关系数据库与nosql

Mongodb与关系型数据库相比的优缺点
与关系型数据库相比，MongoDB的优点：
①弱一致性（最终一致），更能保证用户的访问速度：
举例来说，在传统的关系型数据库中，一个COUNT类型的操作会锁定数据集，这样可以保证得到"当前"情况下的精确值。这在某些情况下，例 如通过ATM查看账户信息的时候很重要，但对于Wordnik来说，数据是不断更新和增长的，这种"精确"的保证几乎没有任何意义，反而会产生很大的延 迟。他们需要的是一个"大约"的数字以及更快的处理速度。
但某些情况下MongoDB会锁住数据库。如果此时正有数百个请求，则它们会堆积起来，造成许多问题。我们使用了下面的优化方式来避免锁定：
每次更新前，我们会先查询记录。查询操作会将对象放入内存，于是更新则会尽可能的迅速。在主/从部署方案中，从节点可以使用"-pretouch"参数运行，这也可以得到相同的效果。
使用多个mongod进程。我们根据访问模式将数据库拆分成多个进程。
②文档结构的存储方式，能够更便捷的获取数据。
对于一个层级式的数据结构来说，如果要将这样的数据使用扁平式的，表状的结构来保存数据，这无论是在查询还是获取数据时都十分困难。
③内置GridFS，支持大容量的存储。
GridFS是一个出色的分布式文件系统，可以支持海量的数据存储。
内置了GridFS了MongoDB，能够满足对大数据集的快速范围查询。
④内置Sharding。
提供基于Range的Auto Sharding机制：一个collection可按照记录的范围，分成若干个段，切分到不同的Shard上。
Shards可以和复制结合，配合Replica sets能够实现Sharding+fail-over，不同的Shard之间可以负载均衡。查询是对 客户端是透明的。客户端执行查询，统计，MapReduce等操作，这些会被MongoDB自动路由到后端的数据节点。这让我们关注于自己的业务，适当的 时候可以无痛的升级。MongoDB的Sharding设计能力最大可支持约20 petabytes，足以支撑一般应用。这可以保证MongoDB运行在便宜的PC服务器集群上。PC集群扩充起来非常方便并且成本很低，避免了"sharding"操作的复杂性和成本。
⑤第三方支持丰富。(这是与其他的NoSQL相比，MongoDB也具有的优势)
现在网络上的很多NoSQL开源数据库完全属于社区型的，没有官方支持，给使用者带来了很大的风险。
而开源文档数据库MongoDB背后有商业公司10gen为其提供供商业培训和支持。
而且MongoDB社区非常活跃，很多开发框架都迅速提供了对MongDB的支持。不少知名大公司和网站也在生产环境中使用MongoDB，越来越多的创新型企业转而使用MongoDB作为和Django，RoR来搭配的技术方案。
⑥性能优越：
在使用场合下，千万级别的文档对象，近10G的数据，对有索引的ID的查询不会比mysql慢，

而对非索引字段的查询，则是全面胜出。 mysql实际无法胜任大数据量下任意字段的查询，而mongodb的查询性能实在让我惊讶。写入性能同样很令人满意，同样写入百万级别的数 据，mongodb比我以前试用过的couchdb要快得多，基本10分钟以下可以解决。补上一句，观察过程中mongodb都远算不上是CPU杀手。

与关系型数据库相比，MongoDB的缺点：
①mongodb不支持事务操作。
__所以事务要求严格的系统（如果银行系统）肯定不能用它。(这点和优点①是对应的)
②mongodb占用空间过大。
__关于其原因，在官方的FAQ中，提到有如下几个方面：
1、空间的预分配：为避免形成过多的硬盘碎片，mongodb每次空间不足时都会申请生成一大块的硬盘空间，而且申请的量从64M、128M、256M那 样的指数递增，直到2G为单个文件的最大体积。随着数据量的增加，你可以在其数据目录里看到这些整块生成容量不断递增的文件。
2、字段名所占用的空间：为了保持每个记录内的结构信息用于查询，mongodb需要把每个字段的key-value都以BSON的形式存储，如果 value域相对于key域并不大，比如存放数值型的数据，则数据的overhead是最大的。一种减少空间占用的方法是把字段名尽量取短一些，这样占用 空间就小了，但这就要求在易读性与空间占用上作为权衡了。我曾建议作者把字段名作个index，每个字段名用一个字节表示，这样就不用担心字段名取多长 了。但作者的担忧也不无道理，这种索引方式需要每次查询得到结果后把索引值跟原值作一个替换，再发送到客户端，这个替换也是挺耗费时间的。现在的实现算是 拿空间来换取时间吧。
3、删除记录不释放空间：这很容易理解，为避免记录删除后的数据的大规模挪动，原记录空间不删除，只标记"已删除"即可，以后还可以重复利用。
4、可以定期运行db.repairDatabase()来整理记录，但这个过程会比较缓慢
③MongoDB没有如MySQL那样成熟的维护工具，这对于开发和IT运营都是个值得注意的地方。

