数据库实用教程（第四版）配套资料
习题参考答案

1.1 人工管理阶段数据管理各有哪些特点？
答：人工管理阶段主要有四个特点：　　数据不保存在计算机内；没有专用的软件对数据进行管理；只有程序的概念，没有文件的概念；数据面向程序。

1.2 文件系统阶段的数据管理各有哪些特点？
答：文件系统阶段主要有五个特点：　　数据以“文件”形式长期保存；数据的逻辑结构与物理结构有了区别；文件组织已多样化；数据面向应用；对数据的操作以记录为单位。

1.3 文件系统阶段的数据管理有些什么缺陷？试举例说明。
答：主要有三个缺陷：数据冗余；数据不一致性；数据联系弱。
例如学校里教务处、财务处、保卫处建立的文件中都有学生详细资料，譬如联系电话，家庭住址等。这就是“数据”冗余；如果某个学生搬家，就要修改三个部门文件中的数据，否则会引起同一数据在三个部门中不一致；产生上述问题的原因是这三个部门的文件中数据没有联系。

1.4 数据管理的数据库阶段产生的标志是哪三件事情？
答：进入数据库阶段的标志是 20 世纪 60 年代末发生的三件事件：
- 1968 年 IBM 公司研制的 IMS 系统是一个典型的层次 DBS；
- 1969 年美国 CODASYL 组织 DBTG 报告，提出网状 DBS 的概念；
- 1970 年美国 IBM 公司的 E.F.Codd 发表论文，提出关系模型的思想。

1.5 数据库阶段的数据管理有哪些特点？
答：主要有五个特点：
- 采用数据模型表示复杂的数据结构；
- 有较高的数据独立性；
- 为用户提供了方便的用户接口；
- 提供了四个方面的数据控制功能；
- 对数据的操作以数据项为单位，增加了系统的灵活性。

1.6 什么是数据独立性？在数据库中有哪两级独立性？
答：数据独立性是指应用程序与 DB 的数据结构之间相互独立。在物理结构改变时，尽量不影响应用程序，称为物理数据独立性；在逻辑结构改变时，尽量不影响应用程序，称为逻辑数据独立性。

1.7 试解释 DB、DBMS 和 DBS 三个概念。
答：DB 是长期存储在计算机内、有组织的、统一管理的相关数据的集合。
DBMS 是位于用户与 OS 之间的一层数据管理软件，它为用户或应用程序提供访问 DB 的方法。

