简单题

子句执行顺序(搬运

from select where group order

@MYSQL

零碎

与NOT IN等价的操作符是 <>ALL

SQL中一组具有相同权限的用户称为 角色

索引项的顺序与表中记录的物理顺序一致的索引，称之为 聚簇索引

DBS**中，DBMS和**OS之间的**关系是DBMS调用OS**

许多类型的数据依赖，最重要的是 **函数依赖 + 多值依赖**

SQl 语言是（非过程化)的语言，易学习

在SQL SERVER中声明局部变量时，变量名前面的字符为 @

在数据库中产生数据不一致的根本原因是 **数据冗余**

反映现实世界中实体及实体间联系的信息模型是 **E-R 模型**

下面列出的数据库管理技术发展的三个阶段中，没有专门的软件对数据进行管理的是 **人工管理阶段**

关联子查询通常会和EXISTS操作符一起使用，用来检查在子查询中是否存在满足条件的行

要保证数据库的数据独立性，需要修改的是[2分]
A 模式与外模式
B 模式与内模式
C 三级模式之间的两层映射
D 三层模式

答案 C

判断空值用“IS NULL”

单列排序、多列排序

- ```
  order by <列名i> [asc | desc]
  ```
- 默认 asc
- 多列排序：从左到右，左边相同右边才按order by指定顺序排序

HAVING 和 where语句的用途，HAVING作用于分组，对分组进行限制输出，Where作用于整个表，用来选择满足条件的行

在关系数据库设计中，设计关系模式是（逻辑设计阶段）的任务

如果对关系emp（eno，ename，salary）成功执行下面的SQL语句，其结果是 在emp表上按salary升序创建了一个聚簇索引
  ```

```
CREATE  CLUSTER  INDEX  name_index  ON  emp（salary）
```

主键一定是唯一性索引。但是一张表中可以有多个唯一索引，所以唯一索引不一定是主键

关于触发器叙述错误的是（ B ）。

(A) 触发器是不需要调用的，当触发事件发生时它就会被激活

(B) 触发器**不可以同步数据库的相关表进行级联更改**

(C) 触发器是一类特殊的存储过程

(D) 触发器能作用在数据库、服务器及表上。

在 SQL 语言查询语句中，SELECT 子句实现关系代数的 **投影运算**

在 SQL 语言查询语句中，WHERE 子句实现关系代数的 **选择运算**

定义一个存储过程查询学生某门课程的成绩和学分，存储过程的首部如下形式：

CREATE PROCEDURE student_info @name char(8),@kc_name char(16)。其中形参@name表示学生姓名，@kc_name表示课程名，则下列执行方式中，正确的是（ D ）。

(A) EXECUTE student_info('李明', '数据库原理')

(B) EXEC student_info '数据库原理','李明'

(C) EXEC student_info 李明,数据库原理

(D) EXECUTE student_info '李明', '数据库原理'

下列聚合函数中不忽略空值(null)的是 COUNT (*) 计算个数，包含空值个数

关系模式 R 中的属性全部是主属性，则 R 至少是 3NF

A 2NF

B 3NF

C BCNF

D 4NF

消除了部分函数依赖的 1NF 的关系模式，必定是 2NF

在对关系模式进行规范化的过程中，为得到一组 3NF 关系需对 2NF 关系进行投影，消除原关系中非主属性对关键字的
A 传递函数依赖

关系模式R中每个非平凡函数依赖X->Y，X必包含候选码，则R必定是 BCNF

关系模式 R 中的属性全部是主属性，则 R 至少是 3NF

有两个关系模式，分别为职工（职工号，职工名，部门号，职务，工资）、设备（设备号，职工号，设备名，数量），其中职工号、设备号分别是职工关系和设备关系的主键，且设备关系的职工号的取值来自职工关系的职工号，则两个关系的属性中，哪个是外键（ C ）。

（A）职工关系的职工号　　（B）职工关系的设备号

（C）设备关系的职工号　　（D）设备关系的设备号

从E-R图导出关系模型时，如果实体间的联系是M：N的，下列说法中正确的是（）。
A 将N方码和联系的属性纳入M方的属性中
B 将M方码和联系的属性纳入N方的属性中
C 增加一个关系表示联系，其中纳入M方和N方的码
D 在M方属性和N方属性中均增加一个表示级别的属性

答案 C

一个 1：n 只能与n端实体所对应的关系模式合并

在关系数据库设计中，对关系进行规范化处理，使关系达到一定的范式，例如达到 3NF，这是（）阶段的任务

A 需求分析阶段
B 概念设计阶段
C 物理设计阶段
D 逻辑设计阶段

答案 D

以下说法正确的是
A 游标中对应的查询只能是单表查询
B 游标可以逐行读取数据
C 当@@fetch_states值为0时，表示读取失败
D close mycursor用来完全释放游标所占用的内存

答案 B

如果要在员工工资表中限制工资的输入范围，应使用（ CHECK）约束

可以在不删除表的情况下删除所有的行。这意味着表的结构、属性和索引都是完整的：

