**MySQL**

## 1.事务四大特性

原子性：不可分割的操作单元，事务中所有操作，要么全部成功；要么撤回到执行事务之前的状态

一致性：如果在执行事务之前数据库是一致的，那么在执行事务之后数据库也还是一致的；

隔离性：事务操作之间彼此独立和透明互不影响。事务独立运行。这通常使用锁来实现。一个事务处理后的结果，影响了其他事务，那么其他事务会撤回。事务的100%隔离，需要牺牲速度。

持久性：事务一旦提交，其结果就是永久的。即便发生系统故障，也能恢复。

## 2.MySQL的事务隔离级别

未提交读(Read Uncommitted)：允许脏读，其他事务只要修改了数据，即使未提交，本事务也能看到修改后的数据值。也就是可能读取到其他会话中未提交事务修改的数据

提交读(Read Committed)：只能读取到已经提交的数据。Oracle等多数数据库默认都是该级别 (不重复读)。

可重复读(Repeated Read)：可重复读。无论其他事务是否修改并提交了数据，在这个事务中看到的数据值始终不受其他事务影响。

串行读(Serializable)：完全串行化的读，每次读都需要获得表级共享锁，读写相互都会阻塞

**MySQL数据库(InnoDB引擎)默认使用可重复读（ Repeatable read)**

## 3.什么是索引，四大类索引分别是什么

数据库索引，是数据库管理系统中一个排序的数据结构，以协助快速查询、更新数据库表中数据。索引的实现通常使用 B\_TREE。B\_TREE 索引加速了数据访问，因为存储引擎不会再去扫描整张表得到需要的数据；相反，它从根节点开始，根节点保存了子节点的指针，存储引擎会根据指针快速寻找数据。

index  ----  普通索引,数据可以重复，没有任何限制。  
unique ---- 唯一索引,要求索引列的值必须唯一，但允许有空值；如果是组合索引，那么列值的组合必须唯一。

primary key ---- 主键索引,是一种特殊的唯一索引，一个表只能有一个主键，不允许有空值，一般是在创建表的同时创建主键索引。

组合索引 ----  在多个字段上创建的索引，只有在查询条件中使用了创建索引时的第一个字段，索引才会被使用。

fulltext ---- 全文索引,是对于大表的文本域：char，varchar，text列才能创建全文索引，主要用于查找文本中的关键字，并不是直接与索引中的值进行比较。fulltext更像是一个搜索引擎，配合match against操作使用，而不是一般的where语句加like。

注:全文索引目前只有MyISAM存储引擎支持全文索引，InnoDB引擎5.6以下版本还不支持全文索引

## 4.数据库三范式

第一范式：1NF是对属性的原子性约束，要求字段具有原子性，不可再分解；(只要是关系型数据库都满足1NF)

第二范式：2NF是在满足第一范式的前提下，非主键字段不能出现部分依赖主键；解决：消除复合主键就可避免出现部分以来，可增加单列关键字。

第三范式：3NF是在满足第二范式的前提下，非主键字段不能出现传递依赖，比如某个字段a依赖于主键，而一些字段依赖字段a，这就是传递依赖。解决：将一个实体信息的数据放在一个表内实现。

## 5.存储引擎 MyISAM和InnoDB区别

1）InnoDB支持事务，MyISAM不支持。

2）MyISAM适合查询以及插入为主的应用，InnoDB适合频繁修改以及涉及到安全性较高的应用。

3）InnoDB支持外键，MyISAM不支持。

4）从MySQL5.5.5以后，InnoDB是默认引擎。

5）MyISAM支持全文类型索引，而InnoDB不支持全文索引。

6）InnoDB中不保存表的总行数，select count(\*) from table时，InnoDB需要扫描整个表计算有多少行，但MyISAM只需简单读出保存好的总行数即可。注：当count(\*)语句包含where条件时MyISAM也需扫描整个表。