DBS 是实现有组织地、动态地存储大量关联数据、方便多用户访问的计算机硬件、软件和数据资源组成的系统，即采用数据库技术的计算机系统。

1.8 分布式数据库系统有哪些特点？

答：DDBS 主要有三个特点：

- 数据物理上分布在各地，但逻辑上是一个整体；
- 每个场地既可以执行局部应用，也可以执行全局应用；
- 各地的计算机由数据通信网络相连接。

1.9 分面向对象数据库系统各有哪些特点？

答：面向对象数据系统主要有两个特点：

- 面向对象数据模型能完整地描述现实世界的数据结构，能表达数据间嵌套、递归的联系。
- 具有面向对象技术的封装性和继承性的特点，提高了软件的可重用性。

2.1 名词解释

- 逻辑数据：指程序员或用户用以操作的数据形式。
- 物理数据：指存储设备上存储的数据。
- 联系的元数：与一个联系有关的实体集个数，称为联系的元数。
- 1:1 联系：如果实体集 E1 中每个实体至多和实体集 E2 中的一个实体有联系，反之亦然，那么 E1 和 E2 的联系称为“1:1 联系”。
- 1:N 联系：如果实体集 E1 中每个实体可以与实体集 E2 中任意个（零个或多个）实体有联系，而 E2 中每个实体至多和 E1 中一个实体有联系，那么 E1 和 E2 的联系是“1:N 联系”。
- M:N 联系：如果实体集 E1 中每个实体可以与实体集 E2 中任意个（零个或多个）实体有联系，反之亦然，那么 E1 和 E2 的联系称为“M:N 联系”。
- 数据模型：能表示实体类型及实体间联系的模型称为“数据模型”。
- 概念数据模型：独立于计算机系统、完全不涉及信息在计算机中的表示、反映企业组织所关心的信息结构的数据模型。
- 结构数据模型（或逻辑数据模型）：与 DBMS 有关的，直接面向 DB 的逻辑结构、从计算机观点对数据建模的数据模型。
- 层次模型：用树型（层次）结构表示实体类型及实体间联系的数据模型称为层次模型。
- 网状模型：用有向图结构表示实体类型及实体间联系的数据模型称为网状模型。
- 关系模型：用二维表格表达实体集的数据模型。
- 外模式：是用户用到的那部分数据的描述。
- 概念模式：数据库中全部数据的整体逻辑结构的描述。
- 内模式：DB 在物理存储方面的描述。
- 外模式/模式映象：用于定义外模式和概念模式之间数据结构的对应性。
- 模式/内模式映象：用于定义概念模式和内模式之间数据结构的对应性。
- 数据独立性：应用程序和 DB 的数据结构之间相互独立，不受影响。
- 物理数据独立性：在 DB 的物理结构改变时，尽量不影响应用程序。
- 逻辑数据独立性：在 DB 的逻辑结构改变时，尽量不影响应用程序。
- 主语言：编写应用程序的语言（如 C 一类高级程序设计语言），称为主语言。

• DDL：定义 DB 三级结构的语言，称为 DDL。

• DML：对 DB 进行查询和更新操作的语言，称为 DML。

• 过程性语言：用户编程时，不仅需要指出“做什么”，还需要指出“怎么做”的语言。

• 非过程性语言：用户编程时，只需指出“做什么”，不需要指出“怎么做”的语言。

• DD（数据字典）：存放三级结构定义的 DB，称为 DD。

• DD 系统：管理 DD 的软件系统，称为 DD 系统。

2.2 逻辑记录与物理记录，逻辑文件与物理文件有些什么联系和区别？

答：逻辑数据是用户用以操作的数据形式，是抽象的概念化数据。物理数据是实际存放在存储设备上的数据。

逻辑数据与物理数据在结构上可以差别很大，需通过两级映象来进行数据传输和格式转换。

从以上的解释可以看出，逻辑记录和逻辑文件是用户在程序中使用的记录和文件，而物理记录和物理文件是指磁盘上的记录和文件。逻辑记录、文件与物理记录、文件在结构、组成上有很大的差异，而数据管理软件就是通过三级结构两级映象来实现逻辑数据与物理数据之间的转换。

2.3 设某商业集团数据库有三个实体集。一是“商品”实体集，属性有商品号、商品名、规格、单价等；二是“商店”实体集，属性有商店号、商店名、地址等；三是“供应商”实体集，属性有供应商编号、供应商名、地址等。

供应商与商品之间存在“供应”联系，每个供应商可供应多种商品，每种商品可向多个供应商订购，每个供应商供应每种商品有个月供应量；商店与商品间存在“销售”联系，每个商店可销售多种商品，每种商品可在多个商店销售，每个商店销售每种商品有个月计划数。

