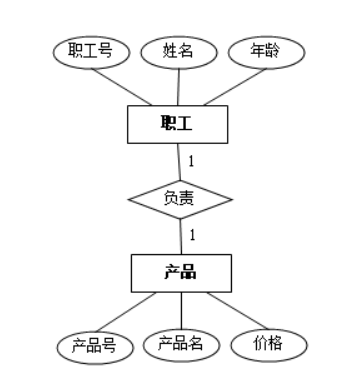
## 概念结构：ER图 逻辑结构：关系模式（二维表）

### 概念结构向逻辑结构转换

1. **若实体间的联系是1:1 可以在两个实体类型转换成的两个关系模式中任意一个关系模式中加入另一个关系模式的码和联系属性的类型**
2. **若实体间的联系是1:n 则在n端实体类型转换的关系模式中加入1端实体类型的码和练习类型的属性**
3. **若实体间的联系是n:m 则将联系类型也转换成关系模式，其属性为两段实体的码加上联系类型的属性，而码为两段实体码的组合**

**例一**

**方案一**

**职工（职工号，姓名，年龄，产品号）**

**产品（产品号，产品名，价格）**

**方案二**

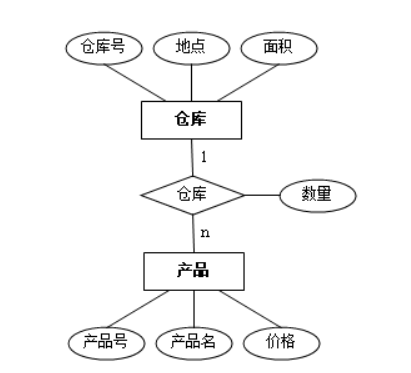
**职工（职工号，姓名，年龄）**

**产品（产品号，产品名，价格，职工号）**

**将上面两个方案进行比较 方案一中由于**

**并不是每个员工都负责产品，就会造成产品号**

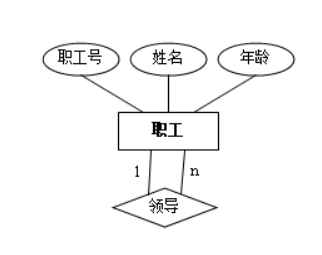
**属性的NULL值比较多，所以方案二较合理**



**例二**

**仓库（仓库号，地点，面积）**

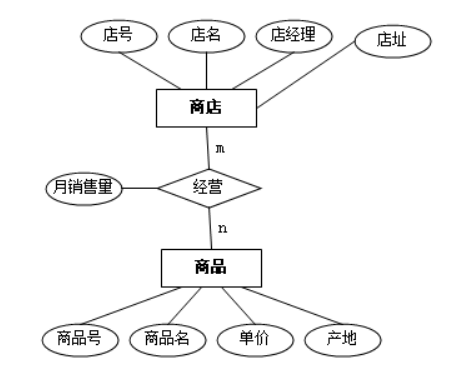
**产品（产品号，产品名，价格，仓库号，数量）**

**例三**

**职工（职工号，姓名，年龄，领导工号）**

**由于不能有同名字段故将领导的‘职工号’改为**

**‘领导工号’**

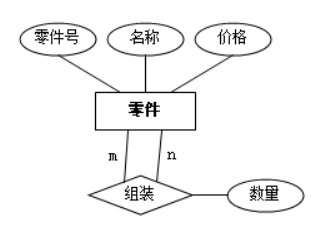


**例四**

**商店（电号，店名，店址，店经理）**

**商品（商品号，商品名，单价，产地）**

**经营（店号，商品号，月销售量）**



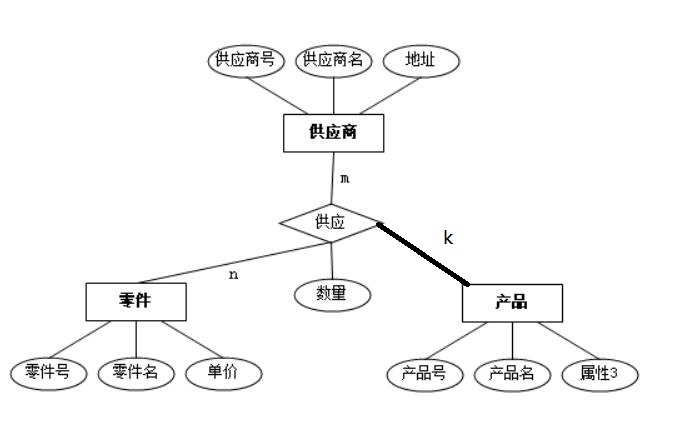
**例五**

**零件（零件号，名称，价格）**

**组装（组装件号，零件号，数量）**

**‘组装件号’为组装后的复杂零件号，**

**由于不能有同名字段故改为‘组装件号’**

****

**例六**

**供应商（供应商号，供应商名，地址）**

**零件（零件号，零件名，单价）**

**产品（产品号，产品名，型号）**

**供应（供应商号，零件号，产品号，数量）**

## 复制已有表结构（只复制结构，不复制内容）：

create table 新表明 like 表名

## 显示表结构：

1.DESCRIBE student;

2.DESC student;

3.SHOW COLUMNS FROM student

## 查看表创建时的语句：

SHOW CREATE table student; 与 SHOW CREATE table student\g

字段在上排横着， 下面跟对应数据

SHOW CREATE table student\G

字段在左侧竖着，数据在右侧横着

## 设置表属性（选项）：

基本语法： alter table 表名 表选项 值

注意：不要轻易修改表选项

修改表名：rename table 旧表名 to 新表名；

新增字段：alter table 表名 add 新字段名 字段类型【first/after字段名】

