Mysql学习

一.关闭和启动数据库

net stop mysql56 停止mysql服务

net start mysql56 启动mysql服务

ps：mysql56是服务名要和管理工具>服务> mysql56的名称保持一致

二.连接数据库

1.连接远程数据库

mysql –h localhost –uroot -proot

例子：mysql –h 192.198.20.200 –uroot –proot

mysql –h localhost –uroot -proot

ps: -h表示连接服务器的IP地址，localhost表示本机的也可是远程的地址，

–uroot表示-u用户名，-proot表示 -密码

2.连接本地数据库

mysql –u用户名 -p密码

mysql -uroot -proot

3.没有密码

myspl –uroot –p

三．基本配置

show databases; 查看有哪几个数据库

create database mysqldata; 创建一个数据库

drop database mysqldata; 删除一个数据库

use test; 使用哪个数据库

show tables; 显示数据库里的表

select version(); 查看版本

select current\_date; 查看日期

select now(); 查看时间

select user(); 查看用户

select database(); 查看当前数据库

show status; 查看运行状态

show create database test; 查看数据库的编码类型，test为数据库名

alter database 表名 character set utf8; 修改数据库字符集

select connection\_id(); 查看链接号

show processlist; 查看当前用户链接数据库的信息

四.目录结构及Notifier介绍

五．Mysql存储引擎

show engines\G; 查看默认存储引擎，默认是InnoDB

六．数值类型

数值数据类型

1.数值类型

整形



TINYINT 默认是有符号的，存储范围在 -128->127之间

unsigned 是无符号，影响范围

M ： 代表宽度，(在zerofill时才有意义)

Zerofill：用零填充(如果某列用了zerofill ，数据默认就是unsigned )

例子：TINYINT(5) zerofill

小数型(浮点型/定点型)

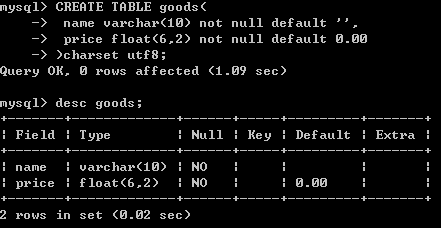


1. Float浮点型

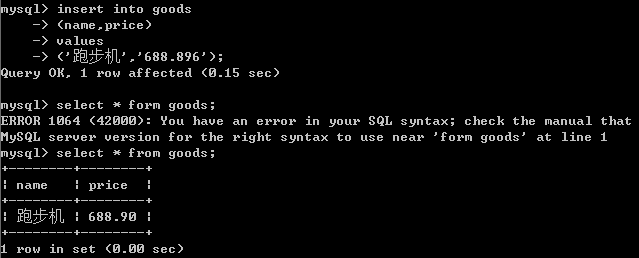
ps：Float(M,D):M代表精度(总位数)，D代表标度(小数位),

例：float(6,2) 表示：能存6个长度数字，有2位小数。如-9999.99到9999.99之间

创建小数表



添加数据



ps：会发现在添加688.896时，数据库会进行四舍五入，把最后一位舍去了。

此时用的四舍五入不是数学里的四舍五入，而是银行里的四舍五入。满五进一而是取决要舍弃哪位数的前一位数

前一位数为1,3,7,9时 后一位数舍弃

前一位数为2,4,6,8时 后一位数进一

2. DOUBLE定点型

decimal(M,D): M代表精度(总位数)，D代表标度(小数位),

decimal更精确

1.字符串类型



char：称为定长，声明时 char(M),M代表宽度（字符串长度），可容纳的字符数

ps:例char(6):表示可以插入6个长度的字符，在utf8的编码方式下，可插入6个汉字，

不足6个字符时，也会占6个字符空间，有点浪费空间。在固定长度下使用，如

四字成语表。

varchar：称为变长，声明时vachar(M),M代表宽度（字符串长度），可容纳的字符数

ps:例varchar(8):表示可以插入8个长度的字符，在utf8的编码方式下，可插入8个汉字，

不足8个字符时，会存多少，占用多少个字符空间，不浪费空间。在任何场景下都可以使用。

char和varchar的区别？？？？？？？？？？？？？



区别1. 两个类型的存储的范围不同。

char定长类型

char(M),M带表宽度（字符长度），0<=M<=255之间

例1.char(10),侧能输入10个字符，也就是能保存10个汉字

Varchar变长类型

varchar(M)，M代表宽度,0<=M<=65535(以asci字符为)，utf8可保存22000个左右

varchar存入多少字符！占用多少字符

区别2.两个类型，实占空间不一样，利用也率不一样。

char定长类型

存M个字符，占M个空间

Varchar变长类型

存M个字符，占M+(1到2)字节

区别3.两个类型，对字符末尾的空格处理不一样

char定长类型

存入末尾带空格的字符，取出来末尾没有空格。

例：存入：123&nbsp; &nbsp 取出：123

Varchar变长类型

存入末尾带空格的字符，取出来末尾也带有空格。

例：存入：123&nbsp; &nbsp 取出：123&nbsp; &nbsp

两个类型比较：char定长类型：速度快些

什么时候选char（定长），什么时候选（varchar）边长？？？？？？？？？？？？？

1. 从空间的利用效率考虑

如：四字成语表，适合用char(4)

个人简介，微博140字，适合用varchar(140)

2.从速度考虑

如：用户名：char

text：文本类型，可以存储比较大的文本段，收索速度稍慢

因此，如果不是特别大的内容，剪影用char，varchar来代替

text不用加默认值（加了也没用）

3.时间日期内向



Timestamp YYYYMMDDHHMMSS 时间戳类型（每更新一次记录一次）

首先日期分为

年——>year

年—月—日—>date

时—分—秒—>time

年—月—日——时—分—秒—> datetime

1.year(年)类型

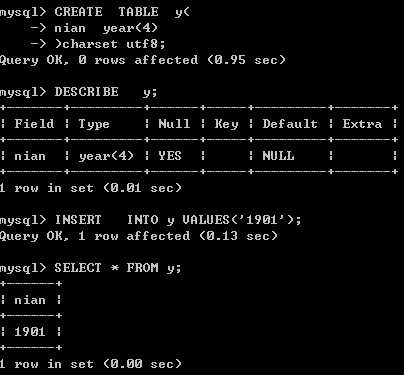
占1个字节，表示 1901年—2155年之间，（其中有个0000年，表示输入一个不合法年份时转换成0000年）

如果年份输入2位：

“00—69”表示2000年—2069年，计算机的元年从1970开始

“70—99”表示1970年—1999年

例子：

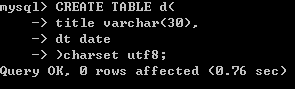


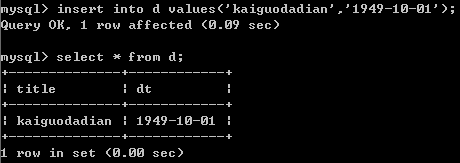
1. 日期类型（年-月-日）data

data类型的典型 1992-08-12

data类型的存储范围：公元1000-01-01—>9999-12-31

例子：插入日期类型数据

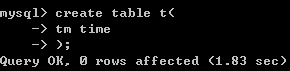


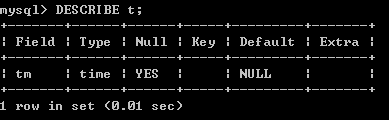


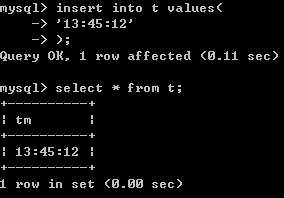
3.时间(time)类型：典型的格式 时—分—秒(hh-mm-ss)

日期类型范围：-838:59:59—>838:59:59

例子：







4.年—月—日 时—分—秒 (datetime) 类型 ,典型格式:1998-05-03 14：32：24

日期类型范围：公元 1000—01—01 00:00:00—>9999—12—31 23:59:59

日期类型可以设置默认值

例子：创建日期表

create table gjfuser(

name varchar(20) not null default ' ',

egetime datetime not null default ' 1000-01-01 00:00:00'

