MongoDB 面试题

以下面试题，基于网络整理，和自己编辑。具体参考的文章，会在文末给出所有的链接。

如果胖友有自己的疑问，欢迎在星球提问，我们一起整理吊吊的 MongoDB 面试题的大保健。

而题目的难度，尽量按照从容易到困难的顺序，逐步下去。

> ：对于 MongoDB 的面试题的准备，我的想法是看过一遍即可，更大的重心在于 MySQL 的面试题的准备。
>
> 因为，从目前的技术风向标来说，大家在技术选项上，更偏向 MySQL + 分库分表，解决海量数据存储的问题。同时 NewSQL 也慢慢走进大家的视野之中，例如 TiDB 。

什么是 NoSQL 数据库?

NoSQL 是非关系型数据库，NoSQL = Not Only SQL 。

📣 NoSQL 和 RDBMS 有什么区别?

- 关系型数据库，采用的结构化的数据。
- NoSQL 采用的是键值对的方式存储数据。

📣 在哪些情况下使用和不使用 NoSQL 数据库?

> ：如下是我从网上找到的答案，怎么看着会觉得怪怪的。

优先考虑使用 NoSQL 数据库

- 在处理非结构化 / 半结构化的大数据
- 在水平方向上进行扩展
- 随时应对动态增加的数据项

优先考虑关系型数据库

- 在考虑数据库的成熟度
- 分析和商业智能
- 管理及专业性等问题

> ：下面我说下自己的观点

- 信息表(例如，商品信息)，使用 NoSQL 数据库不错，因为 NoSQL 往往提供灵活的数据类型。
- 交易表，使用关系数据库，因为往往需要关注事务性。
- 对 BI 分析等，使用关系数据库，因为周边设施更加完善。

📣 非关系型数据库有哪些?

比较常见的三个是：

- MongoDB
- Redis
- Elasticsearch

什么是 MongoDB？

> MongoDB 是一个介于关系数据库和非关系数据库之间的产品，是非关系数据库当中功能最丰富，最像关系数据库的。
>
> - 它支持的数据结构非常松散，是类似 jSON 的 BSON 格式，因此可以存储比较复杂的数据类型。
> - Mongo 最大的特点是他支持的查询语言非常强大，其语法有点类似于面向对象的查询语言，几乎可以实现类似关系数据库单表查询的绝大部分功能，而且还支持对数据建立索引。

综合来说，MongoDB 可以说是最棒的文档数据库。

📣 MongoDB的特点是什么?

- 面向集合的存储：适合存储对象及 JSON 形式的数据。

  > 这个也是 MongoDB 和 MySQL 之间最基本的级别。

- 动态查询：MongoDB 支持丰富的查询表达式。查询指令使用 JSON 形式的标记，可轻易查询文档中内嵌的对象及数组。
- 完整的索引支持：包括文档内嵌对象及数组。Mongo的查询优化器会分析查询表达式，并生成一个高效的查询计划。
- 查询监视：MongoDB 包含一个监视工具用于分析数据库操作的性能。
- 复制及自动故障转移：MongoDB 数据库支持服务器之间的数据复制，支持主-从模式及服务器之间的相互复制。复制的主要目标是提供冗余及自动故障转移。
- 高效的传统存储方式：支持二进制数据及大型对象（如照片或图片）。
- 自动分片以支持云级别的伸缩性：自动分片功能支持水平的数据库集群，可动态添加额外的机器。

📣 术语和概念?

MySQL 中的许多概念在 MongoDB 中具有相近的类比。本表概述了每个系统中的一些常见概念。

MySQL	MongoDB
库 Database	库 Database
表 Table	集合 Collection
行 Row	文档 Document
列 Column	字段 Field
joins	嵌入文档或者链接

- MongoDB 数据库，可以看成是一个电子化的文件柜，用户可以对文件中的数据运行新增、检索、更新、删除等操作。数据库是一个所有集合的容器，在文件系统中每一个数据库都有一个相关的物理文件。
- MongoDB 集合，就是一组 MongoDB 文档。它相当于关系型数据库（RDBMS）中的表这种概念。集合位于单独的一个数据库中。一个集合内的多个文档可以有多个不同的字段。一般来说，集合中的文档都有着相同或相关的目的。
- MongoDB 文档，由一组 key value 组成。文档是动态模式，这意味着同一集合里的文档不需要有相同的字段和结构。在关系型数据库中 Table 中的每一条记录相当于 MongoDB 中的一个文档。

> 不过一般情况下，相同 Collection 下的 Document 字段是统一的。

📣 MySQL 与 MongoDB 的功能对比?