7）对于自增长的字段，InnoDB中必须包含只有该字段的索引，但是在MyISAM表中可以和其他字段一起建立联合索引。

8）清空整个表时，InnoDB是一行一行的删除，效率非常慢。MyISAM则会重建表。MyisAM使用delete语句删除后并不会立刻清理磁盘空间，需要定时清理，命令：OPTIMIZE table dept;

9）InnoDB支持行锁（某些情况下还是锁整表，如 update table set a=1 where user like ‘%lee%’）

10）Myisam创建表生成三个文件：.frm 数据表结构 、 .myd 数据文件 、 .myi 索引文件，Innodb只生成一个 .frm文件，数据存放在ibdata1.log

现在一般都选用InnoDB，主要是MyISAM的全表锁，读写串行问题，并发效率锁表，效率低，MyISAM对于读写密集型应用一般是不会去选用的。

## 6.CHAR和VARCHAR的区别

CHAR和VARCHAR类型在存储和检索方面有所不同

CHAR列长度固定为创建表时声明的长度，长度值范围是1到255

当CHAR值被存储时，它们被用空格填充到特定长度，检索CHAR值时需删除尾随空格。

## 7.Mysql中有哪几种锁

MyISAM支持表锁，InnoDB支持表锁和行锁，默认为行锁

表级锁：开销小，加锁快，不会出现死锁。锁定力度大，发生锁冲突的概率最高，并发量最低

行级锁：开销大，加锁慢，会出现死锁。锁力度小，发生锁冲突的概率小，并发度最高

## 8.简述内连接，左外连接，右外连接的查询过程

内连接：从左表中取出每一条记录,与右表中的所有记录进行匹配，必须是某个条件,在左表和右表中相同【公共部分】,才会保留结果.否则不保留

左外连接：包含左边表的全部行（不管右边的表中是否存在与它们匹配的行），以及右边表中全部匹配的行。

右外连接：包含右边表的全部行（不管左边的表中是否存在与它们匹配的行），以及左边表中全部匹配的行。

9.SQL语句的提问

MongoDB

1.什么是MongoDB

MongoDB是一个文档数据库，提供好的性能，领先的非关系型数据库。采用BSON存储文档数据。

BSON（）是一种类json的一种二进制形式的存储格式，简称Binary JSON.相对于json多了date类型和二进制数组。

2.什么是集合(表)

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。

3.什么是文档(记录)

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。

4.MongoDB支持主键外键关系吗

默认MongoDB不支持主键和外键关系。 用Mongodb本身的API需要硬编码才能实现外键关联，不够直观且难度  
较大

5.如何执行事务/加锁

mongodb没有使用传统的锁或者复杂的带回滚的事务,因为它设计的宗旨是轻量,快速以及可预计的高性能.可以把它类比成mysql mylsam的自动提交模式.通过精简对事务的支持,性能得到了提升,特别是在一个可能会穿过多个服务器的系统里.

Redis

1.什么是Redis持久化？Redis有哪几种持久化方式？

持久化就是把内存的数据写到磁盘中去，防止服务宕机了内存数据丢失。

Redis 提供了两种持久化方式:RDB（默认） 和AOF

2.Redis 有哪些架构模式？讲讲各自的特点

**单机版**

特点：简单

问题：1.内存容量有限 2.处理能力有限 3.无法高可用。

**主从复制**

Redis 的复制（replication）功能允许用户根据一个 Redis 服务器来创建任意多个该服务器的复制品，其中被复制的服务器为主服务器（master），而通过复制创建出来的服务器复制品则为从服务器（slave）。 只要主从服务器之间的网络连接正常，主从服务器两者会具有相同的数据，主服务器就会一直将发生在自己身上的数据更新同步 给从服务器，从而一直保证主从服务器的数据相同。

