# 数据库

## 1．数据库基础

### （★）1.1 Sql语句

#### **1.1.1 有3 个表（15 分钟）**

Student 学生表(学号，姓名，性别，年龄，组织部门)

Course 课程表(编号，课程名称)

Sc 选课表(学号，课程编号，成绩)

表结构如下：

1）写一个SQL 语句，查询选修了’计算机原理’的学生学号和姓名（3 分钟）

2）写一个SQL 语句，查询’周星驰’同学选修了的课程名字（3 分钟）

3）写一个SQL 语句，查询选修了5 门课程的学生学号和姓名（9 分钟）

答：1）SQL 语句如下：

select stu.sno, stu.sname from Student stu

where (select count(\*) from sc where sno=stu.sno and cno =

(select cno from Course where cname='计算机原理')) != 0;

2）SQL 语句如下：

select cname from Course

where cno in ( select cno from sc where sno =

(select sno from Student where sname='周星驰'));

3）SQL 语句如下：

select stu.sno, stu.sname from student stu

where (select count(\*) from sc where sno=stu.sno) = 5;

#### 1.1.2、有三张表,学生表S,课程C,学生课程表SC,学生可以选修多门课程,一门课程可以被多个学生选修,通过SC 表关联。

1）写出建表语句；

2）写出SQL 语句,查询选修了所有选修课程的学生；

3）写出SQL 语句,查询选修了至少5 门以上的课程的学生。

答：1）建表语句如下（mysql 数据库）：

create table s(id integer primary key, name varchar(20));

create table c(id integer primary key, name varchar(20));

create table sc(

sid integer references s(id),

cid integer references c(id),

primary key(sid,cid)

);

2）SQL 语句如下：

select stu.id, stu.name from s stu

where (select count(\*) from sc where sid=stu.id)

= (select count(\*) from c);

3）SQL 语句如下：

select stu.id, stu.name from s stu

where (select count(\*) from sc where sid=stu.id)>=5;

108、数据库表(Test)结构如下：【基础】

ID NAME AGE MANAGER(所属主管人ID)

106 A 30 104

109 B 19 104

104 C 20 111

107 D 35 109

112 E 25 120

119 F 45 NULL

要求:列出所有年龄比所属主管年龄大的人的ID 和名字?

答：SQL 语句如下：

select employee.name from test employee

where employee.age > (select manager.age from test manager

where manager.id=employee.manager);

109、有如下两张表：【中等难度】

表city： 表state：

CityNo CityName StateNo

BJ 北京（Null）

SH 上海（Null）

GZ 广州GD

DL 大连LN

欲得

到如下结果： City

No City Name State No State Name BJ

北京（Null） （Null） DL

大连LN 辽宁GZ

广州GD 广东SH

上海（Null） （Null） 写相

应的SQL 语句。

答：SQL 语句为：

SELECT C.CITYNO, C.CITYNAME, C.STATENO, S.STATENAME

FROM CITY C, STATE S

WHERE C.STATENO=S.STATENO(+)

ORDER BY(C.CITYNO);

1.比较truncate和delete命令？

1、都是删除

2、truncate不会触发触发器

3、delete可以回滚，truncate不会

4、truncate速度快

2.简述Oracle数据库中rowid和rownum的区别?

rowid在记录创建时就生成了，而且是不变的，直接指向硬件上的存储位置，能用rowid直接访问是最快的，但也是人力所无法做到的。rownum是个伪列，查询的时候除非特别指定，否则不会显示。其主要的用处是控制查询返回的行数，比如在WHERE中加ROWNUM<5，则查询结果如果在5行或以上时，只返回前4行。

3.写一个简单的触发器的示例。

创建触发器那一段代码，在sql 窗口无法正常执行，会报错，必须放到命令窗口去方可

4.写一个简单的函数示例。

oracle写法

create function fn\_plus(x number, y number)

return number is

return x+y;

end fn\_plus;

#### （★★）1.2 存储过程和函数的区别

存储过程是用户定义的一系列sql语句的集合，涉及特定表或其它对象的任务，用户可以调用存储过程，而函数通常是数据库已定义的方法，它接收参数并返回某种类型的值并且不涉及特定用户表。

#### （★★）1.3 事务是什么？

事务是作为一个逻辑单元执行的一系列操作，一个逻辑工作单元必须有四个属性，称为 ACID（原子性、一致性、隔离性和持久性）属性，只有这样才能成为一个事务：

原子性事务必须是原子工作单元；对于其数据修改，要么全都执行，要么全都不执行。

一致性事务在完成时，必须使所有的数据都保持一致状态。在相关数据库中，所有规则都必须应用于事务的修改，以保持所有数据的完整性。事务结束时，所有的内部数据结构（如 B 树索引或双向链表）都必须是正确的。

隔离性由并发事务所作的修改必须与任何其它并发事务所作的修改隔离。事务查看数据时数据所处的状态，要么是另一并发事务修改它之前的状态，要么是另一事务修改它之后的状态，事务不会查看中间状态的数据。这称为可串行性，因为它能够重新装载起始数据，并且重播一系列事务，以使数据结束时的状态与原始事务执行的状态相同。

持久性事务完成之后，它对于系统的影响是永久性的。该修改即使出现系统故障也将一直保持。

#### （★★★）1.4 游标的作用？如何知道游标已经到了最后？

游标用于定位结果集的行，通过判断全局变量@@FETCH\_STATUS可以判断是否到了最后，通常此变量不等于0表示出错或到了最后。

#### （★★★）1.5 触发器

触发器分为事前触发和事后触发，这两种触发有和区别。语句级触发和行级触发有何区别。

事前触发器运行于触发事件发生之前，而事后触发器运行于触发事件发生之后。通常事前触发器可以获取事件之前和新的字段值。

语句级触发器可以在语句执行前或后执行，而行级触发在触发器所影响的每一行触发一次。

## 2. Mysql

#### （★★★）sql语句应该考虑哪些安全性？（新浪网技术部）

防止Sql注入，对特殊字符进行转义、过滤或者使用预编译的sql语句绑定变量。

最小权限原则，特别是不要用 root 账户，为不同的类型的动作或者组建使用不同的账户。

当 sql 运行出错时，不要把数据库返回的错误信息全部显示给用户，以防止泄露服务器和数据库相关信息。

#### （★★★）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（从读写两方面）（新浪网技术部）

索引是一种特殊的文件(InnoDB数据表上的索引是表空间的一个组成部分)，它们包含着对数据表里所有记录的引用指针。

普通索引(由关键字KEY或INDEX定义的索引)的唯一任务是加快对数据的访问速度。

普通索引允许被索引的数据列包含重复的值。如果能确定某个数据列将只包含彼此各不相同的值，在为这个数据列创建索引的时候就应该用关键字UNIQUE把它定义为一个唯一索引。也就是说，唯一索引可以保证数据记录的唯一性。

主键，是一种特殊的唯一索引，在一张表中只能定义一个主键索引，主键用于唯一标识一条记录，使用关键字 PRIMARY KEY 来创建。

索引可以覆盖多个数据列，如像INDEX(columnA, columnB)索引，这就是联合索引。

索引可以极大的提高数据的查询速度，但是会降低插入、删除、更新表的速度，因为在执行这些写操作时，还要操作索引文件。

另外mysql给某字段建立索引以后，where查询时这个字段等于或or或like时索引是不会生效的。

#### （★★★）有一个留言板，用mysql做数据库，用户信息包括：用户名，密码，email，留言内容包括：留言ID，标题，内容，发表时间，状态（新浪网技术部）请实现下列需求：

(1).数据库结构。无需写建表语句，用类似下面的表格，描述清楚即可,注意，要在索引栏中注明是否需要创建索引，以及要创建的索引的类型

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | table\_aaa |  |  |
| 字段名 | 字段说明 | 字段类型 | 索引 |
| name | 姓名 | varchar（64） | 唯一索引 |
| gender | 性别 | enum（‘M’，‘F’） |  |

(2).用一个sql语句查询出发表留言数量大于10条的用户名及其留言数量，查询结果按文章数量降序排列参考答案：用户表结构如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | user |  |  |
| 字段名 | 字段说明 | 字段类型 | 索引 |
| user\_id | 用户编号 | int unsigned | 主键 |
| name | 用户名 | varchar(30) |  |
| password | 密码 | char（32） |  |
| email | 邮箱 | varchar（50） |  |

留言表结构如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | message |  |  |
| 字段名 | 字段说明 | 字段类型 | 索引 |
| message\_id | 留言编号 | int unsigned | 主键 |
| title | 标题 | varchr(100) |  |
| content | 内容 | text |  |
| user\_id | 用户 id | int unsigned | 普通索引 |
| pubtime | 发表时间 | int unsigned |  |
| state | 状态 | tinyint 0 未审核 1 审核 |  |

查询语句如下：

SELECT u.name, COUNT(\*) AS total

FROM user AS u INNER JOIN message AS m

ON u.user\_id = m.user\_id

GROUP BY u.name

HAVING total > 10

ORDER BY total DESC

#### （★★★）两张表 city表和province表。分别为城市与省份的关系表。

city:

|  |  |  |
| --- | --- | --- |
| **id** | **city** | **provinceid** |
| 1 | 广州 | 1 |
| 2 | 深圳 | 1 |
| 3 | 惠州 | 1 |
| 4 | 长沙 | 2 |
| 5 | 武汉 | 3 |

province:

|  |  |
| --- | --- |
| **id** | **province** |
| 1 | 广东 |
| 2 | 湖南 |
| 3 | 湖北 |

(1). 写一条sql语句关系两个表，实现：显示城市的基本信息。

显示字段：城市id ，城市名， 所属省份 。如：

id（城市id） cityname(城市名) privence(所属省份)

……

SELECT c.id AS id,c.city AS cityname,p.province

FROM city c LEFT JOIN province p ON c.provinceid=p.id

(2). 如果要统计每个省份有多少个城市，请用group by 查询出来。

显示字段：省份id ，省份名，包含多少个城市。

SELECT p.id,p.province,count(c.id) AS num

FROM province p LEFT JOIN city c ON p.id = c.provinceid

GROUP BY p.id;