试画出反映上述问题的 ER 图，并将其转换成关系模型。

答：（1）ER 图-1（图 2.3-1）：

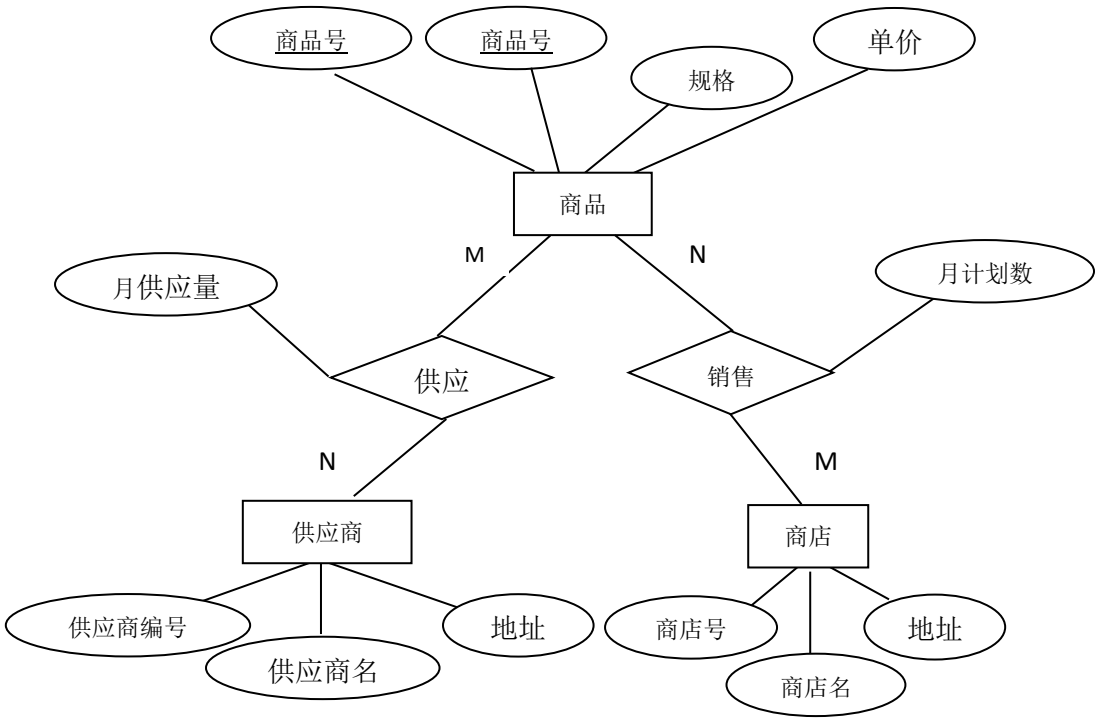


图 2.3-1

ER 图-2（图 2.3-2）：

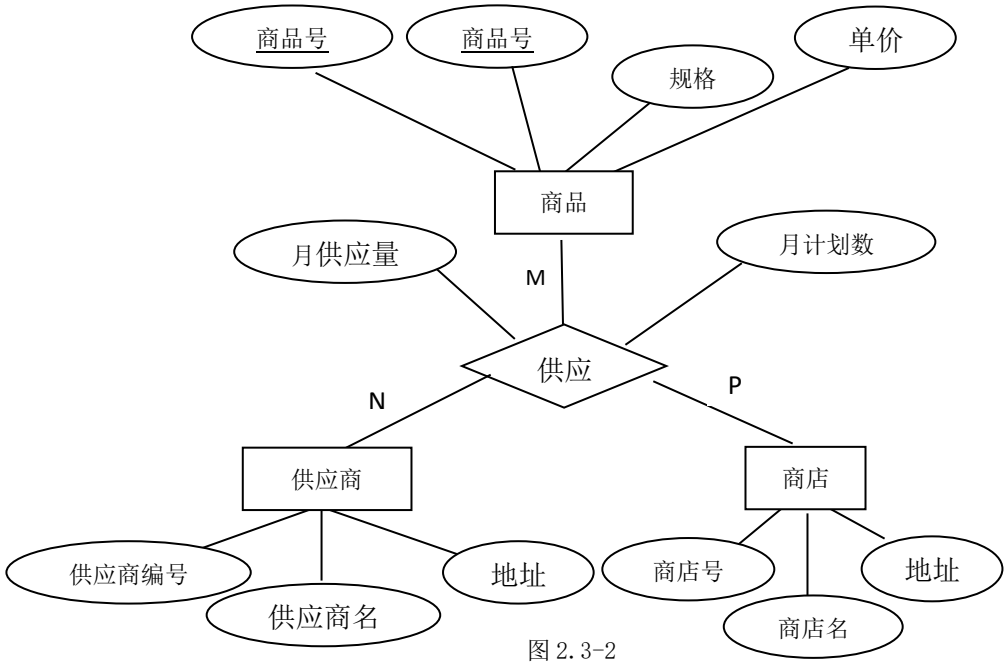


图 2.3-2

（2）ER 图-1 可转换 5 个关系模式：

商店（商店号，商店名，地址）
供应商（供应商号，供应商名，地址）
商品（商品号，商品名，规格，单价）
销售（商店号，商品号，月计划数）
供应（供应商号，商品号，月供应量）

