第五章 NoSQL

第五章考点：

- NoSQL概念及兴起
- NoSQL与关系数据库的比较
- NoSQL的四大类型
- NoSQL的三大基石，CAP，BASE

一、NoSQL简介

NoSQL是一种不同于关系数据库的数据管理系统设计方式，是对非关系型数据库的统称，它采用的是类似键值、列族、文档等非关系模型
NoSQL数据库有以下三个特点：

- 灵活的可拓展性：
  NoSQL数据库在设计之初是为了满足横向扩展需求，因此天生具有良好的横向水平拓展能力
- 灵活的数据模型：
  采用键值、列组等非关系数据模型，允许一个数据元素里存储不同类型的数据
- 与云计算紧密融合
  凭借横向拓展能力，充分利用云计算基础设施，很好地将数据库融入到云计算环境中，构建基于NoSQL的云数据库服务

二、NoSQL与关系数据库的比较

比较标准	RDBMS	NoSQL	备注
数据库原理	完全支持	部分支持	RDBMS有关系代数理论作为基础 NoSQL没有统一的理论基础
数据规模	固定	超大	RDBMS很难实现横向拓展，纵向拓展的空间也比较有限，性能会随着数据规模的增大而降低 NoSQL可以很容易通过添加更多的设备来支持更大规模的数据
数据库模式	固定	灵活	RDBMS需要定义数据库模式，严格遵守数据定义和相关的约束条件 NoSQL不存在数据库模式，可以自由灵活定义并存储不同类型的数据

比较标准	RDBMS	NoSQL	备注
查询效率	快	可以实现高效的简单查询，但是不具备高度结构化查询等特性，复杂查询的性能不尽人意	RDBMS借助索引机制可以实现快速查询 很多NoSQL没有面向复杂查询的索引，虽然NoSQL可以使用MapReduce来实现加速查询，但是在复杂查询方面的性能仍不如RDBMS
一致性	强一致性	弱一致性	RDBMS严格遵守事物ACID模型，可以确保事物强一致性 很多NoSQL数据库放松了对事物ACID四性的要求，而是遵守BASE模型，只能保证最终一致性
数据完整性	容易实现	很难实现	任何一个RDBMS都可以很容易实现数据完整性，比如通过主键或者非空约束来实现实体完整性，通过主键、外键来实现参照完整性，通过约束或者触发器来实现用户自定义完整性 但是在NoSQL数据库却无法实现
扩展性	一般	好	RDBMS很难实现横向扩展，纵向扩展的空间比较有限 NoSQL在设计之初就充分考虑了横向扩展的需求，可以很容易通过添加廉价设备来实现扩展
可用性	好	很好	RDBMS在任何时候都以保证数据一致性为优先目标，其次才是优化系统性能，随着数据规模的增大，RDBMS为了保证严格的一致性，只能提供相对较弱的可用性 大多数NoSQL都只能提供较高的可用性
标准化	是	否	RDBMS已经标准化（SQL) NoSQL还没有行业标准
技术支持	高	低	RDBMS经过几十年的发展已经非常成熟，Oracle等大型厂商都可以提供很好的技术支持 NoSQL在技术支持方面仍处于起步阶段，还不成熟，缺乏有力的技术支持
可维护性	复杂	复杂	RDBMS需要专门的数据库管理员（DBA)维护

总结

- 关系数据库
  优势：以完善的关系代数理论作为基础，有严格的标准，支持事务ACID四性，借助索引机制可以实现高效的查询，技术成熟，有专业公司的技术支持。
  劣势：可扩展性较差，无法较好支持海量数据存储，数据模型过于死板、无法较好支持Web2.0应用，事务机制影响了系统的整体性能等。

- NoSQL数据库
  优势：可以支持超大规模数据存储，灵活的数据模型可以很好地支持Web2.0应用，具有强 大的横向扩展能力等。
  劣势：缺乏数学理论基础，复杂查询性能不高，大都不能实现事务强一致性，很难实现数据 完整性，技术尚不成熟，缺乏专业团队的技术支持，维护较困难等。
  关系数据库和NoSQL数据库各有优缺点，彼此无法取代。
  关系数据库应用场景：电信、银行等领域的关键业务系统，需要保证强事务一致性。
  NoSQL数据库应用场景：互联网企业、传统企业的非关键业务（比如数据分析）。

三、NoSQL四大类型

典型的NoSQL数据库包括键值数据库、列组数据库、文档数据库和图数据库

1. 键值数据库
   键值数据库会使用一个哈希表，这个表中有一个特定的Key和一个指针指定特定的Value。Key可以用来定位Value，即存储和检索具体的Value。Value对数据库而言是透明不可见的，不能对Value进行检索和查询，只能通过Key进行查询