区别	关系型数据库	非关系型数据库（Nosql）
存储方式	表格式存储。 存储在表的行和列中。他们之间很容易关联协作存储，提取数据很方便	而Nosql数据库则与其相反，他是大块的组合在一起。 通常存储在数据集中，就像文档、键值对或者图结构。
存储结构	**结构化数据。** 数据表都预先定义了结构（列的定义），结构描述了数据的形式和内容。这一点对数据建模至关重要，虽然预定义结构带来了可靠性和稳定性（优点），但是修改这些数据比较困难（缺点）。	而Nosql数据库基于动态结构，使用与**非结构化数据**。因为Nosql数据库是动态结构，可以很容易适应数据类型和结构的变化。
存储规范	数据存储为了更高的规范性，把数据分割为最小的关系表以避免重复，获得精简的空间利用。虽然管理起来很清晰，但是单个操作设计到多张表的时候，数据管理就显得有点麻烦	而Nosql数据存储在平面数据集中，数据经常可能会重复。单个数据库很少被分隔开，而是存储成了一个整体，这样整块数据更加便于读写
存储扩展	系型数据库是**纵向扩展**，也就是说想要提高处理能力，要使用速度更快的计算机。因为数据存储在关系表中，操作的性能瓶颈可能涉及到多个表，需要通过提升计算机性能来克服。虽然有很大的扩展空间，但是最终会达到纵向扩展的上限	而Nosql数据库是**横向扩展**的，它的存储天然就是分布式的，可以通过给资源池添加更多的普通数据库服务器来分担负载。
查询方式	结构化查询语言来操作数据库（就是我们通常说的SQL） **关系型数据库表中主键** 关系型数据库使用预定义优化方式（比如索引）来加快查询操作	以块为单元操作数据，使用的是非结构化查询语言（UnQl），它是没有标准的 **Nosql中存储文档的ID** 更简单更精确的数据访问模式
事务	**遵循ACID规则**（原子性(Atomicity)、一致性(Consistency)、隔离性(Isolation)、持久性(Durability)）支持对事务原子性细粒度控制，并且易于回滚事务。	**遵循BASE原则**（基本可用（Basically Availble）、软/柔性事务（Soft-state ）、最终一致性（Eventual Consistency）） Nosql数据库是在CAP（一致性、可用性、分区容忍度）中任选两项，因为基于节点的分布式系统中，很难全部满足，所以对事务的支持不是很好，虽然也可以使用事务，但是并不是Nosql的闪光点。
性能	为了维护数据的一致性付出了巨大的代价，读写性能比较差。在面对高并发读写性能非常差，面对海量数据的时候效率非常低。	Nosql存储的格式都是key-value类型的，并且存储在内存中，非常容易存储，而且对于数据的 一致性是 弱要求。Nosql无需sql的解析，提高了读写性能。
授权方式	关系型数据库通常有SQL Server，Mysql，Oracle。大多数的关系型数据库都是付费的并且价格昂贵，成本较大。	主流的Nosql数据库有redis，memcache，MongoDb。 而Nosql数据库通常都是开源的。

数据库类型	特性	优点	缺点
关系型数据库 SQLite、Oracle、mysql	1、关系型数据库，是指采用了关系模型来组织 数据的数据库； 2、关系型数据库的最大特点就是事务的一致性； 3、简单来说，关系模型指的就是二维表格模型， 而一个关系型数据库就是由二维表及其之间的联系所组成的一个数据组织。	1、容易理解：二维表结构是非常贴近逻辑世界一个概念，关系模型相对网状、层次等其他模型来说更容易理解； 2、使用方便：通用的SQL语言使得操作关系型数据库非常方便； 3、易于维护：丰富的完整性(实体完整性、参照完整性和用户定义的完整性)大大减低了数据冗余和数据不一致的概率； 4、支持SQL，可用于复杂的查询。	1、为了维护一致性所付出的巨大代价就是其读写性能比较差； 2、固定的表结构； 3、高并发读写需求； 4、海量数据的高效率读写；
非关系型数据库 MongoDb、redis、HBase	1、使用键值对存储数据； 2、分布式； 3、一般不支持ACID特性； 4、非关系型数据库严格上不是一种数据库，应该是一种数据结构化存储方法的集合。	1、无需经过sql层的解析，读写性能很高； 2、基于键值对，数据没有耦合性，容易扩展； 3、存储数据的格式：nosql的存储格式是key,value形式、文档形式、图片形式等等，文档形式、图片形式等等，而关系型数据库则只支持基础类型。	1、不提供sql支持，学习和使用成本较高； 2、无事务处理，附加功能bi和报表等支持也不好；

四类NoSQL数据库比较

分类	相关产品	典型应用	数据模型	优点	缺点
键值（key-value）	Tokyo Cabinet/Tyrant、Redis、Voldemort、Berkeley DB	内容缓存，主要用于处理大量数据的高访问负载	一系列键值对	快速查询	存储的数据缺少结构化
列存储数据库	Cassandra, HBase, Riak	分布式的文件系统	以列簇式存储，将同一列数据存在一起	查找速度快，可扩展性强，更容易进行分布式扩展	功能相对局限
文档型数据库	CouchDB, MongoDb	Web应用（与Key-Value类似，Value是结构化的）	一系列键值对	数据结构要求不严格	查询性能不高，而且缺乏统一的查询语法。
图形(Graph)数据库	Neo4J, InfoGrid, Infinite Graph	社交网络，推荐系统等。专注于构建关系图谱	图结构	利用图结构相关算法	需要对整个图做计算才能得出结果，不容易做分布式的集群方案