) charset utf8;



注意在实际开发中，很少用日期类型来表示一个需要精确到秒的时间，

因为：虽然日期时间类型能精确到秒，而且方便查看，

用时间戳来表示datetime类型的日期

用int类型存储时间戳，方便计算，对于显示来说，也可以方便格式化，

可以直接调用java，php里的格式化日期方法！

补充：枚举类型

//创建表

create table t2(

age int,

gender enum('男', '女')

)charest utf8;

enum('男', '女') //枚举类型就是把提前要输入的，先保存起来

//插入数据

insert into t2 values('男'); //成功

insert into t2 values('女 '); //成功

insert into t2 values('大神 '); //报错

做个小例子

姓名 ：char(4) //名字一般3到4个字符，用定长类型

年龄 ：tinyint unsigned //用整形无符号位

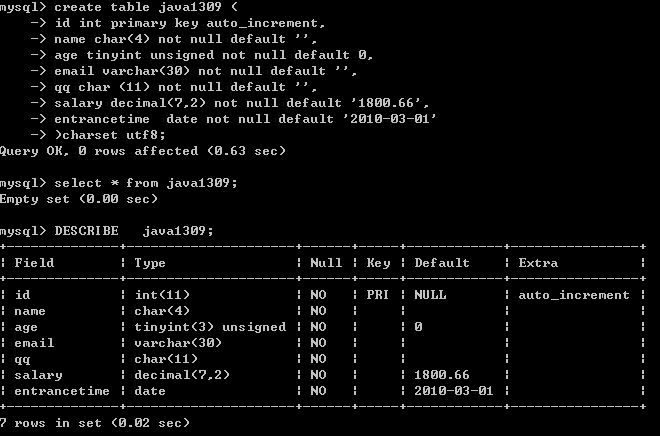
email : varchar(30) //邮箱最长不过30个字符，用变成类型

QQ: char(11) //QQ最长11位，用定长类型

intioro：varchar(500) //简介最长500字，用变成类型

salary：decimal(7,2) //工资最多10000.就用decimal类型

entrancetime : data //入学日期 2014-10-1



id int primary key auto\_increment,

id：表示属性值唯一

int：表示数据类型

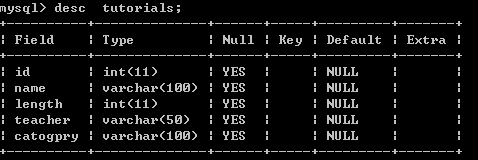
primary key：表示id是主键

auto\_increment：表示值自动增长

DESCRIBE 表的名称; 查看创建表后的结构

或者简写

DESC 表的名称; 查看创建表后的结构



Field:表示表有哪些属性。

Type:表示每个属性值的数据类型

Null：表示属性值是否可以为空，YES表示当前这一列可以不保存数据

Key：表示是不是表格的主键

Default：表示是否设置默认值，NULL表示没有设置默认值

Extra:

查看所有表

SHOW TABLES; 查看所有表

删除一个表

DROP TABLE 表名;

删除多个表

DROP TABLE 表1, 表2;

修改表

1. 重命名

语法：rename table 原表名 to 新表名;

语法：ALTER TABLE 原表名 RENAME 新表名;

例子：把tutorials 修改成tutorials123;

2.给表格添加一列（字段）

ALTER TABLE 表名 ADD 列名 列值的数据类型;

例子，添加link列

ALTER TABLE tutorials ADD link varchar(100);

ps:有符号，无符号的用法

2.1增加列，创建一列，保存的数据类型为 00001这样的

alter table myuser add length tinyint(5) zerofill;

可以使用after新增的列在哪一列后面

alter table myuser add length tinyint(5) zerofill after age;[ 让新增的列在age后面 ]

放在第一列

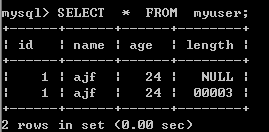
alter table myuser add length tinyint(5) zerofill first

zerofill:不足5位的位置补0

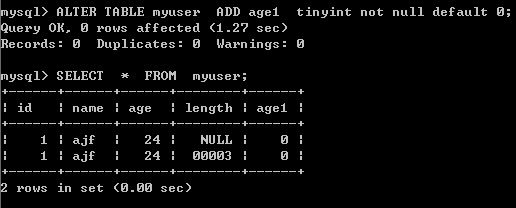
2.2添加一条数据记录

insert into myuser(id,name,age,length) values (1,'ajf',24,3);

2.3查看一下，length的属性值就变成了00003这样的



ps：给列设置默认值

alter table myuser add age1 tinyint not null default 0;

not null default 0;表示不能为空默认值为0

2.3显示所有列信息

show full columns from 表名;

3.删除表格一列

alter table 表名 drop column 列名;

例子，删除link列

alter table tutorials drop column link;

4.修改列的数据类型（数据类型要和值得类型一致）

alter table 表名 modify column 列名 新的列的数据类型;

例子

alter table tutorials modify column teacher varchar(100);

5.重命名一个列（数据类型要和值得类型一致）

alter table 数据表名 change 原列名 新列名 新列数据类型;

例子：

alter table tutorials change name myname int(20);

向表格中插入数据

insert into 表名称 values(值1,值2,…..); 给每个列都插入值

或者

insert into 表名称(列1,列2,……) values (值1,值2,……); 给指定的列插入值

在或者插入多列

insert into 表名

(列名,列名 ,列名)

values

(列值 1, 列值 1,列值 1),

(列值 2, 列值 2,列值 2),

(列值 3 ,列值 3, 列值 3);

例子

1.向表格中每列都插入数据

insert into tutorials values(1,'sjk',10,'aa','bb');

2.向指定的列插入数据

insert into tutorials(id,myname) values (2,'node');

3.同时插入多列数据

insert into java1309

(name,age,email)

values

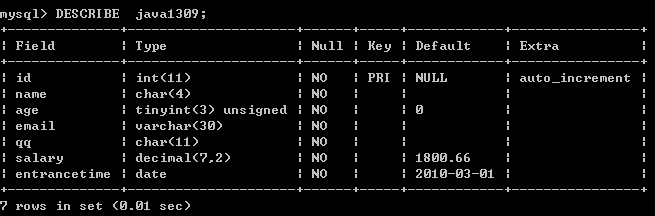
('张飞',90,'490771426@qq.com'),

('马超',80,'490771426@qq.com'),

('赵云',70,'490771426@qq.com');

查看表结构

describe java1309;





更改表中的列值

UPDATE 表名称

SET

要修改列的名称=要修改列的值 ,

要修改列的名称=要修改列的值

WHERE

要查询列的名称=要查询列的值;

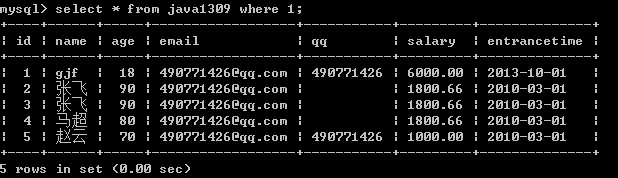
ps:（要查询列的名称=要查询列的值）值是唯一的



补充：select \* from1309 where 1;

在sql执行的时候，从上往下执行，因为条件为1，因此每向下执行

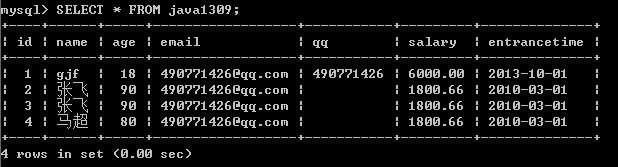
执行一次都返回true，所有就找到全部数据



删除一条记录

DELETE FROM 表名 WHERE 要删除的列名=要删除的列值;