#### （★★）MySQL数据库中的字段类型varchar和char的主要区别是什么？哪种字段的查找效率要高，为什么?

区别一，定长和变长

char 表示定长，长度固定，varchar表示变长，即长度可变

当所插入的字符串超出它们的长度时，视情况来处理，如果是严格模式，则会拒绝插入并提示错误信息，如果是宽松模式，则会截取然后插入。如果插入的字符串长度小于定义长度时，则会以不同的方式来处理，如char（10），表示存储的是10个字符，无论你插入的是多少，都是10个，如果少于10个，则用空格填满。而varchar（10），小于10个的话，则插入多少个字符就存多少个。varchar怎么知道所存储字符串的长度呢？实际上，对于varchar 字段来说，需要使用一个（如果字符串长度小于255）或两个字节（长度大于255）来存储字符串的长度。

区别之二，存储的容量不同对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。

VARCHAR的最大有效长度由最大行大小和使用的字符集确定。整体最大长度是65,532 字节

最大有效长度是 65532 字节，在varchar存字符串的时候，第一个字节是空的，不存任何的数据，然后还需要两个字节来存放字符串的长度。所以有效长度就是 65535 - 1 - 2

= 65532

由字符集来确定，字符集分单字节和多字节

Latin1一个字符占一个字节，最多能存放 65532个字符

GBK 一个字符占两个字节， 最多能存 32766 个字符 UTF8一个字符占三个字节， 最多能存 21844 个字符注意，char和varchar 后面的长度表示的是字符的个数，而不是字节数。

两相比较，char的效率高，没有碎片，尤其更新比较频繁的时候，方便数据文件指针

的操作。但不够灵活，在实际使用时，应根据实际需求来选用合适的数据类型。

相关题目：若一个表定义为 **create table t1(c int, c2 char(30), c3 varchar(N)) charset=utf8;** 问

**N** 的最大值又是多少？（65535 - 1 - 2 - 4 - 30 \* 3 ）/3

#### （★★★）设有成绩表如下所示，查询两门及两门以上不及格同学的平均分。

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 姓名 | 科目 | 分数 |
| 1 | 张三 | 数学 | 90 |
| 2 | 张三 | 语文 | 50 |
| 3 | 张三 | 地理 | 40 |
| 4 | 李四 | 语文 | 55 |
| 5 | 李四 | 政治 | 45 |
| 6 | 王五 | 政治 | 30 |

#创建一个成绩表

CREATE TABLE grade( id INT UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY, name VARCHAR(10) NOT NULL, subject VARCHAR(10) NOT NULL, score TINYINT UNSIGNED NOT NULL

);

#插入记录

INSERT INTO grade(name,subject,score) VALUES('张三','数学',90);

INSERT INTO grade(name,subject,score) VALUES('张三','语文',50);

INSERT INTO grade(name,subject,score) VALUES('张三','地理',40);

INSERT INTO grade(name,subject,score) VALUES('李四','语文',55);

INSERT INTO grade(name,subject,score) VALUES('李四','政治',45);

INSERT INTO grade(name,subject,score) VALUES('王五','政治',30);

#查询语句

SELECT name,AVG(score),SUM(score<60) AS gk

FROM grade

GROUP BY name HAVING gk>=2;

#### （★）为了记录足球比赛的结果，设计表如下， team：参赛队伍表

|  |  |  |
| --- | --- | --- |
| 字段名称 | 类型 | 描述 |
| teamID | int | 主键 |
| teamname | varchar(20) | 队伍名称 |

match：赛程表

|  |  |  |
| --- | --- | --- |
| 字段名称 | 类型 | 描述 |
| matchID | int | 主键 |
| hostTeamID | int | 主队的 ID |
| gusetTeamID | int | 客队的 ID |
| marchResult | varchar(20) | 比赛结果 |
| matchTime | date | 比赛日期 |

其中，match 赛程表中的 hostTeamID 与 guestTeamID 都和 team 表中的 teamID 关联，查出

2006-6-1 到 2006-7-1 之间举行的所有比赛，并且用以下形式列出：拜仁 2:0 不莱梅 2006-6-21

#创建参赛队伍表

CREATE TABLE team( teamID INT UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY, teamName VARCHAR(30) NOT NULL

);

#向参赛队伍表中插入记录

INSERT INTO team(teamName) VALUES('拜仁');

INSERT INTO team(teamName) VALUES('不莱梅');

INSERT INTO team(teamName) VALUES('皇家马德里');

INSERT INTO team(teamName) VALUES('巴塞罗那');

INSERT INTO team(teamName) VALUES('切尔西');

INSERT INTO team(teamName) VALUES('曼联');

INSERT INTO team(teamName) VALUES('AC 米兰');

INSERT INTO team(teamName) VALUES('国际米兰');

#创建 match 赛程表

CREATE TABLE mat( matchID INT UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEy, hostTeamID INT UNSIGNED NOT NULL, guestTeamID INT UNSIGNED NOT NULL, matchResult VARCHAR(20) NOT NULL, matchTime DATE NOT NULL

);

#向赛程表中插入几条记录

INSERT INTO mat(hostTeamID,guestTeamID,matchResult,matchTime)

VALUES(1,2,'3:1','2006-6-15');

INSERT INTO mat(hostTeamID,guestTeamID,matchResult,matchTime)

VALUES(3,4,'2:2','2006-6-28');

INSERT INTO mat(hostTeamID,guestTeamID,matchResult,matchTime) VALUES(5,6,'0:2','2006-7-10');

INSERT INTO mat(hostTeamID,guestTeamID,matchResult,matchTime)

VALUES(7,8,'5:3','2006-5-30');

#查询语句

SELECT t1.teamName,matchResult,t2.teamName,matchTime FROM mat

LEFT JOIN team AS t1

ON hostTeamID = t1.teamID

LEFT JOIN team AS t2

ON guestTeamID = t2.teamID

WHERE matchTime BETWEEN '2006-6-1' AND '2006-7-1';

#### （★★）请简述项目中优化sql语句执行效率的方法?

1. 尽量选择较小的列
2. 将where中用的比较频繁的字段建立索引
3. select子句中避免使用‘\*’
4. 避免在索引列上使用计算、not in 和<>等操作
5. 当只需要一行数据的时候使用limit 1
6. 保证单表数据不超过200W，适时分割表。针对查询较慢的语句，可以使用explain 来分析该语句具体的执行情况。
7. 避免改变索引列的类型。
8. 选择最有效的表名顺序，from字句中写在最后的表是基础表，将被最先处理，在from子句中包含多个表的情况下，你必须选择记录条数最少的表作为基础表。
9. 避免在索引列上面进行计算。
10. 尽量缩小子查询的结果

#### （★★）使用java写一段简单查询，查出所有姓名为“张三”的内容并打印出来

|  |  |  |
| --- | --- | --- |
| 表 user |  |  |
| name tel | content | date |
| 张三 13333663366 | 大专毕业 | 2006-10-11 |
| 张三 13612312331 | 本科毕业 | 2006-10-15 |
| 张四 021-55665566 | 中专毕业 | 2006-10-15 |

请根据上面的题目完成代码：

select \* from user where name='张三'

#### （★）写出 SQL语句的格式 : 插入 ，更新 ，删除 （卓望）

表名 user

name tel content date

张三 13333663366 大专毕业 2006-10-11 张三 13612312331 本科毕业 2006-10-15 张四 021-55665566 中专毕业 2006-10-15

(a).有一新记录(小王 13254748547 高中毕业 2007-05-06)请用SQL语句新增至表中

(b).请用sql语句把张三的时间更新成为当前系统时间 (c).请写出删除名为张四的全部记录

INSERT INTO user(name,tel,content,date)VALUES('小王','13254748547','高中毕业

','2007-05-06');

UPDATE user SET date = date\_format(now(),'%Y-%m-%d') WHERE name = '张三'

DELETE FROM user WHERE name = '张四'

#### （★★）数据库中的事务是什么?

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，即使只有一个操作失败，事务也不成功。如果所有操作完成，事务则提交，其修改将作用于所有其他数据库进程。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。ACID 四大特性,原子性、隔离性、一致性、持久性。

#### （★）从表 login 中选出 name 字段包含 admin 的前 10 条结果所有信息的 sql 语句 （酷讯）

SELECT \* FROM login WHERE name LIKE ‘%admin%’LIMIT 10;

#### （★★）表中有A B C三列,用SQL语句实现：当A列大于B列时选择A列，否则选择B列，当 B列大于C列时选择B列否则选择C列。

使用case语句，如下：

SELECT CASE WHEN A > B THEN A ELSE B END,

CASE WHEN B > C THEN B ELSE C END

FROM table

#### （★★★）取得最新一次添加记录（假设id为主键，并且是自增类型）所产生的id的函数是什么？

mysqli\_insert\_id();如果上一查询没有产生 AUTO\_INCREMENT 的值，则 mysqli\_insert\_id() 返回 0。

#### （★）在平常mysql优化方面，最基本的也是最重要的优化是（）。（奇矩互动）

查询优化

#### （★★★）用户互为好友的SNS存储结构怎么设计。（亿邮）

首先是有用户表，如下：

CREATE TABLE user( id INT UNSIGNED NOT NULL PRIMARY KEY AUTO\_INCREMENT, username VARCHAR(30) NOT NULL DEFAULT '' COMMENT '用户名', email VARCHAR(50) NOT NULL DEFAULT '' COMMENT '邮箱', password CHAR(32) NOT NULL DEFAULT '' COMMENT '密码'

)engine=MyISAM charset=utf8 comment='用户表';

其次是用户间的关系，如下：

CREATE TABLE relation( rel\_id INT UNSIGNED NOT NULL PRIMARY KEY AUTO\_INCREMENT, fuid INT UNSIGNED NOT NULL COMMENT '关注人的 id', suid INT UNSIGNED NOT NULL COMMENT '被关注人的 id', relation\_type ENUM('S','D') NOT NULL DEFAULT 'S' COMMENT '关系，s 为关注，

d 表示为好友'

)engine=MyISAM charset=utf8 comment='用户关系表';

