面试题总结

1. MySQL中InnoDB与MySIAM的区别?

2. 简单描述MySQL中索引、主键、唯一索引、联合索引的区别，对数据库的性能有什么影响?

主要考点：
MySQL索引的基础和类型
延伸：MySQL索引的创建原则
延伸：MySQL索引的注意事项

《一》MySQL索引的基础：
索引类似于书籍的目录，要想找到一本书的某个特定主题，需要先查找书的目录，定位对应的页码。
存储引擎使用类似的方式进行数据查询，先去索引当中找到对应的值，然后根据匹配的索引找到对应的数据行。

《二》MySQL索引对数据库的性能有什么影响？
1》大大减少服务器需要扫描的数据量
2》帮助服务器避免排序和临时表
3》将随机I/O变成顺序I/O
4》大大提高查询速度，降低写的速度、占用磁盘

《三》索引的使用场景
1》对于非常小的表，大部分情况下全表扫描效率更高
2》中到大型表，索引非常有效
3》特大型表。建立和使用索引的代价将随之增长，可以使用分区技术来解决。

《四》索引的类型 - 都是实现在存储引擎层的
普通索引：最基本的索引，没有任何约束限制
唯一索引：于普通索引类似，但具有唯一性约束
主键索引：特殊的唯一索引，不允许有空值

各类索引之间的区别
一个表只能有一个主键索引，可以有多个唯一索引；
主键索引一定是唯一索引，唯一索引不是主键索引；
主键可以与外键构成参照完整性约束，防止数据不一致；
复合索引：将多个列组合在一起创建索引，可以覆盖多个列；
外建索引：只有InnoDB类型的表才可以使用外键索引，保证数据的一致性、完整性和实现级联操作；
全文索引：MySQL自带的全文索引只能用于MySIAM，并且只能对英文进行全文检索。

《四》MySQL索引的创建原则
1》最适合索引的列是出现在where子句中的列，或者是连接子句中的列，而不是出现在select关键字后的列；
2》索引列的基数越大，索引的效果越好；
3》对字符串进行索引，应该制定一个前缀长度，可以节省大量的索引空间；
4》根据情况创建符合索引，符合索引可以提高查询效率
5》避免创建过多索引，索引会额外占用磁盘空间，降低写操作效率。
6》主键尽可能选择较短的数据类型，可以有效减少索引的磁盘占用、提高查询效率。

《五》MySQL索引的注意事项
1》复合索引遵循前缀原则 key(a,b,c)

```
where a =1 and b =2 and c = 3
```

```
where a =1 and b =2
where a =1
where a =1 and c =3

2》like查询，% 不能再前，可以使用全文索引
3》column is null 可以使用索引
4》如果MySQL估计使用索引比全表扫描更慢，会放弃使用索引。
5》如果or前的条件中的列有索引，后面的没有，索引都不会被用到
6》列类型是字符串，查询时一定要给值加引号，否则索引失效。
```

3. SQL语句考察

问题：
有A(id,sex,par,c1,c2)，B(id,age,c1,c2)两张表，其中A.id与B.id关联，现在要求写出一条sql语句，将B中age>50的记录的c1,c2更新到A表中统一记录中的c1,c2字段中。

答案：

```
update A set A.c1=B.c1, A.c2=B.c2 where A.id=B.id and B.age>50

update A inner join B on A.id=B.id set A.c1=B.c1, A.c2=B.c2 where B.age>50
```

《一》MySQL的关联查询语句
1》 交叉连接(cross join)：笛卡尔积
2》 内连接（inner join）
（1）等值连接：on A.id = B.id
（2）不等值连接：on A.id > B.id
（3）自连接：select * from A t1 inner join A t2 on t1.id = t2.id
3》 外连接（left/right join）
（1）left join:以左表为主表查询出左表数据，按照on后边的关联条件匹右表，没有匹配到的用NULL填充，可以简写成 left join
（2）right join:以右表为主表查询出右表数据，按照on后边的关联条件匹左表，没有匹配到的用NULL填充，可以简写成 right join
4》 联合查询(union/union all)
（1）union: select * from A union select * from B 就是把多个结果集集合在一起，以union前的结果为基准，需注意联合查询的列数一定相等，相同的记录行会合并。
（2）union all: 不会合并重复的记录行
5》 全连接（full join）
MySQL不支持全连接，可以使用left join + union + right join 联合使用
select * from A left join B on A.id = B.id Union
select * from A right join B on A.id = B.id

4. MySQL查询优化

问题：
请简述项目中优化SQL语句执行效率的方法，从哪些方面？SQL语句性能如何分析？

先说明如何　定位低效sql语句，然后根据SQL语句可能低效的原因进行排查，先从索引着手，如果索引没有问题，考虑一下几个方面，如数据访问的问题，长难查询语句的问题以及特定类型优化的问题，进行逐一回答。