例子：删除第5条数据



清空一张表,删除一张表中的所有信息

DELETE \* FROM 表名称;

约束（主键[pk] 外键[fk] 唯一[uk]）

例子1(主键):创建teacher表没有指定主键名，需要添加id为主键，不能重复

create table teacher(

id int unsigned not null auto\_increment,

name varchar(30),

age tinyint,

primary key(id)

);

添加主键约束(前提主键内不能有重复值)：

alter table teacher add constraint pk primary key(id);

或

alter table teacher add constraint primary key(id);

移除主键约束：

alter table teacher drop primary key;

例子2(外键):创建学生表使tid是老师表的主键。这样的键叫外键

create table student(

id int unsigned not null auto\_increment,

name varchar(30),

tid int unsigned,/\*建立外键约束，指向teacher表的id \*/

primary key(id)

)

create table teacher(

id int unsigned not null auto\_increment,

name varchar(30),

primary key(id)

)

--增加外键(指向另一张表的主键，或唯一约束键.两张表以上使用)

1.

alter table student add constraint fk foreign key (tid) references teacher(id);

给student 表添加约束外键，外键key是tid。引用teacher表id值

2.

alter table student add constraint fk foreign key (tid) references teacher(id) on delete set null;

给student 表添加约束外键，外键key是tid。引用teacher表id值,当引用被删除时自动补null。

3.

alter table student add constraint fk foreign key (tid) references teacher(id) on delete cascade;

给student 表添加约束外键，外键key是tid。引用teacher表id值,当引用被删除时自动删除关联表。

如 把老师表中的某条数据删除时，也会自动删除学生表中关联学生数据。也支持级联修改

4.

alter table student add constraint fk foreign key (tid) references teacher(id) on delete set null on update cascade;

支持级联删除，级联修改。

--删除外键

alter table student drop foreign key fk;

--删除索引

alter table student drop index fk;

执行

insert into teacher values(null,’李老师’), (null,’张老师’);

insert into student values(null,’李同学’,1), (null,’张同学’,2);

--取消关联

ps:学生表中的tid字段必须是老师表中的id字段。老师表不能删除，因为学生表使用了外键关联。

只能使用更新学生表的tid字段，然后在删除

update student set tid=3 where id in (1,2);

把学生表中id为1和2 行中tid字段更新设置为3或null，这样就没有关联了，就可以删了

--这样就可以删除老师了

delete from teacher where id=1

添加外键的另一种方式。先创建老师表，在创建学生表，设置老师表的id是学生表的外键。

create table teacher(

id int unsigned not null auto\_increment,

name varchar(30),

primary key(id)

)

create table student(

id int unsigned not null auto\_increment,

name varchar(30),

tid int unsigned,/\*建立外键约束，指向teacher表的id \*/

constraint fk foreign key (tid) references teacher(id) on delete set null on update cascade,

primary key(id)

)

红色文字为给学生表建立外键，引用老师表中的id键

索引：

本质相当于书籍的目录，作为加快数据检索速度。

-- SELECT \* FROM `user`;

-- 建立索引，desc:降序，asc:（默认）升序

-- CREATE INDEX agei ON `user`(age DESC);

-- 删除索引

-- ALTER TABLE `user` DROP INDEX agei;

-- 使用索引,因为排序了，所以速度很快。

-- SELECT \* FROM `user` WHERE age > 11;

-- 索引缺点：占用磁盘空间，降低了插入速度与修改速度。

ps: user：表示表名，agei：表示索引名称

查询数据

1.查寻整张表的数据

SELECT \* FROM 表名称;

例子

SELECT \* FROM tutorials;

2.查寻指定列，所有行 的数据

SELECT

列1,

列2

FROM

表名称;

SQL按条件查询语句

mysql支持的运算符

Mysql运算符主要分为四大类：

算术运算符，比较运算符，逻辑运算符，位操作运算符

1:算术运算符 + - \* / %

2:比较运算符： > , <,=,>=,<=,!=以及in，between and ,is null, greatest ,least,like,regexp等

Between：在某个范围内查询，like：搜索某种模式

3.逻辑运算符 逻辑运算符所得的结果均为1（true）0(false),这类运算符有逻辑非（not或者！），

逻辑与（and 或者&&），逻辑或（or或者||），逻辑异或（XOR）

4: 位操作运算符，参与运算的操作数按二进制位进行运算，包括位与（&），位或（|）位非（~）位异或（^）,左移（<<）,右移（>>）六种。

SELECT \* FROM 表名称 WHERE 列名 运算符 列值;

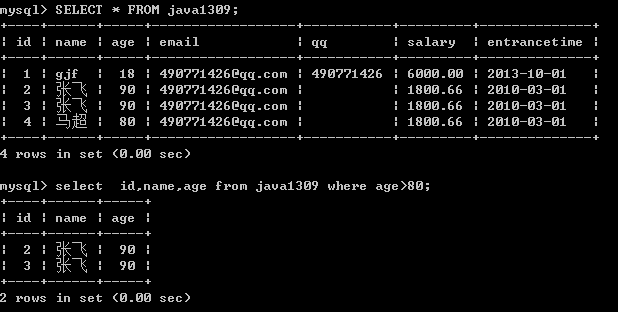
例1.查询整个表中，列名为myname，列值为node的所有条数据。

SELECT \* FROM tutorials WHERE myname='node'; （数据类型是字符串类型时一定是单引号）

例2.查询指定列中，age列大于80的数据

SELECT 列1, 列2, 列3 FROM 表名 WHERE 条件语句;

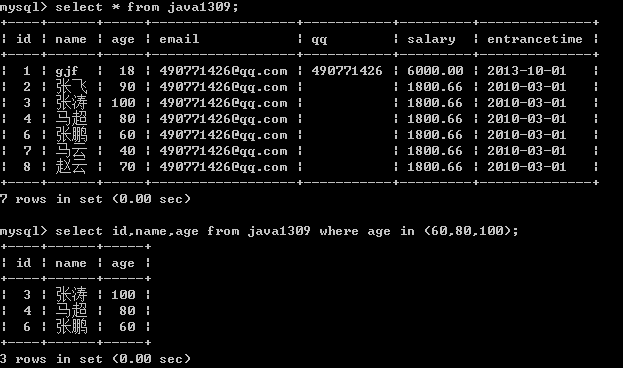
select id,name,age from java1309 where age>80;



例3：查询在多个值之一的数据，是用in查询

select id,name,age from java1309 where age in (60,80,100);

表示 查询年纪在60岁，80岁，100岁三个岁数之一的人

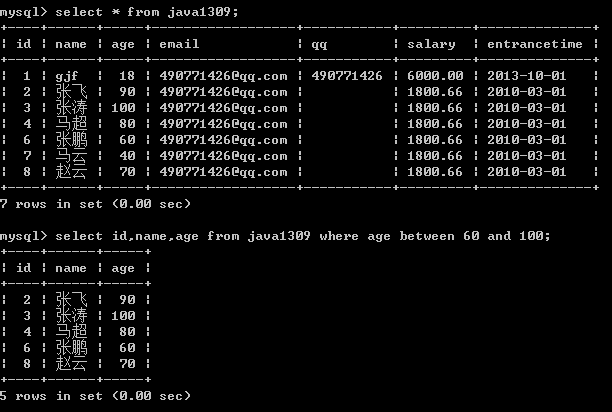


例4：查询什么范围到什么范围至间，使用between and语句，允许等于范围值

select 列名1, 列名2, 列名3… 表名 where 列名 between 范围值1 and 范围值2;

查询年龄在60岁致100岁范围之间的人，使用between and查询

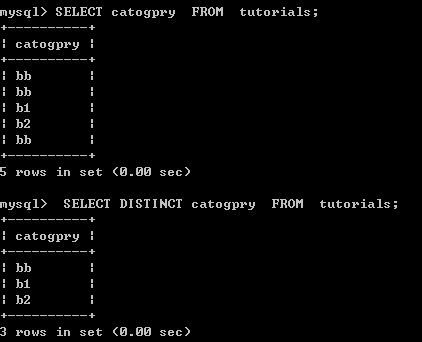
select id,name,age from java1309 where age between 60 and 100;