#### （★★★）假设现在有一个数据库服务器，服务器地址为 192.168.0.110，用户名为 root 密码为 password 请使用 java编写一个连接该数据库的代码 （亿邮）

#### （★★★）简述在MySQL数据库中MyISAM和InnoDB的区别 （亿邮）

区别于其他数据库的最重要的特点就是其插件式的表存储引擎。切记：存储引擎是基于表的，而不是数据库。  
InnoDB与MyISAM的区别：  
InnoDB存储引擎，主要面向OLTP(Online Transaction Processing，在线事务处理)方面的应用，是第一个完整支持ACID事务的存储引擎(BDB第一个支持事务的存储引擎，已经停止开发)。  
特点：  
行锁设计、支持外键；  
支持类似于Oracle风格的一致性非锁定读(即：默认情况下读取操作不会产生锁)；  
InnoDB将数据放在一个逻辑的表空间中，由InnoDB自身进行管理。从MySQL4.1版本开始，可以将每个InnoDB存储引擎的表单独存放到一个独立的ibd文件中；  
InnoDB通过使用MVCC(多版本并发控制：读不会阻塞写，写也不会阻塞读)来获得高并发性，并且实现了SQL标准的4种隔离级别(默认为REPEATABLE级别)；  
InnoDB还提供了插入缓冲(insert buffer)、二次写(double write)、自适应哈希索引(adaptive hash index)、预读(read ahead)等高性能和高可用的功能；  
InnoDB采用了聚集(clustered)的方式来存储表中的数据，每张标的存储都按主键的顺序存放(如果没有显式的在建表时指定主键，InnoDB会为每一行生成一个6字节的ROWID，并以此作为主键)；  
InnoDB表会有三个隐藏字段：除了上面提到了6字节的DB\_ROW\_ID外，还有6字节的DB\_TX\_ID(事务ID)和7字节的DB\_ROLL\_PTR(指向对应回滚段的地址)。这个可以通过innodb monitor看到；  
MyISAM存储引擎是MySQL官方提供的存储引擎，主要面向OLAP(Online Analytical Processing,在线分析处理)方面的应用。  
特点：  
不支持事务，支持表所和全文索引。操作速度快；  
MyISAM存储引擎表由MYD和MYI组成，MYD用来存放数据文件，MYI用来存放索引文件。MySQL数据库只缓存其索引文件，数据文件的缓存交给操作系统本身来完成；  
MySQL5.0版本开始，MyISAM默认支持256T的单表数据；

#### （★★★）现在有下面一个查询语句 select \* from tabname where id=2 and password='abc’如何判断它是现在是最优的。（亿邮）

可以使用 explain select \* from tabname where id=2 and password='abc’来分析其执行情况。

#### （★★★）请问如何在Mysql操作中如何写入utf8格式数据 （亿邮）

首先确保数据库中的表是基于utf8编码的，其次java文件是utf8编码，在执行mysql 操作之前，对于要操作的文本如果是utf8 编码，则可以直接操作，如果是其它编码，则可以使用函数将其转化为utf8编码，然后写入。

#### （★★★）mysql 中 varchar的最大长度是多少？用什么类型的字段存储大文本？date 和 datetime和timestamp什么区别？怎么看数据库中有哪些sql正在执行？ （卓望）

varchar的最大有效长度由最大行大小和使用的字符集确定。整体最大长度是65532字节。在varchar存字符串的时候，第一个字节是空的，不存任何的数据，然后还需要两个字节来存放字符串的长度。所以有效长度就是 65535 - 1 - 2 = 65532。

由字符集来确定，字符集分单字节和多字节，如果是单字节，如 latin1，则最多可以存放65532个字符，如果是多字节，如GBK则可以存放32766个字符，UTF8则可以存放21844 个字符。

存储大文本可以使用text类型。

date表示日期，其范围为 1000-01-01 ~ 9999-12-31

datetime表示日期时间，其范围为 1000-01-01 00:00:00 ~ 9999-12-31 23:59:59

timestamp 是 unix 时间戳的日期时间表示方式，其范围较小为 1970-01-01

00:00:00~2038-01-19 03:14:07 ,timestamp具备自动初始化和自动更新功能。

查看数据库中正在执行的 sql 语句可以使用日志，也可以使用 show processlist 命令。

#### （★★★）对于关系型数据库而言，索引是相当重要的概念，请回答有关索引的几个问题：

a)、索引的目的是什么？

1.快速访问数据表中的特定信息，提高检索速度

2.创建唯一性索引，保证数据库表中每一行数据的唯一性。

3.加速表和表之间的连接

4.使用分组和排序子句进行数据检索时，可以显著减少查询中分组和排序的时间

b)、索引对数据库系统的负面影响是什么？

负面影响：创建索引和维护索引需要耗费时间，这个时间随着数据量的增加而增加；索引需要占用物理空间，不光是表需要占用数据空间，每个索引也需要占用物理空间；当对表进行增、删、改、的时候索引也要动态维护，这样就降低了数据的维护速度。

c)、为数据表建立索引的原则有哪些？

1、在最频繁使用的、用以缩小查询范围的字段上建立索引。

2、在频繁使用的、需要排序的字段上建立索引

d)、 什么情况下不宜建立索引？

1、对于查询中很少涉及的列或者重复值比较多的列，不宜建立索引。

2、对于一些特殊的数据类型，不宜建立索引，比如文本字段（text）等。

#### （★★★）请举例说明在你的开发中用什么方法来加快页面的加载速度？

要用到服务器资源时才打开，及时关闭服务器资源，数据库添加索引，页面可生成静态，图片等大文件单独服务器。使用代码优化工具。

#### （★★★）如何设计或配置Mysql，才能达到高效使用的目的？

1、数据库设计方面，设计结构良好的数据库，允许部分数据冗余。选取最适用的字段属性，尽可能把字段设置为NOTNULL,这样在查询的时候，数据库不用去比较NULL值。

2、系统架构设计方面，表散列，把海量数据散列到几个不同的表里面，集群，数据库查询和写入分开。

写高效的sql语句，以提高效率。

使用连接（join）来代替子查询。

使用联合（union）来代替手动创建的临时表。

所得皆必须，只从数据库取必须的数据。

必要的时候用不同的存储引擎，比如Innodb 可以减少死锁，HEAP可以提高一个数量级的查询速度。

使用事务

使用外键

使用索引。

#### （★★★）设定网站的用户数量在千万级，但是活跃用户的数量只有1%，如何通过优化数据库提高活跃用户的访问速度？

我们可以根据用户的活跃程度，把活跃的用户提取出来放到另外一张表里面，每次活跃的用户登录的时候就直接到活跃用户表中进行查询，这样就提高了数据库的查询速度。

#### （★★★）SQL注入漏洞产生的原因？如何防止？

SQL注入产生的原因：程序开发过程中不注意规范书写sql语句和对特殊字符进行过滤，导致客户端可以通过全局变量POST和GET提交一些sql语句正常执行。

防止SQL注入的方式：

1. 开启配置文件中的magic\_quotes\_gpc 和 magic\_quotes\_runtime设置
2. 执行sql语句时使用addslashes进行sql语句转换
3. Sql语句书写尽量不要省略双引号和单引号。
4. 过滤掉sql语句中的一些关键词：update、insert、delete、select、 \* 。
5. 提高数据库表和字段的命名技巧，对一些重要的字段根据程序的特点命名，取不易被猜到的。
6. 控制错误信息，不要在浏览器上输出错误信息，将错误信息写到日志文件中。

#### （★）你常用到的 mysql 命令？

Show databases

Show tables

Insert into 表名() values()

Update 表名 set 字段=值 where ...

Delete from 表名 where ...

Select \* from 表名 where 条件 order by ...

Desc/asc limit ... Group by ... Having ...

#### （★）mysql常用命令的应用？

**进入 l mysql 管理命令行的命令？**

Mysql -uroot -p 回车 密码

**show databases; 这个命令的作用？**

显示当前 mysql 服务器中有哪些数据库

**show create database mysql; 这个命令的作用？**

显示创建数据库的 sql 语句

**show create table user; 这个命令的作用？**

显示创建表的 sql 语句

**desc user; 这个命令的作用？**

查询 user 表的结构

**explain select \* from user; 这个命令的作用？**

获取 select 相关信息

**show processlist; 这个命令的作用？**

显示哪些线程正在运行

**SHOW VARIABLES; 这个命令的作用？**

显示系统变量和值

**SHOW VARIABLES like ’ %conn% ’ 这个命令的作用？**

显示系统变量名包含 conn 的值

**LEFT JOIN 写一个 SQL 语句？**

SELECT A.id,A.class FROM A LEFT JOIN B ON A.cid=B.id

#### （★★）什么是JDBC？ 解释下驱动(Driver)在JDBC中的角色。

JDBC是允许用户在不同数据库之间做选择的一个抽象层。JDBC允许开发者用JAVA写数据库应用程序，而不需要关心底层特定数据库的细节。

JDBC驱动提供了特定厂商对JDBC API接口类的实现，驱动必须要提供java.sql包下面这些类的实现：Connection, Statement, PreparedStatement,CallableStatement, ResultSet和Driver。

#### （★★）Class.forName()方法有什么作用？

初始化参数指定的类，并且返回此类对应的Class 对象

#### （★★）PreparedStatement比Statement有什么优势？

PreparedStatements是预编译的，因此，性能会更好。同时，不同的查询参数值，PreparedStatement可以重用。

#### （★★）数据库连接池是什么意思？ 说出数据连接池的工作机制是什么?

像打开关闭数据库连接这种和数据库的交互可能是很费时的，尤其是当客户端数量增加的时候，会消耗大量的资源，成本是非常高的。可以在应用服务器启动的时候建立很多个数据库连接并维护在一个池中。连接请求由池中的连接提供。在连接使用完毕以后，把连接归还到池中，以用于满足将来更多的请求。

J2EE服务器启动时会建立一定数量的池连接，并一直维持不少于此数目的池连接。

调用：客户端程序需要连接时，池驱动程序会返回一个未使用的池连接并将其表记为 忙。如果当前没有空闲连接，池驱动程序就新建一定数量的连接，新建连接的数量有配置参数决定。

释放：当使用的池连接调用完成后，池驱动程序将此连接表记为空闲， 其他调用就可以使用这个连接。

#### （★★）DBUtils是什么?

DbUtils是Apache组织提供的一个对JDBC进行简单封装的开源工具类库，使用它能够简化JDBC应用程序的开发，同时也不会影响程序的性能。

#### （★★）下面哪个SQL命令用来向表中添加列（）

A.MODIFY TABLE TableName ADD COLUMN ColumnName

B.MODIFY TABLE TableName ADD ColumnName

C.ALTER TABLE TableName ADD COLUMN ColumnName

D.ALTER TABLE TableName ADD ColumnName Type

答:D

#### （★★）数据库中的事务是什么?

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，即使只有一个操作失败，事务也不成功。如果所有操作完成，事务则提交，其修改将作用于所有其他数据库进程。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。ACID 四大特性,原子性、隔离性、一致性、持久性。

