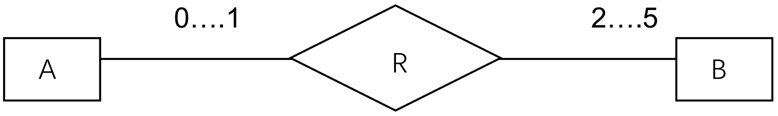
一、选择：

1. The following are functions of a DBMS except (A)

(A)creating and processing forms (B)creating databases

(C)processing data (D)administrating databases

2. For the E-R diagram given below, the mapping cardinality from **A** to **B** is (C)



(A) one-to-many (B) one-to-one (C) many-to-one (D) many-to-many

3. In a database system, which part is it to provide data consistency? (C)

(A) the DBA (B) the user (C) the DBMS (D) the application program

4. To authorize query privilege of table s to database user Wang, and permit Wang to authorize the privilege to others, The SQL sentence is (D) .

(A) GRANT SELECT TO S ON Wang WITH PUBLIC OPTION

(B) GRANT SELECT ON S TO Wang WIT H PUBLIC OPTION

(C) GRANT SELECT ON Wang TO S WITH GRANT OPTION

(D) GRANT SELECT ON S TO Wang WITH GRANT OPTION

5.SQL views can be used to hide: (C)

(A)columns and rows only.

(B)complicated SQL syntax only.

(C) both of the above can be hidden by an SQL view.

(D) None of the above is correct

6. Which order of SQL clauses **EXECUTES** is correct? (C)

A. SELECT-FROM-GROUP-WHERE-ORDER

B. FROM-SELECT-WHERE-GROUP-ORDER

C. FROM-WHERE-GROUP-SELECT-ORDER

D. SELECT-FROM-WHERE-GROUP-ORDER

7. A lossless join decomposition of a relation means: (C)

A. none of the attributes are lost

B. no functional dependencies are lost

C. the natural join of the relations in the decomposition produces the original relation

D. no information of any kind is lost

8. Which of the following commands can be used to remove access privileges associated with a table? (C)

A. DELETE B. REMOVE C. REVOKE D. DROP

9. If all candidate keys for a relation schema consist of only one attribute, the schema is satisfied at least (A)

A. 2NF B. 3NF C. BCNF D. 4NF

10. In levels of Consistency of Transaction in SQL-92, which is “only committed records to be read, and repeated reads of same record must return same value” (C)

A．READ UNCOMMITTED B．READ COMMITTED

C．REPEATABLE READ D．SERIALIZABLE

二、查询

Consider the relational database of a university research project with the following relation schemas, where the primary keys are underlined.

teacher(<u>ID,</u> name, dept_name)

project(<u>ID</u>, name, teacher_ID,budget)

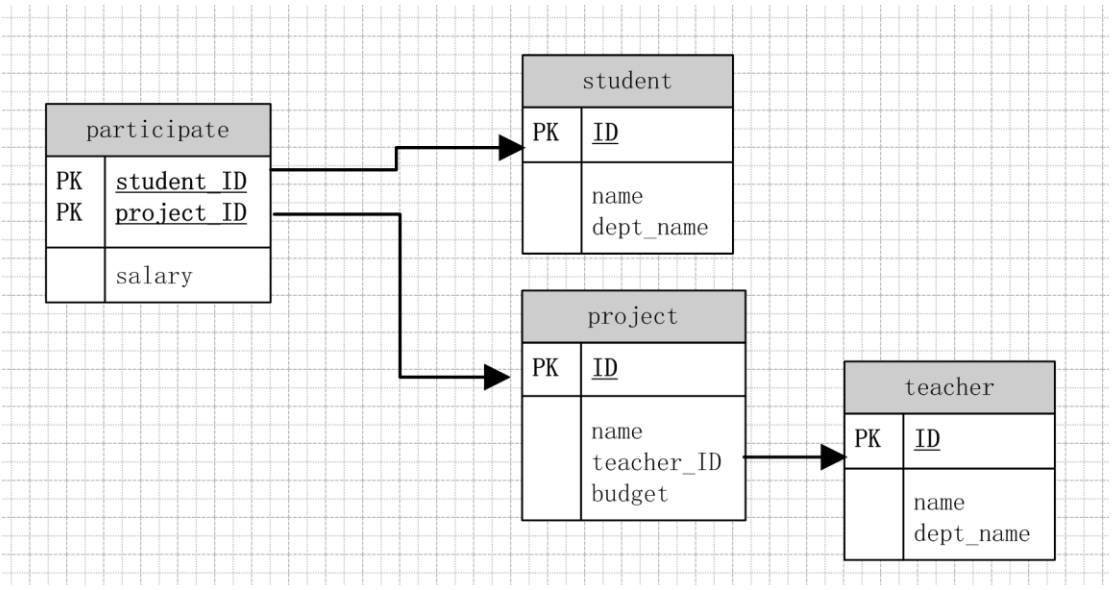
student(<u>ID</u>, name, age, dept_name)

participate(<u>student_ID,project_ID,</u> salary)