将查询结果唯一化（DISTINCT ）

SELECT DISTINCT 要查询的列名称 FROM 表名称;

例子

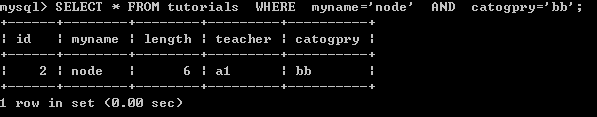


在WHERE条件中使用逻辑组合,返回符合条件的某些记录。

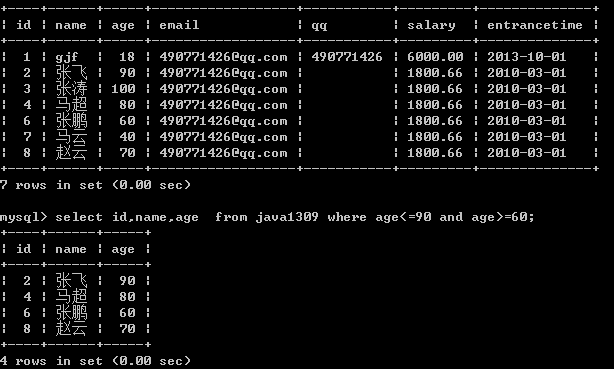
一．SELECT \* FROM 表名称 WHERE 条件1 AND 条件2; ps：AND是逻辑与

例子1 ：使用AND是逻辑与

SELECT \* FROM tutorials WHERE myname='node' AND catogpry='bb';



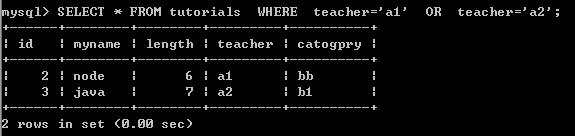
例子2 ：查询年龄在小于90岁与年龄大于60岁之间的人



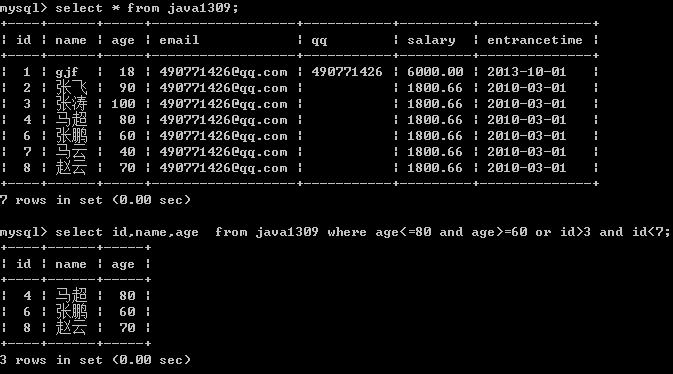
二．SELECT \* FROM 表名称 WHERE 条件1 OR 条件2; ps：OR是逻辑 或

例子1：使用OR是逻辑 或

SELECT \* FROM tutorials WHERE teacher='a1' OR teacher='a2';



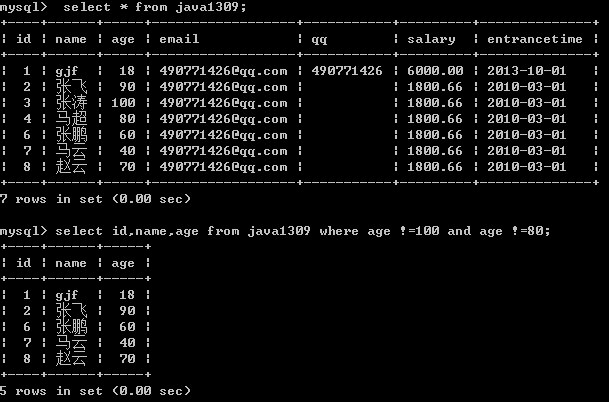
例子2：查询年龄在小于等于80岁与年龄大于等于60岁范围，或，id大于3与id<7的范围人



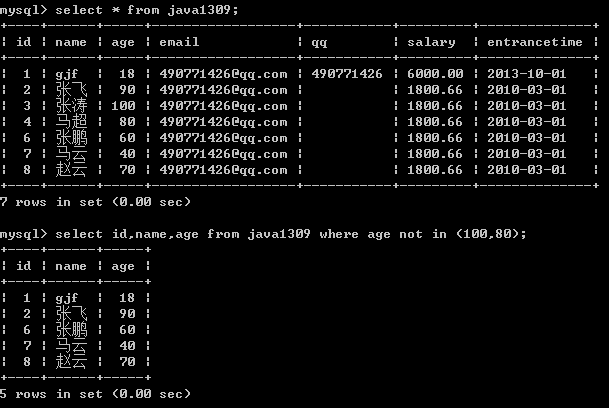
三.不等于 != 或 not in

例子1：查询年龄不等于100与年龄不等于80的人

select id,name,age from java1309 where age !=100 and age !=80;



例子2：查询年龄不等于100岁和年龄不等于80岁的人



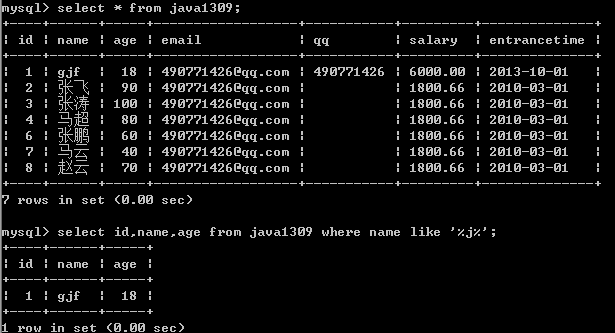
LIKE查询（模糊查询）

like：像

%：表示任意个或多个字符串

SELECT \* FROM 表名称 WHERE 列名称 LIKE '%要查询含有的字符 %';

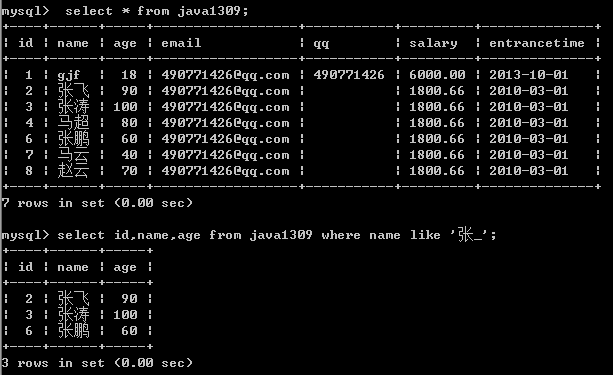
例子1.查询name中含有字符j的人



\_ :表示任意字符

SELECT \* FROM 表名称 WHERE 列名称 LIKE '\_中间的字符\_';

例2：把第一位为张开头的，第二位为任意字符的名字查询出来。



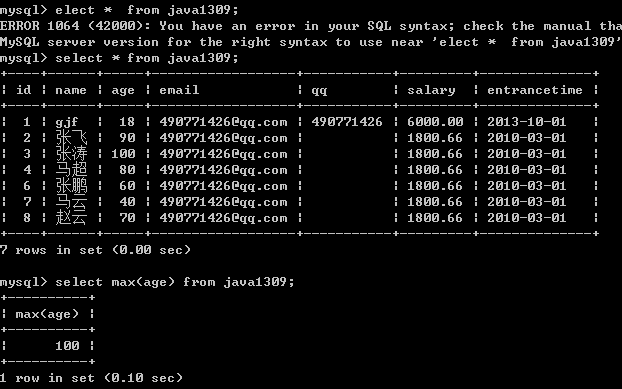
分组查询 group by

一般用于统计的场合

select max(列名) from 表名;