修改字段名：alter table 表名 change 旧字段名 新字段名 字段类型 【位置】

修改字段类型 alter table 表名 modify 字段名 新类型 【新属性】【新位置】

删除字段：alter table 表名 drop 字段名；

删除表结构 drop table 表名【,表名2...】可以同时删除多个数据表

## 插入操作：

insert into 表名 【（字段列表）】 values （对应字段列表）

eg：INSERT INTO st\_teacher (name,age) VALUES('Jack',30)

insert into 表名 values （）向表中所有字段插入数据；

## 查询操作：

查询全部数据： select \*from 表名：//\* 表示匹配所有字段

查询表中部分字段; select 字段列表 from 表名 // 字段列表用 ，隔开

简单条件查询数据：select 字段列表/\* from 表名 where 字段名 = 值

## 删除操作：

delete from 表名 where 条件

## 更新操作：

update 表名 set 字段名 = 新值 where条件

sets names 字符集

## 数据类型

无符号设定：（没有负数） 在类型之后加上 unsigned

显示长度（可以自己设定）tinyint（2）tinyint（3） 最长可显示3位

Zerofill 从左侧开始填充0 负数的时候不能使用zerofill 一旦使用zerofill 相当于确定该字段为unsigned

Decimal（M，D） 定点数 M总长度不超过65 D小数长度 不超过30 整数部分精确

如果涉及到钱可能用

时间日期类型：

Date 三个字节 YYYY-mm-dd 范围 ：1000-01-01——9999-12-12

Time 能够表示某个特定时间 三个字节 HH：ii：ss 范围很大 用来表示时间段

Datetime 八个字节 YYYY-mm-dd HH：ii：ss 范围 1000-01-01 00:00:00——9999-12-12 23:59:59

不能为空 默认值当前时间 但数据被更新时 这个字段自动更新为当前最新的时间

Timestamp 时间戳 只表示从格林威治时间开始 格式： YYYY-mm-dd HH：ii：ss

Year 一个字节 范围 1900——2155 插入方式 00——99/0000

两位数 69以下 为20+ 70以上 为19+

字符串：

Char：char（L）L范围：0——255 中英文一样

Varchar （L） 0——65535 要记录数据长度每个varchar数据产生后系统会增加1到2个字节 保存空间长度

char和varchar储存对比（utf8 一个字符都会占用三个字节）

Char（2） Varchar（2） Char所占字节 Varchar所占字节

A A 2\*3=6 1\*3+1 = 4

AB AB 2\*3=6 2\*3+1 = 7

当数据长度超过255个字符用text

Text：

Enum：(‘’,’’)

Set：存储到字段中的是数字

一个字节 8个选项

CREATE TABLE my\_set(

hoppy SET('篮球','足球','羽毛球','网球','乒乓球','冰球','排球','好球')

)charset utf8;

INSERT INTO my\_set VALUES('篮球,网球')

