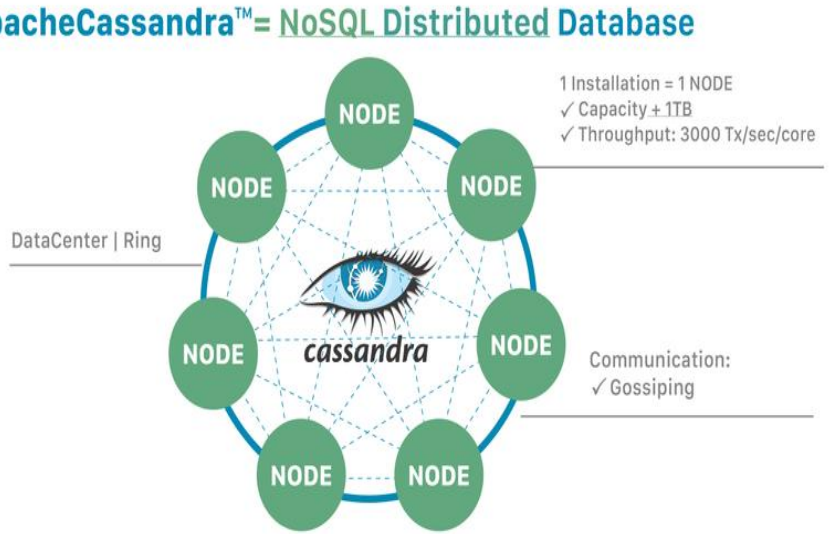
- Hầu hết các CSDL NoSQL hỗ trợ Java, trong đó có MongoDB và Hbase.
- **MongoDB** phân phối một thư viện riêng dành cho việc kết nối với Java.
- Thư viện tài liệu có sẵn: https://www.mongodb.com/docs/drivers/java-drivers/
- Thư viện này được đóng gói dưới dạng .jar. Sau khi tải về, chỉ cần thêm đường dẫn tới thư viện vào trong classpath của ứng dụng là có thể sử dụng.

3.2 NoSQL với Python

- **Python** là một ngôn ngữ lập trình mạnh và khá mềm dẻo. Việc truy vấn đến các CSDL NoSQL bằng các thư viện có sẵn.
- Chẳng hạn như thư viện **pymongo** (cần phải cài đặt thêm CSDL MongoDB): https://www.mongodb.com/languages/python/pymongo-tutorial
- Sau khi tải về và cài đặt vào trong thư viện của Python, ta có thể dễ dàng sử dụng thư viện này.

3.3 NoSQL với PHP

- PHP là một ngôn ngữ được sử dụng rất nhiều trong thiết kế và phát triển các ứng dụng Web. Ngoài việc tương thích rất tốt với các CSDL quan hệ RDBMS, PHP cũng được rất nhiều các CSDL NoSQL hỗ trợ.
- Giống như Python, Apache Cassandra cũng hỗ trợ PHP với một thư viện có tên phpcassa.
- **Cassandra** là dự án mã nguồn mở của Apache, có thể tích hợp với các mã nguồn mở Apache khác như: Hadoop, Apache Pig và Apache Hive.



