应用特征Yes / No

应用不需要事务及复杂 join 支持必须 Yes

新应用，需求会变，数据模型无法确定，想快速迭代开发？

应用需要2000-3000以上的读写QPS（更高也可以）？

应用需要TB甚至 PB 级别数据存储?

应用发展迅速，需要能快速水平扩展?

应用要求存储的数据不丢失?

应用需要99.999%高可用?

应用需要大量的地理位置查询、文本查询？

如果上述有1个 Yes，可以考虑 MongoDB，2个及以上的 Yes，选择MongoDB绝不会后悔。

MySQL是关系型数据库，使用简单，方便，有统一的SQL语句，支持事务一致性。

MongoDB是文档式数据库，必须使用专用的API进行操作，不过大部分操作都绕不过增删改查，学习起来也方便，不支持事务一致性。

MySQL里数据模型是二维线性表，里面每一个元素都是不可再分的原子，而且同一列的数据类型是相同的。

MongoDB数据模型就比较灵活，一张表就是一个collection，里面的每一个数据都是一个document，一般MongoDB的管理软件都把一个数据用json格式来显示。

分布式

1.MongoDB适用于以下场景：

  a.网站数据：mongo非常适合实时的插入，更新与查询，并具备网站实时数据存储所需的复制及高度伸缩性。

  b.缓存：由于性能很高，mongo也适合作为信息基础设施的缓存层。在系统重启之后，由mongo搭建的持久化缓存可以避免下层的数据源过载。

  c.大尺寸、低价值的数据：使用传统的关系数据库存储一些数据时可能会比较贵，在此之前，很多程序员往往会选择传统的文件进行存储。

  d.高伸缩性的场景：mongo非常适合由数十或者数百台服务器组成的数据库。

  e.用于对象及JSON数据的存储：mongo的BSON数据格式非常适合文档格式化的存储及查询。

2.不适合的场景：

  a.高度事物性的系统：例如银行或会计系统。传统的关系型数据库目前还是更适用于需要大量原子性复杂事务的应用程序。

  b.传统的商业智能应用：针对特定问题的BI数据库会对产生高度优化的查询方式。对于此类应用，数据仓库可能是更合适的选择。

  c.需要SQL的问题。

————————————————

版权声明：本文为CSDN博主「stalin\_」的原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/stalin\_/article/details/83819860

二、特点：

1.面向集合存储，易于存储对象类型的数据。

2.模式自由。没有固定的模式，文档的键不会事先定义，也不会写死。所以更改数据模型相当简便。

3.支持动态查询。

4.支持完全索引，包含内部对象。

5.支持复制与对象恢复。

6.使用高效的二进制数据存储，支持大型对象（如视频等）

7.自动处理碎片，以支持云计算层次的扩展性。

8.文档存储格式为BSON。

9.支持当前大多数主流编程语言的驱动程序。

————————————————

版权声明：本文为CSDN博主「stalin\_」的原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/stalin\_/article/details/83819860

mongodb的性能和mysql差不多，存储方式不一样，mysql还支持事务等等，mongodb是json对象存储，可以存对象（可以内嵌（会有数据冗余，查询速度快点）也可以引用）

1、MongoDB的应用场景

1）表结构不明确且数据不断变大

MongoDB是非结构化文档数据库，扩展字段很容易且不会影响原有数据。内容管理或者博客平台等，例如圈子系统，存储用户评论之类的。

2）更高的写入负载

MongoDB侧重高数据写入的性能，而非事务安全，适合业务系统中有大量“低价值”数据的场景。本身存的就是json格式数据。例如做日志系统。

3）数据量很大或者将来会变得很大

Mysql单表数据量达到5-10G时会出现明细的性能降级，需要做数据的水平和垂直拆分、库的拆分完成扩展，MongoDB内建了sharding、很多数据分片的特性，容易水平扩展，比较好的适应大数据量增长的需求。

4）高可用性

自带高可用，自动主从切换（副本集）

不适用的场景

1）MongoDB不支持事务操作，需要用到事务的应用建议不用MongoDB。

2）MongoDB目前不支持join操作，需要复杂查询的应用也不建议使用MongoDB。

2、关系型数据库和非关系型数据库的应用场景对比

关系型数据库适合存储结构化数据，如用户的帐号、地址：

1）这些数据通常需要做结构化查询，比如join，这时候，关系型数据库就要胜出一筹

2）这些数据的规模、增长的速度通常是可以预期的

3）事务性、一致性

NoSQL适合存储非结构化数据，如文章、评论：

1）这些数据通常用于模糊处理，如全文搜索、机器学习

2）这些数据是海量的，而且增长的速度是难以预期的，

3）根据数据的特点，NoSQL数据库通常具有无限（至少接近）伸缩性

4）按key获取数据效率很高，但是对join或其他结构化查询的支持就比较差

OLTP

OLAP

作者：李小翀

链接：https://www.zhihu.com/question/26518864/answer/33150665

来源：知乎

著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。

传统数据库侧重交易处理，即OLTP，关注的是多用户的同时的双向操作，在保障即时性的要求下，系统通过内存来处理数据的分配、读写等操作，存在IO瓶颈。

OLTP（On-Line Transaction Processing，联机事务处理）系统也称为生产系统，

它是事件驱动的、面向应用的，比如电子商务网站的交易系统就是一个典型的OLTP系统。

OLTP的基本特点是：

数据在系统中产生；

基于交易的处理系统（Transaction-Based）；

每次交易牵涉的数据量很小；

对响应时间要求非常高；

用户数量非常庞大，主要是操作人员；

数据库的各种操作主要基于索引进行。

分析型数据库是以实时多维分析技术作为基础，即侧重OLAP，对数据进行多角度的模拟和归纳，从而得出数据中所包含的信息和知识。

OLAP（On-Line Analytical Processing，联机分析处理）是基于数据仓库的信息分析处理过程，是数据仓库的用户接口部分。OLAP系统是跨部门的、面向主题的，其基本特点是：

本身不产生数据，其基础数据来源于生产系统中的操作数据（OperationalData）；

基于查询的分析系统；

复杂查询经常使用多表联结、全表扫描等，牵涉的数据量往往十分庞大；

响应时间与具体查询有很大关系；

用户数量相对较小，其用户主要是业务人员与管理人员；

支持基于内存查询的数据库，减少磁盘IO交互。支持复杂多变数据存储结构与类型。支持集群架构保证高可用。支持复杂的SQL查询，例如基于预算，来搜索满足条件的航班;或者基于航班时刻(上下午)，来搜索满足条件的航班等。

综合以上标准进行评估，最后我们选择了MongoDB。