	MySQL	MongoDB
丰富的数据模型	否	是
动态 Schema	否	是
数据类型	是	是
数据本地化	否	是
字段更新	是	是
易于编程	否	是
复杂事务	是	否
审计	是	是
自动分片	否	是

- 这是一个比较老的比较。
- MySQL5.7 版本开始，也提供了 JSON 类型的数据格式。
- MongoDB4.0 版本开始，也提供了事务功能。
- 关于是否易于编程，这个看每个人的自己的想法，实际都还可以。

MongoDB 有哪些存储引擎?

从 《MongoDB Documentation —— Storage Engines》 中，我们看到 MongoDB 有三种存储引擎：

- WiredTiger Storage Engine **默认**
- In-Memory Storage Engine
- MMAPv1 Storage Engine **(Deprecated as of MongoDB 4.0)**

具体的对比，可以看看 《MongoDB 存储引擎选择》 文章。

- 生产环境下，基本使用的都是 WiredTiger Storage Engine ，因为各方面性能都优于 MMAPv1 Storage Engine 。
- 如果真的需要 In-Memory Storage Engine 的情况，Redis 或者是一个更加的选择。

MongoDB 支持哪些数据类型?

- String
- Integer
- Double
- Boolean
- Object
- ObjectId
- Arrays
- Min/Max Keys

- Datetime
- Code
- Regular Expression
- ... 等等

📣 **为什么要在 MongoDB 中用"Code"数据类型?**

"Code" 类型，用于在文档中存储 JavaScript 代码。决绝绝大多数业务场景下，我们并不会使用这个数据类型。

📣 **为什么要在 MongoDB 中用"Regular Expression"数据类型?**

"Regular Expression" 类型，用于在文档中存储正则表达式。

📣 **为什么在 MongoDB 中使用"ObjectId"数据类型?**

"ObjectId" 数据类型，用于存储文档 id 。关于 ObjectId 的组成，可以看看 《MongoDB 深究之 ObjectId》 。

- 另外，ObjectId 天然就是分布式主键的实现，所以在 MongoDB 分片后，依然可以继续使用。

当然，对于绝大数业务场景下，我们还是希望能够使用自增 ID ，可以参看 《Java 中实现 MongoDB 自增主键 ID》 。

- 另外，使用 SnowFlake 等等，也是一种方式。

📣 **如何理解 MongoDB 中的 GridFS 机制，MongoDB 为何使用 GridFS 来存储文件?**

GridFS 是一种将大型文件存储在 MongoDB 中的文件规范。使用 GridFS 可以将大文件分隔成多个小文档存放，这样我们能够有效的保存大文档，而且解决了 BSON 对象有限制的问题。

当然，实际生产环境下，建议使用专门的文件服务器，例如 FastDFS、TFS 等，而不是使用 MongoDB GridFS 。

📣 **MongoDB 支持存储过程吗？如果支持的话，怎么用?**

MongoDB 支持存储过程，它是 Javascript 写的，保存 `db.system.js` 表中。

当然，和 MySQL 一样，实际场景下，不会使用存储过程。

MongoDB 为什么选择 B-Tree 索引?