## 列属性

Null

Default 默认值

CREATE TABLE my\_default(

name varchar(10) NOT NULL,

age INT DEFAULT 18

)charset utf8

DESC my\_default

INSERT INTO my\_default （name ）VALUES（'Jeck'）

INSERT INTO my\_default VALUES('Tom',DEFAULT)

comment 注释说明

comment'字段描述'

用SHOW CREATE TABLE my\_comment 能查看

主键 （不允许为空）

primary key 值具有唯一性

primary key = （）

表后增加： alter table 表名 add primary key（字段）

查看主键

1 查看表结构 desc 表名

2 查看表的创建语句 show create table 表名

删除主键 alter table drop primary key

复合主键

主键约束 当前字段数据不能为空 且不能重复

主键分类

业务主键 具有业务意义

逻辑主键 自然曾长得整型

自动增长：

auto\_increment

通常用于逻辑主键

修改自增长 alter table my\_auto auto\_increment = 19

只能改大 不能该小

外键约束 表A外键字段取值 要么是NULL 要么是来自表B 主键字段的取值

Foreign key （表A字段名列表）references 表B（字段名列表）

On delete cascade/set null/no action /restrict

On update cascade/set null/no action /restrict

（cascade：父表记录的删除或修改操作会自动删除或修改子表中与之对应的记录

Set null：父表记录的删除或修改操作会将子表中与之对应的记录自动设置为NULL

No action：父表记录的删除或修改操作如果子表存在与之对应的记录，操作失败

Restrict：与no action 功能相同 且为默认值；

）

Alter table 表名add foreign key （表A字段名列表）references 表B（字段名列表）

字段列表一定要加（）

唯一键 unique 一张表中可以有多个 允许字段数据为Null 不为空的情况下不允许重复

创建唯一键：1在字段后加 unique 2 在所有字段后加 unique key(字段列表) alter table 表名 add unique key（字段列表）

检查约束 check

将 sage 设置为15以上检查约束

create table student(

sno char(8),

sage int check(sage>=15)

)

查看唯一键 1desc 表名

index 关键字 索引 唯一键是索引一种（提升查询效率）

删除唯一键 alter table 表名 drop index 唯一键名字

## ————高级数据操作————

多数据插入：insert into 表名 （字段列表）values （值列表）（值列表）....

INSERT into my\_auto values(151,'张三'),(121,'李四'),(131,'王五')

## 主键冲突（解决方案）：

1 主键冲突更新：

insert into st\_student values ('stu0004','小婷') on DUPLICATE key update st\_name = '小婷'

2主键冲突替换

Replace into values （值列表）

REPLACE into st\_student values ('stu0001','夏洛')

蠕虫复制

（注意主键冲突）： 一分为二 成倍增加 从已有的数据中获取数据 插入到新表中

inert into 表名 select \*/字段列表 from 表名

insert into my\_simple select name from my\_simple

通常是重复数据

可以再短期内快速增加数据量 测试表的数率

跟新数据

Update 表名 set 字段名 = 新值 where 判断条件

UPDATE st\_student set class\_id = 1 where st\_id = 'stu0001'

UPDATE st\_student set class\_id =2 WHERE st\_id in('stu0002','stu0003')

如果没有条件是全表更新 limit 限制数量

Update 表名 set 字段名 = 新值 limit数量

update my\_simple set name = 'q' limit 3

## 删除数据（高级）

可以用limit 限制数量

Delete from 表名

Delete 无法重置 auto\_increment（无法重1开始）

Truncate 表名： 重置自增长

查询数据

select select选项 字段列表 from 数据源 where条件 group by 分组 having 条件 order by 排序 limit限制

select选项 （系统如何对待查询结果）

1.All 默认 保存所有记录

2.Distinct 去重 去除重复数据 只保留一条 条件：（所有字段都相同）

字段列表

查询的时候有重名字段时 设置别名; 字段名 【as】别名

select DISTINCT name as name1,name name2 from my\_simple（同一个表可以拿多次）

### from 数据源

from为前面的查询提供数据

数据源只要求是一个符合二维表结构数据即可

单表数据 select DISTINCT name as name1,name name2 from my\_simple

多表数据

from 表1，表2.....

