从SQL到NoSQL

徐辰
华东师范大学
数据科学与工程学院
cxu@dase.ecnu.edu.cn

DaSE
Data Science
& Engineering

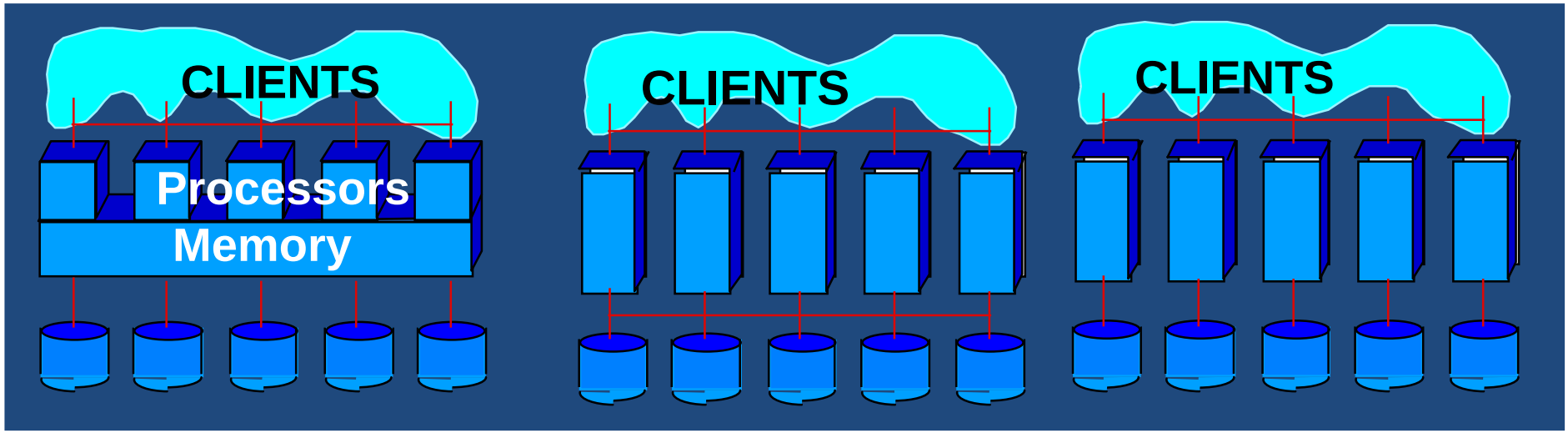
分布式数据库

并行数据库

Shared Memory
Shared Everything

Shared Disk

Shared Nothing



扩展能力 / Scalability

- Scale Up：移植到更强大的服务器上
  - 昂贵
  - 扩展能力有限
  - Easy
- Scale Out：增加节点（commodity computer）
  - 性价比高
  - 扩展能力更强 / 弹性
  - Difficult

NoSQL


APACHE
HBASE
Cassandra
riak
CouchDB
relax
mongoDB
HYPERTABLE INC
Neo4j
redis

NoSQL分类

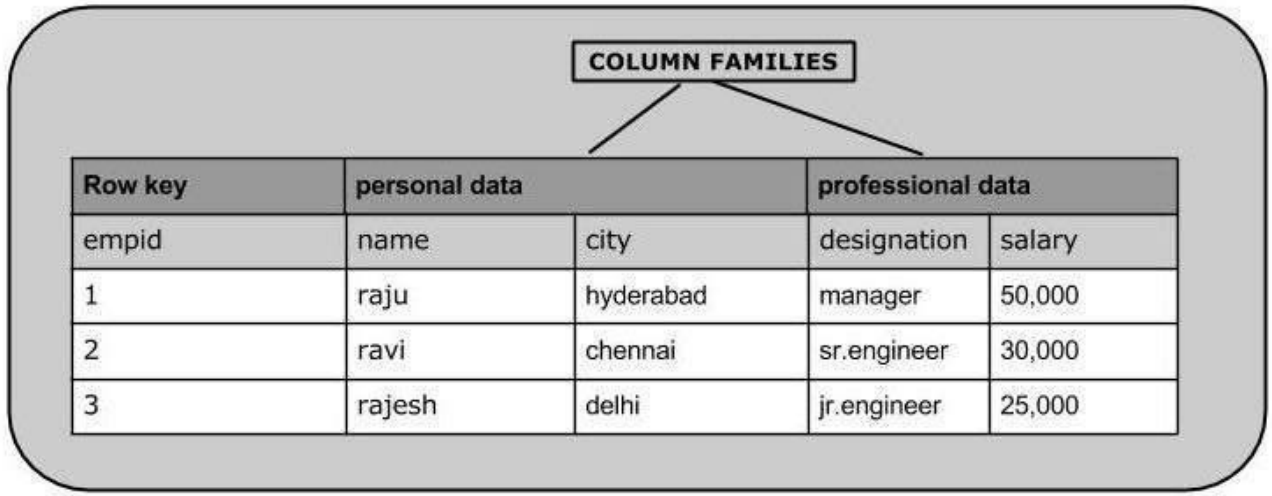
- Key-value stores
  - Riak, Voldemort, Redis
- Wide-column stores
  - Cassandra, HBase
- Document databases
  - MongoDB, CoachBase
- Graph databases
  - Neo4J, HyperGraphDB