Note: Each project has a teacher as a leader, represented by attribute teacher_ID.

注意 1：如果没有给模式图，需要自己绘制，找出所有参照关系，以便后面作连接!

注意 2：查询连接中是不同的属性名时，不能随便用自然连接!



1、 **Give a relational algebra expression for each of the following queries:**

“关系代数”语言表达查询：

select: project: union: set difference: –，Cartesian product: x，rename

选用最基本的操作：并，差，选择，投影，笛卡尔积（rename：一般不涉及）。

注意：集合“并”可以由 “交”和“差”表示出来！

注意：一般增加了考“聚集”操作和“除法”操作（课件里有讲）

(1) List the Ids and names of all projects whose leader is the teacher of the “Software” department.

列出所有项目中其指导教师是“软件“系教师的指导老师的标识 ID 和姓名 name

分析：输出：教师的 ID, name，条件：有指导项目的“软件”系的教师。输入：教师和项目表。

最简单思路：就是作连接后进行筛选，最后投影。

注意： 不能用自然连接，因为 project 和 teacher 相同的 ID 不是同一属性！

$$\Pi_{\text{id,name}}\sigma_{\text{project.teacher\_id=teacher.id}}(\sigma_{\text{dept\_name='software'}}(\text{teacher}) \times \text{project})$$

$$\Pi_{\text{teacher.id,name}}\sigma_{\text{project.teacher\_id=teacher.id}}(\sigma_{\text{dept\_name='software'}}(\text{teacher})\times \text{project})$$

(2)List the Ids and names of all members who have participated in the project named “ AI” .

列出参与了项目名为’AI’的所有学生的 ID 和 name

分析：输出表是 student. 筛选条件设计项目名在 project 里，通过 participate 才能关联起来，所以需要连接，筛选和投影。

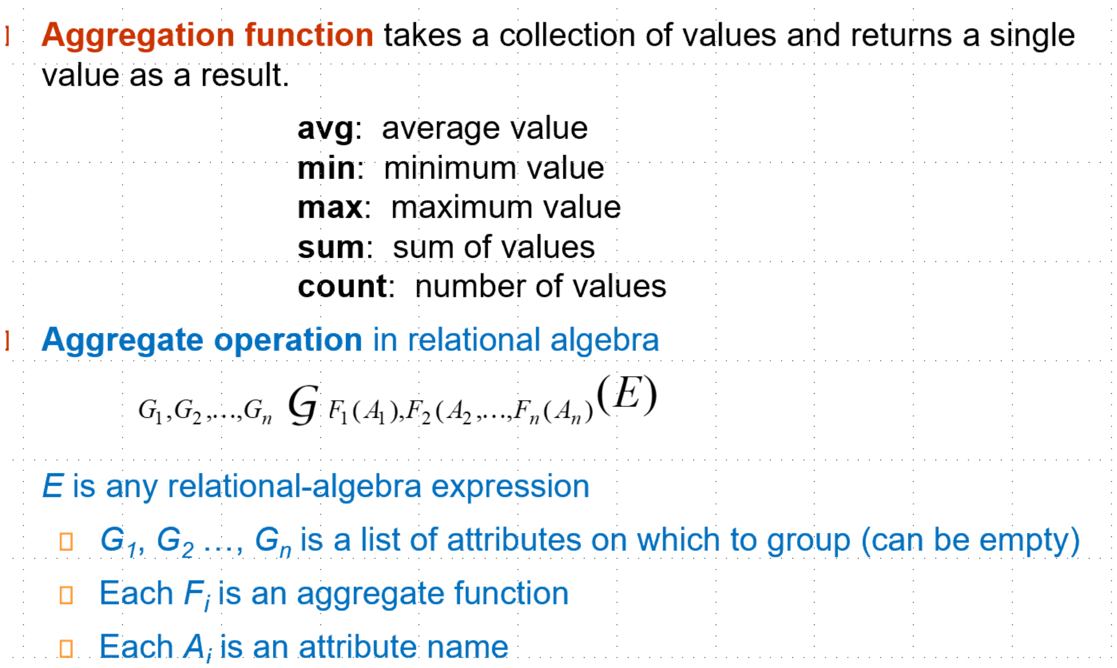
$$\Pi_{\text{id,name}}\sigma_{\text{project.name='AI'}\wedge\text{student.id=participate.studentd\_id}}$$
$$(\sigma_{\text{project.id=participate.project\_id}}(\text{project}) \times(\text{participate})) \times(\text{student}))$$