相关产品	Redis、Riak、SimpleDB、Chordless、Scalaris、Memcached
数据模型	键/值对： 键是一个字符串对象； 值可以是任意类型的数据，比如整型、字符型、数组、列表、集合等。
典型应用	涉及频繁读写、拥有简单数据模型的应用； 内容缓存，比如会话、配置文件、参数、购物车等； 存储配置和用户数据信息的移动应用。
优点	扩展性好，灵活性好，大量写操作时性能高
缺点	无法存储结构化信息，条件查询效率较低
不适用情形	不是通过键而是通过值来查：键值数据库根本没有通过值查询的途径。 需要存储数据之间的关系：在键值数据库中，不能通过两个或两个以上的键来关联数据。 需要事务的支持：在一些键值数据库中，产生故障时，不可以回滚。
使用者	百度云数据库（Redis）、GitHub（Riak）、BestBuy（Riak）、Twitter（Redis和Memcached）、StackOverFlow（Redis）、Instagram （Redis）、Youtube（Memcached）、Wikipedia（Memcached）

2. 列族数据库

相关产品	BigTable、HBase、Cassandra、HadoopDB、GreenPlum、PNUTS
数据模型	列族数据模型
典型应用	分布式数据存储与管理； 数据在地理上分布于多个数据中心的应用程序； 可以容忍副本中存在短期不一致情况的应用程序； 拥有动态字段的应用程序； 拥有潜在大量数据的应用程序，大到几百TB的数据。
优点	查找速度快，可扩展性强，容易进行分布式扩展，复杂性低
缺点	功能较少，大都不支持强事务一致性
不适用情形	需要ACID事务支持的情形，Cassandra等产品就不适用
使用者	Ebay（Cassandra）、Instagram（Cassandra）、NASA（Cassandra）、Twitter（Cassandra and HBase）、Facebook（HBase）、Yahoo!（HBase）

3. 文档数据库

文档其实是一个数据记录，这个记录能对包含的数据类型和内容进行“自我描述”

数据是不规则的，每一条记录包含所有的有关存储对象的信息而没有任何外部引用，这个记录是自包含的

这使得记录很容易完全移动到其他服务器，因为这条记录的所有信息都包含在里面了。不需要考虑还有信息在别的表没有一起迁移走

同时，因此在移动过程中，只有移动的那一条记录需要操作，而不像关系型中每一个有关联的表都需要锁住来保证一致性，这样一来ACID的保证就会变得更快速，读写的速度也会有更大的提升

相关产品	MongoDB、CouchDB、Terrastore、ThruDB、RavenDB、SisoDB、RaptorDB、CloudKit、Perservere、Jackrabbit。
数据模型	键/值 值（value）是版本化的文档。
典型应用	存储、索引并管理面向文档的数据或者类似的半结构化数据。比如，用于后台具有大量读写操作的网站、使用JSON数据结构的应用、使用嵌套结构等非规范化数据的应用程序。
优点	性能好（高并发），灵活性高，复杂性低，数据结构灵活。提供嵌入式文档功能，将经常查询的数据存储在同一个文档中，既可以根据键来构建索引，也可以根据内容构建索引。
缺点	缺乏统一的查询语法。
不适用情形	在不同的文档上添加事务。文档数据库并不支持文档间的事务，如果对这方面有需求则不应该选用这个解决方案。
使用者	百度云数据库（MongoDB）、SAP（MongoDB）、Codecademy（MongoDB）、Foursquare（MongoDB）、NBC News（RavenDB）

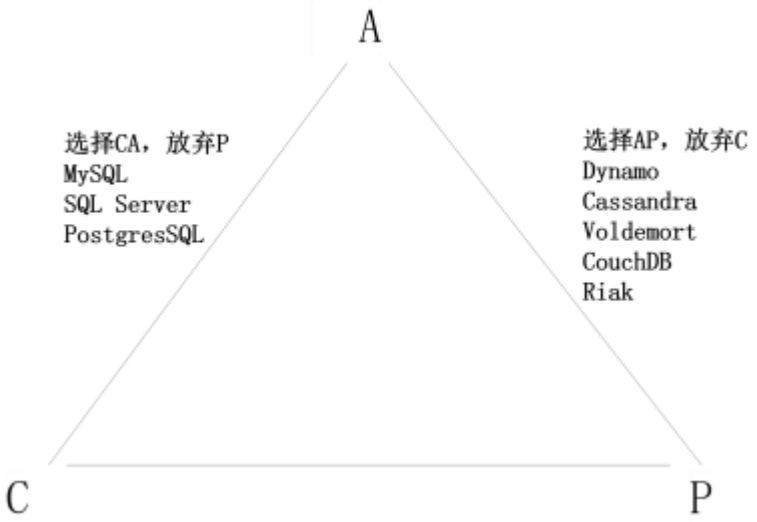
4. 图数据库

相关产品	Neo4J、OrientDB、InfoGrid、Infinite Graph、GraphDB
数据模型	图结构
典型应用	专门用于处理具有高度相互关联关系的数据，比较适合于社交网络、模式识别、依赖分析、推荐系统以及路径寻找等问题。
优点	灵活性高，支持复杂的图形算法，可用于构建复杂的关系图谱。
缺点	复杂性高，只能支持一定的数据规模。
使用者	Adobe（Neo4J）、Cisco（Neo4J）、T-Mobile（Neo4J）