#### （★★★）pdate和saveOrUpdate的区别？

update()和saveOrUpdate()是用来对跨Session的PO进行状态管理的。

update()方法操作的对象必须是持久化了的对象。也就是说，如果此对象在数据库中不存在的话，就不能使用update()方法。

saveOrUpdate()方法操作的对象既可以使持久化了的，也可以使没有持久化的对象。如果是持久化了的对象调用saveOrUpdate()则会 更新数据库中的对象；如果是未持久化的对象使用此方法,则save到数据库中。

#### （★★★）如果某段与数据库交互的程序运行较慢你将如何处理?

一是首先提高数据库的查询速度，比如增加索引，优化表的结构。

二是优化程序代码，如果查询比较多，可以尽量用条件查询，减少查询语句，比如能用一条查询语句就不用两条。

三就是提高服务器的速度，优化服务器，把不必要的进程关掉。

#### （★）以下哪个不是与Mysql服务器相互作用的通讯协议()

A.TCP/IP B.UDP C.共享内存 D.Socket

答: B

#### （★）下面有关jdbc statement的说法错误的是？

1. JDBC提供了Statement、PreparedStatement 和 CallableStatement三种方式来执行查询语句，其中 Statement 用于通用查询， PreparedStatement 用于执行参数化查询，而 CallableStatement则是用于存储过程
2. 对于PreparedStatement来说，数据库可以使用已经编译过及定义好的执行计划，由于 PreparedStatement 对象已预编译过，所以其执行速度要快于 Statement 对象”
3. PreparedStatement中，“?” 叫做占位符，一个占位符可以有一个或者多个值
4. PreparedStatement可以阻止常见的SQL注入式攻击

答: C

#### （★）下面哪一项不是加载驱动程序的方法？

1. 通过DriverManager.getConnection方法加载
2. 调用方法 Class.forName
3. 通过添加系统的jdbc.drivers属性
4. 通过registerDriver方法注册

答: A

#### （★★）数据库题（\*）学生表St (id ‘学号’，name ‘姓名’，cid ‘班级id’) 分数表Sc (id ‘学号’，kid ‘科目id’，score‘分 数’) 科目表K (id ‘科目id’，name ‘科目名’，tid ‘老师id’) 列出有二门以上(含两门)不及格课程的学生姓名及其平均成绩

SELECT

res.sname,

AVG(res.score) average

FROM

(SELECT

st.`id` as id,

st.`name` as sname,

sc.`score` as score,

k.`name` as kname

FROM

st,

sc,

k

WHERE

st.`id`=sc.`id` and sc.`kid` = k.`id` and score<60

)as res

GROUP BY

res.`id`

HAVING

COUNT(res.id)>=2

#### （★）一个表中有多个重复id的记录，通过SQL语句把存在重复id的记录查出来， 并显示每个id共有多少条重复记录数。

Select

id,

count(id)

from

table

group by

id

#### （★★）表A,表B结构如下:

表A基础信息表

|  |  |  |
| --- | --- | --- |
| Name | Code | Type |
| 用户代码 | Usercode | Interger |
| 用户名称 | Username | Varchar(50) |
| 住址 | Address | Varchar(100) |

表B账户信息表

|  |  |  |
| --- | --- | --- |
| Name | Code | Type |
| 用户代码 | Usercode | Interger |
| 账号 | Account | Varchar(50) |
| 开户银行 | bank | Varchar(100) |

A表与B表为一对多关系，请使用一条SQL语句查询用户代码在10至10000之间的其开户银行信息，要求开户银行信息不可重复。

Select

A.Usercode,

A.Username,

A.Address,

B.Account,

B.bank

From

A,B

Where

A.Usercode=B.Usercode and A.Usercode>=10 and A.Usercode<=10000

#### （★）写一个SQL，查出user表中姓名重名的名字。例如:

User表

|  |  |
| --- | --- |
| id | name |
| 1 | 张三 |
| 2 | 李四 |
| 3 | 王五 |
| 4 | 张三 |
| 5 | 李四 |
| 6 | 赵六 |

查询重名的结果：张三、李四

SELECT

id,

`name`,

count(`name`) as sum

FROM

table

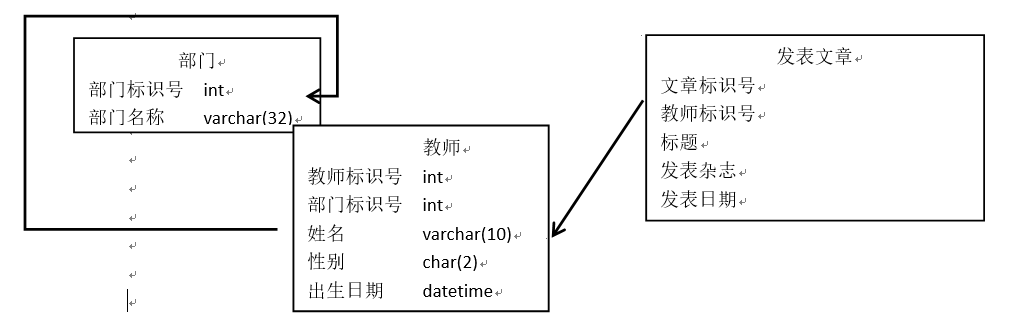
GROUP BY

`name`

HAVING

sum>=2

#### （★★）数据库物理模型如下图所示：



请写出查询部门名称为“高一语文组”的所有女教师发表的文章信息的SQL语句，要求结果集（包含的）

请写出查询部门名称为“高一语文组”的所有女教师发表的文章信息的SQL请句，要求结果集(包含的数据项有：部门名称、教师姓名、性别、发表文章标题，发表杂志、发表日期)以发表日期为降序排序。

Select

教师.姓名，

教师.性别，

部门.部门名称，

发表文章.标题，

发表文章.发表杂志，

发表文章.发表日期

From

教师，部门，发表文章

Where

教师.教师标识号=发表文章.教师标识号 and

教师.部门标识号=部门.部门标识号 and

部门.部门名称=“高一语文组”and

教师.性别=“女“

Order by

发表文章.发表日期 desc

#### （★★）表结构

表名 1 cardapply

字段（字段名/类型/长度）

g\_applyne varcher 8;//申请单号(关键字)

g\_applydate bigint 8;//申请日期

g\_state varchar 2;申请状态

表2 cardapplydetail

字段（字段名/类型/长度）

g\_applyne varcher 8;//申请单号(关键字)

g\_name varchar 30;//申请人姓名

g\_idcard varchar 18;申请人身份证号

g\_state varchar 2;申请状态

其中，两个表的关联字段为申请单号。

题目:

查询身份证号码为440401430103082的申请日期

select

cardapplydetail.g\_idcard,

cardapply.g\_applydate

from

cardapply,

cardapplydetail

where

cardapply.g\_applyne = cardapplydetail.g\_applyne and

cardapplydetail.g\_idcard = '440401430103082';

查询同一个身份证号码有两条以上记录的身份证号码及记录个数、

select

g\_idcard,

count(g\_idcard)

from

cardapplydetail

group by g\_idcard

having count(g\_idcard)>=2

4. 删除cardapplydetail表中所有姓李的记录

delete from cardapplydetail where g\_name like "李%"

#### （★★）简述 数据库查询语句中，左连接left join与右连接right join的区别

左连接left join:以左表为基准，根据on条件过滤连接生成临时表，on后面的过滤条件对左表无效

右连接right join:以右表为基准，根据on条件过滤连接生成临时表，on后面的过滤条件对右表无效

#### （★★）数据库题

Student (S#，Sname，Sage，Ssex)学生表

Course (C#,Cname,T#)课程表

SC (S#,C#,score) 成绩表

Teacher (T#,Tname) 教师表

1写出学生建表语句

CREATE TABLE `student` (

`S#` varchar(255) CHARACTER NOT NULL,

`Sname` varchar(255) CHARACTER NULL DEFAULT NULL,

`Sage` int(10) NULL DEFAULT NULL,

`Ssex` varchar(2) CHARACTER DEFAULT NULL,

PRIMARY KEY (`S#`) USING BTREE

)

2查询姓“张”的学生名单

select \* from Student where Sname like '张%'

3查询所有同学的学号、姓名、选课数、总成绩

select

Student.S#,

Sname,

count(score),

sum(score)

from

Student,SC

where

Student.S# = SC.S#

group by

Student.S#

4查询选修 “叶平”老师所教课程的学生中，成绩最高的学生姓名及其成绩

select

Student.Sname,

max(score)

from

Student,Course,SC,Teacher

where

Student.S#=SC.S# and

Course.C#=SC.C# and

Teancher.Tname='叶平'

group by

Course.Cname

5在数据库中，查看表中重复记录

select \* from vitae a

where (a.peopleId,a.seq) in (select peopleId,seq from vitae group by peopleId,seq having count(\*) > 1)

and rowid not in (select min(rowid) from vitae group by peopleId,seq having count(\*)>1)

#### （★★）编写sql语句，根据下面两个表，选择条件：所在地是非伦敦的并且职业是医生或护士，且分数大于20的人员。

并按年龄从大到小输出所有列：ID Name Age Score Gender Job City

注：id表示员工号，Gender表示性别

表1 ID Name Age Score Gender

1 Kevin 34 50 male

2 lack 20 20 female

表2 ID Job City

1 docto Lodon

2 nuse NewYork

3 teacher Lodon

答：

select

表1.ID,

Name,

Age,

Score,

Gender,

Job,

City

from

表1,表2

where

表1.ID = 表2.ID and

City <> 'Lodon' and

(Job ='docto' or Job = 'nuse') and

Score >20

#### （★）什么是数据库的主键。在msyql中如何创建主键，请写出sql语句

关系型数据库中的一条记录中有若干个属性，若其中某一个属性组(注意是组)能唯一标识一条记录，该属性组就可以成为一个主键

如：新建表：

create table [表名]

(

[自动编号字段] int IDENTITY (1,1) PRIMARY KEY ,

[字段1] nVarChar(50) default \'默认值\' null ,

[字段2] ntext null ,

[字段3] datetime,

[字段4] money null ,

[字段5] int default 0,

[字段6] Decimal (12,4) default 0,

[字段7] image null ,

#### （★★）为什么要使用数据库连接池?

使用原因：

数据库连接是非常占用资源的，尤其是在高并发的情况下，如果每次都去建立数据库连接就会有性能问题，也会影响一个应用程序的延展性，针对这个问题，连接池出现了，连接池就是为了解决这个问题的。

连接池原理：

数据库连接池负责分配、管理、释放连接，它保证应用程序可以重复使用同一个连接而不需要每次都建立数据库连接，如果数据库连接时间超过设置的最长数据库连接时间会自动释放链接，为了避免因为没有释放链接而导致的数据库连接遗漏，因此，数据库连接池可以明显的提高数据库的连接性能。