```
DELETE FROM table_name
```

或者：

```
DELETE * FROM table_name
```

若属性A是关系R的主键（码），则属性A不能取空值（NULL），这是满足（ A ）。

（A）实体完整性规则（B）参照完整性规则（C）用户定义完整性规则（D）域完整性规则

SQL SERVER提供的单行注释语句是使用（ -- ）开始的一行内容，多行注释：/* *

下列关于关系数据模型的术语中，（ D ）术语所表达的概念与二维表中的“行”的概念最接近。

（A）属性　　（B）域　　（C）关系　　（D）元组

列值为空值（NULL），则说明这一列（ A ）

（A）数值是未知的 （B）数值为0 （C） 数值为空格 （D）不存在

在SQL SERVER中全局变量前面的字符为（ @@ ）

若有语句table_1 RIGHT JOIN table_2 ON 连接条件，这种连接会生成哪种结果集（ B ）。

（A）包括table_1中的所有行，不包括table_2的不匹配行

（B）包括table_2中的所有行，不包括table_1的不匹配行

（C）包括两个表的所有行

（D）只包括table_1和table_2满足条件的行

在DB应用中，一般一条SQL语句可产生或处理一组记录，而DB可通过（ B ）技术实现一次只能处理一条记录。

（A）指针 （B）游标 （C）数组 （D）栈

在表或视图上执行除了（ D ）以外的语句都可以激活触发器。

（A）INSERT （B）DELETE （C）UPDATE （D）SELECT

哪条SQL语句表示将计算列sal*12生成别名Annual Salary（ ）

（A）SELECT sal*12 'Annual Salary' FROM emp

日志文件用于存放恢复数据库中用的所有日志信息、每个数据库至少拥有一个日志文件，也可以拥有多个日志文件，扩展名为ldf。（ √ ）

数据库中只存放视图的定义。（ √ ）

使用索引可以加快查询语句的执行速度，使用视图可以简化查询语句的编写’。（ √ ）

SQL语言中，“>any”等价于“>max”。（ x ）（如果子查询返回的结果为空时，all 与max 的写法不等价,否则等价）

CREATE INDEX语句中，使用CLUSTERED来建立簇索引。（ x ）

正确示例 Cluster，没有过去式