《一》如何查找分析查询速度慢的原因？分析方法有哪些？

（1）记录慢查询的日志

分析查询日志，不要直接打开慢查询日志进行分析，这样比较浪费时间和精力，可以使用pt-query-digest工具进行分析

（2）使用**show** profile

set profiling=1;开启，服务器上执行的所有语句会检测消耗的时间，并存入到临时表中。

show profiles;

查询临时表的主键ID， **show** profile **for query** 临时表**ID** 可以查到某条**sql**语句在什么地方发生了慢的操作

（3）使用**show status**

show status会返回一些计数器；

show global status 查看服务器级别的所有计数

根据这些计数器大概分析一下查询慢的原因

（4）使用**show full processlist**

观察是否有大量线程处于不正常的状态和特征

（5）使用 **explain/desc** 分析**sql**语句

《二》分析原因后，如何进行优化？

1》优化查询过程中的数据访问

（1）访问数据太多导致查询性能下降

（2）确定应用程序是否在检索大量数据超过需要的数据，可能是太多行或列

（3）避免使用以下**SQL**语句：

查询不需要的记录，使用**limit**解决

多表关联返回全部列，指定要查询的列

总是取出全部列，因为**select** * 无法使优化器完成索引覆盖扫描的优化

重复查询相同的数据，可以缓存数据，下次直接可以读取缓存。

（4）是否扫描额外的记录？

使用**explain**来进行分析，如果发现查询需要扫描大量的数据但返回少数的行，可以通过如下技巧进行优化：

使用覆盖索引扫描，把所有用的列放到索引当中，这样存储索引不需要回表获取对应行就可以返回结果

改变数据库和表的结构，修改数据表范式

重写**SQL**语句，让优化器以更优的方式执行查询

2》优化长难的查询语句

（1）切分查询：将一个大的查询分为多个小的相同查询

（2）分解关联查询：

可以将一条**sql**语句分解成多条**SQL**语句去执行，这样可以使缓存的效率更高

执行单个查询可以减少锁的竞争

在应用层做关联可以更容易对数据库进行拆分，从而查询效率就会提高

较少冗余记录的查询

3》查询特定类型的查询语句

（1）优化**count()**查询

count(*) 中的 * 会忽略所有的列，直接统计所有的列，因此不要使用**count**(列)

MyISAM当中，没有任何**where**条件的**count(*)**非常快，当有**where**条件时，MyISAM的**count**统计不一定比其他表引擎快，可以使用**explain**查询近似值，用近似值代替**count(*)**

使用缓存

（2）优化关联查询

确定**on**或者**using**子句的列上有索引

确保**order by**和**group by**中只有一个表的序列，这样MySQL才有可能使用索引

（3）优化子查询

尽量使用关联查询替代子查询

（4）优化**group by** 和 **distinct**
这两种查询均可使用索引来优化是最有效的方法
关联查询中，使用标识列进行分组效率会更高
如果不需要**order by**,进行**group by**时使用**order by** null, MySQL 不会再进行文件排序

（5）优化**limit**分页
limit偏移量大的时候，查询效率很低，可以记录上一次查询的最大**ID**，下次查询的时候根据该**ID**进行查询。

（6）优化**union**查询
union all 的查询效率比 **union** 高

5. MySQL高可用和高可扩展

问题：
1. 简述MySQL分表操作和分区的工作原理，分别说说分区和分表的使用场景和各自的优缺点？
答案简述：充分了解分区和分表的工作原理和实用场景，可根据分区、分表以及MySQL复制、负载均衡的适用场景来根据情况进行回答。

2. 设定网站的用户数量在千万级，但是活跃的用户数只有1%，如何通过优化数据库来提高活跃用户的访问速度？
答案简述：
使用分区，让活跃用户分在一个区，不活跃用户分在另一个区，查询的时候只查询活跃用户的那个区。
使用分库分表，通过水平分割的方式，将活跃用户放在一个表中，不活跃用户放在另一个表中，查询的时候只查询活跃用户表。

《一》分区表的原理
（1）工作原理：
<1>对用户而言，分区表是一个独立的逻辑表，但底层MySQL将其分成了多个物理子表，这对用户来说是透明的，每个分区表都会使用一个独立的表文件。

<2>创建表时使用partition by 子句定义每个分区存放的数据，执行查询时，优化器会根据分区定义过滤那些没有我们需要数据的分区，这样查询只需要查询所需数据在的分区即可

<3>分区的主要目的：将数据按照一个较粗的粒度分在不同的表中，这样可以将相关的数据存放在一起，而且如果想一次性删除整个分区的数据也很方便。

（2）使用场景：
<1>表非常大，无法全部存在内存中，或者只在表的最后有热点数据，其他都是历史数据。
<2>分区表的数据更容易维护，可以对独立的分区进行独立的操作
<3>分区表的数据可以分布到不同的机器上，从而高效实用资源。
<4>可以使用分区表来避免某些特殊的瓶颈
<5>可以备份和恢复独立的分区

