mysql是一个**[关系型数据库管理系统](https://baike.baidu.com/item/%E5%85%B3%E7%B3%BB%E5%9E%8B%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F/696511" \t "/Users/xiaozhu_wt/Documents\\x/_blank)**

参考链接：

https://www.cnblogs.com/frankielf0921/p/5930743.html

https://www.cnblogs.com/limuzi1994/p/9684083.html

**一、sql语句：**

1、 现在学生小测验成绩表table1（id，uid，name，score，stdate), 其属性分别表示自增id、学生学号、学生姓名、成绩和统计日期。查询每天成绩最高分，输出日期、最高分。

select stdate, max(score) from table1 group by stdate;

2、用一条SQL语句：查询出每门课都大于80分的学生姓名

select distinct name from table group by name having count(kecheng)>=3 and min(fenshu)>=80;

或者：

　select name from (select name,min(score) from student group by name having min(score)>80) stu;

select count(ordid) from odertable group by(uderid) bettewn ;

3、求工资第二高的人：

select max(Salary) SecondHighestSalary from Employee where Salary <> (select max(Salary) from Employee );

4、 建一个mysql表：

CREATE TABLE `quicker\_perf\_plan` (

`id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '自身id',

`jobname` varchar(255) DEFAULT NULL COMMENT '脚本名称',

`createdata` datetime DEFAULT CURRENT\_TIMESTAMP COMMENT '创建时间',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=4896 DEFAULT CHARSET=utf8mb4;

5、 CREATE INDEX 实例

本例会创建一个简单的索引，名为 "PersonIndex"，在 Person 表的 LastName 列：

CREATE INDEX PersonIndex ON Person (LastName)

如果您希望以降序索引某个列中的值，您可以在列名称之后添加保留字 DESC：

CREATE INDEX PersonIndex ON Person (LastName DESC)

假如您希望索引不止一个列，您可以在括号中列出这些列的名称，用逗号隔开：

CREATE INDEX PersonIndex ON Person (LastName, FirstName)

**二、数据库的事务**

事务（transaction）是作为一个单元的一组有序的数据库操作。如果组中的所有操作都成功，则认为事务成功，如果一个操作失败，则事务将回滚，该事务所有操作的影响都将取消。

**事务的四大特性：**

1、原子性：原子性是指事务包含的所有操作要么全部成功，要么全部失败回滚，因此事务的操作如果成功就必须要完全应用到数据库，如果操作失败则不能对数据库有任何影响。

2、一致性：是指事务必须使数据库从一个一致性状态变换到另一个一致性状态，也就是说一个事务执行之前和执行之后都必须处于一致性状态。举例来说，假设用户A和用户B两者的钱加起来一共是1000，那么不管A和B之间如何转账、转几次账，事务结束后两个用户的钱相加起来应该还得是1000，这就是事务的一致性。

3、隔离性：是当多个用户并发访问数据库时，比如同时操作同一张表时，数据库为每一个用户开启的事务，不能被其他事务的操作所干扰，多个并发事务之间要相互隔离。关于事务的隔离性数据库提供了多种隔离级别。

4、持久性：是指一个事务一旦被提交了，那么对数据库中的数据的改变就是永久性的，即便是在数据库系统遇到故障的情况下也不会丢失提交事务的操作。例如我们在使用JDBC操作数据库时，在提交事务方法后，提示用户事务操作完成，当我们程序执行完成直到看到提示后，就可以认定事务已经正确提交，即使这时候数据库出现了问题，也必须要将我们的事务完全执行完成。否则的话就会造成我们虽然看到提示事务处理完毕，但是数据库因为故障而没有执行事务的重大错误。这是不允许的。

**四大隔离级别 （mcc锁实现）**

1.Read uncommitted(未授权读取、读未提交)： 如果一个事务已经开始写数据，则另外一个事务则不允许同时进行写操作，但允许其他事务读此行数据。该隔离级别可以通过“排他写锁”实现。这样就避免了更新丢失，却可能出现脏读。也就是说事务B读取到了事务A未提交的数据。

2.Read Committed（授权读取、读提交，可能引起脏读） ：读取数据的事务允许其他事务继续访问该行数据，但是未提交的写事务将会禁止其他事务访问该行。该隔离级别避免了脏读，但是却可能出现不可重复读。事务A事先读取了数据，事务B紧接了更新了数据，并提交了事务，而事务A再次读取该数据时，数据已经发生了改变。

3.Repeatable read（可重复读取） ：可重复读是指在一个事务内，多次读同一数据。在这个事务还没有结束时，另外一个事务也访问该同一数据。那么，在第一个事务中的两次读数据之间，即使第二个事务对数据进行修改，第一个事务两次读到的的数据是一样的。这样就发生了在一个事务内两次读到的数据是一样的，因此称为是可重复读。读取数据的事务将会禁止写事务（但允许读事务），写事务则禁止任何其他事务。这样避免了不可重复读取和脏读，但是有时可能出现幻象读。（读取数据的事务）这可以通过“共享读锁”和“排他写锁”实现。

4. Serializable（序列化）：提供严格的事务隔离。它要求事务序列化执行，事务只能一个接着一个地执行，但不能并发执行。如果仅仅通过“行级锁”是无法实现事务序列化的，必须通过其他机制保证新插入的数据不会被刚执行查询操作的事务访问到。序列化是最高的事务隔离级别，同时代价也花费最高，性能很低，一般很少使用，在该级别下，事务顺序执行，不仅可以避免脏读、不可重复读，还避免了幻像读。