$$\Pi_{\text{id,name}}\sigma_{\text{student.id=participate.studentd\_id}}$$
$$(\sigma_{\text{project.id=participate.project\_id}}(\sigma_{\text{project.name='AI'}}(\text{project}))\times\text{participate}) \times(\text{student}))$$

(3) List the IDs of all students who have taken more than five projects

列出参与了 5 个项目以上的学生的 ID

分析：首先判定有聚集操作（超出本科版，课件里有讲 **Extended Relational-Algebra-Operations**）



- **Aggregation function** takes a collection of values and returns a single value as a result.

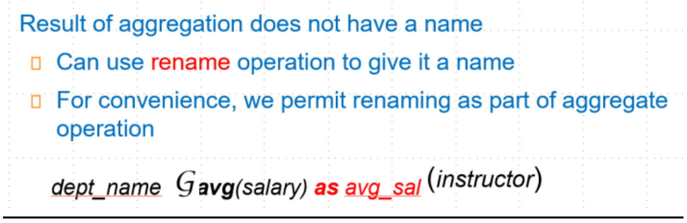
  avg: average value
  min: minimum value
  max: maximum value
  sum: sum of values
  count: number of values

- **Aggregate operation** in relational algebra

$$_{G_1,G_2,\ldots,G_n}\ \mathcal{G}\ _{F_1(A_1),F_2(A_2,\ldots,F_n(A_n)}(E)$$

E is any relational-algebra expression

- *G_1, G_2 …, G_n* is a list of attributes on which to group (can be empty)
- Each *F_i* is an aggregate function
- Each *A_i* is an attribute name



Result of aggregation does not have a name

- Can use rename operation to give it a name
- For convenience, we permit renaming as part of aggregate operation

$$_{dept\_name}\ \mathcal{G}\ _{\mathbf{avg}(salary)\ \mathbf{as}\ avg\_sal}(instructor)$$

$$\Pi_{student\_id}\ \sigma_{number\_project>5}\ (_{student}\ \mathcal{G}\ _{count\ distinct(student\_id)\ as\ num\_project}(participate)$$

(4) List the IDs and names of all students who have participated all projects that the student with ID “12345” has participated.

列出参与了学生 ID=’12345’的学生所参与的全部项目的学生的 ID，name，
分析：关系代数表达形如若查询诸如“选修了全部课程”的学生、“使用了全部零件”的工程等，需用除法操作实现！（参见 LEC8 CHAPTER6 的 PPT 中 **Additional Operations*）！**

$$\Pi_{id,name}\sigma_{student\_id=student.id}$$

$$(\Pi_{student\_id,project\_id}(participate)\div\Pi_{project\_id}\sigma_{student\_id='12345'}(participate)\times student)$$

$$\Pi_{id,name}\ \sigma_{student\_id=student.id}$$

$$(\Pi_{student\_id,project\_id}(participate)\div\Pi_{project\_id}\sigma_{student\_id='12345'}(participate))$$

2、　Write SQL statements to perform the following commands:

"SQL"语言表达查询:

注意：　重点是考察初级 SQL 的灵活使用

条件语句：like 字符串匹配"%"，"-"，　between…and：数值范围含等号。

嵌套查询：融合集合运算，多表连接，聚集函数等。

(1)List Ids and names of all projects with a budget greater than 10,000 and with a name include "Software".

列出项目预算大于 10000 的并且项目名包含"软件"的项目 ID，name

分析：条件在 project 中都可以找到字段，所以输入只 project，考察 where 条件写法!

```
select id,name from project where budget>10000 and name like ' %Software %';
```

(2)List Ids and names of all student of "Software" department who have NOT participated in any project.