3.4 Ưu, nhược điểm của NoSQL

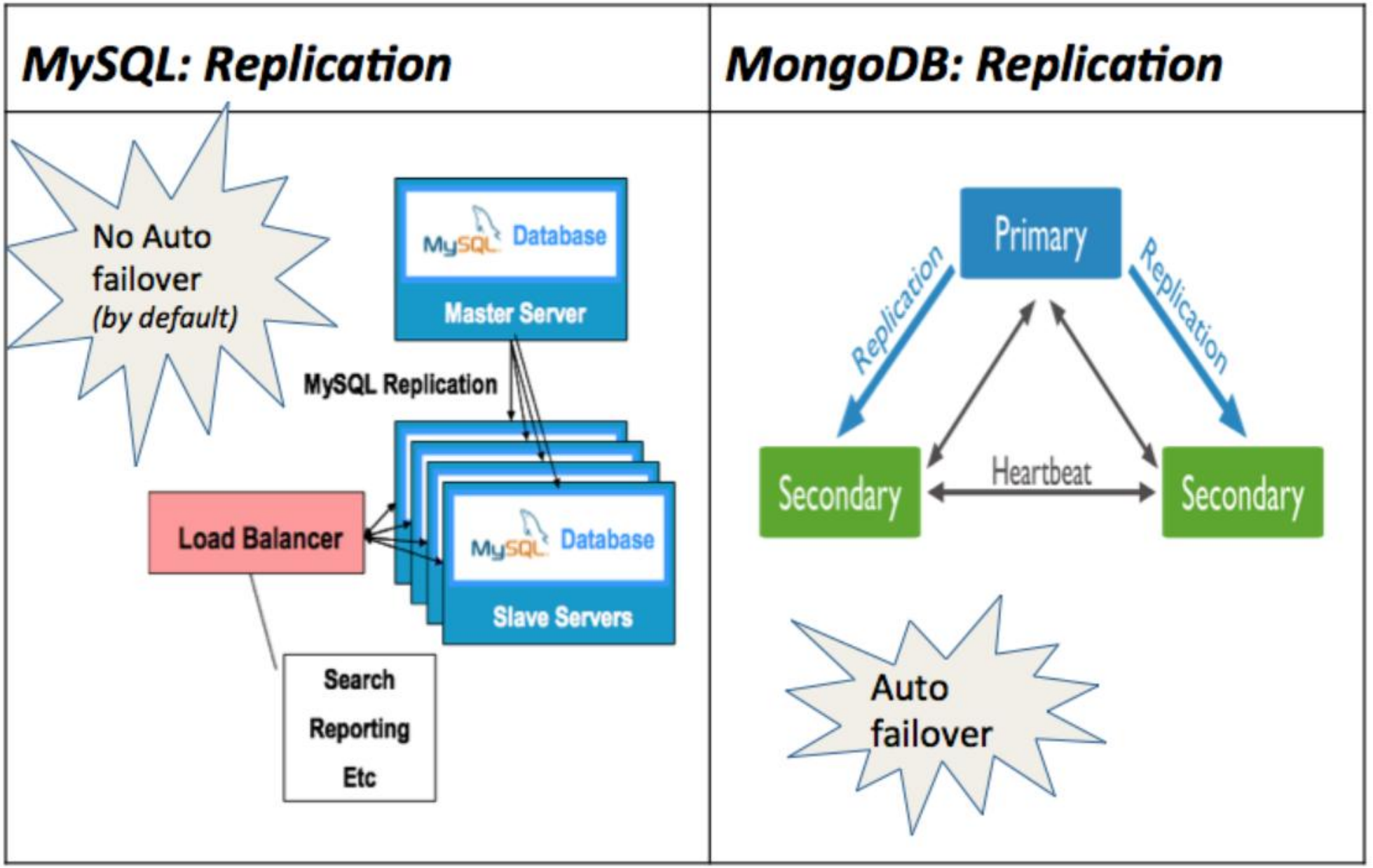
Ưu điểm	Nhược điểm
• Dễ dàng mở rộng quy mô dữ liệu. • Giải quyết tốt các CSDL có kích thước và truy cập lớn • Không thực sự cần các DB Admin • Có hiệu quả kinh tế cao • Mô hình dữ liệu linh hoạt	• Khả năng tin cậy • Sự hỗ trợ bên phía nhà sản xuất • Tính tương thích • Quản trị • Chuyên môn của người quản lý

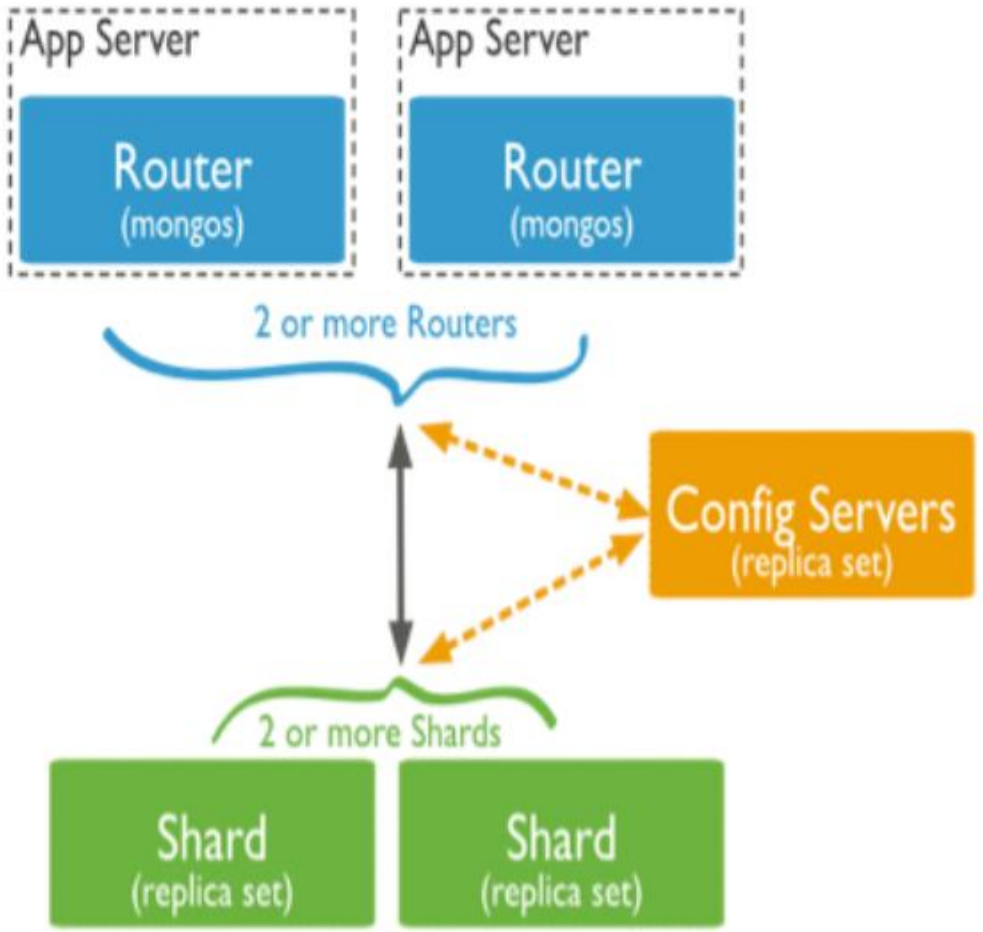
3.5 NoSQL có thể thay thế SQL không?