结果：两张表的记录数相乘 字段数拼接

动态数据：

from后面跟的不是一个实体表 而是结果表

from （select 字段列表 from 表）as 别名

select \* from (select int\_1,int\_5 from my\_int)as int\_my

where字句 获取数据时进行条件筛选

Group by 子句 根据指定的字段 将数据分组 目的是为了统计

group by 字段名

是为了分组后进行数据统计的 将数据按照指定字段分组后只保留每组第一条记录

利用一些统计函数

count（）统计没组数量 如果统计目标是字段 不统计空字段 count（\*）为统计记录个数

avg （）求平均值 （注意先分组）

sum（）求和

max（）求最大值

min（）求最小值

select class\_id,count(\*),max(st\_age),min(st\_hight),avg(st\_age) from st\_student group by class\_id

Group\_concat 为了将分组中指定的字段进行合并（字符串拼接）

select class\_id,GROUP\_CONCAT(st\_name),count(\*),max(st\_age),min(st\_hight),avg(st\_age) from st\_student group by class\_id

多分组

将分组的数据进行再分组

group by 字段1，字段2....

select class\_id,gender,count(\*),group\_concat(st\_name) from st\_student group by class\_id,gender

### 分组排序

默认升序

语法：group by 字段【asc|desc】，字段【asc|desc】 asc升序（默认） desc降序

select class\_id,gender,count(\*),group\_concat(st\_name) from st\_student group by class\_id,gender desc

回溯统计

语法： group by 【asc|desc】with rollup

Having 语句 进行数据条件筛选

having 在group by语句之后 可以针对分组数据进行统计筛选 where不行

where 不能使用聚合函数 聚合函数是用在group by分组的时候 where已经运行完毕

Having 在group by 分组之后 可以使用聚合函数或者字段别名（where是从表中取出数据，别名是在数据进入到内存之后才有的）

select class\_id,count(\*) as number from st\_student group by class\_id having number>=4

select class\_id,count(\*) as number from st\_student group by class\_id having count(\*)>=4

## Order by

根据校对规则对数据进行排序

order by 字段【asc|desc】 asc升序（默认）

班级学生按照升高排序 ：select \* from st\_student order by st\_hight

也可以进行多字段排序

order by 字段1【asc|desc】，字段2【asc|desc】 select \* from st\_student order by class\_id desc,st\_hight

## limit子句

1：记录数限制 限制获取数量 语法：limit 数量 select \* from st\_student limit 2

2：分页 利用limit来限制获取指定区间的限制 语法 limit offset，length offset：冲那开始 length最多 获取多少条

select \* from st\_student limit 2,2

## 运算符

### 1算数运算符

在myself 除法运算结果用浮点数表示 除数为零结果为null null 进行运算都为null

select int\_1-int\_2 from表名

### 2条件运算符 通常用来限定结果

= 进行相等判断 没有== 也可以用<=>

select \*from 表名 where 字段名 >=20

特殊应用

select '1' <=>1,0.02<=>0 结果 1 0

数据会转换成同数据 在比较 结果1为真 0位假

between 条件1 and 条件2

select \* from st\_student where st\_hight between 10 and 100

闭区间 条件1必须小于条件2

### 逻辑运算符

and or not

select \* from st\_student where st\_age>=20 and st\_age<=30

### in运算符

在什么里面 用来代替 = 当结果不是一个值 而是结果集的时候

语法 in（结果1，结,2....）

### is 运算符

专门用来判断 字段是否为null

语法： is null/is not null

select \* from 表名 where 字段名 is null

### like运算符

用来模糊匹配（匹配字符串）

语法 like 匹配模式

匹配模式中 有两种占位符

\_ 匹配对应的单个字符 select \* from st\_student where st\_name like '小\_' 小童

% 匹配多个字符 select \* from st\_student where st\_name like '小%' 小白兔

联合查询

（将多个查询结果合并到一起（纵向合并）字段数不变 多个查询的记录数合并）

应用场景

1将同一张表中不同结果（需要对应多条查询语句来实现）合并到一起展示数据

男生身高升序 女生身高降序 排序

(select \* from st\_student where gender = '女' order by st\_hight desc limit 10)

union

(select \* from st\_student where gender = '男' order by st\_hight asc limit 10)

