**1. 概念**

SQL (Structured Query Language) 数据库，指关系型数据库。主要代表：SQL Server，Oracle，MySQL。

NoSQL（Not Only SQL）泛指非关系型数据库。主要代表：MongoDB，HBase。

**2.诞生的原因**

目前世界上主流的存储系统大部分还是采用了关系型数据库，其主要有一下优点：

1.事务处理—保持数据的一致性；

2.由于以标准化为前提，数据更新的开销很小（相同的字段基本上只有一处）；

3.可以进行Join等复杂查询。

关系型数据库的缺点：

1. 扩展困难：由于存在类似Join这样多表查询机制，使得数据库在扩展方面很艰难;

2. 读写慢：当数据量达到一定规模时数据库读写速度下降严重，MySql单表数据量超过500万时IO速度会快速下降;

3. 成本高：企业级数据库的License价格很惊人，并且随着系统的规模，而不断上升;

4. 有限的支撑容量：现有关系型解决方案还无法支撑Google这样海量的数据存储;

所以人们希望有这样的数据库：

1. 低延迟的读写速度：应用快速地反应能极大地提升用户的满意度;

2. 海量的数据和流量：对于搜索这样大型应用而言，需要利用PB级别的数据和能应对百万级的流量;

3. 大规模集群的管理：系统管理员希望分布式应用能更简单的部署和管理;

4. 庞大运营成本的考量：IT经理们希望在硬件成本、软件成本和人力成本能够有大幅度地降低;

为了满足这种需求NoSQL数据库出现了，

现在主流的NoSQL数据库有HBase、Cassandra、MongoDB和Redis等。

**2.NoSQL 优缺点**

**优点**

1.**简单的扩展**：

2.**快速的读写**：主要例子有Redis，由于其逻辑简单，而且纯内存操作，使得其性能非常出色，单节点每秒可以处理超过10万次读写操作;

3.**低廉的成本**：

**缺点**

1. 不提供对SQL的支持：如果不支持SQL这样的工业标准，将会对用户产生一定的学习和应用迁移成本;

2. 支持的特性不够丰富：现有产品所提供的功能都比较有限，大多数NoSQL数据库都不支持事务，也不像MS SQL Server和Oracle那样能提供各种附加功能，比如BI和报表等;

3. 现有产品的不够成熟：大多数产品都还处于初创期，和关系型数据库几十年的完善不可同日而语;

**3.NoSQL 使用场景**

与关系型数据库相比，NoSQL数据库适于以下场景

**1) 数据库表schema经常变化**

比如在线商城，维护产品的属性经常要增加字段，这就意味着ORMapping层的代码和配置要改，如果该表的数据量过百万，新增字段会带来额外开销（重建索引等）。NoSQL应用在这种场景，可以极大提升DB的可伸缩性，开发人员可以将更多的精力放在业务层。

**2)数据库表字段是复杂数据类型**

对于复杂数据类型，比如SQL Sever提供了可扩展性的支持，像xml类型的字段。NoSQL以json方式存储，提供了原生态的支持，在效率方便远远高于传统关系型数据库。

**3)高并发数据库请求**

此类应用常见于web2.0的网站，很多应用对于数据一致性要求很低，而关系型数据库的事务以及大表join反而成了”性能杀手”。

**4)海量数据的分布式存储**

海量数据的存储如果选用大型商用数据，如Oracle，那么整个解决方案的成本是非常高的，要花很多钱在软硬件上。NoSQL分布式存储，可以部署在廉价的硬件上，是一个性价比非常高的解决方案。

**What is MongoDB?**

MongoDB is one of NoSQL databases, which is based on distributed documents storage. Many people think that it is the most similar to the relational database due to the fact that it has the rich functions.

**Main features:**

There are also following features in MongoDB beside the common features of the NoSQL database.

1. It is easy to install, study and use MongoDB.
2. MongoDB supports creating index in order to speed up query.
3. MongoDB can spread data to other nodes in the network automatically in the cause of more storage and processing power.
4. Mongo supports rich query expressions.
5. MongoDB use JSON documents in order to store records.
6. MongoDB supports many kinds of programming language, including RUBY，PYTHON，JAVA，C++，PHP，C# and so forth.

**Some conception of MongoDB compared with SQL**

|  |  |
| --- | --- |
| SQL | MongoDB |
| database | database |
| table | collection |
| row | document |
| column | field |
| index | index |
| table joins | N/A |
| primary key | primary key |

**Database**

## There may be more than one independent database in one MaogoDB instance. There are independent collections and authority in every database.

## Document

A document consists of a set of key-value pairs. The document of MongoDB does not need the same fields, and the same field does not need the same data type.

collection

A set of documents is a collection, it likes the table of relational database. We can insert documents which have different fields and data type into a document.