```
CREATE CLUSTER INDEX name_index ON emp（salary）
```

关系R和S的元组个数分别为100和300，若关系T是R和S的笛卡尔积，则关系T的元组个数为400。（ x ）（100*300）

1. 打开SQL SERVER后，当前默认的数据库为 Master ，若需使student数据库成为当前数据库，可以使用语句 USE student
2. 如果两个实体之间具有M：N联系，则将它们转换为关系模型的结果是 3 个表。
3. 在 SQL SERVER的事务处理中，开启一个事务可用BEGIN TRANSACTION，提交事务和撤销事务应使用 COMMIT TRAN 和ROLLBACK TRAN 。
4. 游标处理步骤包括： 声明游标 、 打开游标 、 读取数据 、关闭游标、 删除游标
5. 指出下列缩写的含义：DBMS 数据库管理系统 、DBA 数据库管理员
6. SELECT语句进行查询，若希望查询的结果不出现重复元组，应在SELECT子句中使用 DISTINCT 保留字
7. 将数据库从SQL SERVER实例中删除，即在逻辑上将数据文件和日志文件与服务器相脱离，但文件并不从磁盘上删除，此操作称为 数据库分离，可通过 附加 将其重新加载到SQL SERVER实例中
8. “向EMP表增加一个Telephone列，其数据类型为长度为10的字符串”的SQL语句是：ALTER TABLE EMP ADD Telephone char(10)
9. SQL语言的数据操纵语句包括SELECT、INSERT、UPDATE和DELETE，最重要的也是使用最频繁的语句是 select
10. SQL SERVER的数据库文件分为 数据文件和 日志文件
11. 在SQL SERVER中，按触发事件不同，触发器可分为 DDL 触发器 和 DML 触发器
12. 为了使通过视图插入新行时，元组满足视图定义的条件，在定义视图时必须加上

> `WITH CHECK OPTION` 子句。

简答题

零碎概念

数据字典稍微看看

范式

最高能第几范式

函数依赖集 `F={S,D,D->M}`，显然满足第一范式的“属性唯一”的要求。

码：`(S,D)`

主属性：`S、D`，非主属性：`M`

存在非主属性对`M`对码 `(S,D)` 的函数部分依赖，所以，不满足第二范式的要求。

解法

[还算可以的教程]

根据题目语义画依赖->标识是最好的

第一范式好看，就是不可再分

第二范式是不存在非主属性对于码的**部分函数依赖(从码的一部分就能推出非主属性)**，对于只有一个码的情况就肯定满足了，如果是两个码的话**很可能不需要两个码就能完全确定一个非主属性**，因此达不到第二范式

3NF在2NF的基础之上，消除了非主属性对于码的传递函数依赖

> 对于学生表，主码为学号，主属性为学号，非主属性为姓名、系名和系主任。因为 学号 → 系名，同时 系名 → 系主任

至少要有三个属性才可能存在传递函数依赖关系

BCNF

存在着主属性对于码的部分函数依赖与传递函数依赖。（在此例中就是存在主属性【仓库名】对于码【（管理员，物品名）】的部分函数依赖。

解决办法就是要在 3NF 的基础上消除主属性对于码的部分与传递函数依赖。
仓库（仓库名，管理员）
库存（仓库名，物品名，数量）

总结：

1NF：字段是最小的的单元不可再分

2NF：满足1NF，表中的字段必须完全依赖于全部主键而非部分主键（一般我们都会做到）

3NF：满足2NF，非主键外的所有字段必须互不依赖

函数依赖

函数依赖中寻找候选键

对属性进行分类

L类：**只出现在→左边的属性，必为候选键之一**，若X+包含了R的所有属性，则X是唯一的候选键
R类：只出现在→右边的属性，则它是非主属性
N类：在→两边都没出现过的属性，它是主属性
LR类：在→左右两边都出现过的属性，且X+包含了R的所有属性，则X是唯一的候选键

例题

设有关系模式R（A,B,C,D,E），给定数据依赖如下：AB→C，A→D，D→E。

（1）列出R的码；主码或候选码都简称为码

A、B

(2) R为第几范式? 说明理由;

第一范式

∵各分量均不可再分，码A→非主属性D,而AB→D也能成立，存在非主属性对码的部分函数依赖

∴不能是2NF，只能是1NF

(3) 如果R不属于3NF，规范化R为3NF，并阐述规范化过程。

step①消除部分函数依赖：解决办法是用投影把关系模式R分解为两个关系模式：
R1（A，B，C）和 R2（A，D，E）
step②消除传递函数依赖： 解决办法是将R2分解为：AD（A，D）和 DE（D，E）
分解后的关系模式AD与DE中不再存在传递依赖

大题

零散大题

查询学号为95001学生选的课号,成绩

注意：条件SNO=95001　学号不要加引号
　　CNO=1　　课号不要加引号
　　SDEPT='IS' 系名加单引号(英文单引号)
　　SSEX='男'　性别，姓名，课名等同上

声明一个长度为16的字符型变量，变量名为cname，并赋初值为“数据库原理与应用”。请按要求写出相应的语句。

