mysql复习
一:复习前的准备
1:确认你已安装wamp
2:确认你已安装ecshop,并且ecshop的数据库名为shop

二　　基础知识:
1.数据库的连接
mysql -u -p -h
-u 用户名
-p 密码
-h host主机

2:库级知识
2.1 显示数据库: show databases;
2.2 选择数据库: use dbname;
2.3 创建数据库: create database dbname charset utf8;
2.3 删除数据库: drop database dbname;

3: 表级操作:
3.1 显示库下面的表
show tables;

3.2 查看表的结构:
desc tableName;

3.3 查看表的创建过程:
show create table  tableName;

3.4 创建表:
  create table tbName (
列名称1  列类型  [列参数]  [not null default ],
....列2...
....
列名称N  列类型  [列参数]  [not null default ]
)engine myisam/innodb charset utf8/gbk


3.4的例子:
create table user (
    id int auto_increment,
    name varchar(20) not null default '',
    age tinyint unsigned not null default 0,
   index id (id)
   )engine=innodb charset=utf8;
注:innodb是表引擎,也可以是myisam或其他,但最常用的是myisam和innodb,
charset 常用的有utf8,gbk;


3.5 修改表
3.5.1　修改表之增加列:
alter table tbName
add 列名称1　列类型  [列参数]  [not null default ]  #(add之后的旧列名之后的语法和创建表时的列声明一样)

3.5.2　修改表之修改列
alter table tbName
change 旧列名　新列名　列类型  [列参数]  [not null default ]
(注:旧列名之后的语法和创建表时的列声明一样)

3.5.3　修改表之减少列:
alter table tbName
drop 列名称;


3.5.4　修改表之增加主键
alter table tbName add primary key(主键所在列名);
例:alter table goods add primary key(id)
该例是把主键建立在id列上

3.5.5　修改表之删除主键
alter table tbName  drop primary key;

3.5.6　修改表之增加索引
alter table tbName add [unique|fulltext] index 索引名(列名);

3.5.7　修改表之删除索引
alter table tbName drop index 索引名;

3.5.8　清空表的数据
truncate tableName;

4:列类型讲解
列类型:
        整型:tinyint (0~255/-128~127) smallint (0~65535/-32768~32767) mediumint int bigint (参考手册11.2)
        参数解释:
        unsigned 无符号(不能为负)  zerofill 0填充  M 填充后的宽度
        举例:tinyint unsigned;
             tinyint(6) zerofill;
数值型
        浮点型:float double
        格式:float(M,D)  unsigned\zerofill;


字符型
        char(m) 定长
        varchar(m)变长
        text

| 列 | 实存字符i | 实占空间 | 利用率 |
| --- | --- | --- | --- |
| char(M) | 0<=i<=M | M | i/m<=100% |
| varchar(M) | 0<=i<=M | i+1,2 | i/i+1/2<100% |

| | | |
| --- | --- | --- |
| 日期时间类型 | year | YYYY 范围:1901~2155. 可输入值2位和4位(如98,2012) |
| | date | YYYY-MM-DD 如:2010-03-14 |
| | time | HH:MM:SS 如:19:26:32 |
| | datetime | YYYY-MM-DD HH:MM:SS 如:2010-03-14 19:26:32 |
| | timestamp | YYYY-MM-DD HH:MM:SS 特性:不用赋值,该列会为自己赋当前的具体时间 |

5:增删改查基本操作

5.1 插入数据

```
insert into 表名(col1,col2,……) values(val1,val2……); -- 插入指定列
insert into 表名 values (,,,,); -- 插入所有列
insert into 表名 values -- 一次插入多行
(val1,val2……),
(val1,val2……),
(val1,val2……);
```

5.3修改数据

```
update tablename
set
col1=newval1,
col2=newval2,
...
...
colN=newvalN
where 条件;
```

5.4，删除数据 delete from tablenaeme where 条件;