**How to use MongoDB**

There is a table persons in a relational database, and the table has 3 fields name, age, and gender:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **fields** | **id** | **name** | **age** | **gender** |
| **value** | 5b53fcce89d0192c20f257bd | Tom | 15 | male |

In mongoDB it can be expressed as:

{

\_id: ObjectId(“5b53fcce89d0192c20f257bd”),

name: “Tom”,

ago: 15,

gender: “male”

}

**Connecting to a database:**

mongodb://admin:123456@localhost/

**username is admin, password is 123456 and hostname is localhost.**

**Creating a database:**

**use persondb**

**Persondb is database name, if persondb is not exist then create it, or switched to db persondb.**

****Delete a database:****

**use persondb**

**db.dropDatabase()**

****Create/delete a collection:****

**use persondb**

**db.createCollection("collection1") //create collection1**

**db.collection1.drop() //delete collection1**

When you insert documents into a document, the document would be created automatically.

For example:

db.my\_collection.insert({name: “Tom”, ago: 15, gender: “male”})

The collection, my\_collection would be created automatically.

Insert documents:

db.my\_collection.insert({name: “Tom”, ago: 15, gender: “male”})

db.my\_collection.insert({name: “Mary”, ago: 11, gender: “fmale”})

db.my\_collection.insert({name: “Kate”, ago: 20, gender: “fmale”})

db.my\_collection.insert({name: “Mark”, ago: 30, gender: “male”})

db.my\_collection.insert({name: “Jimmy”, ago: 8, gender: “male”})

**NoSQL和关系数据库结合**

In actual development，it is a good solution to combine with the benefit of NoSQL and relational database. For example, if we want to store some articles. there are three fields including title, author and content. Normally,there are a lot of words , sometimes they may be more than 10 thousands, in content field. Now we have a oracle and a mongoDB. If we only use the oracle database to store those articles, it would put a lot of IO pressure on it in the case of high system concurrency. So, we can consider to make the oracle cooperate with the mongoDB to solve the problem. First of all, we can create a table called ‘articles’ which includes id, title, author, and contentId 4 fields in Oracle. Then, we create a collection called ‘ArticleContents’ which is used to save the content of article in the MongoDB.

If we want to get the content of a article by the id of the article, we can do the following steps.

First step: we get article content id from oracle:SELECT id as articleContentId FROM articles where id =’articleId’

Second step: we get the content of the article by articleContentId: db.ArticleContents.find({id:’articleContentId’})

For this, the IO pressure of the oracle can be reduced, and the preference of the system would experience a climb.

**4. NoSQL 与 SQL 的区别**

**SQL 数据库**

在使用之前需要定义表的一个模式

在表中存储相关联的数据

支持 join 多表查询

提供事务

使用一个强声明性语言查询

提供足够的支持,专业技能和工具

**NoSQL 数据库**

将相关联的数据存储在类似 JSON 格式，名称-值

可以保存没有指定格式的数据

保证更新一个文档 – 但不是多个文档

提供出色的性能和可伸缩性

使用 JSON 数据对象查询

**a 存储方式**

SQL数据存在特定结构的表中；

而NoSQL则更加灵活和可扩展，存储方式可以是JSON文档、哈希表或者其他方式。

**b 表/集合数据的关系**

SQL中，必须定义好表和字段结构后才能添加数据，例如定义表的主键(primary key)，索引(index),触发器(trigger),存储过程(stored procedure)等。表结构可以在被定义之后更新，但是如果有比较大的结构变更的话就会变得比较复杂。

在NoSQL中，数据可以在任何时候任何地方添加，不需要先定义表。

NoSQL也可以在数据集中建立索引。以MongoDB为例，会自动在数据集合创建后创建唯一值\_id字段，这样的话就可以在数据集创建后增加索引。从这点来看，NoSQL可能更加适合初始化数据还不明确或者未定的项目中。

**b 外部数据存储**

SQL中如果需要增加外部关联数据的话，规范化做法是在原表中增加一个外键，关联外部数据表。

NoSQL中除了这种规范化的外部数据表做法以外，我们还能用如下的非规范化方式把外部数据直接放到原数据集中，以提高查询效率。缺点也比较明显，更新审核人数据的时候将会比较麻烦。

**c SQL中的JOIN查询**

SQL中可以使用JOIN表链接方式将多个关系数据表中的数据用一条简单的查询语句查询出来。

NoSQL未提供对多个数据集中的数据做查询。

**d 数据耦合性**

SQL中不允许删除已经被使用的外部数据，例如审核人表中的"熊三"已经被分配给了借阅人熊大，那么在审核人表中将不允许删除熊三这条数据，以保证数据完整性。

而NoSQL中则没有这种强耦合的概念，可以随时删除任何数据。