DB

MySQL部分

1. 数据库的三范式是什么?

2. 如何获取当前数据库版本?

mysql --version

3. 说一下 ACID 是什么?

是一个支持事务数据库的四种必须特性：
A：原子性（要么全部完成，要么全部不完成）
C：一致性（事务必须始终保持系统处于一致的状态）
I：隔离性（执行事务，使它们好像是系统在给定时间内执行的唯一操作，串行执行）
D：持久性（在事务完成以后，该事务对数据库所作的更改保存在数据库之中，并不会被回滚。）

4. char 和 varchar 的区别是什么?

char 表示定长，长度固定，varchar表示变长，即长度可变。

对 char 来说，最多能存放的字符个数 255，和编码无关。而 varchar 呢，最多能存放 65532 个字符。

5. mysql 的内连接、左连接、右连接有什么区别?

MYSQL中可以通过内外键链接,将有关系的表中数据合并到一起进行条件筛选。

inner join：当进行内连接时,系统会自动忽略两个表中对应不起来的数据:只显示所有有关联的数据。例如有一个学生信息表student和一个学生成绩表score，学生信息表中有一个学号为007的学生，而成绩表中007没有数据，只有学生信息。那么在内连接查询所有成绩时，就不会显示007的信息。同理，如果学生成绩表中有一个008信息，而学生信息表中，只有008学号而没有其他008信息，那么，查询结果也不会有008信息。

left join：例如我们先在有一句SQL语句：SELECT * FROM students LEFT JOIN score ON st.sid=sc.stu_id;那么student表就是左表，score就是右表，查询结果中左侧显示student表中的信息，右侧显示score表信息。与inner join不同的地方是，007的成绩信息也会显示，只不过显示NULL来代替。而008在student表中没有信息，那么008就不会显示出来。

right join：例如SELECT * FROM score sc LEFT JOIN students st ON st.sid=sc.stu_id; 这句SQL里面，score是左表，student是右表，007在score中没有信息，因此不显示，而008在score中有信息，那么008就会显示，而且008那行记录红的student信息都为NULL。

6. mysql 索引是怎么实现的?

索引：是一种类似字典目录的快速查询机制，作用在表中某些字段上，但存储独立于表。

优点：可快速定位到数据；可将数据项进行排序；加快表与表之间的连接；在查询过程中提高系统性能

缺点：增删索引和维护消耗时间，占用额外空间。

实现：B树或者B+树

索引的使用时机：表经常进行select操作，表很大，记录内容多。
不使用的时机：经常进行update、delete、insert，表很小。

分类：

- 唯一索引：不允许两行有相同索引值

- 主键索引：主键中每个值唯一，并且不能为空
- 聚集索引：表中各行的物理顺序与键值索引顺序相同
- 非聚集索引：指定表逻辑顺序，数据存储在一个位置，索引存在另一个位置

索引的创建：

```
create index index_name on table_name(column_name);
```

索引的删除：

```
ALTER table table_name Drop index index_name;
```

7. 说一下数据库的事务隔离？以及MYSQL中的事务隔离级别

事务的隔离性就是指，多个并发的事务同时访问一个数据库时，一个事务不应该被另一个事务所干扰，每个并发的事务间要相互进行隔离。

假设我们现在有这样一张表（T），里面记录了很多牛人的名字，我们不进行事务的隔离看看会发生什么呢？

第一天，事务A访问了数据库，它干了一件事情，往数据库里加上了新来的牛人的名字，但是没有提交事务。
insert into T values (4, '牛D');
这时，来了另一个事务B，他要查询所有牛人的名字。
select Name from T;
这时，如果没有事务之间没有有效隔离，那么事务B返回的结果中就会出现“牛D”的名字。这就是“脏读（dirty read）”。

第二天，事务A访问了数据库，他要查看ID是1的牛人的名字，于是执行了
select Name from T where ID = 1;
这时，事务B来了，因为ID是1的牛人改名字了，所以要更新一下，然后提交了事务。
update T set Name = '不牛' where ID = 1;
接着，事务A还想再看看ID是1的牛人的名字，于是又执行了
select Name from T where ID = 1;
结果，两次读出来的ID是1的牛人名字竟然不相同，这就是不可重复读（unrepeatable read）。