2在数据量大的情况下，会对表进行分表操作 需要对每张表进行部分数据统计

使用联合查询将数据存放到一起显示

QQ1表获取在线数据

QQ2表获取在线数据 将所有在线数据显示出来

语法

select 语句

union 【union选项】

select 语句；

union选项 1 distinct 去重 （默认） 2 all

注意 ： union

select \* from st\_student

union （默认）

select \* from st\_student

select \* from st\_student

union all

select \* from st\_student

order by 的使用

1在联合查询中 如果要使用 order by 那么对应的select语句必须要使用（）

2 order在联合查询中 必须配合limit

(select \* from st\_student where gender = '女' order by st\_hight desc limit 10)

union

(select \* from st\_student where gender = '男' order by st\_hight asc limit 10)

limit 限字数 通常使用一个较大的数

## 连接查询

将多张表连在一起进行查询

意义 保证数据完整性

分类 交叉连接 内连接 外连接 自然连接

内连接： inner join 从一张表中取出所有记录去另一张表中匹配（必须保证匹配到才会保留数据）

从一张表中取出所有记录去另一张表中匹配

利用匹配条件进行匹配

匹配到 保留 继续向下

匹配失败 向下 全部失败 结束

语法：表1【inner】join 表2 on 匹配条件 select \* from st\_student inner join my\_class on class\_id = id

表的设计通常容易产生同名字段 通常使用（条件）表名.字段名

select \* from st\_student inner join my\_class on st\_student.class\_id = my\_class.id

如果条件中使用到对应的表名比较长 可以通过表的别名来简化

select \* from st\_student as s inner join my\_class as m on s.class\_id = m.id

当给表取别名时 前面查询时要用 别名.字段名

select sno,tscore.kno,kname,credit,score from tscore ts join tcourse tc on ts.kno = tc.kno X

select sno,tc.kno,kname,credit,score from tscore ts join tcourse tc on ts.kno = tc.kno √

外连接 outer join 按照某一张表作为主表（表中所有记录都会保留）根据条件去连接另外一张表 从而得到目标数据

left join 左表为主表 right join 右表为主表

左连接 主表 left join 从表 on 连接条件

右连接 从表 right join 主表 on 连接条件

select \* from st\_student as s right join my\_class as m on s.class\_id = m.id

特点：主表数据记录一定会保存

using 关键字

在连接查询中用来代替on

前提 两张表中的字段同名

语法： 表1 【inner，left，right】join using （同名字段列表）

**视图**

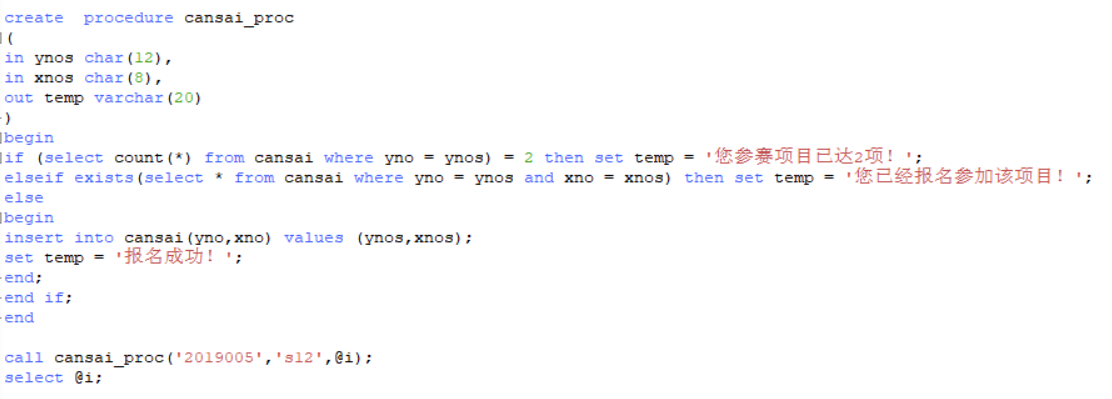
（有计算列一定要起别名）

create view view\_CJ (学号,课程号,成绩)

as select students.sno,cno,grade

from students,score where students.saddress = '上海' and students.sno = score.sno

**存储过程**



有多个end 或 end if 之后一定要加 ；

7.假设score表中是以前选修的课程并已考试取得成绩，现在有一个selcourses表用于保存本次选课的信息，向selcourses表中添加一行数据（一个学号，对应一个课程号）就表示该生选修该课程，  
请创建存储过程，实现选课功能的控制：  
输入：学号，课程号    -- 表示某学生选修某课程  
1）以前选修过的课程，本次不能再选；  
2）本次也不能重复选课；  
3）本次选课总数不能超过5门,否则，输出 你已经超过5门；  
4）不违反以上规则则保存该选课数据。