5.5， select 查询

| | | | |
| --- | --- | --- | --- |
| （1） | 条件查询 | where | a. 条件表达式的意义，表达式为真，则该行取出<br>b. 比较运算符 = , !=, < > <= >=<br>c. like , not like ('%'匹配任意多个字符,'_'匹配任意单个字符)<br>in , not in , between and<br>d. is null , is not null |
| （2） | 分组 | group by | 一般要配合5个聚合函数使用:max,min,sum,avg,count |
| （3） | 筛选 | having | |
| （4） | 排序 | order by | |
| （5） | 限制 | limit | |

6: 连接查询

6.1， 左连接

```
.. left join .. on
table A left join table B on tableA.col1 = tableB.col2 ;
```

例句:

```
select 列名 from table A left join table B on tableA.col1 = tableB.col2
```

2. 右链接: right join
3. 内连接: inner join

左右连接都是以在左边的表的数据为准,沿着左表查右表.
内连接是以两张表都有的共同部分数据为准,也就是左右连接的数据之交集.

7 子查询

where 型子查询:内层sql的返回值在where后作为条件表达式的一部分
例句: select * from tableA where colA = (select colB from tableB where ...);

from 型子查询:内层sql查询结果,作为一张表,供外层的sql语句再次查询
例句:select * from (select * from ...) as tableName where ....

8: 字符集

客服端sql编码 character_set_client
服务器转化后的sql编码 character_set_connection
服务器返回给客户端的结果集编码 character_set_results
快速把以上3个变量设为相同值: set names 字符集

存储引擎 engine=1\2
1 Myisam 速度快 不支持事务 回滚
2 Innodb 速度慢 支持事务,回滚

①开启事务 start transaction
②运行sql;
③提交,同时生效\回滚 commit\rollback

触发器 trigger
监视地点:表
监视行为:增 删 改
触发时间:after\before
触发事件:增 删 改

创建触发器语法

```
create trigger tgName
after/before insert/delete/update
```

```
        on tableName
        for each row
        sql; -- 触发语句

  删除触发器:drop trigger tgName;
```

索引
提高查询速度,但是降低了增删改的速度,所以使用索引时,要综合考虑.
索引不是越多越好,一般我们在常出现于条件表达式中的列加索引.
值越分散的列，索引的效果越好

索引类型
primary key主键索引
index 普通索引
unique index 唯一性索引
fulltext index 全文索引

综合练习:
连接上数据库服务器
创建一个gbk编码的数据库
建立商品表和栏目表,字段如下:

商品表:goods
goods_id  --主键,
goods_name -- 商品名称
cat_id  -- 栏目id
brand_id -- 品牌id
goods_sn -- 货号
goods_number -- 库存量
shop_price  -- 价格
goods_desc  --商品详细描述

栏目表:category
cat_id --主键
cat_name -- 栏目名称
parent_id -- 栏目的父id

建表完成后,作以下操作:
删除goods表的goods_desc 字段,及货号字段
并增加字段:click_count  -- 点击量

在goods_name列上加唯一性索引
在shop_price列上加普通索引
在clcik_count列上加普通索引
删除click_count列上的索引

对goods表插入以下数据:

| goods_id | goods_name | cat_id | brand_id | goods_sn | goods_number | shop_price | click_count |
|---|---|---|---|---|---|---|---|
| 1 | KD876 | 4 | 8 | ECS000000 | 10 | 1388.00 | 7 |
| 4 | 诺基亚N85原装充电器 | 8 | 1 | ECS000004 | 17 | 58.00 | 0 |
| 3 | 诺基亚原装5800耳机 | 8 | 1 | ECS000002 | 24 | 68.00 | 3 |
| 5 | 索爱原装M2卡读卡器 | 11 | 7 | ECS000005 | 8 | 20.00 | 3 |
| 6 | 胜创KINGMAX内存卡 | 11 | 0 | ECS000006 | 15 | 42.00 | 0 |
| 7 | 诺基亚N85原装立体声耳机HS-82 | 8 | 1 | ECS000007 | 20 | 100.00 | 0 |
| 8 | 飞利浦9@9v | 3 | 4 | ECS000008 | 17 | 399.00 | 9 |
| 9 | 诺基亚E66 | 3 | 1 | ECS000009 | 13 | 2298.00 | 20 |
| 10 | 索爱C702c | 3 | 7 | ECS000010 | 7 | 1328.00 | 11 |
| 11 | 索爱C702c | 3 | 7 | ECS000011 | 1 | 1300.00 | 0 |
| 12 | 摩托罗拉A810 | 3 | 2 | ECS000012 | 8 | 983.00 | 14 |
| 13 | 诺基亚5320 XpressMusic | 3 | 1 | ECS000013 | 8 | 1311.00 | 13 |
| 14 | 诺基亚5800XM | 4 | 1 | ECS000014 | 4 | 2625.00 | 6 |
| 15 | 摩托罗拉A810 | 3 | 2 | ECS000015 | 3 | 788.00 | 8 |
| 16 | 恒基伟业G101 | 2 | 11 | ECS000016 | 0 | 823.33 | 3 |
| 17 | 夏新N7 | 3 | 5 | ECS000017 | 1 | 2300.00 | 2 |
| 18 | 夏新T5 | 4 | 5 | ECS000018 | 1 | 2878.00 | 0 |
| 19 | 三星SGH-F258 | 3 | 6 | ECS000019 | 0 | 858.00 | 7 |
| 20 | 三星BC01 | 3 | 6 | ECS000020 | 13 | 280.00 | 14 |
| 21 | 金立 A30 | 3 | 10 | ECS000021 | 40 | 2000.00 | 4 |
| 22 | 多普达Touch HD | 3 | 3 | ECS000022 | 0 | 5999.00 | 15 |
| 23 | 诺基亚N96 | 5 | 1 | ECS000023 | 8 | 3700.00 | 17 |
| 24 | P806 | 3 | 9 | ECS000024 | 148 | 2000.00 | 36 |
| 25 | 小灵通/固话50元充值卡 | 13 | 0 | ECS000025 | 2 | 48.00 | 0 |
| 26 | 小灵通/固话20元充值卡 | 13 | 0 | ECS000026 | 2 | 19.00 | 0 |
| 27 | 联通100元充值卡 | 15 | 0 | ECS000027 | 2 | 95.00 | 0 |
| 28 | 联通50元充值卡 | 15 | 0 | ECS000028 | 0 | 45.00 | 0 |
| 29 | 移动100元充值卡 | 14 | 0 | ECS000029 | 0 | 90.00 | 0 |
| 30 | 移动20元充值卡 | 14 | 0 | ECS000030 | 9 | 18.00 | 1 |
| 31 | 摩托罗拉E8 | 3 | 2 | ECS000031 | 1 | 1337.00 | 5 |
| 32 | 诺基亚N85 | 3 | 1 | ECS000032 | 1 | 3010.00 | 9 |

三　　查询知识
注:以下查询基于ecshop网站的商品表(ecs_goods)
在练习时可以只取部分列,方便查看.

1: 基础查询 where的练习:

查出满足以下条件的商品
1.1:主键为32的商品

```
select goods_id,goods_name,shop_price
     from ecs_goods
     where goods_id=32;
```
1.2:不属第3栏目的所有商品
```
select goods_id,cat_id,goods_name,shop_price  from ecs_goods
     where cat_id!=3;
```

1.3:本店价格高于3000元的商品

```
 select goods_id,cat_id,goods_name,shop_price  from ecs_goods
     where shop_price >3000;
```

1.4:本店价格低于或等于100元的商品
```
select goods_id,cat_id,goods_name,shop_price  from ecs_goods where shop_price <=100;
```