第三天，事务A访问了数据库，他想要看看数据库的牛人都有哪些，于是执行了
select * from T;
这时候，事务B来了，往数据库加入了一个新的牛人。
insert into T values(4, '牛D');
这时候，事务A忘了刚才的牛人都有哪些了，于是又执行了。
select * from T;
结果，第一次有三个牛人，第二次有四个牛人。
相信这个时候事务A就蒙了，刚才发生了什么？这种情况就叫“虚读（phantom problem）”。

事务隔离级别：

- transcation_serializable：避免重复读、脏读和虚读
- transcation_repeatable_read:避免重复读、脏读
- transcation_read_commited:避免脏读

- transcation_uncommited:什么都避免不了

8. 说一下 mysql 常用的引擎?

MyISAM用于事务处理、
Innodb用于查询处理、
Memory用于查询临时表

9. 说一下 mysql 的视图

是一种虚表，建立在已有表上。视图无真正的数据，程序员操作视图时，最终会转换成对基表的操作。一个基表可以有0个会多个视图。视图的作用：简化用户的操作使查询变得简单，隐藏了数据表结构的复杂性，具有定制性是不同的用户看到各自的数据。视图可以作为一种安全机制。通过视图用户只能查看和修改他们所能看到的数据。

mysql中视图的创建语句：

```
create view view_name AS select column_name(s) From table_name where condition;
```

删除视图的语句

```
Drop view view_name ;
```

10. 说一下乐观锁和悲观锁?

悲观锁和乐观锁都是在并发控制主要采用的技术手段。悲观锁性能较低

1. 悲观锁：假定发生并发冲突，屏蔽一切违反数据完整性的操作。例如：查询完数据，将事务锁起来，直到事务提交，使用数据库锁来实现
2. 乐观锁：假定发生并发冲突，只在提交操作时检查是否违反数据完整性。例如：在修改数据时，把事务锁起来，通过vesion来实现或者时间戳

11. mysql 的约束?

- NOT NULL: 非空
- primary key: 主键
- unique：不可重复
- check：用于控制字段范围
- foreign key：外键

12. 如何做 mysql 的性能优化?

主要有SQL优化和数据库结构优化：

1. sql优化：

- 查询三个无关表的时候，将记录最少的放在最后一位
- 查询三个有关表的时候，将引用最多的放在最后一位
- select语句减少使用*

- 用Truncate代替delete
- 多使用内部函数，例如用contact()来代替||
- 多使用commit，合理使用索引，使用>=代替>，用IN代替OR

2. 数据库结构优化：
   范式优化、反范式优化、拆分表

13. 什么是存储过程？优点是什么？

存储过程就像编程语言的函数，将一些计算代码封装起来代替了大量的SQL语句，可以进行调用，提高执行效率，降低网络通信量。

缺点：每个数据库存储过程的语法都不一样，维护困难。业务逻辑放在数据库中，难以迭代。

13. drop、 delete和truncat？

- drop table :删除表，无回滚
- truncate table：删除表中的内容保留表结构，无回滚
- delete from table：删除表中的内容，一行一行删除，也可以指定删除某行记录

14. 数据库运行于那种模式下可以防止数据丢失？

归档模式下，只要其归档日志不丢失，可有效防止数据丢失。

Redis部分

1. redis 是什么？都有哪些使用场景？

redis是一种键值对类型的内存数据库,读取速度非常快，定时异步地将数据写入到硬盘上。
它支持多种数据类型，比如String、List、Hash、Set、SortedSet。通常单个value能存储1GB数据，而String类只能存储512mb数据。
redis是单线程的，在同一台机器上可部署多个实例，每一个实例都具有232个key值，每一种数据类型都能存出232个元素值。缺点是受物理内存的限制，无法用来做海量数据的处理。

使用场景有，消息队列。会话缓存。订阅发布。排行榜。计数器。用来当消息队列的时候。使用链表实现一个先进先出的循环队列。

redis当做缓存来使用的时候，通过一致性哈希来实现动态扩容和缩容。当做持久化存储时，必须使用固定的keys-to-nodes，节点一旦确定，并不能改变，除非有数据再平衡系统。