create table selcourses (  
   sno char(12),  
   cno char(4)  
)engine = InnoDB charset=utf8

create procedure cun\_procedure(in xh char(12),in ch char(4),out psh varchar(20))  
begin   
if exists(select \* from score where sno=xh and cno=ch) then set psh = '以前已选该课程';  
elseif exists(select \* from selcourses where sno=xh and cno=ch) then set psh = '已选该课程';  
elseif (select count(\*) from selcourses)>=5 then set psh = '你已超过5门';  
else insert into selcourses(sno,cno) values(xh,ch);  
end if;  
end;  
call cun\_procedure('2015001','c11',@i);  
select @i;

1. 创建存储过程。

功能：根据输入的类型输出相应的男生人数或女生人数。

存储过程名：CountNumBysex

输入参数：1 、2、其他数字

输出信息：如果输入type=1，输出男学生的人数，如果输入type=2，输出女学生的人数，如果输入其他的数字，输出Error Input!

create procedure CountNumBysex

(

in type int,

out mount varchar(20)

)

begin

if(type=1) then select count(\*) into mount from tstudent where ssex ='男';

elseif(type=2) then select count(\*) into mount from tstudent where ssex = '女';

else set mount = 'Error Input!';

end if;

end

call CountNumBysex(3,@i);

select @i;

**触发器**

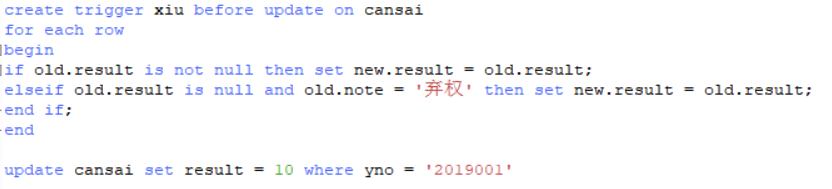
同一个表用before 不同表用after

在 before中set new.列名 = 值 √

在 after 中 set new.列名 = 值 ×

Update激活触发器语句中不能用update更改数值不然会一直触发触发器 可以用set代替

修改变量要加上set



2.给课程表添加一个列selcount，smallint类型，表示选修该课程的人数，然后创建“t\_insert”触发器，实现当成绩表中插入一条记录时，课程表中相应课程的选修人数自动加1.

alter table courses add selcount smallint;

create trigger t\_insert after insert on score  
for each row   
begin  
 if(select selcount from courses where cno = new.cno)is null then update courses set selcount = 0 where cno = new.cno;

 end if;  
 update courses set selcount = selcount + 1 where cno = new.cno;

end   
insert into score values('2015002','c04',80);

3. 创建“t\_update”触发器，实现对成绩修改的限制，要求成绩不能从及格改为不及格，否则继续存储原成绩；成绩也不能提高曾超过20%，  
   否则继续存储原成绩，如果没有违反以上规则，允许修改。  
create trigger t\_update before update on score   
for each row   
begin   
if (old.grade>=60 and new.grade<60) then set new.grade = old.grade;  
elseif (new.grade > old.grade\*1.2) then set new.grade = old.grade;

end if;

end  
update score set grade = 80 where sno = '2015001' and cno = 'c01';

4. 创建触发器t\_check,实现对课程学分取值的约束，限制学分的取值只能是1-20之间，当向课程表中添加记录时，  
如果学分取值小于1，则取1学分，如果学分大于20，则取20学分，以此保证学分均在1-20之间。

create trigger t\_check before insert on courses   
for each row   
begin  
if(new.credit<1) then set new.credit = 1;  
elseif (new.credit>20) then set new.credit = 20;  
end if;

end   
insert into courses values('c11','java',60,-9,'必修','3')

**高级操作**

### 添加用户

Create user ‘用户名’@’主机名’

Indentified by ‘密码’

create user 'xh1'@'localhost'

identified by '123456'

### 权限授予

Grant select

On 表名

To ‘用户名’@’主机名’

### 备份

Select \* from 表名 into outfile ‘stu.txt’