```
declare @cname char(16)
```

Set @cname='数据库原理与应用'或SELECT @cname='数据库原理与应用'

假设存在名为AAA的数据库，包括S (SNO CHAR(8), SN VARCHAR(8), AGE INT, DEPT VARCHAR(20), DATET DATETIME) 和SC (SNO CHAR(8), CN VARCHAR(10), GRADE NUMERIC(5,2)) 两张表。请设计一存储过程PROC3，利用存储过程可以修改SC表中学号为@s1、课程名为@c1的学生成绩，使其值为@g1。

```
CREATE PROC PROC3 @S1 CHAR(8), @c1 VARCHAR(10), @g1 NUMERIC(5,2)

AS

UPDATE SC

SET GRADE=@g1

WHERE SNO=@s1 AND CN=@c1
```

查询缺少成绩的学生的学号和相应的课程号。

```
   SELECT Sno, Cno FROM SC WHERE Grade ( ) NULL[2分]
```

A =

B !=

C IS

D IS NOT

答案 C

查询选2号课的学号,成绩

```
    SELECT Sno, Grade
    FROM SC
    WHERE Cno= ( )[2分]
```

A 2

B "2"

C '2'

D AC答案都对

E ABC答案都对

答案 D　数字选择 正常或者''都可以

查询95001的选课门数.

```
select      A
from     B
where     C   ;
```

答案

A count(*)

B sc

C sno=95001

采用嵌套方式查询“小林书局”这个经销商批发的图书名。

```
    SELECT BNAME FROM BOOK WHERE BNO IN
   (SELECT BNO  FROM  WHOLESAL  WHERE DNO IN
        (SELECT  DNO FROM DEALER WHERE DNAME= ‘小林书局’)
   )
```

写出SQL语句　查询共有多少个系

```
    A
    B   ;
```

答案

A select count(distinct sdept)

B from student

写出SQL语句　求各个系名及相应的学生人数 ？　注意每个子句一行

```
     A
     B
     C   ;
```

答案

A select sdept,count(*)

B from student

C group by sdept

写出SQL语句,查询学生人数>1000的系名，注意每个子句一行

```
    A
    B
    C
    D    ;
```

答案

A select sdept

B from student

C group by sdept

D having count(*)>1000

查询选1号课的姓名

```
 SELECT SNAME
 FROM STUDENT
 WHERE    A
    (      B
             C
              D
    );
```

答案

A SNO IN

B SELECT SNO

C FROM SC

D WHERE CNO=1

查询95001选修的课名(用EXISTS子查询实现)

```
SELECT     A
 FROM     B
 WHERE     C    (
     SELECT     D
     FROM     E
     WHERE CNO=    F
       AND  SNO=   G
 );
```

答案

A CNAME

B COURSE

C EXISTS

D *

E SC

F COURSE.CNO

G 95001

查询数据库原理课成绩>90的学号(用EXISTS子查询实现)

```
SELECT     A
FROM     B
WHERE       C       AND
        D    (
    SELECT     E
    FROM     F
    WHERE CNO=    G
        AND CNAME='数据库原理'
);    [每空2分]
```

答案

A SNO

B SC

C GRADE>90

D EXISTS

E *

F COURSE

G SC.CNO

查询选修了全部课程的学生姓名