在 《精尽 MySQL 面试题》 中，我们已经看到 MySQL 使用的是 B+Tree 索引。

- B+Tree 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) 。
- B-Tree 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) 。

我们知道，尽可能少的磁盘 IO 是提高性能的有效手段。MongoDB 是聚合型数据库，而 B-Tree 恰好 key 和 data 域聚合在一起。

至于 MongoDB 为什么使用 B-Tree 而不是 B+Tree ，可以从它的设计角度来考虑，它并不是传统的关系性数据库，而是以 JSON 格式作为存储的 NoSQL ，目的就是高性能，高可用，易扩展。

- MySQL 由于使用 B+ 树，数据都在叶节点上，每次查询都需要访问到叶节点，而 MongoDB 使用 B-Tree ，所有节点都有 data 域，只要找到指定索引就可以进行访问，无疑单次查询平均快于 MySQL 。

具体的讨论，可以看看 《为什么有关 MongoDB 采用 B 树索引，以及 MySQL B+ 树做索引? 》 上的讨论。当然，也觉得这个回答可能有点牵强。列出这个问题的目的在于，让我们知道 MongoDB 并不是使用 B+Tree 来实现索引。

📣 **MongoDB 在 A:{B,C} 上建立索引，查询 A:{B,C} 和 A:{C,B} 都会使用索引吗?**

因为 MongoDB 使用 B-Tree 索引，实际上查询和 MySQL 的 B+Tree 索引是基本一致的。

- A:{B,C} 上，可以完整使用索引。
- A:{C,B} 上，只能部分使用索引，只有 A 部分。

什么是 MongoDB 聚合操作?

聚合操作，能够处理数据记录并返回计算结果。聚合操作能将多个文档中的值组合起来，对成组数据执行各种操作，返回单一的结果。它相当于 SQL 中的 `COUUNT(*)` 组合 `GROUP BY` 。

对于 MongoDB 中的聚合操作，应该使用 aggregate 方法。具体使用，可以看看 《MongoDB Documentation —— aggregate》 。

MongoDB 如何实现高可用?

和 MySQL 一样，MongoDB 也提供了其复制方案，为实现高可用提供了基础。目前，MongoDB 支持两种复制模式：

- Master / Slave ， 主从复制，角色包括 Master 和 Slave 。
- Replica Set ， 复制集复制，角色包括 Primary 和 Secondary 以及 Arbiter 。

生产环境下，只会使用 Replica Set 复制级的方式，实现 MongoDB 的高可用。主要因为，它提供了良好的自动故障切换功能。具体的搭建方式，可以看看 《MongoDB 集群搭建及使用》 。

📣 什么是 Primary?

Primary，它是当前备份集群(replica set)中负责处理所有写入操作的主要节点/成员。在一个复制集集群中，当失效备援(failover)事件发生时，Secondary 中的一个成员会变成新的 Primary。

📣 什么是 Secondary?

Secondary ，从当前的 Primary上 复制相应的操作。它是通过跟踪复制 `oplog(local.oplog.rs)` 做到的。

📣 MongoDB 如何实现读写分离?

在程序中，我们可以配置读取 Secondary 节点的数据，从而实现读写分离。例如说，每天晚上的 T+1 数据统计，可以读取 Secondary 节点的数据。具体可以参见 《Mongodb 的读写分离使用 Replica Sets 来实现》 文章。

📣 什么是 MongoDB 的延迟节点?

参见文章 《Mongodb 延迟复制节点配置》 。

另外，MongoDB 的延迟节点，是不对外提供服务，所以在其上做全量备份，也是非常不错的选择。

MongoDB 如何实现分片?

MongoDB 分片，是将数据水平切分到不同的物理节点。当应用数据越来越大的时候，数据量也会越来越大。当数据量增长时，单台机器有可能无法存储数据或可接受的读取写入吞吐量。利用分片技术可以添加更多的机器来应对数据量增加以及读写操作的要求。

> 或者，我们可以将 MongoDB 分片理解成内置的分库分表功能。

具体如何使用，可以看看 《MongoDB 分片》 文章。

📣 我应该启动一个集群分片(sharded)还是一个非分片集群的 MongoDB 环境?

为开发便捷起见，我们建议以非集群分片(unsharded)方式开始一个 MongoDB 环境，除非一台服务器不足以存放你的初始数据集。从非集群分片升级到集群分片(sharding)是无缝的，所以在你的数据集还不是很大的时候没必要考虑集群分片(sharding)。

另外，引入 MongoDB 分片后，会带来相应的运维复杂性，所以在 MongoDB 复制集能够支撑当前业务的情况下，不要过早的使用 MongoDB 分片。

📣 分片(Shard)和复制(replication)是怎样工作的?