数据库连接池在初始化的时候会放入一定数量的连接，这个连接是由最小连接数决定的，就算没有用到这些连接，这个连接也会放在连接池中。如果连接数超过最大连接数，那么会放入队列中等待释放链接再使用。

连接池的优点，也就是为啥要使用连接池？

1、资源重用

2、更快的系统反应速度

3、新的分配方式

4、统一的连接管理，避免数据库连接泄漏

#### （★★）用一条sql语句查询出每门课都大于80分的学生姓名

Name class score

张三 语文 81

张三 数学 75

李四 语文 76

李四 数学 90

王五 语文 81

王五 数学 100

王五 英语 90

Select Name from stuScore where score>80

#### （★★）Mysql优化从A账户转账到B账户，可以拆解为几个原子步骤？每个步骤有何风险？如何避免？

帐号不存在、金额不足

事务回滚

1 A帐号检查 帐号不存在、金额不足

2 B帐号检查 帐号不存在、金额不足

3 A账号建立转出帐

4 B账号建立转入帐

5确认修改账号金融额

#### （★★）MySql的存储引擎和索引选择有哪些考虑？

MySQL的存储引擎包括：MyISAM、InnoDB、BDB、MEMORY、MERGE、EXAMPLE、NDBCluster、ARCHIVE、CSV、BLACKHOLE、FEDERATED等，其中InnoDB和BDB提供事务安全表，其他存储引擎都是非事务安全表。

最常使用的2种存储引擎:

1.Myisam是MySQL的默认存储引擎，当create创建新表时，未指定新表的存储引擎时，默认使用Myisam。每个MyISAM在磁盘上存储成三个文件。文件名都和表名相同，扩展名分别是.frm（存储表定义）、.MYD(MYData，存储数据)、.MYI(MYIndex，存储索引)。数据文件和索引文件可以放置在不同的目录，平均分布io，获得更快的速度。

2.InnoDB存储引擎提供了具有提交、回滚和崩溃恢复能力的事务安全。但是对比Myisam的存储引擎，InnoDB写的处理效率差一些并且会占用更多的磁盘空间以保留数据和索引。

MySQL目前主要有以下几种索引类型：

1.普通索引:

2.唯一索引

3.主键索引

4.组合索引

5.全文索引

#### （★★）下列属于关系型数据库的是(AB)

A. Oracle B.MySql C.IMS D.MongoDB

#### （★★）sql中对某一字段order，如果该字段的值不是唯一的，可能会产生什么问题？

order by 排序值相同结果不唯一，可能导致分页查询的重复或者导致分页查询查询条数不同结果显示发生变化，数据库order by排序不唯一，会导致最终的数据排序是不是稳定的 ，建议在 order by 最后，增加 PK 列或 ROWID，rowid还能增加查询效率！

#### （★★）SQL中内连接和外连接的区别是什么？

内连接的语法：

SELECT XXX FROM XXX INNER JOIN XXX ON XXX;

内连接指的是把表连接时表与表之间匹配的数据行查询出来，就是两张表之间数据行匹配时，要同时满足ON语句后面的条件才行。

左连接和右连接，又叫左外连接和右外连接。

我们来看一下左连接的语法

SELECT XXX FROM XXX LEFT OUTER JOIN XXX ON XXX

#### （★★）左连接的意思是，无论是否符合ON语句后面的表连接条件都会把左边那张表的记录全部查询出来，右边的那张表只匹配符合条件的数据行。右连接则与之相反（这里同样OUTER 可以省略）。(★★)两张表A和B，要查出Name在A中不在B中的记录，并按Age倒序排列，写出SQL语句

SELECT

centerTable.IDA,

centerTable.nameA,

centerTable.AgeA

FROM

(

select

A.ID as IDA,

A.`Name` as nameA,

A.Age as AgeA,

B.`Name` as nameB

from

A

left join

B

on

A.`Name`=B.` Name`

) centerTable

WHERE

ISNULL(centerTable.nameB)

表A

|  |  |  |
| --- | --- | --- |
| ID | Name | Age |
| 1 | Li | 26 |
| 2 | Wang | 24 |
| 3 | Zhang | 30 |
| 4 | Liu | 29 |

表B

|  |  |  |
| --- | --- | --- |
| ID | Name | Score |
| 1 | Wang | 90 |
| 2 | Zhao | 65 |
| 3 | Zhang | 87 |

#### （★★）按下述要求写出相应的SQL语句。

用户表T1

|  |  |
| --- | --- |
| U\_id | U\_name |
| 1 | 张三 |
| 2 | 李四 |

收入表T2

|  |  |  |  |
| --- | --- | --- | --- |
| S\_id | S\_date | U\_id | S\_income |
| 1001 | 20121201153611 | 1 | 100.00 |
| 1002 | 20121209183609 | 1 | 300.00 |
| 1003 | 20121230090001 | 2 | 800.00 |
| 1004 | 20121231103159 | 2 | 750.00 |

请用一个sql查询12月份收入的总额超过1500的用户

SELECT

income.U\_id,

t1.U\_name

FROM

(SELECT

U\_id,

SUM(S\_income) as incomeSum

FROM

T2

WHERE

SUBSTRING(S\_date,5,2)=12

GROUP BY

U\_id) as income,

T1

WHERE

income.incomeSum>1500

and

t1.U\_id=income.U\_id

;

结果

|  |  |
| --- | --- |
| 2 | 李四 |

#### （★★）存储过程和函数的区别？

答：存储过程和函数目的是为了 可重复地 执行操作数据库的sql语句的集合。区别是写法和调用上。

相同点：1.创建语法结构相似，都可以携带多个传入参数和传出参数。2.都是一次编译，多次执行。

不同点：1.存储过程定义关键字用procedure，函数定义用function。2.存储过程中不能用return返回值，但函数中可以，而且函数中必须有return子句。3.执行方式略有不同，存储过程的执行方式有两种（1.使用execute2.使用begin和end），函数除了存储过程的两种方式外，还可以当做表达式使用，例如放在select中（select f1() form dual;）。

总结：如果只有一个返回值，用存储函数，否则，一般用存储过程

#### （★★）假如系统中有如下四张表：

1） 文档表（Doc）：编号（docid）、名称（docname）、描述（docdesc）：docid为主键，docname有索引：

2） 学生表（XS）：学生代码（XSDM）、学生代码（XSMC）：

3） 学科表（XX）：学科代码（XKDM）、学科名称（XKMC）、最高分数（ZGF）：

4） 成绩表（CI）：学生代码（XSDM）、学科代码（XKDM）、成绩（CI）

问题1写出您最常用SQL语句的关键字？

答：distinct、between、in、order by、join、group by、having

问题2以下查询的处理速度（返回数据量相同）从高到低排列？

A、 select \* from doc where docname = ‘test’

B、 select \* from doc where docid = 50

C、 select \* from doc where docdesc = ‘test’

问题3列出有不及格（<低于60分）的学生姓名（SQL语句）

问题4统计每个学生的总分（SQL语句）

问题5列出没有不及格（>=60分）的学生姓名及平均分（SQL语句）

#### （★★）Student（SNO，Sname，Sage，Ssex）学生表

Course（CNO，Cname，TNO）课程表

SC（SNO，CNO，score）成绩表

Teacher（TNO，Tname）教师表

问题：

1. 查询“001“课程比”002“课程成绩高的所有学生的学号（SNO）；

答：select

a.sno

from

(select sno,score from SC where cno='001') a,

(select sno,score from SC where cno='002') b

where a.score>b.score and a.sno=b.sno;

1. 查询平均成绩大于60分的同学的学号和平均成绩；

答：select SNO,avg(score) from sc group by SNO having avg(score) >60;

1. 查询所有同学的学号、姓名、选课数、总成绩；

答：select Student.SNO,Student.Sname,count(SC.CNO),sum(score)

from Student left Outer join SC on Student.SNO=SC.SNO

group by Student.SNO,Sname

1. 查询姓“李“的老师的个数；

答：select count(distinct(Tname))

from Teacher where Tname like '李%';

1. 查询没学过“叶平“老师课的同学的学号（SNO）、姓名（Sname）；

答：select Student.SNO,Student.Sname from Student

where SNO not in (select distinct(SC.SNO) from SC,Course,Teacher where SC.CNO=Course.CNO and Teacher.Tno=Course.Tno and Teacher.Tname='叶平');

#### （★）公司补贴发放流水表的结构和数据如下，要求根据TName查询出重复发放的记录，写出sql语句

ID TName Salary

1 张三 1200

2 李四 1150

3 王五 1200

4 张三 1200

答：select ID, TName, Salary

from A

where TName in (select TName from A group by TName having count(1) >= 2)

#### （★★）数据表设有A、B、C三个字母的联合索引，那么查询WHERE B=？ANDC=?是否能用上该索引？同时列出能使用该索引的全部组合?

答：a) where a=1 and b =2 and c=3 b) where b=2 c) where a >1 and b>2 and c=3 d) where a=1 and b>2 and c=3 e) where c=3 order by a f) where b=2 order by a a 使用联合索引abc b 没有使用索引，因为最左前缀 c 使用索引 c d 使用索引abc e 使用索引c 并且排序使用a索引排序 f 没有使用索引

#### （★）写出SQL语句查询以下表结构里平均分大于70的学生姓名与平均成绩

学生表Student（SID,SNAME）课程表Course(CID,CNAME)

成绩表Score(SID,CID,SCORE)

答：select SNAME, SCORE from Student a

left join(

select SID,avg(SCORE) SCORE from Score group by SID

)b

on a. SID =b. SID

where b. SCORE >70

#### （★★）设计一个数据表实现无限极分类目录的存放，要求写出各个字段的名称与作用

大类1

L二级小类1

L三级小类1

L四级小类1

L五级小类1

大类2

二级小类2

大类3

CREATE TABLE tdb\_goods\_types(

type\_id SMALLINT UNSIGNED PRIMARY KEY AUTO\_INCREMENT,

type\_name VARCHAR(20) NOT NULL,

parent\_id SMALLINT UNSIGNED NOT NULL DEFAULT 0

);

这种数据表至少存在3个字段，第一个是分类的id，第二个字段是分类的名称，第三个字段是他父类的id，也就是说他实际上是通过自身的连接来实现的。

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('家用电器',DEFAULT);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('电脑、办公',DEFAULT);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('大家电',1);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('生活电器',1);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('平板电视',3);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('空调',3);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('电风扇',4);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('饮水机',4);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('电脑整机',2);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('电脑配件',2);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('笔记本',9);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('超级本',9);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('游戏本',9);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('CPU',10);