列出"software"系没有参与项目的学生 ID 和 name

分析：输出取自 student，筛选条件是"没有参与项目"涉及 participate，并且是没有参与。如果直接连接 student 和 participate 得到的是"参与项目"的，所以应该嵌套子查询~集合成员资格进行筛选。

```
select id,name from student
where dept_name='Software' and id not in
                            (select distinct student_ID from participate );
```

若采用"集合差"运算：注意：except 和 except all 的区别：

```
 （select id,name from student
   where dept_name='Software'）
except
   (select id,name
   from student, participate
   where id=student_ID);
```

(3)List Ids and names of all projects and the number of students who have participated in it .Sort the results in descending order based on the number of students.

列出项目的 ID,name 和参与该项目的学生数。并且输出结果按学生数降序排列。

分析：涉及聚集函数因为有统计学生数，主要涉及 pariticipate 表，分组条件是 project_ID。考虑输出字段在 project 表，所以需要和 student 表作连接。考虑到输出结果"学生数"需要作为排序条件，所以需要取别名以方便表达。（注意不要写成自然连接，因为相同属性在各表中名称不同）

```
select id,name,count(*) as stu_nums
from participate,project
where participate.project_id=project.id group by id order by stu_nums desc;
```

(4)List Ids of projects whose students earn a higher salary on average than the average salary at project with　id　'001'.

列出比项目ID=’001’的平均salary高的项目ID

分析：涉及聚集函数，并且包含having条件筛选。

```
select   project_id   from   participate   group by project_id
having avg (salary) >(select avg (salary) from participate where project_id= ’001')
```

(5) List the names, sum salary of all students who have the most (highest) sum salary.

列出所有获得项目最高sum(salary)的学生的姓名及其sum(salary)

分析：本类题涉及聚集函数的，嵌套子查询中的**集合比较**!

聚集函数的筛选条件是：输出学生的sum(salary) >=all（所有学生的sum(salary)）。

```
select student.name,sum(salary)
from student, participate
where student.id= participate.student_id
group by student.id
having sum(salary)>= all
        (select sum(salary)
        from student, participate
        where student.id= participate.student_id
        group by student.id );
```

```
select student.name,sum(salary)
from student, participate
where student.id= participate.student_id
group by student.id
having sum(salary)>= all
        (select sum(salary)
        from participate
        group by student.id );
```

(6)List the names of all students who have participated in all projects led by the teacher whose name is ‘Enstein’;.

列出参与了由教师“Enstein”所指导的**所有项目**的学生姓名 name

分析：类似于关系代数除法，SQL 常采用**嵌套子查询的空关系测试（exists 或 not exists）实现**（注意：因为没有数量，不能用集合比较（>=all））该查询转换为“ 不存在一个项目是“Enstein”指导的，而该学生没有选！ ”

注意：需要采用“相关子查询”，参考 PPT LEC4 教材 3.8.3 空关系测试

```
select name
 from student s
where not exists
(
   select project.id from teacher,project
```

```
        where   teacher.id=project.teacher_id and teacher.name='Enstein'
        except
        select participate.project_id
        from student, participate
        where student.id=participate.student_id and student.id=s.id;
    )
```

三、规范化

1. **Consider the following relational schema:**

project_material=(prjno, prjname, prjbudget, begindate, enddate, materialNo, materialName,materialquantity)

It contains information about the material usage of projects.

The Attribute prjno is the No. of project, The Attribute prjname is the name of project ,The Attribute prjbudget is the budget of project.

The Attribute materialquantity is quantities of materials used in a projerct.

- **Each project has unique project No.**
- **Different projects may have same project name, project budget, project begindate and project enddate.**
- **Each project use various materials.**

The following is an instance of the schema:

prjNo	prjName	prjBudget	begindate	enddate	Material No	Material Name	Material quantity
101	1# project	10000	2020/01/01	2021/01/01	10001	Cement	100
101	1# project	10000	2020/01/01	2021/01/01	10002	Steel	120
101	1# project	10000	2020/01/01	2021/01/01	10003	Sand	120
102	2# project	20000	2020/09/01	2021/07/01	10001	Cement	100
102	2# project	20000	2020/09/01	2021/07/01	10003	Sand	110
103	1# project	10000	2020/01/01	2021/01/01	10001	Cement	100

(1) **Based on above, Identify functional dependencies of *project_material*.**

分析：主要根据前面描述的语义（加点的黑色文字），辅以 instance 找函数依赖。

prjNo→prjName,prjBudget,begindate,enddate

materialNo→materialName

prjNo,materialNo→materialquantity

(2) Based on above, Identify the **candidate key(s)** of **project_material**.