四、NoSQL的三大基石

NoSQL的三大基石是CAP，最终一致性，BASE

1. CAP
   所谓的CAP理论指的是：
   - C(Consistency)：一致性，是指任何一个读操作总是能够读到之前完成的写操作的结果，也就是说在分布式环境中，多点的数据是一致的，或者说，所有节点在同一时间具有相同的数据
   - A(Availability)：可用性，是指快速获取数据，可以在确定的时间内返回操作结果，保证每个请求不管成功或者失败都有响应
   - P(Tolerance of Network Partition)：分区容忍性，是指当出现网络分区的情况时（即系统中的一部分节点无法和其他节点进行通信），分离的系统也能正常运行，也就是说，系统中任意信息的丢失或者失败不会影响系统的继续运作
     CAP理论告诉我们，一个分布式系统不肯同时满足一致性，可用性和分区容忍性这三个需求，最多只能同时满足其中两个



处理CAP问题的时候可以有三个选择：

- CA：强调一致性(C)与可用性(A)，放弃分区容忍性(P)，最简单的做法是将所有的事务相关的内容都放在同一台机器上，很显然这种做法会影响系统的可拓展性。传统的数据库（MySQL，SQL Server和PostgreSQL）都采用这种设计原则，拓展性都比较差
- CP：也就是强调一致性(C)和分区容忍性(P)，放弃可用性(A)，当出现网络分区的情况时，收影响的服务需要等待数据一致，因此在等待期间无法对外提供服务
- AP：也就是强调可用性(A)和分区容忍性（P)，放弃一致性(C)允许系统返回不一致的例子。例如对于微博会立即发布但是不会立即被读到，网站一般采用AP进行设计；牺牲一致性换取可用性，在面临对同一数据的并发读写操作一般写入修改后的值会传递失败，导致依然读取的旧值

2. BASE

对于Web2.0场景中，对数据一致性的要求并不是很高，为了获得系统的高可用性，可以考虑适当牺牲一致性和分区容忍性

BASE的基本含义是基本可用(Basically Available)，软状态(Soft-state)和最终一致性(Eventual consistency)

- 基本可用：是指一个分布式系统的一部分发生问题变得不可用的时候，其他部分仍然可以正常使用，也就是允许分区失败的情形出现。
- 软状态：软状态与硬状态相对应，硬状态可以保证数据一致性，即保证数据一直是正确的，软状态指的是状态有一段时间不同步，具有一定的滞后性
- 最终一致性：一致性包含强一致性和弱一致性，对于强一致性执行完更新操作后，后续的读操作就可以保证读到更新后的最新数据，反之不能保证的就是弱一致性。最终一致性是弱一致性的特例，允许更新后暂时读不到更新后的数据，但是经过一段时间之后必须读到更新后的数据

相比BASE，还有ACID，一个数据库事务具有ACID四性：

- A（Atomicity）：原子性，是指事务必须是原子工作单元，对于其数据修改，要么全都执行，要么全都不执行。
- C（Consistency）：一致性，是指事务在完成时，必须使所有的数据都保持一致状态。
- I（Isolation）：隔离性，是指由并发事务所做的修改必须与任何其它并发事务所做的修改隔离。

- D（Durability）：持久性，是指事务完成之后，它对于系统的影响是永久性的，该修改即使出现致命的系统故障也将一直保持。

五、文档数据库MongoDB

1. MongoDB简介
   MongoDB是由C++编写的，是一个基于分布式文件存储的开源数据库系统，属于文档数据库
   在高负荷的情况下，添加更多的节点，可以保证服务器性能
   MongoDB旨在为Web应用提供可拓展的高性能数据存储解决方案
   MongoDB将数据存储为一个文档，数据结构由键值（key=>value）对组成
   MongoDB文档类似JSON对象，字段值可以包含其他文档，数组以及文档数组
   主要特点：
   - 提供了一个面向文档存储，操作起来比较简单和容易。
   - 可以设置任何属性的索引来实现更快的排序。
   - 具有较好的水平可扩展性。
   - 支持丰富的查询表达式，可轻易查询文档中内嵌的对象及数组。•可以实现替换完成的文档（数据）或者一些指定的数据字段。
   - MongoDB中的Map/Reduce主要是用来对数据进行批量处理和聚合操作。
   - 支持各种编程语言:RUBY，PYTHON，JAVA，C++，PHP，C#等语言。
   - MongoDB安装简单。
2. MongoDB概念解析
   数据库
   - 一个mongodb中可以建立多个数据库。
   - MongoDB的默认数据库为"db"，该数据库存储在data目录中。
   - MongoDB的单个实例可以容纳多个独立的数据库，每一个都有自己的集合和权限，不同的数据库也放置在不同的文件中。
     文档
   - 文档是一个键值(key-value)对(即BSON)。MongoDB 的文档不需要设置相同的字段，并且相同的字段不需要相同的数据类型，这与关系型数据库有很大的区别，也是MongoDB 非常突出的特点。
     集合
   - 集合就是 MongoDB 文档组，类似于 RDBMS （关系数据库管理系统：Relational Database Management System)中的表格。
   - 集合存在于数据库中，集合没有固定的结构，这意味着你在集合中可以插入不同格式和类型的数据，但通常情况下我们插入集合的数据都会有一定的关联性。