INSERT tdb\_goods\_types(type\_name,parent\_id) VALUES('主机',10);

#### （★★）学生成绩表

ID 姓名：name 课程：subject 分数：score 学号：stuid

1. 张三 数学 89 1
2. 张三 语文 80 1
3. 张三 英语 70 1
4. 李四 数学 90 2
5. 李四 语文 70 2
6. 李四 英语 80 2

计算每个人的总成绩并排名(要求显示字段：姓名，总成绩)请编写sql

select name,sum(score) as allscore from stuscore group by name order by allscore;

统计如下：学号 姓名 语文 数学 英语 总分 平均分

select name,subject,score,sum(score) as allscore,avg(score) as avgscore from stuscore group by name;

#### （★）表的主键特点中，说法错误的是：D

A:主键每一列都必须为空

B:主键每一列都必须唯一

C:一个表只能定义一个主键

D:主键可以定义在表级或列级

#### （★）数据库中只存放视图的？(C)

A:操作 B:定义 C：限制 D:对应的数据

#### （★）在数据库中，产生数据不一致的根本原因是（D）

A:数据冗余 B:数据存储量太大

C:没有严格保护数据 D:未对数据进行完整性控制

[解析] 由于数据冗余，有时修改数据时，一部分数据修改，而另一部分没有修改，造成同一种数据有多个值，产生数据不一致。

#### （★）执行如下两个查询，结果为（A）

SELECT ename name,sal salary FROM emp order by salary;

SELECT ename name,sal “SALARY” FROM emp order by sal ASC;

A:两个查询结果完全一致 B:第二个查询结果正确，第一个查询结果错误

C: 第一个查询结果正确，第二个查询结果错误 D:两个查询结果不同

#### （★）下面属于关系型数据库的是（AB）

A、Oracle B、Mysql C、IMS D、MongoDB

#### （★★）在mysql数据库中case when函数的作用是什么？写出一个完整的casewhen的例子

case when语句，用于计算条件列表并返回多个可能结果表达式之一。

CASE 具有两种格式：简单 CASE 函数将某个表达式与一组简单表达式进行比较以确定结果。CASE 搜索函数计算一组布尔表达式以确定结果。 两种格式都支持可选的 ELSE 参数。

例子：

表a

|  |  |  |
| --- | --- | --- |
| id | name | age |
| 1 | Li | 26 |
| 2 | Wang | 24 |
| 3 | Zhang | 30 |
| 4 | Liu | 29 |

SELECT \*,CASE WHEN a.age>25 THEN a.age ELSE 'null' END as result from a

结果

|  |  |  |  |
| --- | --- | --- | --- |
| id | name | age | result |
| 1 | Li | 26 | 26 |
| 2 | Wang | 24 | null |
| 3 | Zhang | 30 | 30 |
| 4 | Liu | 29 | 29 |

## 3. 连接池

### （★）4.1**、基本概念及原理**

由上面的分析可以看出，问题的根源就在于对数据库连接资源的低效管理。我们知道，对于共享资源，有一个很著名的设计模式：资源池（Resource Pool）。该模式正是为了解决资源的频繁分配?释放所造成的问题。为解决上述问题，可以采用数据库连接池技术。数据库连接池的基本思想就是为数据库连接建立一个“缓冲池”。预先在缓冲池中放入一定数量的连接，当需要建立数据库连接时，只需从“缓冲池”中取出一个，使用完毕之后再放回去。我们可以通过设定连接池最大连接数来防止系统无尽的与数据库连接。更为重要的是我们可以通过连接池的管理机制监视数据库的连接的数量?使用情况，为系统开发?测试及性能调整提供依据

### （★）**4.2、服务器自带的连接池**

JDBC的API中没有提供连接池的方法。一些大型的WEB应用服务器如BEA的WebLogic和IBM的WebSphere等提供了连接池的机制，但是必须有其第三方的专用类方法支持连接池的用法。

### （★★★）4.3**、并发问题** 为了使连接管理服务具有最大的通用性，必须考虑多线程环境，即并发问题。这个问题相对比较好解决，因为Java语言自身提供了对并发管理的支持，使用synchronized关键字即可确保线程是同步的。使用方法为直接在类方法前面加上synchronized关键字，如： public synchronized Connection getConnection（） **3.2、多数据库服务器和多用户** 对于大型的企业级应用，常常需要同时连接不同的数据库（如连接Oracle和Sybase）。如何连接不同的数据库呢？我们采用的策略是：设计一个符合单例模式的连接池管理类，在连接池管理类的唯一实例被创建时读取一个资源文件，其中资源文件中存放着多个数据库的url地址（）?用户名（）?密码（）等信息。如tx.url=172.21.15.123：5000/tx\_it，tx.user=yang，tx.password=yang321。根据资源文件提供的信息，创建多个连接池类的实例，每一个实例都是一个特定数据库的连接池。连接池管理类实例为每个连接池实例取一个名字，通过不同的名字来管理不同的连接池。 对于同一个数据库有多个用户使用不同的名称和密码访问的情况，也可以通过资源文件处理，即在资源文件中设置多个具有相同url地址，但具有不同用户名和密码的数据库连接信息

### （★★★）**4.4.事务处理**

我们知道，事务具有原子性，此时要求对数据库的操作符合“ALL-ALL-NOTHING”原则,即对于一组SQL语句要么全做，要么全不做。   
在Java语言中，Connection类本身提供了对事务的支持，可以通过设置Connection的AutoCommit属性为false,然后显式的调用commit或rollback方法来实现。但要高效的进行Connection复用，就必须提供相应的事务支持机制。可采用每一个事务独占一个连接来实现，这种方法可以大大降低事务管理的复杂性。

### （★）4.5连接池的分配与释放

连接池的分配与释放，对系统的性能有很大的影响。合理的分配与释放，可以提高连接的复用度，从而降低建立新连接的开销，同时还可以加快用户的访问速度。   
  
对于连接的管理可使用空闲池。即把已经创建但尚未分配出去的连接按创建时间存放到一个空闲池中。每当用户请求一个连接时，系统首先检查空闲池内有没有空闲连接。如果有就把建立时间最长（通过容器的顺序存放实现）的那个连接分配给他（实际是先做连接是否有效的判断，如果可用就分配给用户，如不可用就把这个连接从空闲池删掉，重新检测空闲池是否还有连接）；如果没有则检查当前所开连接池是否达到连接池所允许的最大连接数（maxConn）,如果没有达到，就新建一个连接，如果已经达到，就等待一定的时间（timeout）。如果在等待的时间内有连接被释放出来就可以把这个连接分配给等待的用户，如果等待时间超过预定时间timeout,则返回空值（null）。系统对已经分配出去正在使用的连接只做计数，当使用完后再返还给空闲池。对于空闲连接的状态，可开辟专门的线程定时检测，这样会花费一定的系统开销，但可以保证较快的响应速度。也可采取不开辟专门线程，只是在分配前检测的方法。

### （★★）4.6.连接池的配置和实现

连接池中到底应该放置多少连接，才能使系统的性能最佳？系统可采取设置最小连接数（minConn）和最大连接数（maxConn）来控制连接池中的连接。最小连接数是系统启动时连接池所创建的连接数。如果创建过多，则系统启动就慢，但创建后系统的响应速度会很快；如果创建过少，则系统启动的很快，响应起来却慢。这样，可以在开发时，设置较小的最小连接数，开发起来会快，而在系统实际使用时设置较大的，因为这样对访问客户来说速度会快些。最大连接数是连接池中允许连接的最大数目，具体设置多少，要看系统的访问量，可通过反复测试，找到最佳点。   
  
如何确保连接池中的最小连接数呢？有动态和静态两种策略。动态即每隔一定时间就对连接池进行检测，如果发现连接数量小于最小连接数，则补充相应数量的新连接,以保证连接池的正常运转。静态是发现空闲连接不够时再去检查。

### （★★★）4.7 连接池的实现

#### 4.7.1 连接池模型

本文讨论的连接池包括一个连接池类（DBConnectionPool）和一个连接池管理类（DBConnetionPoolManager）。连接池类是对某一数据库所有连接的“缓冲池”，主要实现以下功能：①从连接池获取或创建可用连接；②使用完毕之后，把连接返还给连接池；③在系统关闭前，断开所有连接并释放连接占用的系统资源；④还能够处理无效连接（原来登记为可用的连接，由于某种原因不再可用，如超时，通讯问题），并能够限制连接池中的连接总数不低于某个预定值和不超过某个预定值。   
  
连接池管理类是连接池类的外覆类（wrapper）,符合单例模式，即系统中只能有一个连接池管理类的实例。其主要用于对多个连接池对象的管理，具有以下功能：①装载并注册特定数据库的JDBC驱动程序；②根据属性文件给定的信息，创建连接池对象；③为方便管理多个连接池对象，为每一个连接池对象取一个名字，实现连接池名字与其实例之间的映射；④跟踪客户使用连接情况，以便需要是关闭连接释放资源。连接池管理类的引入主要是为了方便对多个连接池的使用和管理，如系统需要连接不同的数据库，或连接相同的数据库但由于安全性问题，需要不同的用户使用不同的名称和密码。

## 4.（★★★）数据库优化

### 当只要一行数据时使用Limit 1

答：当你查询表的有些时候，你已经知道结果只会有一条结果，但因为你可能需要去fetch游标，或是你也许会去检查返回的记录数。

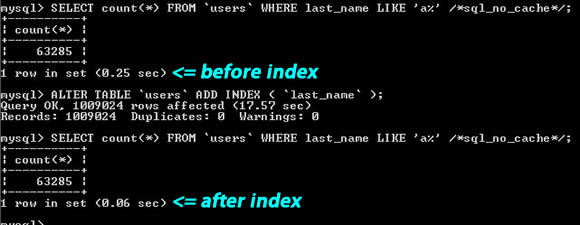
在这种情况下，加上 LIMIT 1 可以增加性能。这样一样，MySQL数据库引擎会在找到一条数据后停止搜索，而不是继续往后查少下一条符合记录的数据。