查询年龄最大的值

select max(age) from java1309;



查询每个栏目下最贵的商品

根据栏目的icat\_id分组查询,在配合聚合函数

select 列名1,max(要查询数据的列) from 表名 group by 按哪个查询的列名分组;

要查询数据的列：要当成变量来看，可以进行加减乘除运算，写出更强大的sql语句

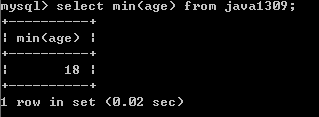


数据源



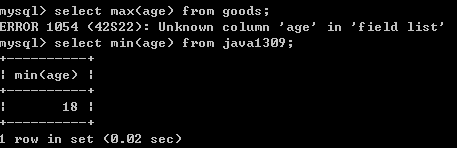
查询年龄最大的值。

select max(age) from java1309;

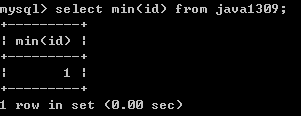


查询年龄最小的值

select min(age) from java1309;

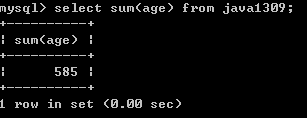


查询创建最早的数据

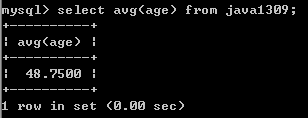


计算这么多人的年龄和

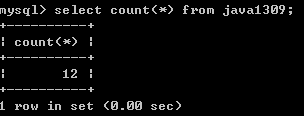
select sum(age) from java1309;



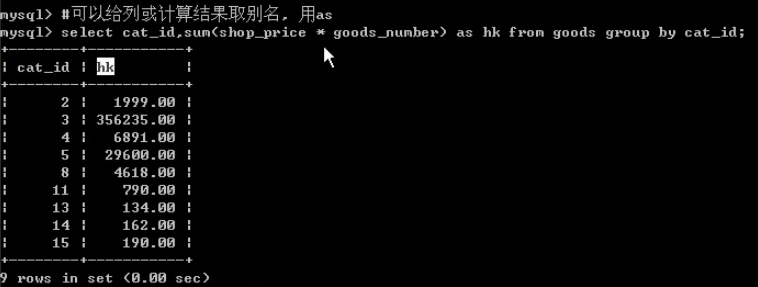
计算这么多人的平均年龄；



计算有多少个人，其实就是查看有多少行。



as：可以给列取别名



having筛选

把年龄小于50岁的查询出来，在在结果中筛选出qq号是490771426的人。

select name,age,qq from java1309 where age<50 having qq='490771426';

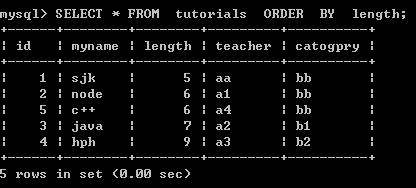


对查询结果按指定列进行排序（数字按从小到大排序，字符串按首字母在英文中的排序）

1.正序

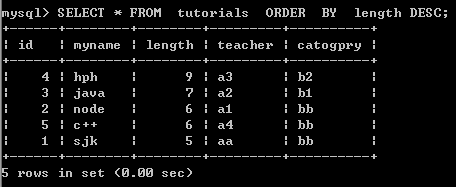
SELECT \* FROM 表名称 ORDER BY 列名称 ASC;

例子：将length列按正序（从小到大）排序



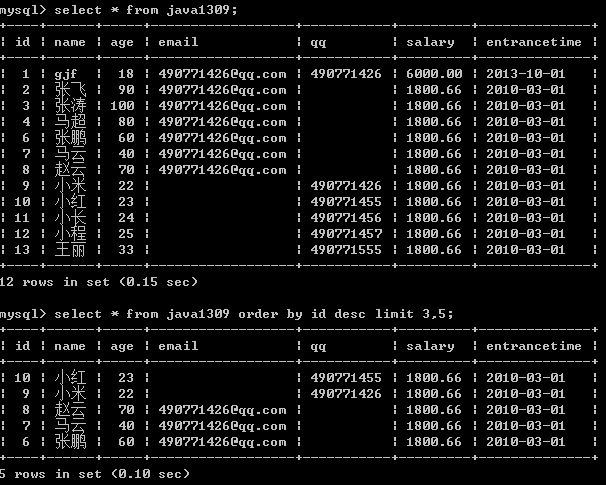
2.倒序 SELECT \* FROM 表名称 ORDER BY 列名称 DESC;

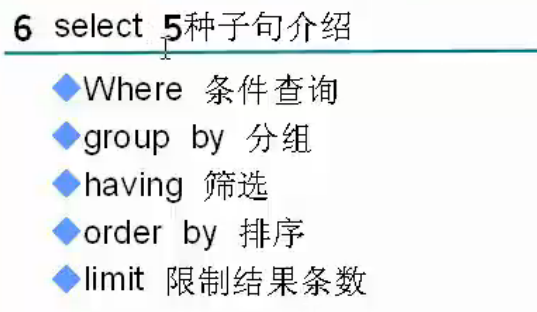
例子：将length列按倒序（从大倒序）排序



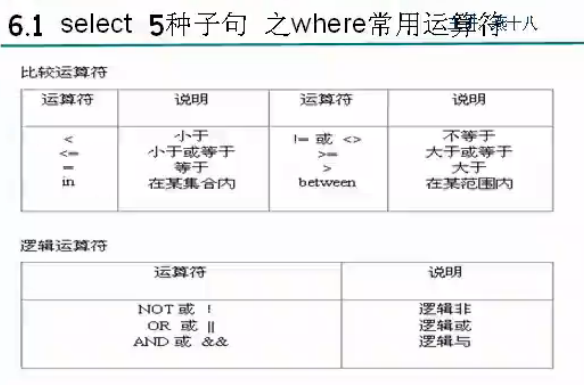
只取查询前几行limit[offset] N，offset表示偏移量，不写默认为0，N表示取出条数

将按id列的倒序排列，筛选出从第3行到往后的5行数据显示

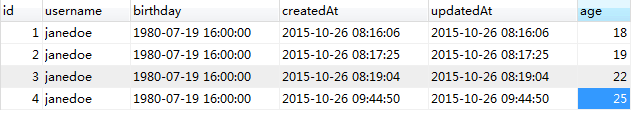




where主要在列上做筛选，跟查询条件表达式



数据表



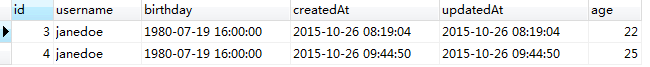
1.查询年纪大于22岁的

select id,username,birthday,createdAt,updatedAt,age from users where age>22



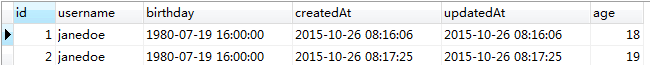
2.查询年纪大于等于22岁的

select id,username,birthday,createdAt,updatedAt,age from users where age>=22



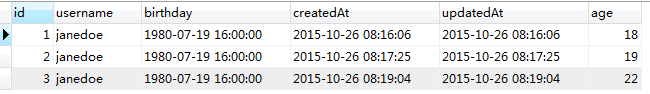
3. 查询年纪小于22岁的

select id,username,birthday,createdAt,updatedAt,age from users where age<22



4. 查询年纪小于等于22岁的

select id,username,birthday,createdAt,updatedAt,age from users where age<=22

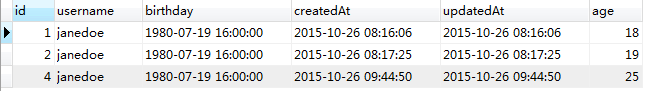


5. 查询年纪等于22岁的

select id,username,birthday,createdAt,updatedAt,age from users where age=22

6. 查询年纪不等于22岁的

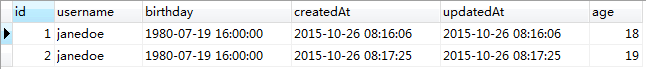
select id,username,birthday,createdAt,updatedAt,age from users where age!=22