1.5:取出第4栏目或第11栏目的商品(不许用or)
```
select goods_id,cat_id,goods_name,shop_price  from ecs_goods
     where cat_id in (4,11);
```


1.6:取出100<=价格<=500的商品(不许用and)
```
select goods_id,cat_id,goods_name,shop_price  from ecs_goods
     where shop_price between 100 and 500;
```


1.7:取出不属于第3栏目且不属于第11栏目的商品(and,或not in分别实现)
```
select goods_id,cat_id,goods_name,shop_price  from ecs_goods     where cat_id!=3 and cat_id!=11;

select goods_id,cat_id,goods_name,shop_price  from ecs_goods     where cat_id not in (3,11);
```



1.8:取出价格大于100且小于300,或者大于4000且小于5000的商品()
```
select goods_id,cat_id,goods_name,shop_price  from ecs_goods where shop_price>100 and shop_price <300 or shop_price >4000 and shop_price <5000;
```



1.9:取出第3个栏目下面价格<1000或>3000,并且点击量>5的系列商品
```
select goods_id,cat_id,goods_name,shop_price,click_count from ecs_goods where
cat_id=3 and (shop_price <1000 or shop_price>3000) and click_count>5;
```

1.10:取出第1个栏目下面的商品(注意:1栏目下面没商品,但其子栏目下有)
```
select goods_id,cat_id,goods_name,shop_price,click_count from ecs_goods
     where cat_id in (2,3,4,5);
```

1.11:取出名字以"诺基亚"开头的商品
```
select goods_id,cat_id,goods_name,shop_price  from ecs_goods     where goods_name like '诺基亚%';
```


1.12:取出名字为"诺基亚Nxx"的手机
```
select goods_id,cat_id,goods_name,shop_price  from ecs_goods
   where goods_name like '诺基亚N__';
```


1.13:取出名字不以"诺基亚"开头的商品
```
select goods_id,cat_id,goods_name,shop_price from ecs_goos
     where goods_name not like '诺基亚%';
```

1.14:取出第3个栏目下面价格在1000到3000之间,并且点击量>5 "诺基亚"开头的系列商品
```
select goods_id,cat_id,goods_name,shop_price  from ecs_goods where
cat_id=3 and shop_price>1000 and shop_price <3000 and click_count>5 and goods_name like '诺基亚%';


select goods_id,cat_id,goods_name,shop_price  from ecs_goods where
shop_price between 1000 and 3000 and cat_id=3  and click_count>5 and goods_name like '诺基亚%';
```


一道面试题
有如下表和数组
把num值处于[20,29]之间,改为20
num值处于[30,39]之间的,改为30

mian表
```
+------+
| num  |
+------+
|    3 |
|   12 |
|   15 |
|   25 |
|   23 |
|   29 |
|   34 |
|   37 |
|   32 |
|   45 |
|   48 |
|   52 |
+------+
```

练习题:
把good表中商品名为'诺基亚xxxx'的商品,改为'HTCxxxx',
提示:大胆的把列看成变量,参与运算,甚至调用函数来处理 .
substring(),concat()