（3）缺点：
<1>一个表最多有1024个分区
<2>可以使用列分区，分区表表达式必须是整数
<3>分区字段中如果有主键和唯一索引列，那么主键列和唯一列都必须包含进来
<4>分区表无法使用外键约束
<5>对现有表结构进行修改
<6>所有的分区都必须使用相同的存储引擎
<7>分区函数中可以使用的表达式和函数会有一些限制
<8>某些存储引擎不支持分区

<9>对于MyISAM的分区表，不能使用load index into cache
<9>对于MyISAM表，使用分区表时需要打开更多的文件描述符，会降低查询效率

《二》分库分表的原理
（1）工作原理：
<1>通过一些hash算法或者工具实现将一张表垂直或水平进行物理切分

（2）使用场景：
<1>单表记录条数达到百万到千万级别时
<2>解决表锁的问题

（3）分表方式：
<1>水平分割-原理：表很大，分割后可降低在查询时需要读的数据和索引的页数，同时也降低了索引的层数，提高查询速度。
<2>水平分割-使用场景：表中的数据本身就具有独立性，例如表中分别记录各个地区的数据或者不同时期的数据，特别是有些数据常用，有些不常用；需要把数据存放在多个介质上
<3>水平分割-缺点：
给应用增加复杂度，通常查询是需要多个表名，查询所有数据都需要union操作
在许多数据库应用中，这种复杂度会超过它带来的优点，查询时会增加读一个索引层的磁盘次数，从而降低查询效率

<1>垂直分割-原理：把主键和一些列放在一个表，然后把主键跟另外的列放在另一个表中

<2>垂直分割-使用场景：
如果一个表中某些列常用，而另外一些列不常用
可以使数据行变小，一个数据页能存储更多的数据，查询时减少I/O次数
<3>水平分割-缺点：
管理冗余列，查询所有数据需要join操作

（4）分表的缺点：
<1>有些分表的策略是基于应用层的逻辑算法、一旦逻辑算法改变，整个分表逻辑都会改变，扩展性差。
<2>对于应用层来说，逻辑算法无疑增加开发成本

《三》MySQL复制原理及负载均衡
（1）MySQL主从复制原理
<1> 在主库上把数据（insert、update、delete）更改到二进制文件（binlog）中
<2> 从库将主库的binlog日志复制到自己的中继日志relaylog中
<3> 从库读取中继日志中的事件，将其重放到从库数据中。

（2）主从复制解决的问题
<1> 数据分布：随意停止或开始复制，并在不同地理位置分布数据备份。
<2> 负载均衡：降低单个服务器的压力
<3> 高可用和故障切换：帮助应用程序避免单点失败
<4> 升级测试：可以将高版本的MySQL作为从库

6. MySQL的安全性

问题：1. sql语句应该考虑哪些安全性问题？
答案：说sql注入的优化方案

问题：2. 为什么使用PDO和MySQLi连接数据库比mysql函数库更加安全？
答案：支持预处理

《一》SQL查询的安全方案

（1）使用预处理语句防止sql注入
（2）写入数据库的数据要进行特殊字符的转义
（3）查询的错误信息不要返回给用户，将错误记录到日志上。
（4）php端尽量使用pdo对数据库进行操作，pdo拥有对预处理语句很好的支持方法，MySQLi也有，但是扩展性不如PDO，效率略高于pdo

《二》MySQL安全的设置
（1）定期做数据库备份
（2）不给查询用户root权限，合理分配权限。
（3）关闭远程访问为数据库的权限
（4）修改root口令，不用默认口令，修改为复杂的口令
（5）删除多余的用户
（6）修改root用户名称
（7）限制一般用户浏览其他数据库
（7）限制用户对数据文件的访问权限

7. 程序功能设计

问题：1. 编写一个在线留言本，实现用户的在线留言功能，留言信息要存储的数据库中，要求设计表结构，及使用PHP编码完成
答案：表结构设计、编码能力

8. web资源防盗链

《一》什么是防盗链？
（1）盗链：自己的页面上展示了一些不在自己服务器上的内容
小站盗用大站的图片、视频、音乐等资源

《二》防盗链的工作原理
（1）通过referer或签名，网站可以检测目标网页访问的来源网页，如果是资源文件，则可以跟踪到显示它的网页地址
（2）一旦检测到来源不是本站即进行阻止或返回指定的页面。
（3）通过计算签名的方式，判断请求是否合法，如果合法则显示，否则返回错误信息。

《三》防盗链的实现方法
（1）referer
Nginx模块 ngx_http_referer_module用于阻挡来源非法的域名请求

Nginx指令 valid_referers，全局变量 $valid_referer

（2）伪造referer
可以使用加密签名解决
使用第三方模块HttpAccesKeyModule实现nginx防盗链
accesskey on|off 模块开关
accesskey_hashmethod md5|sha-1 签名加密方式
accesskey_arg GET 参数名称