```
SELECT  SNAME
FROM     A
WHERE     B    (
     SELECT     C
     FROM     D
     WHERE     E
     ( SELECT     F
       FROM     G
       WHERE    SC.SNO=   H
           AND  SC.CNO=   I
 ));
```

答案

A STUDENT

B NOT EXISTS

C *

D COURSE

E NOT EXISTS

F *

G SC

H STUDENT.SNO

I COURSE.CNO

查询被所有学生选修的课程名

```
SELECT  CNAME
FROM     A
WHERE     B    (
     SELECT     C
     FROM     D
     WHERE     E
     ( SELECT     F
       FROM     G
       WHERE    SC.SNO=   H
           AND  SC.CNO=   I
 ));
```

答案

A COURSE

B NOT EXISTS

C *

D STUDENT

E NOT EXISTS

F *

G SC

H STUDENT.SNO

I COURSE.CNO

将图书表中书号为1001的图书删除。

```
DELETE FROM BOOK WHREE BNO='1001'
```

将价格在20元以下的图书价格统一调整为21.80元。

```
UPDATE BOOK SET PRICE=21.8 WHERE PRICE<20
```

创建存储过程，输入书名，输出价格

创建：

```
CREATE PROC myproc @bname varchar(20),@price int output

As

SELECT @price=PRICE from BOOK where BNAME=@bname
```

调用：

```
DECLARE @price int

EXEC myproc '计算机应用',@price output
```

创建触发器，当表BOOK插入操作时，显示'trigger is working'。

```
create trigger mytrigger on mbook after insert

as

       declare @str char(50)

       set @str='trigger is working'

       print @str
```

SQL综合编程题

建表

```
create table 表名 (

列名 类型,

列名 类型,

primary key(列名, 列名...),  //表示主码

foreign key(列名) references 表名 (列名)  //表示外码

)
```

除了上面的primary key 和foreign key之外，还有三个可以之际跟在类型后面的约数条件:

not null //不允许为空.

unique(列名) //不允许重复.

cherk(条件) //通常用来指定输入数据的范围，例如: grade int cherk(grade >= 0 and grade <=100) 或者 sex char(4) cherk(sex = '男' or sex = '女')

删除表

drop table 表名 cascade
添加列

alter table 表名 add 列名 类型
删除列

alter table 表名 drop column 列名
修改列的类型

alter table 表名 alter column 列名 类型

对数据的操作

添加数据:

insert into 表名 values (,,,)(,,,)(,,,)
删除数据:

delete from 表名 where 条件
修改数据:

update 表名 set 变量名 = 数据值 where 条件
把where和后面的条件去掉就是修改所有人的某个属性

排序:

排序本质上是对查找到的数据通过比较某个或某几个属性进行排序，所以要用到selete

selete 列名1，列名2，列名3... form 表名 order by 列名2 //order by后只写列名2或者在列名2后加asc，表示将查找结果按照列名2进行升序排列

selete 列名1，列名2，列名3... form 表名 order by 列名2 desc //在列名2后加desc表示将查找结果按照列名2进行降序排列

selete * form 表名 order by 列名1，列名2 desc //将整个表按照列名1升序排列，如果列名1相同则按照列名2降序排列

查找数据:

查找名字中带有吴的人：selete 列名1 from 表名 where 列名1 = '%吴%' // %是匹配任意字符串

查找员工名和公司名(分别在员工表和公司表，两个表的共同列是公司编号)：selete 员工表.名字列，公司表.公司列 from 员工表，公司表 where 员工表.公司编号 = 公司表.公司编号 //更多的表和两个表一个道理

统计总人数大于2的班级，男生和女生分别有多少人，列出班级，性别，人数：

```
select 班级列, sum(case when 性别列 = '男' then 1 else 0 end) as 男生人数, sum(case wh
en 性别列 = '女' then 1 else 0 end) as 女生人数
from  班级表
group by 班级列
having (count(班级列) > 2);
```

视图的建立和表类似

```
create view 视图名 as 列名1, 列名2, 列名3...
form 表名
where 条件
with check option
```

like 模糊查询

①%：表示任意长度

例：查询所有姓刘的学生的姓名、学号和性别