2　　分组查询group:
2.1:查出最贵的商品的价格

```
select max(shop_price) from ecs_goods;

2.2:查出最大(最新)的商品编号
select max(goods_id) from ecs_goods;

2.3:查出最便宜的商品的价格
select min(shop_price) from ecs_goods;

2.4:查出最旧(最小)的商品编号
select min(goods_id) from ecs_goods;

2.5:查询该店所有商品的库存总量
select sum(goods_number) from ecs_goods;

2.6:查询所有商品的平均价
 select avg(shop_price) from ecs_goods;

2.7:查询该店一共有多少种商品
 select count(*) from ecs_goods;


2.8:查询每个栏目下面
最贵商品价格
最低商品价格
商品平均价格
商品库存量
商品种类
提示:(5个聚合函数,sum,avg,max,min,count与group综合运用)
select cat_id,max(shop_price) from ecs_goods  group by cat_id;


3 having与group综合运用查询:
3.1:查询该店的商品比市场价所节省的价格
select goods_id,goods_name,market_price-shop_price as j
     from ecs_goods ;


3.2:查询每个商品所积压的货款(提示:库存*单价)
select goods_id,goods_name,goods_number*shop_price  from ecs_goods

3.3:查询该店积压的总货款
select sum(goods_number*shop_price) from ecs_goods;

3.4:查询该店每个栏目下面积压的货款.
select cat_id,sum(goods_number*shop_price) as k from ecs_goods group by cat_id;

3.5:查询比市场价省钱200元以上的商品及该商品所省的钱(where和having分别实现)
select goods_id,goods_name,market_price-shop_price  as k from ecs_goods
where market_price-shop_price >200;

select goods_id,goods_name,market_price-shop_price  as k from ecs_goods
having k >200;

3.6:查询积压货款超过2W元的栏目,以及该栏目积压的货款
select cat_id,sum(goods_number*shop_price) as k from ecs_goods group by cat_id
having k>20000

3.7:where-having-group综合练习题
有如下表及数据
+------+---------+-------+
| name | subject | score |
+------+---------+-------+
| 张三 | 数学    |    90 |
| 张三 | 语文    |    50 |
| 张三 | 地理    |    40 |
| 李四 | 语文    |    55 |
| 李四 | 政治    |    45 |
| 王五 | 政治    |    30 |
+------+---------+-------+

要求:查询出2门及2门以上不及格者的平均成绩

## 一种错误做法
mysql> select name,count(score<60) as k,avg(score) from stu group by name having k>=2;
+------+---+------------+
| name | k | avg(score) |
+------+---+------------+
| 张三 | 3 |    60.0000 |
| 李四 | 2 |    50.0000 |
+------+---+------------+
2 rows in set (0.00 sec)

mysql> select name,count(score<60) as k,avg(score) from stu group by name;
+------+---+------------+
| name | k | avg(score) |
+------+---+------------+
| 张三 | 3 |    60.0000 |
| 李四 | 2 |    50.0000 |
| 王五 | 1 |    30.0000 |
+------+---+------------+
3 rows in set (0.00 sec)

mysql> select name,count(score<60) as k,avg(score) from stu group by name having k>=2;
+------+---+------------+
| name | k | avg(score) |
+------+---+------------+
| 张三 | 3 |    60.0000 |
| 李四 | 2 |    50.0000 |
```

```
+------+---+------------+
2 rows in set (0.00 sec)

#加上赵六后错误暴露
mysql> insert into stu
    -> values
    -> ('赵六','A',100),
    -> ('赵六','B',99),
    -> ('赵六','C',98);
Query OK, 3 rows affected (0.05 sec)
Records: 3  Duplicates: 0  Warnings: 0

#错误显现
mysql> select name,count(score<60) as k,avg(score) from stu group by name having k>=2;
+------+---+------------+
| name | k | avg(score) |
+------+---+------------+
| 张三 | 3 |    60.0000 |
| 李四 | 2 |    50.0000 |
| 赵六 | 3 |    99.0000 |
+------+---+------------+
3 rows in set (0.00 sec)

#正确思路,先查看每个人的平均成绩
mysql> select name,avg(score) from stu group by name;
+------+------------+
| name | avg(score) |
+------+------------+
| 张三 |    60.0000 |
| 李四 |    50.0000 |
| 王五 |    30.0000 |
| 赵六 |    99.0000 |
+------+------------+
4 rows in set (0.00 sec)

mysql> # 看每个人挂科情况
mysql> select name,score < 60 from stu;
+------+------------+
| name | score < 60 |
+------+------------+
| 张三 |          0 |
| 张三 |          1 |
| 张三 |          1 |
| 李四 |          1 |
| 李四 |          1 |
| 王五 |          1 |
| 赵六 |          0 |
| 赵六 |          0 |
| 赵六 |          0 |
+------+------------+
9 rows in set (0.00 sec)

mysql> #计算每个人的挂科科目
mysql> select name,sum(score < 60) from stu group by name;
+------+-----------------+
| name | sum(score < 60) |
+------+-----------------+
| 张三 |               2 |
| 李四 |               2 |
| 王五 |               1 |
| 赵六 |               0 |
+------+-----------------+
4 rows in set (0.00 sec)

#同时计算每人的平均分
mysql> select name,sum(score < 60),avg(score) as pj from stu group by name;
+------+-----------------+---------+
| name | sum(score < 60) | pj      |
+------+-----------------+---------+
| 张三 |               2 | 60.0000 |
| 李四 |               2 | 50.0000 |
| 王五 |               1 | 30.0000 |
| 赵六 |               0 | 99.0000 |
+------+-----------------+---------+
4 rows in set (0.00 sec)

#利用having筛选挂科2门以上的.
mysql> select name,sum(score < 60) as gk ,avg(score) as pj from stu group by name having gk >=2;
+------+------+---------+
| name | gk   | pj      |
+------+------+---------+
| 张三 |    2 | 60.0000 |
| 李四 |    2 | 50.0000 |
+------+------+---------+
2 rows in set (0.00 sec)



4:      order by 与 limit查询
4.1:按价格由高到低排序
select goods_id,goods_name,shop_price from ecs_goods order by shop_price desc;

4.2:按发布时间由早到晚排序
select goods_id,goods_name,add_time from ecs_goods order by add_time;

4.3:接栏目由低到高排序,栏目内部按价格由高到低排序
select goods_id,cat_id,goods_name,shop_price from ecs_goods
     order by cat_id ,shop_price desc;
```