7.in (值1,值2…值n),等于值1到值n任意一个都可以

查询年龄等于18,19之间之间的

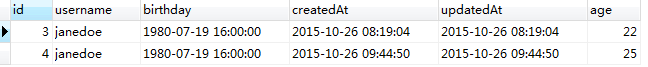
select id,username,birthday,createdAt,updatedAt,age from users where age in (18,19)



8.between 值1 and 值2。表示在值1和值2范围之间的

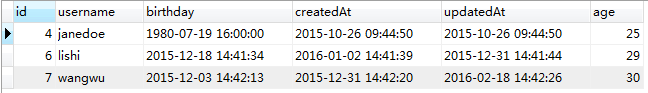
查询年龄在20到25之间的，包括20和25

select id,username,birthday,createdAt,updatedAt,age from users where age between 20 and 25;



9.查询年龄大于大于等于25和年龄小于等于30的

select id,username,birthday,createdAt,updatedAt,age from users where age >= 25 and age <=30 ;

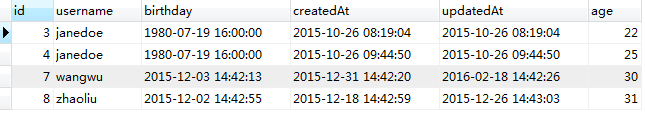


10.查询年龄在20到25之间和30到35之间的

select id,username,birthday,createdAt,updatedAt,age from users where age >= 20 and age <=25 or age>=30 and age<=35 ;

and：和

or:并且



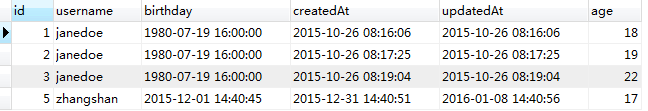
11.not的用法

查询年龄不等于25，30的

select id,username,birthday,createdAt,updatedAt,age from users where age not in (25,30);

in:是等于25,30的

not in：是不等于25,30的



模糊查询 like

Like 像

% 通配符匹配任意字符串

\_ 单个字符的占位符



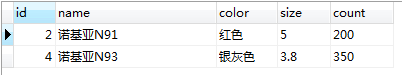
12.查询出带有”诺基亚”关键字的手机

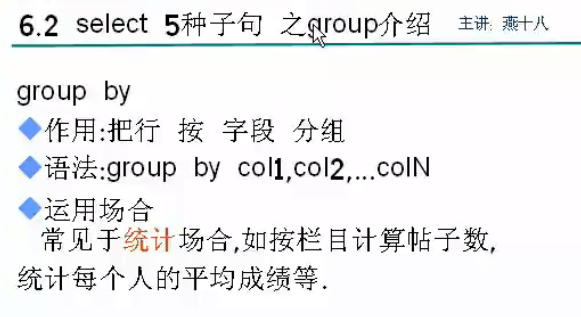
select id,name,color,size,count from goods where name like '诺基亚%';



13.只查询出诺基亚手机。如”诺基亚NXX”系列的手机

select id,name,color,size,count from goods where name like '诺基亚N\_\_';





1. 查询出价格最贵的的手机

select id,name,color,size,count,MAX(price) from goods;

MAX():查询最大值

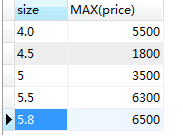


2查询每个size的手机中最贵的手机

select size,MAX(price) from goods group by size;

group by size:按size列分组，把每一组中最大的价格(price)查询出来。

一般配合聚合函数用



1. 查询最新的商品记录，也就是id值最大的

select MAX(id) from goods;

MAX():求最大



1. 查询最便宜的商品价格(price)

select MIN(price) from goods;



MIN():求最小

1. 查询出商品库存总和(count)

select sum(count) from goods;



Sum():求总和

1. 查询出本店商品的平均价格

select avg(count) from goods;



avg():求平均

1. 查询表有多少行

select count(\*) from goods;



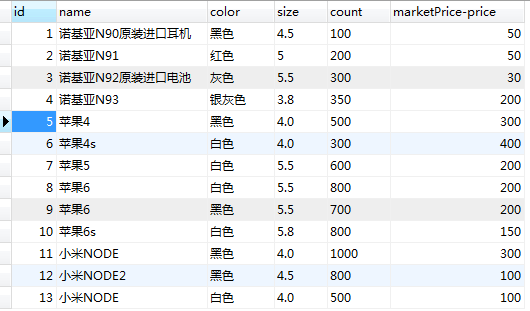
count()：求总行

1. 查询出价格(price)比市场价(marketPrice)便宜多少钱

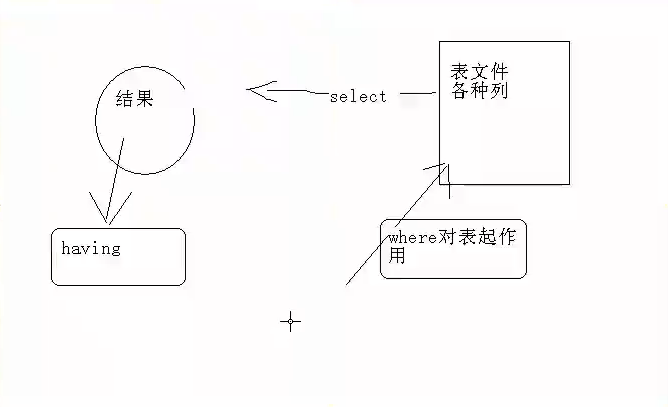
把select后面跟的参数理解成变量，就可以在参数中进行计算了

select id,name,color,size,count,marketPrice-price from goods;

marketPrice-price:在参数中计算变量，得到相差的价格

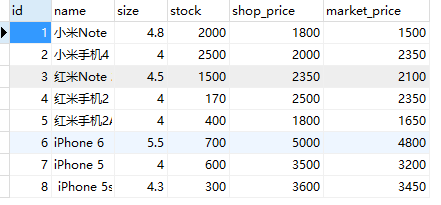


Having的用法



例子1：查询出本店价格比市场价格低多少钱，把低于大于200的价格查询出来。

原表

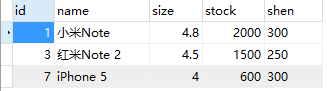


SELECT id,name,size,stock,shop\_price - market\_price as shen from goods having shen>200;

shop\_price - market\_price as shen：把shop\_price(市场价格) - market\_price(本店价格)的值赋值给shen这个变量

as：是把左边的赋值给右边

having shen>200：在把变量大于200筛选出来



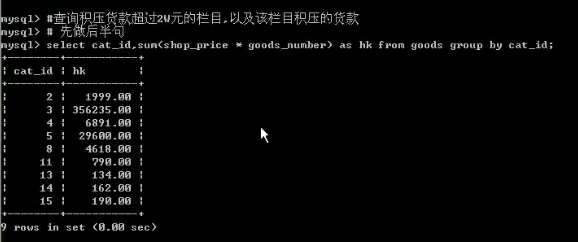
Ps：where:是用来筛选表上的数据的。having :是用来筛选查询结果保存在内存里的数据的，也就是对查询结果在次查询。

