MongoDB & Elastic Search

MongoDB为什么比mysql快

写操作MongoDB比传统数据库快的根本原因是Mongo使用的内存映射技术 - 写入数据时候只要在内存里完成就可以返回给应用程序，这样并发量自然就很高。而保存到硬体的操作则在后台异步完成。注意MongoDB在2.4就已经是默认安全写了（具体实现在驱动程序里），所以楼上有同学的回答说是"默认不安全"应该是基于2.2或之前版本的。

读操作MongoDB快的原因是： 1）MongoDB的设计要求你常用的数据（working set)可以在内存里装下。这样大部分操作只需要读内存，自然很快。 2）文档性模式设计一般会是的你所需要的数据都相对集中在一起（内存或硬盘），大家知道硬盘读写耗时最多是随机读写所产生的磁头定位时间，数据集中在一起则减少了关系性数据库需要从各个地方去把数据找过来（然后Join）所耗费的随机读时间

另外一个就是如@王子亭所提到的Mongo是分布式集群所以可以平行扩展。目前一般的百万次并发量都是通过几十上百个节点的集群同时实现。这一点MySQL基本无法做到（或者要花很大定制的代价）

mongodb索引 采用的wiredTiger 引擎，是按照b-tree(2-3树)的形式来组织的，进行了扩展，叶子节点存储了key 和 数据

B-Tree是为磁盘或其它辅助存储设备而设计的一种数据结构，目的是为了在查找数据的过程中减少磁盘I/O的次数

在整个B-Tree中，从上往下依次为Root结点、内部结点和叶子结点，每个结点就是一个Page，数据以Page为单位在内存和磁盘间进行调度，每个Page的大小决定了相应结点的分支数量，每条索引记录会包含一个数据指针，指向一条数据记录所在文件的偏移量。

为什么 MongoDB （索引）使用B-树而 Mysql 使用 B+树

B-树是一类树，包括B-树、B+树、B*树等，是一棵自平衡的搜索树，它类似普通的平衡二叉树，不同的一点是B-树**允许每个节点有更多的子节点**。B-树是专门为外部存储器设计的，如磁盘，它对于读取和写入大块数据有良好的性能，所以一般被用在文件系统及数据库中

平衡二叉树有很多，如 AVL 树，红黑树等。这些树在一般情况下查询性能非常好，但当数据非常大的时候它们就无能为力了。原因当数据量非常大时，内存不够用，大部分数据只能存放在磁盘上，只有需要的数据才加载到内存中。平衡二叉树高度相对较大log n，逻辑上很近的节点可能会比较远，无法很好利用磁盘预读（局部性原理），故在数据库和文件系统不使用平衡二叉树

B+树是B-树的变种，它与B-树的不同之处在于：

- 在B+树中，key 的副本存储在内部节点，真正的 key 和 data 存储在叶子节点上
- n 个 key 值的节点指针域为 n 而不是 n+1

内节点并不存储 data，所以一般B+树的叶节点和内节点大小不同，而B-树的每个节点大小一般是相同的，为一页。

为了增加 区间访问性，一般会对B+树做一些优化

B树的概念

https://blog.csdn.net/v_JULY_v/article/details/6530142

B-树和B+树的区别

1.B+树内节点不存储数据，所有 data 存储在叶节点导致查询时间复杂度固定为 log n。而B-树查询时间复杂度不固定，与 key 在树中的位置有关，最好为O(1)。

2.B+树叶节点两两相连可大大增加区间访问性，可使用在范围查询等，而B-树每个节点 key 和 data 在一起，则无法区间查找

3.B+树更适合外部存储。由于内节点无 data 域，每个节点能索引的范围更大更精确

为什么 MongoDB 索引选择B-树，而 Mysql 索引选择B+树

- MongoDB 是一种 nosql，也存储在磁盘上，被设计用在 数据模型简单，性能要求高的场合。MongoDB 是聚合型数据库，而 B-树恰好 key 和 data 域聚合在一起
- Mysql 是一种关系型数据库，区间访问是常见的一种情况，而 B-树并不支持区间访问（可参见上图），而B+树由于数据全部存储在叶子节点，并且通过指针串在一起，这样就很容易的进行区间遍历甚至全部遍历

ElasticsSearch vs MongoDB

https://www.sohu.com/a/283594658_505800

es和mongoDB分片及高可用对比

https://www.cnblogs.com/lazio10000/p/8111719.html

es倒排索引

分词后，作为键值，倒排列表为ids主键集合（1.2.3.5）

mongodb是真正的内存数据库，与redis一样?

不一样，mongoDB是关系映射型数据库，真正的数据还是存储在文件中