ER 图-2 可转换 4 个关系模式：

商店（商店号，商店名，地址）
供应商（供应商号，供应商名，地址）
商品（商品号，商品名，规格，单价）
供应（商店号，供应商号，商品号，月计划数，月供应量）

2.4 试述 ER 模型、层次模型、网状模型、关系模型和面向对象模型的主要特点。

答：ER 模型能直接表示实体类型及实体间联系,与计算机系统无关,充分反映用户的需求,用户容易理解。

层次模型的数据结构为树结构，记录之间联系通过指针实现，查询较快，但 DML 属于过程化的，操作复杂。

网状模型的数据结构为有向图，记录之间联系通过指针实现，查询较快，并且容易实现 M:N 联系，但 DML 属于过程化的语言，编程较复杂。

关系模型的数据结构为二维表格，容易为初学者理解。记录之间联系通过关键码实现。DML 属于非过程化语言，编程较简单。

面向对象模型能完整描述现实世界的数据结构，具有丰富的表达能力，能表达嵌套、递归的数据结构。但涉及的知识面较广，用户较难理解，这种模型尚未普及。

2.5 数据之间联系在各种结构数据模型中是怎么实现的？

答：在层次、网状模型中，数据之间的联系通过指针实现的；

在关系模型中，数据之间联系通过外键和主键间联系实现的；

在面向对象模型中，数据之间嵌套、递归联系通过对象标识符（OID）实现的。

2.6 DB 的三级模式结构描述了什么问题？试详细解释。

答：DB 的三级模式结构是对数据的三个抽象级别，分别从外部（用户）级、概念级和内部级去观察数据库。

外部级是用户使用的局部数据库的逻辑结构，其描述称为外模式。

概念级是 DB 的整体逻辑结构，其描述称为概念模式。

内部级是 DB 的物理结构，其描述称为内模式。

2.7 试述概念模式在数据库结构中的重要地位。

答：数据按外模式的描述提供给用户，按内模式的描述存储在磁盘中，而概念模式提供了连接这两级的相对稳定的中间观点，并使得两级的任何一级的改变都不受另一级的牵制。

2.8 什么是数据独立性？其目的是什么？

答：数据独立性是指应用程序与 DB 的数据结构之间相互独立。其目的是在 DB 的物理或逻辑数据结构改变时，尽量不影响应用程序。

2.9 数据独立性与数据联系这两个概念有什么区别？

答：数据独立性是指应用程序和 DB 的数据之间相互独立，不受影响，对系统的要求是“数据独立性要高”，而数据联系是指记录之间的联系，对系统的要求是“数据联系密切”。

2.10 试述 DBMS 的工作模式.

答：DBMS 的工作模式有六点：

- 接受应用程序的数据请求和处理请求；
- 将用户的数据请求转换成低层指令；
- 实现对 DB 的操作；
- 从对 DB 的操作中接受查询结果；
- 对查询结构进行处理；
- 将处理结果返回给用户。