下面的示例，只是为了找一下是否有“中国”的用户，很明显，后面的会比前面的更有效率。（请注意，第一条中是Select \*，第二条是Select 1）

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | // 没有效率的：  $r = mysql\_query("SELECT \* FROM user WHERE country = 'China'");  if (mysql\_num\_rows($r) > 0) {      // ...  }    // 有效率的：  $r = mysql\_query("SELECT 1 FROM user WHERE country = 'China' LIMIT 1");  if (mysql\_num\_rows($r) > 0) {      // ...  } |

### 为搜索字段建索引

答：索引并不一定就是给主键或是唯一的字段。如果在你的表中，有某个字段你总要会经常用来做搜索，那么，请为其建立索引吧。



从上图你可以看到那个搜索字串 “last\_name LIKE ‘a%'”，一个是建了索引，一个是没有索引，性能差了4倍左右。

另外，你应该也需要知道什么样的搜索是不能使用正常的索引的。例如，当你需要在一篇大的文章中搜索一个词时，如： “WHERE post\_content LIKE ‘%apple%'”，索引可能是没有意义的。你可能需要使用[MySQL全文索引](http://dev.mysql.com/doc/refman/5.1/en/fulltext-search.html" \t "https://coolshell.cn/articles/_blank) 或是自己做一个索引（比如说：搜索关键词或是Tag什么的）

### 在Join表的时候使用相当类型的例子，并将其索引

答：如果你的应用程序有很多 JOIN 查询，你应该确认两个表中Join的字段是被建过索引的。这样，MySQL内部会启动为你优化Join的SQL语句的机制。

而且，这些被用来Join的字段，应该是相同的类型的。例如：如果你要把 DECIMAL 字段和一个 INT 字段Join在一起，MySQL就无法使用它们的索引。对于那些STRING类型，还需要有相同的字符集才行。（两个表的字符集有可能不一样）

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | // 在state中查找company  $r = mysql\_query("SELECT company\_name FROM users      LEFT JOIN companies ON (users.state = companies.state)      WHERE users.id = $user\_id");    // 两个 state 字段应该是被建过索引的，而且应该是相当的类型，相同的字符集。 |

### 千万不要Order by rand（）

答：想打乱返回的数据行？随机挑一个数据？真不知道谁发明了这种用法，但很多新手很喜欢这样用。但你确不了解这样做有多么可怕的性能问题。

如果你真的想把返回的数据行打乱了，你有N种方法可以达到这个目的。这样使用只让你的数据库的性能呈指数级的下降。这里的问题是：MySQL会不得不去执行RAND()函数（很耗CPU时间），而且这是为了每一行记录去记行，然后再对其排序。就算是你用了Limit 1也无济于事（因为要排序）

下面的示例是随机挑一条记录

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | // 千万不要这样做：  $r = mysql\_query("SELECT username FROM user ORDER BY RAND() LIMIT 1");    // 这要会更好：  $r = mysql\_query("SELECT count(\*) FROM user");  $d = mysql\_fetch\_row($r);  $rand = mt\_rand(0,$d[0] - 1);    $r = mysql\_query("SELECT username FROM user LIMIT $rand, 1"); |

### 避免Select\*

答：从数据库里读出越多的数据，那么查询就会变得越慢。并且，如果你的数据库服务器和WEB服务器是两台独立的服务器的话，这还会增加网络传输的负载。

所以，你应该养成一个需要什么就取什么的好的习惯。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | // 不推荐  $r = mysql\_query("SELECT \* FROM user WHERE user\_id = 1");  $d = mysql\_fetch\_assoc($r);  echo "Welcome {$d['username']}";    // 推荐  $r = mysql\_query("SELECT username FROM user WHERE user\_id = 1");  $d = mysql\_fetch\_assoc($r);  echo "Welcome {$d['username']}"; |

### 永远为每张表设置一个ID

答：我们应该为数据库里的每张表都设置一个ID做为其主键，而且最好的是一个INT型的（推荐使用UNSIGNED），并设置上自动增加的AUTO\_INCREMENT标志。

就算是你 users 表有一个主键叫 “email”的字段，你也别让它成为主键。使用 VARCHAR 类型来当主键会使用得性能下降。另外，在你的程序中，你应该使用表的ID来构造你的数据结构。

而且，在MySQL数据引擎下，还有一些操作需要使用主键，在这些情况下，主键的性能和设置变得非常重要，比如，集群，分区……

在这里，只有一个情况是例外，那就是“关联表”的“外键”，也就是说，这个表的主键，通过若干个别的表的主键构成。我们把这个情况叫做“外键”。比如：有一个“学生表”有学生的ID，有一个“课程表”有课程ID，那么，“成绩表”就是“关联表”了，其关联了学生表和课程表，在成绩表中，学生ID和课程ID叫“外键”其共同组成主键。

### 使用Enum而不是Varchar

答：[ENUM](http://dev.mysql.com/doc/refman/5.0/en/enum.html" \t "https://coolshell.cn/articles/_blank) 类型是非常快和紧凑的。在实际上，其保存的是 TINYINT，但其外表上显示为字符串。这样一来，用这个字段来做一些选项列表变得相当的完美。

如果你有一个字段，比如“性别”，“国家”，“民族”，“状态”或“部门”，你知道这些字段的取值是有限而且固定的，那么，你应该使用 ENUM 而不是 VARCHAR。

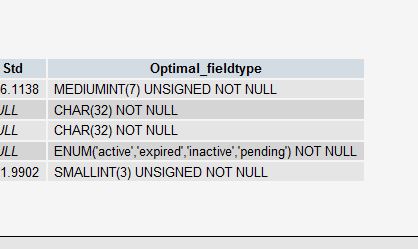
MySQL也有一个“建议”（见第十条）告诉你怎么去重新组织你的表结构。当你有一个 VARCHAR 字段时，这个建议会告诉你把其改成 ENUM 类型。使用 PROCEDURE ANALYSE() 你可以得到相关的建议。

### 从Procedure Analyse（）取得建议

答：[PROCEDURE ANALYSE()](http://dev.mysql.com/doc/refman/5.0/en/procedure-analyse.html) 会让 MySQL 帮你去分析你的字段和其实际的数据，并会给你一些有用的建议。只有表中有实际的数据，这些建议才会变得有用，因为要做一些大的决定是需要有数据作为基础的。

例如，如果你创建了一个 INT 字段作为你的主键，然而并没有太多的数据，那么，PROCEDURE ANALYSE()会建议你把这个字段的类型改成 MEDIUMINT 。或是你使用了一个 VARCHAR 字段，因为数据不多，你可能会得到一个让你把它改成 ENUM 的建议。这些建议，都是可能因为数据不够多，所以决策做得就不够准。

在phpmyadmin里，你可以在查看表时，点击 “Propose table structure” 来查看这些建议



一定要注意，这些只是建议，只有当你的表里的数据越来越多时，这些建议才会变得准确。一定要记住，你才是最终做决定的人。

### 尽可能的使用Not Null

答：除非你有一个很特别的原因去使用 NULL 值，你应该总是让你的字段保持 NOT NULL。这看起来好像有点争议，请往下看。

首先，问问你自己“Empty”和“NULL”有多大的区别（如果是INT，那就是0和NULL）？如果你觉得它们之间没有什么区别，那么你就不要使用NULL。（你知道吗？在 Oracle 里，NULL 和 Empty 的字符串是一样的！)

不要以为 NULL 不需要空间，其需要额外的空间，并且，在你进行比较的时候，你的程序会更复杂。 当然，这里并不是说你就不能使用NULL了，现实情况是很复杂的，依然会有些情况下，你需要使用NULL值。

下面摘自MySQL自己的文档：

*“NULL columns require additional space in the row to record whether their values are NULL. For MyISAM tables, each NULL column takes one bit extra, rounded up to the nearest byte.”*

### Prepared Statements

答：Prepared Statements很像存储过程，是一种运行在后台的SQL语句集合，我们可以从使用 prepared statements 获得很多好处，无论是性能问题还是安全问题。

Prepared Statements 可以检查一些你绑定好的变量，这样可以保护你的程序不会受到“SQL注入式”攻击。当然，你也可以手动地检查你的这些变量，然而，手动的检查容易出问题，而且很经常会被程序员忘了。当我们使用一些framework或是ORM的时候，这样的问题会好一些。

在性能方面，当一个相同的查询被使用多次的时候，这会为你带来可观的性能优势。你可以给这些Prepared Statements定义一些参数，而MySQL只会解析一次。

虽然最新版本的MySQL在传输Prepared Statements是使用二进制形势，所以这会使得网络传输非常有效率。

当然，也有一些情况下，我们需要避免使用Prepared Statements，因为其不支持查询缓存。但据说版本5.1后支持了。

在PHP中要使用prepared statements，你可以查看其使用手册：[mysqli 扩展](http://php.net/manual/en/book.mysqli.php" \t "https://coolshell.cn/articles/_blank) 或是使用数据库抽象层，如： [PDO](http://us.php.net/manual/en/book.pdo.php).

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | // 创建 prepared statement  if ($stmt = $mysqli->prepare("SELECT username FROM user WHERE state=?")) {        // 绑定参数      $stmt->bind\_param("s", $state);        // 执行      $stmt->execute();        // 绑定结果      $stmt->bind\_result($username);        // 移动游标      $stmt->fetch();        printf("%s is from %s\n", $username, $state);        $stmt->close();  } |

### 无缓冲的查询

答：正常的情况下，当你在当你在你的脚本中执行一个SQL语句的时候，你的程序会停在那里直到没这个SQL语句返回，然后你的程序再往下继续执行。你可以使用无缓冲查询来改变这个行为。

关于这个事情，在PHP的文档中有一个非常不错的说明： [mysql\_unbuffered\_query()](http://php.net/manual/en/function.mysql-unbuffered-query.php" \t "https://coolshell.cn/articles/_blank) 函数：

“mysql\_unbuffered\_query() sends the SQL query query to MySQL without automatically fetching and buffering the result rows as mysql\_query() does. This saves a considerable amount of memory with SQL queries that produce large result sets, and you can start working on the result set immediately after the first row has been retrieved as you don’t have to wait until the complete SQL query has been performed.”