Key-Value Store

- 数据组织：<key, value>
- 数据访问方式：
  - Put(key, value)
  - Get(key)
  - Delete(key)
- 例子：

Car	
Key	Attributes
1	Make: Nissan Model: Pathfinder Color: Green Year: 2003
2	Make: Nissan Model: Pathfinder Color: Blue Color: Green Year:2005 Transmission: Auto

Wide Column Store

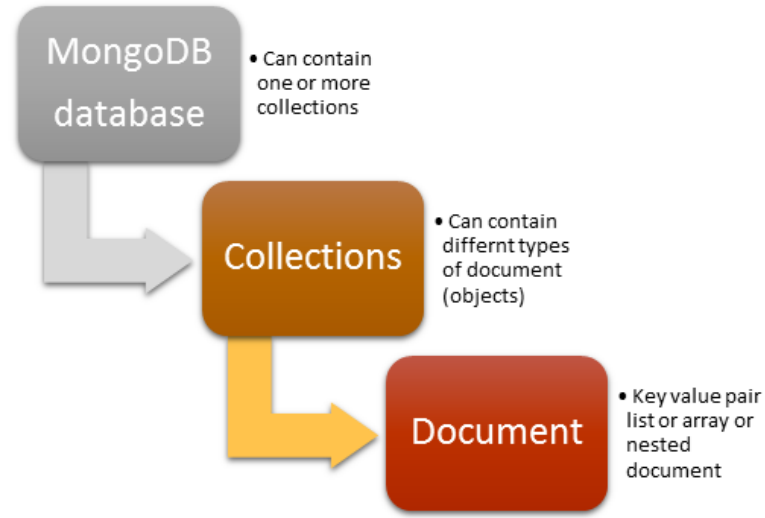


Row key	personal data		professional data	
empid	name	city	designation	salary
1	raju	hyderabad	manager	50,000
2	ravi	chennai	sr.engineer	30,000
3	rajesh	delhi	jr.engineer	25,000