2.11 试述 DBMS 的主要功能。

答：DBMS 的主要功能有 DB 的定义、操纵、保护、维护和数据字典等五个功能。

2.12 叙述 DBMS 对数据库的保护功能。

答：包括 DB 的恢复、并发控制、完整性控制和安全性控制等四个方面。

2.13 试叙述 DBMS 对数据库的维护功能。

答：包括 DB 的数据载入、转换、转储、DB 的改组以及性能监控等功能。这些功能分别由各个实用程序完成。

2.14 从模块结构观察，DBMS 由哪些部分组成？

答：DBMS 由两大部分组成：查询处理器和存储管理器。（解释略）

2.15 DBS 有哪几部分组成？其中 DD 有什么作用？

答：DBS 由 DB、硬件、软件和 DBA 等四个部分组成。（解释略）

在 DBS 中，DD 是存储三级结构描述（即元数据）的 DB。DBMS 的所有工作都要以 DD 中的元数据为依据，也就是所有工作都要通过 DD 访问 DB。

2.16 DBS 中 DD 有什么作用？

答：在 DBS 中，DD 是存储三级结构描述（即元数据）的 DB。DBMS 的所有工作都要以 DD 中的元数据为依据，也就是所有工作都要通过 DD 访问 DB。

2.17 什么是 DBA？DBA 应具有什么素质？DBA 的职责是什么？

答：DBA 是控制数据整体结构的一组人员，负责 DBS 的正常运行，承担创建、监控和维护 DB 结构的责任。

DBA 必须具备下列 4 条素质：熟悉企业全部数据的性质和用途；对所有用户的需求有

充分的了解；对系统的性能非常熟悉；兼有系统分析员和运筹学专家的品质和知识。

DBA 的主要职责有 6 点：定义模式；定义内模式；与用户的联络；定义安全性规则；定义完整性规则；DB 的转储与恢复。

2.18 试对 DBS 的全局结构作详细解释。

答：从四个方面解释：

• 数据库用户有四类：DBA，专业用户，应用程序员，终端用户。

• DBMS 的查询处理器有四个模块：DML 编译器，嵌入型 DML 预编译器，DDL 编译器，查询运行核心程序。

• DBMS 的存储管理器有四个模块：授权和完整性管理器，事务管理器，文件管理器，缓冲区管理器。

• 磁盘存储器中有五种数据结构：数据文件，数据字典，索引文件，统计数据组织和日志。

2.19 使用 DBS 的用户有哪几类？

答：（略，见习题 2.18）

2.20 DBMS 的查询处理器有哪些功能？

答：（略，见习题 2.18）

2.21 DBMS 的存储管理器有哪些功能？

答：（略，见习题 2.18）

2.22 磁盘存储器中有哪五类主要的数据结构？

答：（略，见习题 2.18）

3.1 答：名词解释

关系模型：用二维表格表示实体集，外键和主键表示实体间联系的数据模型，称为关系模型。

关系模式：是对关系的描述，包括模式名、诸属性名、值域名和模式的主键。

关系实例：关系模式具体的值，称为关系实例。

属性：即字段或数据项，与二维表中的列对应。属性个数，称为元数（arity）。

域：属性的取值范围，称为域。

元组：即记录，与二维表中的行对应。元组个数，称为基数（cardinality）。

超键：能惟一标识元组的属性或属性集，称为关系的超键。

候选键：不含有多余属性的超键，称为候选键。

主键：正在使用的、用于标识元组的候选键，称为主键。

外键：属性集 F 是模式 S 的主键，在模式 R 中也出现，那么称 F 是模式 R 的外键。

实体完整性规则：实体的主键值不允许是空值。

参照完整性规则：依赖关系中的外键值或者为空值，或者是相应参照关系中某个主键码。

过程性语言：编程时必须给出获得结果的操作步骤，即指出“干什么”及“怎么干”的语言。

非过程性语言：编程时，只需指出需要什么信息，不必给出具体的操作步骤，即只要指出“干什么”，不必指出“怎么干”的语言。

无限关系：指元组个数为无穷多个的关系。

无穷验证：验证公式真假时需要进行无限次验证。

3.2 **为什么关系中的元组没有先后顺序？**

答：由于关系定义为元组的集合，而集合中的元素是没有顺序的，因此关系中的元组也就没有先后

的顺序（对用户而言）。这样既能减少逻辑排序，又便于在关系数据库中引进集合论的理论。

3.3 为什么关系中不允许有重复元组？

答：每个关系模式都有一个主键，在关系中主键值是不允许重复的。如果关系中有重复元组，那么其主键值肯定相等，起不了惟一标识作用，因此关系中不允许有重复元组。

3.4 关系与普通表格、文件有什么区别？

答：关系与表格、文件相比，有下列 4 个不同点：

属性值不可分解、没有重复的元组、没有行序、使用时有列序。

3.5 笛卡儿积、等值联结、自然联结三者有什么区别？

答：笛卡儿积是关系代数中的一个基本操作，而等值联结和自然联结是关系代数中的组合操作。等值联结是在笛卡儿积的基础上选择满足两个关系中给定属性值相等的元组的集合。自然联接是在两个关系的相同属性组上的等值连接。并且自然联接要在结果中把重复的属性去掉,而等值连接则不必。