分析：候选码主要根据函数依赖，看左部属性的闭包是否包含 R，若是则是候选码。

Candidate Key:（prjNo，materialNo）

(3) Is the relation schema **project_materialin BCNF**? Why? Is it in **3NF**? Why? If it is not in 3NF, bring it to a set of relations at least in 3NF; specify primary keys and referential integrity constraints for each relation.

分析：考察 BCNF 和 3NF 判定方法。3NF 的分解算法,可以判定前面的 F 已经是 Fc.

Not in BCNF and not in 3NF. {prjNo, materialNo } is the candidate key of it, materialNo→materialName, materialNo is not a super(candidate) key.

This gives the following 3NF relations:（关系命名可以不同）

project(prjno, prjname, prjbudget, begindate, enddate)， PK:prjno,

material(materialNo,materialName) PK:materialNo

useage(prjno, materialNo, materialquantity) PK:(prjno, materialNo)

FK: prjno 参考表 project, FK: materialNo 参考表 material.

2.Consider a relation R(A,B,C,D,E,G) with the set of Functional Dependencies

F={BE→G, BD→G, CD→A, CE→G, CDE→AB, BC→A, B→D}

分析：形式化考察规范化理论，函数依赖的闭包，属性的闭包，候选码的计算，最小化函数依赖计算，分解算法等。

(1) Give all candidate keys of R. [3 points]

分析：有 7 个属性，如果根据 R 的子集作为左部求 F 下的属性闭包，看是否包含 R。这种方法下，子集过多不适用。简单方法：只计算 F 中依赖的左部属性闭包，逐步补全。

BE+={BEG}, BD+={BDG}, CD+={CDA}, CE+={CEG}, CDE+={CDEABG}=R,

BC+={BCA}, B+={BDG}

经过观察单个属性，两个属性的闭包都不可能得到 R,

继续看左部 3 个属性的继续查，根据已有左部可以得到这些组合 BED, BEC,BCD,CDE

BED+={BEDG}, BEC+={BECGDA}=R， BCD+={BCDA}，CDE+={CDEABG}=R

没有必要继续找左部 4 个属性的，因为候选码是最小超码。所以候选码只有 2 个：

BCE, CDE

(2) Give a canonical cover of F.

分析：根据求 Fc 的方法：相同左部合并，删除右部多余属性，输出左部多余属性。参见书上 8.4.3 算法。

考察 F={BE→G, BD→G, CD→A, CE→G, CDE→AB, BC→A, B→D}

注意：这里，因为 BC+由其它函数依赖可以求出包含 A，

所以 BC→A 的右端 A 是冗余的！需要去掉 BC→A 这个函数依赖。

根据教科书 7 版 P231 页第 2 行尾：“如果一个函数依赖在它的右侧只包含一个属性，例如 A→C，并且这个属性被证明是多余的，那么我们将得到一个右侧为空的函数依赖。这样的函数依赖应该被删除”！

Fc = {B→GD,CD→A,CE→G,CDE→B}

(3) Is R in 3NF? explain why if it is or decompose it into 3NF if not.

分析：Not in 3NF, **for “CE→G” violate 3NF definition.***左边不是码，右边是非码属性（不含候选码属性。*

decompose it into

R1({B,D,G}, { B→DG }),

R2({A,C,D}, { CD→A }),

R3({C,E,G}, { CE→G }),

R4({B,C,D,E}, { CDE→B }

四、并发控制

Consider the concurrent schedule of transactions T1 and T2.

T1	T2
read (A)	
A=A+1	
	read (B)
write(A)	
	B=B-1
read(B)	
	write(B)
B=B+1	
	read(C)
write(B)	
	C=C+2
	write(C)
	Commit
Commit	

1. Is the schedule conflict serializable? If so, give an equivalent serial schedule. If not, give an explain briefly. **(5 points)**

   No. it is not conflict serializable.

   T1 read(B) AND T2 write(B) is conflict instruction

   T2 write(B) AND T1's write(B) is conflict instruction

   则优先图中有环，所以不是冲突可串行化的调度。

2. Add lock and unlock instructions to Transactions T1 and T2, only consider the 2PL concurrent schedule of transactions T1 and T2, observe the two-phase-locking protocol ,Can it results in a deadlock? **(5 points)**

   T1: lock-X(A), read(A), write(A),lock-X(B),read(B),write(B), unlock(A),unlock(B)

   T2: lock-X(B), read(B), write(B), lock-X(C),read(C),write(C) unlock(B), unlock(C) .