1. **数据库索引**

MySQL目前主要有以下几种索引类型：

1.普通索引index :加速查找

2.唯一索引

主键索引：primary key ：加速查找+约束（不为空且唯一）

唯一索引：unique：加速查找+约束 （唯一）

3.联合索引

-primary key(id,name):联合主键索引

-unique(id,name):联合唯一索引

-index(id,name):联合普通索引

4.全文索引fulltext :用于搜索很长一篇文章的时候，效果最好。

5.空间索引spatial :了解就好，几乎不用

**索引的两大类型hash与btree。**

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树及其变种B+树，是在存储引擎层实现的，而不是在服务器层实现的，所以不同存储引擎具有不同的索引类型和实现。

b/b+树是为了存储而设计的一种平衡多路查找树，高度比二叉树小，能够降低磁盘的I/O次数： n个节点的平衡二叉树的高度为H(即logn)，而B/B+树的高度为logt((n+1)/2)+1；

优点：

第一，通过创建唯一性索引，可以保证数据库表中每一行数据的唯一性。

第二，可以大大加快数据的检索速度，这也是创建索引的最主要的原因。

第三，通过使用索引，可以在查询的过程中，使用优化隐藏器，提高系统的性能。

第四，可以加速表和表之间的连接，特别是在实现数据的参考完整性方面特别有意义。

第五，在使用分组和排序子句进行数据检索时，同样可以显著减少查询中分组和排序的时间。

缺点

第一，创建索引和维护索引要耗费时间，这种时间随着数据量的增加而增加。

第二，索引需要占物理空间。

第三，当对表中的数据进行增加、删除和修改的时候，索引也要动态的维护，这样就降低了数据的维护速度。

1. **为什么说B+树比B树更适合数据库索引？**

4.1 释义：

B Tree 指的是 Balance Tree，也就是平衡树。平衡树是一颗查找树，并且所有叶子节点位于同一层。

B+ Tree 是基于 B Tree 和叶子节点顺序访问指针进行实现，它具有 B Tree 的平衡性，并且通过顺序访问指针来提高区间查询的性能。

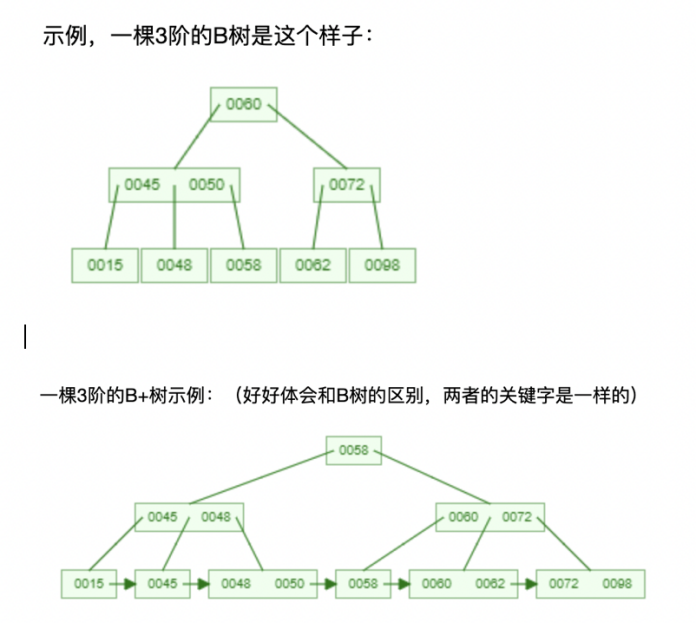
4.2 B+树与B树最大的区别在于：

　　1、B+叶子结点包含全部关键字以及指向相应记录的指针，而且叶结点中的关键字按大小顺序排列，相邻叶结点用指针连接。B树的非叶子节点存储实际记录的指针，而B+树的叶子节点存储实际记录的指针，B+树的叶子节点通过指针连起来了, 适合扫描区间和顺序查找。

　　2、B+非叶结点仅存储其子树的最大（或最小）关键字，可以看成是索引。

4.3 B+树更适合外部存储。由于内结点不存放真正的数据（只是存放其子树的最大或最小的关键字，作为索引），一个结点可以存储更多的关键字，每个结点能索引的范围更大更精确，也意味着B+树单次磁盘IO的信息量大于B树，I/O的次数相对减少。

4.4 MySQL是一种关系型数据库，区间访问是常见的一种情况，B+树叶结点增加的链指针，加强了区间访问性，可使用在区间查询的场景；而使用B树则无法进行区间查找。



1. **mysql的三大引擎**

有InnoDB，MyISAM，Memory，默认是InnoDB。

InnoDB：磁盘表，支持事务，支持行级锁，B+Tree索引

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、MyISAM: 采用表级锁(table-level locking)。InnoDB支持行级锁(row-level locking)和表级锁，默认为行级锁。

4、清空整个表时，MyISAM 会重建表，InnoDB 是一行一行的删除，效率非常慢。

**六、mysql表比较大，查询慢，如何优化？**

1.SQL语句优化（查看慢查询：有无监控报警；show processlist ；查看慢查询日志；explain来了解SQL执行的状态）

2.索引优化

3.数据库结构优化

4.服务器硬件优化（ssd、raid）

5.分表，读写表分离，根据手机号等值模运算分表

6.使用缓存

1. **mysql和nosql的优缺点：**

参考：http://www.mamicode.com/info-detail-861022.html

Mysql 优点：通用化、技术成熟、 可以进行join等复杂查询；

缺点：扩展性差，无法进行分布式存储。

nosql 优点：高并发，支持分布式，易于扩展，可伸缩。

缺点：事务支持较弱、 join等复杂操作能力较弱。