get 'table name', ‘rowid’, {COLUMN ⇒ ‘column family:column name ’}

Document Store

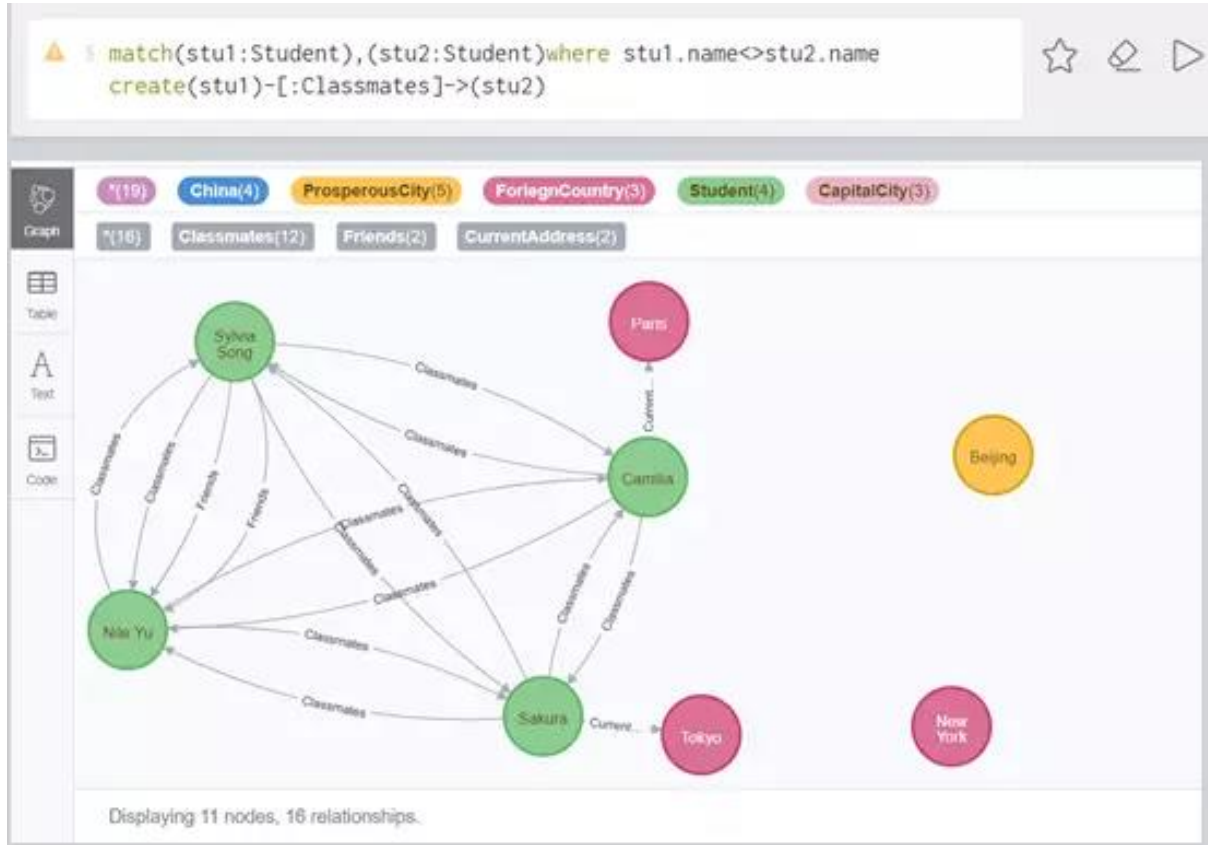
```
{ _id: ObjectID('4bd9e8e17cefd644108961bb'),
  name:'Vivek',
  class : '12th',
  subjects: [ 'physics', 'chemistry', 'math', 'english', 'computer'],
  address: { house_no: '12B',
             block: 'B',
             sector: 12,
             city : 'noida'
           },
  grade: [
           { exam: 'unit test 1',
             score: '60%'
           },
           { exam: 'unit test 2',
             score: '70%‘
           }
         ]
}
```




```
db.marks.find( { name:'student0‘ } )
```

```
db.marks.find( { subjects:{ $in:[ 'sports', 'arts' ] } } )
```

Graph database



数据库可以具备普适扩展能力吗？
即 面对任意形态的数据和负载。

现有系统的扩展能力如何?

- SQL和NoSQL相比谁的扩展能力更好?
  a. NoSQL比SQL具备更强的扩展能力。
  b. SQL和NoSQL的扩展能力无本质区别。
  c. 从NoSQL获得扩展能力更容易。