例2：查询出4寸屏中，比市场价便宜200以上的手机

SELECT id,name,size,stock,shop\_price - market\_price as shen from goods where size=4 having shen>200;

where和having 一起用





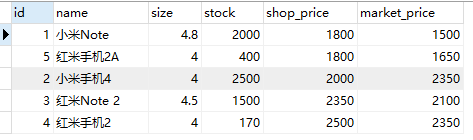
用到了求和sum()，分组group ，having 对查询结果在进行筛选

排序：order by

例子1：在商品表中，按市场价格从小倒大的顺序排序

SELECT id,name,size,stock,shop\_price , market\_price from goods order by shop\_price;

order by：排序



排序：根据字段排序，也可以按升序排序，也可以按降序排序。

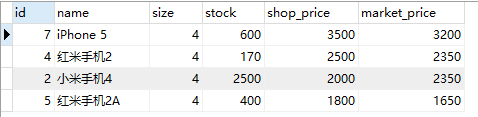
例子2：按屏幕大小排序，把屏幕是4.0的手机，按价格从大倒小排序

SELECT id,name,size,stock,shop\_price , market\_price from goods where size = 4 order by shop\_price desc;

order by：排序，默认正序(升序)排序，从小到大

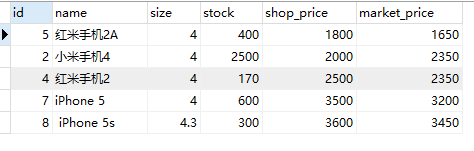
asc：明确指定正序(升序)排序，从小到大，可以不加

desc：倒序(降序)排序，从大倒小



例子3：先按屏幕大小排序，在按市场价格排序

SELECT id,name,size,stock,shop\_price , market\_price from goods order by size asc , shop\_price asc;



Ps：说明order by可以按多字段排序

limit：在语句的最后，起到限制条目的效果。分页就靠这个。

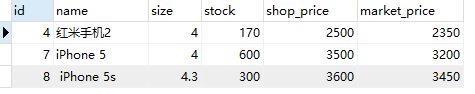
limit：[offsec],[n]

offsec:偏移量。如果该参数不写，就相当从0行开始，向后选取

N：取出条目

例子1：先按屏幕大小排序，在按市场价格排序。在从第2条开始，向后取出3条

SELECT id,name,size,stock,shop\_price , market\_price from goods order by size asc , shop\_price asc limit 2,3;



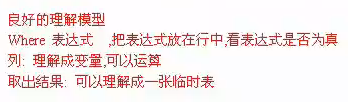
例子2，取市场价最贵的商品。

先排序，在选取

SELECT id,name,size,stock,shop\_price , market\_price from goods order by shop\_price desc limit 0,1;







子查询

where型子查询

把内层查询的结果当作外层查询的比较条件。

from型子查询

把内层的查询结果当成临时表，供外层的sql再次查询。

exists子查询

把外层的查询结果，拿到内层，看内层的查询结果是否成立。

例子1：在商品表中查询出最新的商品。也就是id值最大的记录。

思路：通过max()方法查询出最大的id，把id值保存到一张临时表中。在通过临时表中保存的id值查询出最新的商品记录。

select id,name,size,stock,shop\_price,market\_price from goods where id = ( select max(id) from goods );

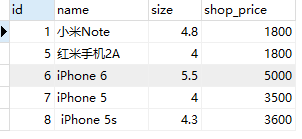


例子2：查询出每种屏幕中最贵的手机

思路：先按屏幕分组查询，把查询结果保存到临时表中。在使用in关键字查询出记录

in：查询在多个值之一的数据，是用in查询

select id,name,size,shop\_price from goods where shop\_price in ( select max(shop\_price) from goods group by size );



例3：用from型的子查询查询出每种屏幕下的最贵的产品

select \* from (select id,name,size,shop\_price,market\_price from goods order by size,shop\_price desc) as lsb group by size;

思路：先按屏幕由小到大排序，在按市场价格由大到小排序，把结果放在临时表中。

select id,name,size,shop\_price,market\_price from goods order by size,shop\_price desc



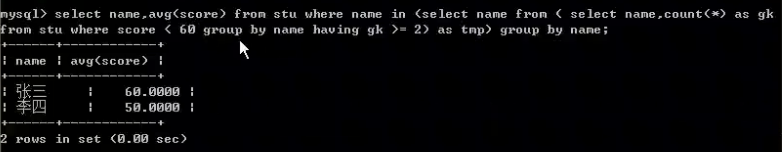
在对临时表进行按size分组，正好得到的是按size分组后得第一条，就得到了结果表。

select \* from (select id,name,size,shop\_price,market\_price from goods order by size,shop\_price desc) as lsb group by size;



例子4：如何利用子查询查询出挂科两门以及两门以上同学的平均分。where型和from型都可以。





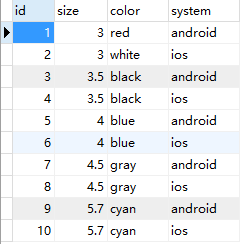
例子5：在商品表goods中，查询出有多少手机的屏幕大小（size），符合属性表goodsAttribute中的屏幕大小（size）。

这个例子很好的用到了exists 的查询

goods表



goodsAttribute表



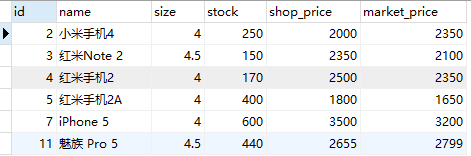
把外层的查询结果，拿到内层，看内层的查询结果与外层的查询结果是否成立。

select \* from goods where exists (select size from goodsAttribute where goods.size = goodsAttribute.size);

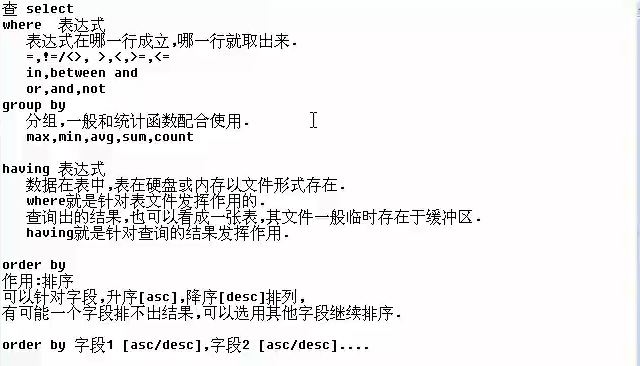
select \* from goods where exists：得到外查询的结果

select size from goodsAttribute：得到内查询的结果

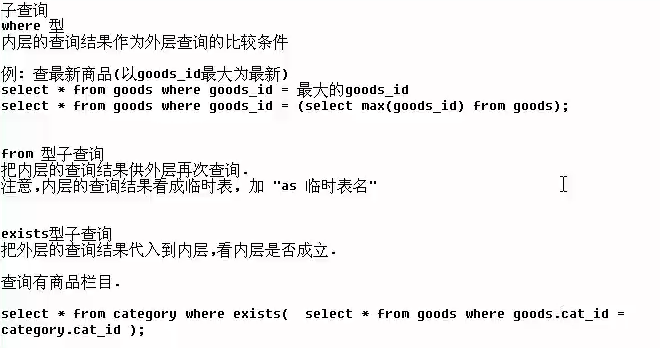
where goods.size = goodsAttribute.size：查看内层的查询结果与外层的查询结果是否成立，筛选出成立的外层结果



总结：







23． union(联合查询)

作用：把两次或多次查询结果合并起来

-- 查收出年龄大于45岁或小于35岁的老师。

select \* from teacher where age > 45 or age < 35;

-- 改用union联合查询实现,把两张表的结果合并成一张表.要求：

select id,name,age from teacher where age > 45 union select id,name,age from teacher where age < 35;

要求：两次查询的列数保持一致

推荐：查询的每一列，相对应的列类型保持一致

当多次sql语句取出的列明不一致时，此时以第一条sql语句为准。

查询扩展：

select ifnull(sgender,’保密’) 性别 from user;

查询user表，把性别查询为null的设置为保密，性别为列的别名

select \* from `user` where sex is null;

查询user表，把性别是null的查询出来。

lect \* from `user` where sex is not null;

查询user表，把性别不是null的查询出来。

select \* from `user` where age not between 14 and 18;

查询user表，把年龄不在14岁到18岁范围之间的取出来。（结果不包含14,18）

select \* from `user` order by rand() limit 2;