每一个分片(shard)是一个分区数据的逻辑集合。分片可能由单一服务器或者集群组成，我们推荐为每一个分片(shard)使用集群。

📣 数据在什么时候才会扩展到多个分片(Shard)里?

MongoDB 分片是基于区域(range)的。所以一个集合(collection)中的所有的对象都被存放到一个块(chunk)中。只有当存在多余一个块的时候，才会有多个分片获取数据的选项。

现在，每个默认块的大小是 64Mb，所以你需要至少 64 Mb 空间才可以实施一个迁移。

📣 当更新一个正在被迁移的块（Chunk）上的文档时会发生什么?

更新操作会立即发生在旧的块（Chunk）上，然后更改才会在所有权转移前复制到新的分片上。

📣 如果一个分片（Shard）停止或很慢的时候，发起一个查询会怎样?

如果一个分片停止了，除非查询设置了 “Partial” 选项，否则查询会返回一个错误。如果一个分片响应很慢，MongoDB 会等待它的响应。

📣 我可以把 moveChunk 目录里的旧文件删除吗?

没问题，这些文件是在分片(shard)进行均衡操作(balancing)的时候产生的临时文件。一旦这些操作已经完成，相关的临时文件也应该被删除掉。

但目前清理工作是需要手动的，所以请小心地考虑再释放这些文件的空间。

📣 如果块移动操作(moveChunk)失败了，我需要手动清除部分转移的文档吗?

不需要，移动操作是一致(consistent)并且是确定性的(deterministic)。

- 一次失败后，移动操作会不断重试。
- 当完成后，数据只会出现在新的分片里(shard)。

为什么要在 MongoDB 中使用分析器?

数据库分析工具(Database Profiler)，会针对正在运行的 mongod 实例收集数据库命令执行的相关信息。

- 包括增删改查的命令以及配置和管理命令。
- 分析器(profiler) 会写入所有收集的数据到 `system.profile` 集合，一个 capped 集合在管理员数据库。

分析器默认是关闭，的你能通过 per 数据库或 per 实例开启。

聊聊 MongoDB 备份?

和 MySQL 备份方式和策略类似，MongoDB 也需要定期的全量备份，以及定期的增量备份。具体可以看看《MongoDB 增量备份方案》和《Mongodb 增量备份脚本与原理》。

📣 journal 回放在条目(entry)不完整时(比如恰巧有一个中途故障了)会遇到问题吗?

每个 journal (group) 的写操作都是一致的，除非它是完整的，否则在恢复过程中它不会回放。

📣 更新操作立刻 fsync 到磁盘?

不会，磁盘写操作默认是延迟执行的。写操作可能在两三秒(默认在60秒内)后到达磁盘。

- 例如，如果一秒内数据库收到一千个对一个对象递增的操作，仅刷新磁盘一次。
- 通过 `syncPeriodSecs` 启动参数，可以进行配置。

> mongod 使用 fsync 操作将数据 flush 到磁盘的时间间隔，默认值为 60（单位：秒），强烈建议不要修改此值。
>
> mongod 将变更的数据写入journal后再写入内存，并间歇性的将内存数据 flush 到磁盘中，即延迟写入磁盘，有效提升磁盘效率。

📣 为什么我的数据文件如此庞大?

MongoDB 会积极的预分配预留空间，来防止文件系统碎片。

彩蛋

许久不用 MongoDB ， 生产基本换成 MySQL ， 因为对事务性的要求更高。

参考与推荐如下文章：

- genuinecx 《MongoDB 经典面试题集锦》
- 搜云库技术团队 《Mysql InnoDB B+树索引和哈希索引的区别？ MongoDB 为什么使用 B-树?》
- 加多 《MongoDB 和 MySQL 对比(译)》
- longz 《MongoDB 面试题问题以及参考答案》