```
select Sname,Sno,Ssex from Student where Sname like ‘刘%’;
```

②_：表示任意单个字符

例：查询姓’欧阳’且全名为三个汉字的姓名和学号

```
select Sname,Sno from Student where Sname like ‘欧阳_’;
```

③若想查的字符本身含有_或%,则在_前加入\（空格） ‘ESCAPE’ \ ' （换码字符）:作用是将\后面的_ 转为普通字符，不表示通配符含义

例：查询以“DB_”开头，且倒数第三个字符为i的课程的详细情况

```
select * from Course where Cname like ‘DB_%i_ _’ ESCAPE ’ \ ';
```

①is null（null：表示不确定）

例：某些学生选修课程后没有参加考试，所以有选修记录，但没有考试成绩。查询缺少成绩的学生的学号和相应的课程号。

```
select Sno,Cno from SC where Grade is null;
```

②is not null

查询所有有成绩的学生的学号和相应的课程号。

```
select Sno,Cno from SC where Grade is not null;
```

聚集函数

（1）count(*)：统计在一个关系中有多少个元组(数目)

例：查询学生总人数

```
select count( * )from Student
```

（2）count([distinct] A)：计算属性A有多少个值（加[distinct]表示有多少个不同的值）

例：查询选修了课程的学生数

```
select count (distinct Sno)  from Sc;
```
这里的distinct表示一个人选多门课程，但是只记录1次

（3）sum([distinct] A)：A的属性上所有元组加一起求和（必须为数值型）

例：查询学生201215012选修课程的总学分数

```
select sum(Credit) from Sc,Course where Sno='201215012' and Sc.Cno=Couse.Cno;
```

（4）avg([distinct] A)：对A的属性上所有元组求平均值（必须为数值型）

例：计算选修1号课程的学生平均成绩

```
select avg(grade) from Sc where Cno='1';
```

（5）max([distinct] A)：求属性A的最大值

例：查询选修1号课程的学生最高分数

```
select max(grade) from Sc where Cno='1';
```

（6）min([distinct] A)：求属性A的最小值

谓词子查询

(1)exists(存在)

(2)not exists(不存在)

例：查询没有选修1号课程的学生姓名

```
select sname from student where not exists(select * from sc where
sno=student.sno and cno='1');
```

select * from sc where sno=student.sno and cno='1'；是指查询选修了1号课程的信息，而题目要求查询不是选修1号课程的信息，所以取它的补集就是没有选修1号课程的信息

集合查询

（1）intersect(交)：满足两个条件

例：查询既选修了1号课程又选修了2号课程的学生。

既选1号课程又选2号课程select sno from sc where cno='1'; and cno=2;这是错误的，不能保证所查的属性既是1号课程又是2号课程，采用集合交，第1步找选修了1号课程select sno from sc where cno='1';第2步找选修了2号课程select sno from sc where cno='2';将两个条件交，度保证能够满足。

```
select sno from sc where cno='1' INTERSECT select sno from sc where cno='2';
```

（2）union（并）：满足一个条件

例：查询选修了1号课程或者选修了2号课程的学生。

或者；采用集合并，第1步找选修了1号课程select sno from sc where cno='1';第2步找选修了2号课程select sno from sc where cno='2';将两个条件并，满足其中1个即可。

```
select sno from sc where cno='1' UNION select sno from sc where cno='2';
```

（3）except（差）：

例：查询计算机科学系的学生与年龄不大于19岁的学生的差集。

直接将两个语句用EXCEPT连接

```
select * from student where sdept='CS' EXCEPT select * from student where sage<=19;
```

派生表

子查询不仅可以出现在where子句中，还可以出现在from子句中，这时子查询生成临时派生表成为主查询的查询对象

插入数据

（1）插入元组

insert into 表名（属性1，属性2…）values（'值1','值2'…）

值和属性要一 一对应

例：将一个新学生元组（学号：121200，姓名：陈东，性别：男，所在系：IS，年龄：18岁）插入student表中