4.4:取出价格最高的前三名商品
```
select goods_id,goods_name,shop_price from ecs_goods order by shop_price desc limit 3;
```

4.5:取出点击量前三名到前5名的商品
```
select goods_id,goods_name,click_count from ecs_goods order by click_count desc limit 2,3;
```

5　　连接查询
5.1:取出所有商品的商品名,栏目名,价格
```
select goods_name,cat_name,shop_price from
ecs_goods left join ecs_category
on ecs_goods.cat_id=ecs_category.cat_id;
```

5.2:取出第4个栏目下的商品的商品名,栏目名,价格
```
select goods_name,cat_name,shop_price from
ecs_goods left join ecs_category
on ecs_goods.cat_id=ecs_category.cat_id
where ecs_goods.cat_id = 4;
```

5.3:取出第4个栏目下的商品的商品名,栏目名,与品牌名
```
select goods_name,cat_name,brand_name from
ecs_goods left join ecs_category
on ecs_goods.cat_id=ecs_category.cat_id
left join ecs_brand
on ecs_goods.brand_id=ecs_brand.brand_id
where ecs_goods.cat_id = 4;
```

5.4: 用友面试题

根据给出的表结构按要求写出SQL语句。
Match 赛程表

| 字段名称 | 字段类型 | 描述 |
|---|---|---|
| matchID | int | 主键 |
| hostTeamID | int | 主队的ID |
| guestTeamID | int | 客队的ID |
| matchResult | varchar(20) | 比赛结果，如（2:0） |
| matchTime | date | 比赛开始时间 |