上面那句话翻译过来是说，mysql\_unbuffered\_query() 发送一个SQL语句到MySQL而并不像mysql\_query()一样去自动fethch和缓存结果。这会相当节约很多可观的内存，尤其是那些会产生大量结果的查询语句，并且，你不需要等到所有的结果都返回，只需要第一行数据返回的时候，你就可以开始马上开始工作于查询结果了。

然而，这会有一些限制。因为你要么把所有行都读走，或是你要在进行下一次的查询前调用[mysql\_free\_result()](http://us2.php.net/manual/en/function.mysql-free-result.php)清除结果。而且， [mysql\_num\_rows()](http://us2.php.net/manual/en/function.mysql-num-rows.php" \t "https://coolshell.cn/articles/_blank) 或 [mysql\_data\_seek()](http://us2.php.net/manual/en/function.mysql-data-seek.php" \t "https://coolshell.cn/articles/_blank) 将无法使用。所以，是否使用无缓冲的查询你需要仔细考虑。

### 把IP地址存成Unsigned Int

答：很多程序员都会创建一个 VARCHAR(15) 字段来存放字符串形式的IP而不是整形的IP。如果你用整形来存放，只需要4个字节，并且你可以有定长的字段。而且，这会为你带来查询上的优势，尤其是当你需要使用这样的WHERE条件：IP between ip1 and ip2。

我们必需要使用UNSIGNED INT，因为 IP地址会使用整个32位的无符号整形。

而你的查询，你可以使用 [INET\_ATON()](http://dev.mysql.com/doc/refman/5.0/en/miscellaneous-functions.html#function_inet-aton) 来把一个字符串IP转成一个整形，并使用 [INET\_NTOA()](http://dev.mysql.com/doc/refman/5.0/en/miscellaneous-functions.html#function_inet-ntoa) 把一个整形转成一个字符串IP。在PHP中，也有这样的函数 [ip2long()](http://php.net/manual/en/function.ip2long.php) 和 [long2ip()](http://us.php.net/manual/en/function.long2ip.php)。

|  |  |
| --- | --- |
| 1 | $r = "UPDATE users SET ip = INET\_ATON('{$\_SERVER['REMOTE\_ADDR']}') WHERE user\_id = $user\_id"; |

### 固定长度的表会更快

答：如果表中的所有字段都是“固定长度”的，整个表会被认为是 [“static” 或 “fixed-length”](http://dev.mysql.com/doc/refman/5.1/en/static-format.html)。 例如，表中没有如下类型的字段： VARCHAR，TEXT，BLOB。只要你包括了其中一个这些字段，那么这个表就不是“固定长度静态表”了，这样，MySQL 引擎会用另一种方法来处理。

固定长度的表会提高性能，因为MySQL搜寻得会更快一些，因为这些固定的长度是很容易计算下一个数据的偏移量的，所以读取的自然也会很快。而如果字段不是定长的，那么，每一次要找下一条的话，需要程序找到主键。

并且，固定长度的表也更容易被缓存和重建。不过，唯一的副作用是，固定长度的字段会浪费一些空间，因为定长的字段无论你用不用，他都是要分配那么多的空间。

使用“垂直分割”技术（见下一条），你可以分割你的表成为两个一个是定长的，一个则是不定长的。

### 垂直分割

答：“垂直分割”是一种把数据库中的表按列变成几张表的方法，这样可以降低表的复杂度和字段的数目，从而达到优化的目的。（以前，在银行做过项目，见过一张表有100多个字段，很恐怖）

**示例一**：在Users表中有一个字段是家庭地址，这个字段是可选字段，相比起，而且你在数据库操作的时候除了个人信息外，你并不需要经常读取或是改写这个字段。那么，为什么不把他放到另外一张表中呢？ 这样会让你的表有更好的性能，大家想想是不是，大量的时候，我对于用户表来说，只有用户ID，用户名，口令，用户角色等会被经常使用。小一点的表总是会有好的性能。

**示例二**： 你有一个叫 “last\_login” 的字段，它会在每次用户登录时被更新。但是，每次更新时会导致该表的查询缓存被清空。所以，你可以把这个字段放到另一个表中，这样就不会影响你对用户ID，用户名，用户角色的不停地读取了，因为查询缓存会帮你增加很多性能。

另外，你需要注意的是，这些被分出去的字段所形成的表，你不会经常性地去Join他们，不然的话，这样的性能会比不分割时还要差，而且，会是极数级的下降。

### 拆分大的Delete 或 Insert语句

答：如果你需要在一个在线的网站上去执行一个大的 DELETE 或 INSERT 查询，你需要非常小心，要避免你的操作让你的整个网站停止相应。因为这两个操作是会锁表的，表一锁住了，别的操作都进不来了。

Apache 会有很多的子进程或线程。所以，其工作起来相当有效率，而我们的服务器也不希望有太多的子进程，线程和数据库链接，这是极大的占服务器资源的事情，尤其是内存。

如果你把你的表锁上一段时间，比如30秒钟，那么对于一个有很高访问量的站点来说，这30秒所积累的访问进程/线程，数据库链接，打开的文件数，可能不仅仅会让你泊WEB服务Crash，还可能会让你的整台服务器马上掛了。

所以，如果你有一个大的处理，你定你一定把其拆分，使用 LIMIT 条件是一个好的方法。下面是一个示例：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | while (1) {      //每次只做1000条      mysql\_query("DELETE FROM logs WHERE log\_date <= '2009-11-01' LIMIT 1000");      if (mysql\_affected\_rows() == 0) {          // 没得可删了，退出！          break;      }      // 每次都要休息一会儿      usleep(50000);  } |

### 越小的列会越快

答：对于大多数的数据库引擎来说，硬盘操作可能是最重大的瓶颈。所以，把你的数据变得紧凑会对这种情况非常有帮助，因为这减少了对硬盘的访问。

参看 MySQL 的文档 [Storage Requirements](http://dev.mysql.com/doc/refman/5.0/en/storage-requirements.html) 查看所有的数据类型。

如果一个表只会有几列罢了（比如说字典表，配置表），那么，我们就没有理由使用 INT 来做主键，使用 MEDIUMINT, SMALLINT 或是更小的 TINYINT 会更经济一些。如果你不需要记录时间，使用 DATE 要比 DATETIME 好得多。

当然，你也需要留够足够的扩展空间，不然，你日后来干这个事，你会死的很难看，参看[Slashdot的例子](http://news.slashdot.org/article.pl?sid=06/11/09/1534204)（2009年11月06日），一个简单的ALTER TABLE语句花了3个多小时，因为里面有一千六百万条数据。

### 选择正确的存储引擎

答：在 MySQL 中有两个存储引擎 MyISAM 和 InnoDB，每个引擎都有利有弊。酷壳以前文章《[MySQL: InnoDB 还是 MyISAM?](https://coolshell.cn/articles/652.html" \t "https://coolshell.cn/articles/_blank)》讨论和这个事情。

MyISAM 适合于一些需要大量查询的应用，但其对于有大量写操作并不是很好。甚至你只是需要update一个字段，整个表都会被锁起来，而别的进程，就算是读进程都无法操作直到读操作完成。另外，MyISAM 对于 SELECT COUNT(\*) 这类的计算是超快无比的。

InnoDB 的趋势会是一个非常复杂的存储引擎，对于一些小的应用，它会比 MyISAM 还慢。他是它支持“行锁” ，于是在写操作比较多的时候，会更优秀。并且，他还支持更多的高级应用，比如：事务。

下面是MySQL的手册

* [target=”\_blank”MyISAM Storage Engine](http://dev.mysql.com/doc/refman/5.1/en/myisam-storage-engine.html)
* [InnoDB Storage Engine](http://dev.mysql.com/doc/refman/5.1/en/innodb.html)

### 使用一个对象关系映射器（Object Relational Mapper）

答：使用 ORM (Object Relational Mapper)，你能够获得可靠的性能增涨。一个ORM可以做的所有事情，也能被手动的编写出来。但是，这需要一个高级专家。

ORM 的最重要的是“Lazy Loading”，也就是说，只有在需要的去取值的时候才会去真正的去做。但你也需要小心这种机制的副作用，因为这很有可能会因为要去创建很多很多小的查询反而会降低性能。

ORM 还可以把你的SQL语句打包成一个事务，这会比单独执行他们快得多得多。

目前，个人最喜欢的PHP的ORM是：[Doctrine](http://www.doctrine-project.org/" \t "https://coolshell.cn/articles/_blank)。

### 小心“永久链接”

答：“永久链接”的目的是用来减少重新创建MySQL链接的次数。当一个链接被创建了，它会永远处在连接的状态，就算是数据库操作已经结束了。而且，自从我们的Apache开始重用它的子进程后——也就是说，下一次的HTTP请求会重用Apache的子进程，并重用相同的 MySQL 链接。

在理论上来说，这听起来非常的不错。但是从个人经验（也是大多数人的）上来说，这个功能制造出来的麻烦事更多。因为，你只有有限的链接数，内存问题，文件句柄数，等等。

而且，Apache 运行在极端并行的环境中，会创建很多很多的了进程。这就是为什么这种“永久链接”的机制工作地不好的原因。在你决定要使用“永久链接”之前，你需要好好地考虑一下你的整个系统的架

### 优化 Mysql 数据库的方法 ?

1. 优化索引、SQL语句、分析慢查询
2. 设计表的时候严格按照数据库的设计范式来设计数据库
3. 我们可以加上memcached缓存，将经常被访问到的数据，但是不需要经常变化的数据放入至memcached缓存服务器里面，这样的话能够节约太强盘I/0（昨天）
4. 还可优化硬件，在硬件层面，我们可以使用更好的一些硬盘（固态硬盘），使用一些磁盘阵列技术（raid 0,raid1,raid5）

raid0 最简单的 （两块硬盘硬盘相加100G + 100 G = 200G）

raid1 镜像卷 把同样的数据写两份。可以随机从A/B里面读取，更高。硬盘坏了一块，数据也不会丢失

raid5 3块硬盘，坏了一块，另外两块也能工作。

1. 如果以上都做速度还是慢，先不要去切分可以使用MySQL内部自带的表分区技术，来将数据分成不同的文件。这样能够让磁盘在读取的时候效率更高。
2. 可以再做垂直分表，可以将不经常读数据放到另外一个表里面去。这样能够节约磁盘IO
3. 如果发现我们的效率还是不够高，我们可以采用主从的方式来将数据读写分离。
4. 数据量特别大，我们优化起来会很困难可以使数数据库中间件的方式，将数据进行分库分表分机器。（原理，数据路由）
5. 此外，我们还可以采用一些更快的存储方式，例如NoSQL来存储一些我们需要经常访问到的数据。从数据库里面取出来后，再到NoSQL取出一些其他的数据。
6. 此外还有一些表引擎 选择，参数优化和一些相关小技巧都是优化MySQL的方式
7. 我们还可以将我们的业务系统在架构级别进行缓存，静态化和分式式。
8. 不使用全文索引，使用xunsearch，ES或者云服务器上的索引。