```
insert into student(sno,sname,ssex,sdept,sage) values ('121200','陈冬', '男','IS',18);
```

若有属性，而没有对应的值要写NULL(赋空值)

(2)插入子查询结果

insert into 表名（属性1，属性2…）子查询

例：对每一个系，求学生的平均年龄，并把结果存入数据库

先建一个新表，一列存放系名，另一列存放学生平均年龄create table dept_age (sdept char(15) Avg_age SMALLINT);按系分组求平均年龄，把系名和平均年龄存入表中

```
insert into dept_age(sdept,Avg_age) select sdept,Avg(sage) from student group by sdept);
```

修改数据

（1）update 表名 set 属性（修改的值）where 条件

例：将学生201200的年龄改为22岁

```
update student set sage=22 where sno='201200';
```

(2) 带子查询的修改 IN

例：将计算机科学系全体学生成绩置0.

两个不同的表，先找在计算机科学系的学生学号，然后再将学号置为0

```
update sc set grade=0 where sno IN (select sno from student where sdept='CS');
```

删除数据

(1) delect from 表名 where 条件

例：删除学号为201200的学生记录

```
delete from student where sno='201200';
```

(2)带子查询的删除语句IN

例：删除计算机科学系所有学生的选课记录

两个不同的表，先找在计算机科学系的学生学号，然后再将选计算机科学的学号删除

```
delete from sc where sno IN (select sno from student where sdept='CS') ;
```

授权

```
grant 权限 (select)  on 对象类型(table,view) 对象名 to 用户 [with grant option]
//可省，加上表示可以将这个权限授予其他用户
```

例：通过角色来实现将一组权限授予一个用户

```
create role R1;
```

使用grant语句让R1拥有student表中的select、update、insert权限

```
grant select ,update,insert on table student to R1;
```

将这个角色授予王平

```
grant R1 to 王平;
```

(5) 对权限收回

```
revoke 权限 (如select) on 对象类型 (如table,view ) to 用户
```

```
revoke select on table student from R1;
```

零散

创建一个断言，限制每个宾馆单人间的房间数不能多于8个。（4分）

```
create assertion roomnum check(8>=all(select count(*) from Room where
type='单人间' group by hotelNo));
```

创建一个视图，包含宾馆号、宾馆名称、所在城市、房间号、房间类型和房间价格，并将该视图的查看权限授予用户U1。（4分）

```
create view h (hotelNo,hotelName,city,roomNo,type,price)
as
(select Hotel.hotelNo,hotelName,city,roomNo,type,price from Hotel,Room
where Hotel.hotelNo = Room.hotelNo);
```

```
grant select on h to U1;
```

建立触发器，当对表Room的price属性进行修改时，若价格增加了10%，则将此次操作记录到另一个表Room_U（roomNo，hotelNo，type(房间类型)，Oldprice，Newprice)中，其中Oldprice是修改前的价格，Newprice是修改后的价格。

```
create trigger Room_T
after update of price on Room
referencing
old row as oldTuple,
new row as newTuple
for each row
when(newTuple.price≥1.1*oldTuple.price)
insert into Room_U (roomNo, hotelNo, type, oldTuple.price, newTuple.price)
```

简单触发器

创建一个简单的触发器gzh_trigger，当执行插入语句之后激活触发器，如果超过某个自己设定的阙值，触发器被触发并给出相应的提示信息

```
delimiter $$
CREATE TRIGGER gzh_trigger #触发器的名称
AFTER INSERT #当执行插入语句之后激活触发器
ON gzh_readers #作用于那个表
FOR EACH ROW  #触发器作用在每条记录上
#以上都是固定的写法，可以直接套
# 触发器需要执行的操作
if NEW.gzh_rnum>50 then
SIGNAL SQLSTATE '45000' #错误状态信息
set message_text="你输入的数字过大，检查后输入";
end if $$
delimiter;


insert into gzh_readers values
(29,'刘小燕','女','42123456789','计算机应用','超时还书',105)
```