注：我们认为NewSQL仅仅是SQL的特殊形态。

大量互联网应用实现了SQL的Scale-out

然而...令人头疼的分库分表

- 数据划分方式的选择
- 数据迁移
- 模式更改
- 应用程序重构
- Debug

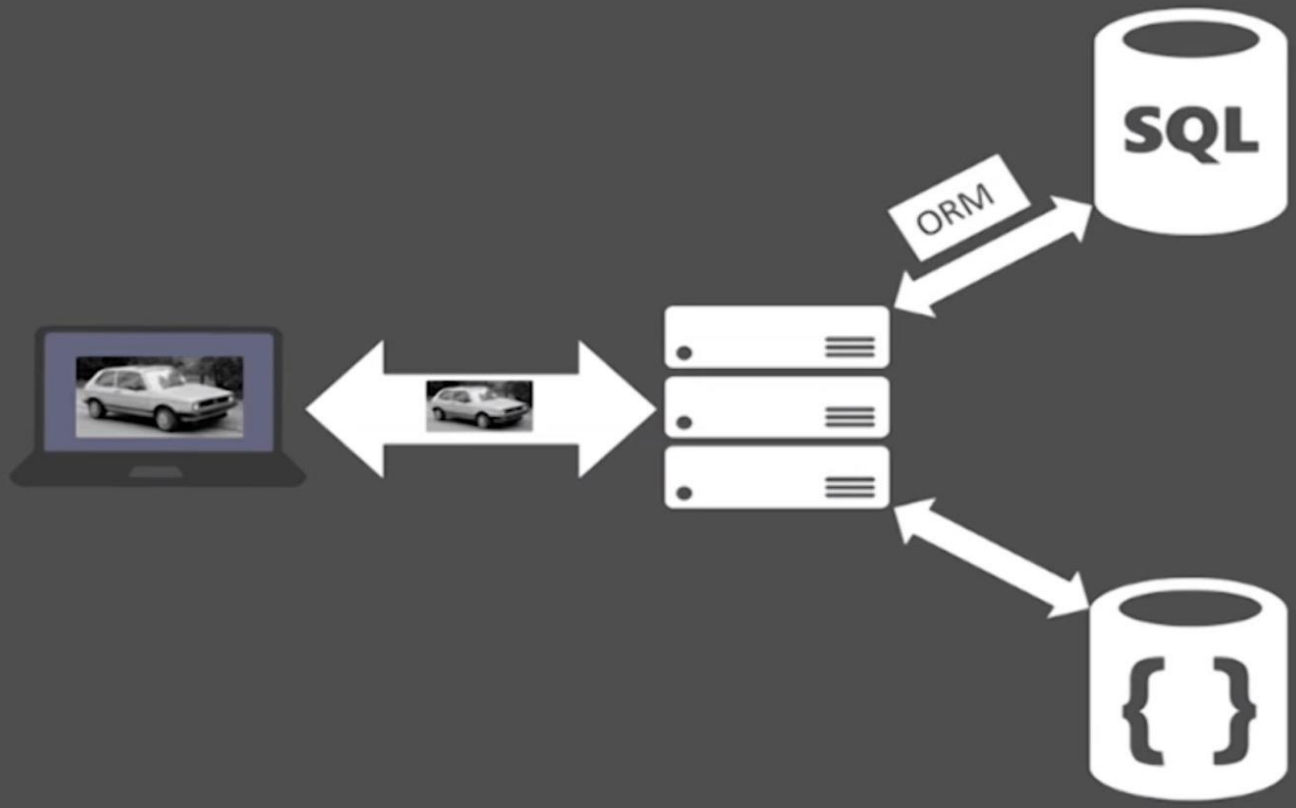
NoSQL和SQL不同的开发过程

- 传统的SQL数据库设计流程
  概念模型(Conceptual Model) → 模式(Schema) →
  物理设计优化(Physical Design) → 分库分表(Sharding)

- NoSQL的数据库设计流程
  应用功能接口(App Operation) → 模式(Schema) →
  横向扩展(Scaling)

- **哪种流程能够获得更好的数据访问局部性？**

Retrain your relational brain



Data normalization

Come as you are

E-Commerce场景

- **基本功能接口**
  - 查找商品
  - 查看商品
  - 添加商品至购物车
  - 付款

  ….

- **基本概念**
  - 商品
  - 用户
  - 购物车

基于概念的模式设计

Product =
{ pid: 456789,
brand: zipper,
name: Copy paper,
price: 24.00,
size: 8.5’ x 11’,
Items_in_stock: 14000
}

Customer =
{ cid: 1357
first_name: woo,
Family_name: yee,
credit_card: xxxxxx,
address: xxxxxx,
email: wyee@gmail.com
}

Cart =
{ oid: 12345678,
Product:[{pid:456789,
quantity:1}, {pid:345567,
quantity:1}],
user: {uid:1357}
}

基于功能接口的模式设计

Product =
{ pid: 456789,
brand: zipper,
name: Copy paper,
price: 24.00,
size: 8.5’ x 11’,
Items_in_stock: 14000
keywords: [{paper} , {print}, {copy}]
}

Customer =
{ cid: 1357
first_name: woo,
Family_name: yee,
credit_card: xxxxxx,
address: xxxxxx,
email: wyee@gmail.com,
cart: {oid: 12345678}
}

Cart =
{ oid: 12345678,
Product:[{pid:456789, name: Copy paper, price: 24.00, quantity:1}, {pid:345567, name:notebook, price: 8.00, quantity:1}],
total_price: 32.00,
user: {uid:1357, name:woo yee, address:xxxxxx}
}

NOSQL的工具特点使得APP开发
从始至终都在考虑扩展性问题。