查询user表，随机的查询出两条。这条语句不建议使用，效率太低，使缓存失效。

rand():表示0到1之间的随机小数。

一条完整的查询语法

select \* from 表名 where 条件 group by 分组 having 分组条件 order by 排序 limit 限制行数;

having没有group by没有意义。

Select distinct name from user; 查询user表中，把重复消除掉，只保留一个。

多表查询

-- insert into teacher(name,age) values('李老师',33),('王老师',34),('赵老师',35),('麻老师',36);

-- insert into student(name,age,tid) values('李同学',33,3),('王同学',34,1),('赵同学',35,4),('麻同学',36,2);

select s.id,s.name studentName,s.age studentAge,t.name teacherName from student s,teacher t where s.tid = t.id;

s: 为student表的别名

t: 为teacher表的别名

查询出student表的id,name,age, teacher表的name字段。条件是student id等于teacher id。

链接查询

和数学中的集合特别向

集合的特性：无序性，唯一性

一张表就是一个集合，一行数据就是集合中的一个元素

左链接(结果以左表为准，去右表找匹配数据，找不到匹配数据用NULL补齐)

语法：

select 列1,列2 ... from

tableA left join tableB

on tableA.列 = tableB.列 [此处生成一张表，完全当成普通的表看]

where group,having.... [后面可以跟任何子句，照常写]

-- 查询学生表老师表，关联查询出等于左边学生id的老师

select s.id,s.name studentName,s.age studentAge,t.name teacherName from student s left join teacher t on s.tid = t.id;

student s left join teacher t: 连接表

s.tid = t.id: 筛选条件

最终得到一张表，后面可以跟各种where group having...子句。

右链接

语法：

select 列1,列2 ... from

tableA right join tableB

on tableA.列 = tableB.列 [此处生成一张表，完全当成普通的表看]

where group,having.... [后面可以跟任何子句，照常写]

总结：左右链接可以相互转换，推荐使用左链接

内链接（查询左右表都有的数据，即 不要左/右中null的那一部分，内链接是左右链接的交集）

语法：

select 列1,列2 ... from

tableA inner join tableB

on tableA.列 = tableB.列 [此处生成一张表，完全当成普通的表看]

where group,having.... [后面可以跟任何子句，照常写]

目前mysql不支持外链接，可以使用union(联合查询)实现

**视图**

视图是由查询结果查询结果形成的一张虚拟表

语法：

create view 视图名称 as 一条select语句;

例子：

create view newTeacher as select id,name,age from teacher where age > 15;

newTeacher：视图名称

select id,name,age from teacher where age > 15：普通的查询语句

备注：以后就可以使用这个虚拟表了

删除视图

drop view newteacher;

用处：

不想把原表的信息开放给开放人员看时！！

把表的权限封闭，开放响应的视图权限，视图里只放部分数据。

在大数据分表时可以用到。

当表的行数超过200万行时，就会变慢，可以把一部分表的数据拆成4张表来存放。

视图的修改

alter view as select .....;

等于是把之前的视图删了重新建立。

视图和表的关系

视图是表的查询结果，自然表的数据改变了，影响视图的结果。

视图改变了呢？

1. 视图增删改会影响到表
2. 但是，视图并不是总是能增删改的

视图的数据与表的数据一一对应时，可以修改

视图必须包含表中没有默认值的列。

视图的algorithm

algorithm = merge/temptable/undefined

merge:当引用视图时，引用视图的语句与定义视图的语句合并。（默认使用）

temptable：当引用视图时，根据视图的创建语句建立一张临时表。

undefined:未定义，自动，让系统帮你选。

merge意味着视图只是一个语句规则，当查询视图时，把查询视图的语句(比如where语句)与创建语句中where子句等合并，分析。形成一条select语句，查询原来的表，得到查询视图结果。

创建视图: create view newTeacher as select id,name,age from teacher where age > 15;

查询视图: select \* from newteacher group by age;

最终会把创建视图和查询视图语句合并一块,执行的是一句话。直接操作的是表。因此什么效果没有。

而temptable是根据创建语句瞬间创建的一张临时表，然后查询视图的语句从该零时表中查询出来。

解决以上问题，需要使用temptable方式。

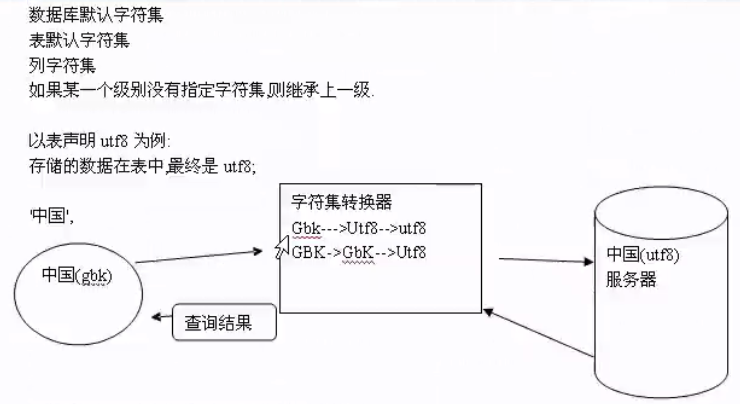
创建视图: create algorithm=temptable view newTeacher as select id,name,age from teacher where age > 15;

查询视图:select \* from newteacher group by age;

执行的是2句话：取数据并放在临时表，然后去查临时表。

**字符集与校对集**

字符集



1. 告诉服务器，我给你发送的数据是什么编码?

set character\_set\_client = utf8; //设置客户端的编码为utf8

1. 告诉转换器，转换成什么编码?

set character\_set\_connection = utf8; //设置转换器的编码为 utf8

1. 查询的结果用什么编码？

set character\_set\_results = utf8; //设置返回结果的编码为 utf8

如果以上3者都为字符集N，测可以简写

set names utf8; //一次性设置

推论：什么时将会乱码?

1.client声明与事实不符

2.results与客户端页面不符的时候

校对集

排序必须按某种规则排，这种规则就是 ”校对集“

指字符的排序规则

一种字符集可以有一个或多个排序规则。

以utf8为例，默认使用utf8\_general\_ci规则，也可以使用二进制来排序，utf\_bin

怎么声明校对集呢?

creat table () charset utf8 collate utf8\_general\_ci;

注意：声明的校对集必须是字符集合法的校对集

触发器

能监视：增删改

触发炒作：增删改

触发器创建语法4要素？

1.监视地点(table)

2.监视事件(insert/updata/delete)

3.触发时间(after/before)

4.触发事件(insert/updata/delete)

语法

create trigger triggerName

after/before insert/update/delete on 表名

for each row

begin

sql语句; (一句或多句，必须在insert/update/delete范围以内)

end;

//商品表

const create\_g = `

create table g(

id int unsigned not null auto\_increment,

name varchar(20) not null,

num int not null,

primary key(id)

)engine=InnoDB character set=utf8;

`;

await query(create\_g);

//订单表

const create\_o = `

create table o(

oid int unsigned not null auto\_increment,

gid int not null,

much int not null,

primary key(oid)

)engine=InnoDB character set=utf8;

`;

await query(create\_o);

//插入商品表

const insert\_g = `

insert into g(

name,

num

) values ('猪',20),('马',25),('牛',30),('羊',20);

`;

await query(insert\_g);

//买3只羊，插入一个订单

const insert\_o = `

insert into o(

gid,

much

) values (4,3);

`;

await query(insert\_o);

//减少羊的库存

const updated\_g = `

update g set num=num-3 where id=4;

`

await query(updated\_g);

//触发器使用的4个条件

/\*

监视地点：o表

监视操作：insert操作

触发操作：update操作

触发时间：after

\*/

const trigger = `

create trigger tg1

after insert on o

for each row

begin

update g set num = num - 3 where id = 2;

end$

`;

//deliniter $;//告诉服务器语句看见￥符号结束

存储引擎与事务

存储过程

备份与恢复