2. redis 的配置安装？

- 在官网山下载redis.tar.gz压缩包，将它解压到目录中
- 然后进入redis目录，进行make编译，完成后redis目录中有对应的src、conf等文件夹
- 进入到src目录中，然后make install 进行安装redis
- 进入到etc目录下，编辑conf文件，将daemonize属性改为yes，表明需要在Linux后台运行
- 进入到bin目录中，输入./redis-server启动redis服务
- 重新开一个终端，输入redis-cli，即可使用redis。redis终端为6379

3. redis 和 memcache 有什么区别？

1、都是内存数据库。不过memcache还可用于缓存其他东西，例如图片、视频等等。
2、Redis支持list，set，hash等数据结构的存储。
3、过期策略memcache指定set key1 0 0 8,表示永不过期。Redis可以通过例如expire设定，例如expire name 10
4、分布式设定memcache集群，利用magent做一主多从;redis可以做一主多从。都可以一主一从
5、redis可以定期保存到磁盘（持久化),memcache挂掉后，数据不可恢复; redis数据丢失后可以通过aof恢复
8、Redis支持数据的备份

4. redis 为什么是单线程的?

因为Redis是基于内存的操作，Redis的瓶颈最有可能是机器内存的大小或者网络带宽，单线程容易实现，而且CPU不会成为瓶颈。

5. redis 支持的数据类型有哪些?

五种：String、 list、 hash、 set、 sortedset、 key

6. redis 支持的 java 客户端都有哪些?

Jedis、 Redisson，常用redisson，因为对Java对象有强力的支持。

7. 怎么保证缓存和数据库数据的一致性?

缓存一致性：使用Hash一致性
数据库持久化存储使用keys-to-nodes方式实现。

8. redis 持久化有几种方式?

两种：使用RDB和AOF。
RDB是指定时间间隔对数据进行快照。
AOF文件末尾记录写操作，可以恢复原始数据

9. redis 如何做内存优化?

两种方式：从物理角度，从代码

- 物理角度，就是增大机器的位数，增大机器的内存
- 代码角度，尽量多使用集合数据类型，将大量的信息写在一个key中。例如，web系统中的用户信息数据，将一个用户所有的信息写在一个散列表

10. redis 淘汰策略有哪些?

1、内存达到上限或者写入大量数据时，会报错
2、回收最小使用
3、回收随机键
4、回收过期键，过期键中包含了过期最小使用键和过期随

11. redis 回收机制应用的算法

有三种LRU、 FIFO、 LRU

LRU：即最近最少使用。根据访问记录

- 实现：用链表实现一个队列来缓存数据，有新数据写入时，将其插到表头；如果有数据被访问，把其也插到表头。这个队列满的时候，或定期进行回收时，将队尾元素删
- 缺点：偶发性、批量数据操作时，会影响redis缓存命中，使得缓存受到污染。缓存命中判断时，必须遍历整个表。

LRU-k：为了解决缓存污染

- 实现：比LRU算法多维护一条队列，来充当访问列表，用于记录数据被访问的次数。当访问次数达到k次，将其写入到缓存队列中。这个访问列表也有相应的回收机制，采用FIFO、LRU进行回收。而在缓存队列中，数据按照访问时间排序，访问时间离当前时间近的都在表头，离当前时间远的都在表尾，淘汰表尾的数据。
- 缺点：k=2时，性能最佳，时间复杂度高，内存消耗大

12.redis集群如何做到一致性，redis集群为了防止部分节点失效，采取的策略

redis集群上不适用一致性hash，而是使用Hash槽，redis会在集群上均匀分布16384个hash槽。每当有k-v写入，会先将key通过crc16算法算出一个结果，再将此结果与16384进行取模运算，这个值就是此k-v对应的Hash槽。

redis为防止部分节点失效，采用了主从复制模型，每个节点都有N-1个复制品。

13.redis事务

事务就是一组命令的集合，redis事务是顺序执行，具有排他性，一致性，但一致性不强，无回滚，通过队列保存命令。

- MULTI 开启事务，告知Redis接下来的命令存入到一个事务中。
- EXEC 执行事务，当一个事务命令输入完成，用EXEC来执行这个事务。
- DISCARD 取消事务的执行。
- watch 对一个key或者多个key进行监视，如果在执行事务前，key值被修改，那么事务就会被打断
- unwatch 对一个key或者多个key取消监视