特点：1、master/slave 角色2、master/slave 数据相同3、降低 master 读压力在转交从库

问题：1.无法保证高可用2.没有解决 master 写的压力

**哨兵**

Redis sentinel 是一个分布式系统中监控 redis 主从服务器，并在主服务器下线时自动进行故障转移。其中三个特性：1.监控（Monitoring）：Sentinel  会不断地检查你的主服务器和从服务器是否运作正常。2.提醒（Notification）： 当被监控的某个 Redis 服务器出现问题时， Sentinel 可以通过 API 向管理员或者其他应用程序发送通知。3.自动故障迁移（Automatic failover）： 当一个主服务器不能正常工作时， Sentinel 会开始一次自动故障迁移操作。

特点：1、保证高可用2、监控各个节点3、自动故障迁移

缺点：1.主从模式，切换需要时间丢数据2.没有解决 master 写的压力

**集群（proxy 型）**

Twemproxy 是一个 Twitter 开源的一个 redis 和 memcache 快速/轻量级代理服务器； Twemproxy 是一个快速的单线程代理程序，支持 Memcached ASCII 协议和 redis 协议。

特点：1、多种 hash 算法：MD5、CRC16、CRC32、CRC32a、hsieh、murmur、Jenkins 2、支持失败节点自动删除3、后端 Sharding 分片逻辑对业务透明，业务方的读写方式和操作单个 Redis 一致

缺点：1.增加了新的 proxy，需要维护其高可用。2.failover 逻辑需要自己实现，其本身不能支持故障的自动转移可扩展性差，进行扩缩容都需要手动干预

**集群（直连型）**

从redis 3.0之后版本支持redis-cluster集群，Redis-Cluster采用无中心结构，每个节点保存数据和整个集群状态,每个节点都和其他所有节点连接。

特点：1、无中心架构（不存在哪个节点影响性能瓶颈），少了 proxy 层。2、数据按照 slot 存储分布在多个节点，节点间数据共享，可动态调整数据分布。3、可扩展性，可线性扩展到 1000 个节点，节点可动态添加或删除。4、高可用性，部分节点不可用时，集群仍可用。通过增加 Slave 做备份数据副本5、实现故障自动 failover，节点之间通过 gossip 协议交换状态信息，用投票机制完成 Slave到 Master 的角色提升。

缺点：1、资源隔离性较差，容易出现相互影响的情况。2、数据通过异步复制,不保证数据的强一致性

3.使用过Redis分布式锁么，它是怎么实现的？

先拿setnx来争抢锁，抢到之后，再用expire给锁加一个过期时间防止锁忘记了释放。

4.使用过Redis做异步队列么，你是怎么用的？有什么缺点？

一般使用list结构作为队列，rpush生产消息，lpop消费消息。当lpop没有消息的时候，要适当sleep一会再重试。

缺点：在消费者下线的情况下，生产的消息会丢失，得使用专业的消息队列如rabbitmq等。

5.什么是缓存穿透？如何避免？什么是缓存雪崩？何如避免？

**缓存穿透**

一般的缓存系统，都是按照key去缓存查询，如果不存在对应的value，就应该去后端系统查找（比如DB）。一些恶意的请求会故意查询不存在的key,请求量很大，就会对后端系统造成很大的压力。这就叫做缓存穿透。

**如何避免**

1：对查询结果为空的情况也进行缓存，缓存时间设置短一点，或者该key对应的数据insert了之后清理缓存。

2：对一定不存在的key进行过滤。可以把所有的可能存在的key放到一个大的Bitmap中，查询时通过该bitmap过滤。

**缓存雪崩**

当缓存服务器重启或者大量缓存集中在某一个时间段失效，这样在失效的时候，会给后端系统带来很大压力。导致系统崩溃。

**如何避免**

1：在缓存失效后，通过加锁或者队列来控制读数据库写缓存的线程数量。比如对某个key只允许一个线程查询数据和写缓存，其他线程等待。

2：做二级缓存，A1为原始缓存，A2为拷贝缓存，A1失效时，可以访问A2，A1缓存失效时间设置为短期，A2设置为长期

3：不同的key，设置不同的过期时间，让缓存失效的时间点尽量均匀。