Team 参赛队伍表

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

Match的hostTeamID与guestTeamID都与Team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 不来梅 2006-6-21

```
mysql> select * from m;
+-----+------+------+------+------------+
| mid | hid  | gid  | mres | matime     |
+-----+------+------+------+------------+
|   1 |    1 |    2 | 2:0  | 2006-05-21 |
|   2 |    2 |    3 | 1:2  | 2006-06-21 |
|   3 |    3 |    1 | 2:5  | 2006-06-25 |
|   4 |    2 |    1 | 3:2  | 2006-07-21 |
+-----+------+------+------+------------+
4 rows in set (0.00 sec)

mysql> select * from t;
+------+----------+
| tid  | tname    |
+------+----------+
|    1 | 国安     |
|    2 | 申花     |
|    3 | 传智联队 |
+------+----------+
3 rows in set (0.00 sec)

mysql> select hid,t1.tname as hname ,mres,gid,t2.tname as gname,matime
    -> from
    -> m left join t as t1
    -> on m.hid = t1.tid
    -> left join t as t2
    -> on m.gid = t2.tid;
+------+----------+------+------+----------+------------+
| hid  | hname    | mres | gid  | gname    | matime     |
+------+----------+------+------+----------+------------+
|    1 | 国安     | 2:0  |    2 | 申花     | 2006-05-21 |
|    2 | 申花     | 1:2  |    3 | 传智联队 | 2006-06-21 |
|    3 | 传智联队 | 2:5  |    1 | 国安     | 2006-06-25 |
|    2 | 申花     | 3:2  |    1 | 国安     | 2006-07-21 |
+------+----------+------+------+----------+------------+
4 rows in set (0.00 sec)
```

6　　union查询
6.1:把ecs_comment,ecs_feedback两个表中的数据,各取出4列,并把结果集union成一个结果集.

6.2:3期学员碰到的一道面试题
A表:
```
+------+------+
| id   | num  |
+------+------+
| a    |    5 |
| b    |   10 |
| c    |   15 |
| d    |   10 |
+------+------+
```

B表:
```
+------+------+
| id   | num  |
+------+------+
| b    |    5 |
| c    |   15 |
| d    |   20 |
| e    |   99 |
+------+------+
```

```
mysql> # 合并 ,注意all的作用
mysql> select * from ta
    -> union all
    -> select * from tb;
+------+------+
| id   | num  |
+------+------+
| a    |    5 |
| b    |   10 |
| c    |   15 |
| d    |   10 |
| b    |    5 |
| c    |   15 |
| d    |   20 |
| e    |   99 |
+------+------+
```

要求查询出以下效果:
```
+------+----------+
| id   | sum(num) |
+------+----------+
| a    |        5 |
| b    |       15 |
| c    |       30 |
| d    |       30 |
| e    |       99 |
+------+----------+
```

参考答案:
```
mysql> # sum,group求和
mysql> select id,sum(num) from (select * from ta union all select * from tb) as tmp group by id;
+------+----------+
| id   | sum(num) |
+------+----------+
| a    |        5 |
| b    |       15 |
| c    |       25 |
| d    |       30 |
| e    |       99 |
+------+----------+
5 rows in set (0.00 sec)
```

7: 子查询:
7.1:查询出最新一行商品(以商品编号最大为最新,用子查询实现)
```
select goods_id,goods_name from
     ecs_goods where goods_id =(select max(goods_id) from ecs_goods);
```

7.2:查询出编号为19的商品的栏目名称(用左连接查询和子查询分别)
7.3:用where型子查询把ecs_goods表中的每个栏目下面最新的商品取出来
```
select goods_id,goods_name,cat_id from ecs_goods where goods_id in (select max(goods_id) from ecs_goods group by cat_id);
```
7.4:用from型子查询把ecs_goods表中的每个栏目下面最新的商品取出来
```
select * from (select goods_id,cat_id,goods_name from ecs_goods order by goods_id desc) as t group by cat_id;
```

创建触发器:

```
 CREATE  trigger tg2
after insert on ord
for each row
update goods set goods_number=goods_number-new.num where id=new.gid

CREATE trigger tg3
after delete on ord
for each row
update goods set goods_number=good_number+old.num where id=old.gid


CREATE  trigger tg4
after update on ord
for each row
update goods set goods_number=goods_number+old.num-new.num where id=old.gid
```

**2012-03-25更新,添加了面试案例**