- Mặc dù CSDL NoSQL đã thể hiện nhiều tính năng vượt bậc, tuy nhiên vẫn còn 1 chặng đường dài để NoSQL bắt chước tính nhất quán và độ tin cậy của CSDL SQL.
- Có nghĩa là **NoSQL** có hỗ trợ **SQL** cùng với nhiều tính năng hơn và nó không thể thay thế CSDL SQL truyền thống.
- Sự lựa chọn hoàn toàn dựa trên các yêu cầu về cấu trúc dữ liệu, xử lý truy vấn và khả năng mở rộng cho một ứng dụng.

MongoDB Administration Checklist for MySQL DBAs

- Dưới đây phác thảo các khái niệm MySQL điển hình và các tác vụ DBA (bên trái) cho các khái niệm MongoDB tương ứng (bên phải)
- Link tham khảo về MongoDB: https://www.mongodb.com/docs/
- Kiến trúc:
  - Replication
  - Sharding


MySQL: Replication
No Auto failover (by default)
MySQL Database
Master Server
MySQL Replication
Load Balancer
MySQL Database
Slave Servers
Search
Reporting
Etc
MongoDB: Replication
Primary
Replication
Replication
Secondary
Heartbeat
Secondary
Auto failover

MySQL: ***sharding, custom solution***	***MongoDB:*** ***out of the box sharding***
• Custom solutions? • Fabric? • MaxScale? • Other?	

MySQL: Grants

```
mysql> grant all on *.*
to user@localhost
identified by 'pass';
```

MongoDB: createUser

```
> use products
db.createUser( {
          user: "accountUser",
          pwd: "password",
          roles: [ "readWrite", "dbAdmin" ]
})
```

MySQL: Index

```
mysql> show keys from zipsG
**** 1. row ****
        Table: zips
   Non_unique: 0
     Key_name: PRIMARY
  Seq_in_index: 1
  Column_name: id
    Collation: A
  Cardinality: 0
     Sub_part: NULL
       Packed: NULL
         Null:
   Index_type: BTREE
```

MongoDB: Index

```
> db.zips.getIndexes()
[
        {
                "v" : 1,
                "key" : {
                        "_id" : 1
                },
                "name" : "_id_",
                "ns" : "zips.zips"
        }
]
```

MongoDB có những ưu điểm sau:

- Dễ học, quản lý bằng command line hoặc GUI như phpMoAdmin
- Linh động, không cần phải định nghĩa cấu trúc dữ liệu trước khi lưu trữ
- Khả năng mở rộng tốt, cân bằng tải cao
- Miễn phí

```
{
_id : ObjectId("4db31fa0ba3aba54146d851a"),
username : "joegunchy",
email : "joe@mysite.org",
age : 26,
is_admin : true,
created : "Sun Apr 24 2011 01:52:58 GMT+0700 (BDST)"
}
```

MongoDB document

Table: *users*

id	username	email	created_at
256	joegunchy	joe@mysite.com	2011-04-24 01:52:58

Collection: *users*

```
{
    _id         : ObjectId("4db31fa0ba3aba54146d851a")
    username    : "joegunchy"
    email       : "joe@mysite.org"
    created_at  : "Sun Apr 24 2011 01:52:58 GMT+0700 (BDST)"
}
```

So sánh Table và Collection

HẾT CHƯƠNG 6