   It can not results in a deadlock, for T1 lock A and B, T2 lock B and C.

五、数据库设计

For a simplified application on Supermarket Management, there are three entity sets: **Supermarket**, **Product**, and **Warehouse**.

The attributes of Supermarket includes: SupermarketNo, SMName, SMAddress, Telephone, Manager; and the attributes of Product contains: ProductNo, PName, Price, Producer, ProduceDate; and the attributes of Warehouse contains: WarehouseNo, WarehouseName, WAddress, Adminstrator.

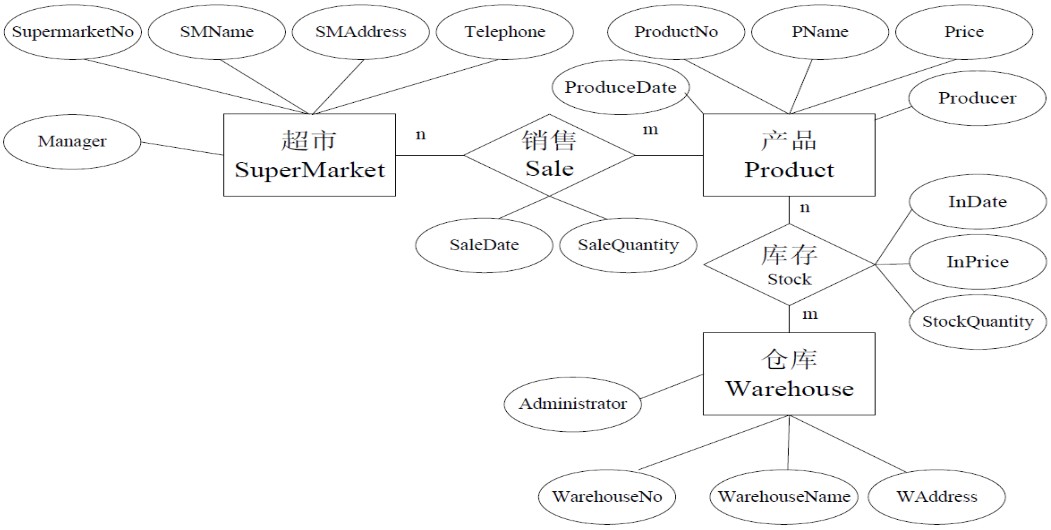
And two relationship sets: Supermarket and Product related through a binary relationship set Sale, Supermarket sales products ,record the product's sale quantities every day ; and Product and Warehouse related through a binary relationship set Stock. **The warehouse only input(store) each product once a day. The supermarket can sale only each product once a day.**
The two relationship sets have the following attributes respectively: Sale has attributes: SaleDate and SaleQuantity; Stock has attributes: InDate, InPrice, and StockQuantity.

分析：联系的约束是哪种需要考虑，超市可以销售多种产品，产品也可以在多个超市销售，所以是多对多。The warehouse only input(store) each product once a day. Stock has attributes: InDate, InPrice, and StockQuantity. 根据描述，仓库它可以进多种产品，一个产品也可以进入不同仓库，所以仓库和产品也是多对多。

（只能用一种 ER 图的表示法，不能多种混用）

注意： 销售，库存 这两个联系： 多值复合？
（下图需要修改联系的属性为多值）

1. Please give the corresponding ER diagram. (8 points)



3. Create the corresponding relational schemas, and point out the primary keys and the foreign keys. (8 points)

Supermarket=(SupermarketNo, SMName, SMAddress,Telephone,Manager)
Product=(ProductNo,PName,Producer,ProduceDate)
WareHouse=(WarehouseNo,WarehouseName,WaAddress,Administrator)
Sale=(SupermarketNo, ProductNo, SaleDate,SaleQuantity);
FK1: SupermarketNo–> Supermarket(SupermarketNo)
FK2: ProductNo–> Product(ProductNo)
Stock=(ProductNo,WarehouseNo,InDate,InPrice,StockQuantity);
FK1: ProductNo–> Product(ProductNo)
FK2: WarehouseNo–> Warehouse (WarehouseNo)

3. Use SQL statements to define Product table, and give proper integrity constraints.(4 points)

```
Create table Product(
    ProductNo char(10),    //长度合理即可
    PName var char(20) not null,
    Price decimal(8,2),
    Producer varchar(60),
    ProduceDate date,
    primary key (ProductNo),
    check Price>0)
```