3.6 设有关系 R 和 S：

R：

A	B	C
3	6	7
2	5	7
7	2	3
4	4	3

S:

A	B	C
3	4	5
7	2	3

计算 R∪S，R-S，R∩S，R×S， $\pi_{3,2}$（S），$\sigma_{B<'5'}(R)$，$R \underset{2<2}{\bowtie} S$，$R \bowtie S$。

答：R∪S 的结果是：

A	B	C
3	6	7
2	5	7
7	2	3
4	4	3
3	4	5

R-S 的结果是：

A	B	C
3	6	7
2	5	7
4	4	3

R∩S 的结果是：

A	B	C
7	2	3

$\pi_{3,2}$（S）

C	D
5	4
3	2

R×S 的结果是：

A	B	C	D	E	F
3	6	7	3	4	5
3	6	7	7	2	3
2	5	7	3	4	5
2	5	7	7	2	3
7	2	3	3	4	5

7	2	3	7	2	3
4	4	3	3	4	5
4	4	3	7	2	3

$\sigma_{B<'5'}(R)$ 的结果：

A	B	C
7	2	3
4	4	3

$R \overset{}{\bowtie}_{2<2} S$ 的结果：

A	B	C	D	E	F
7	2	3	3	4	5

$R \bowtie S$ 的结果：

A	B	C
7	2	3

3.7 如果 R 是二元关系，那么下列元组表达式的结果是什么？

{t|(∃u)(R(t) ∧R(u) ∧(t[1] ≠u[1]∨t[2] ≠u[2]))}

答：当 R 的元组数≥2 时，R 中每个元组都存在与之不相同的元组，因此表达式的结果为关系 R；

当 R 的元组数为 0 或 1 时，表达式的结果为空关系。

3.8 假设 R 和 S 分别是三元和二元关系，试把表达式 $\pi_{1,5}(\sigma_{2=4\vee 3=4}(R\times S))$转换成等价的:

①汉语查询句子;②元组表达式;③域表达式。

答：①汉语查询句子:在关系 R 和 S 的笛卡尔积中，选取第 2 个属性值与第 4 个属性值相等，或者第 3 个属性值与第 4 个属性值相等的那些元组，再取第 1 列和第 5 列组成新的关系。

②等价的元组表达式是：

{t|(∃u)(∃v)(R(u)∧S(v)∧t[1]=u[1]∧t[2]=u[2]∧t[3]=u[3]∧t[4]=v[1]∧ t[5]=v[2])}

与 $\sigma_{2=4 \vee 3=4}$（R×S）等价的元组表达式是：

{t|(∃u)(∃v)(R(u) ∧ S(v) ∧ t[1]=u[1] ∧ t[2]=u[2] ∧ t[3]=u[3] ∧ t[4]=v[1] ∧ t[5]=v[2] ∧ (t[2]=t[4]∨ t[3]=t[4]))}

与 $\pi_{1,5}$（ $\sigma_{2=4 \vee 3=4}$（R×S））等价的元组表达式是：

{w|(∃t)(∃u)(∃v)(R(u) ∧S(v) ∧t[1]=u[1] ∧t[2]=u[2] ∧t[3]=u[3] ∧t[4]=v[1] ∧t[5]=v[2] ∧ (t[2]=t[4]∨t[3]=t[4]) ∧w[1]=t[1]∧w[2]=t[5])}

再对上述元组表达式化简（消去 t）可得：

{w|(∃u)(∃v)(R(u)∧S(v)∧(u[2]=v[1]∨u[3]=v[1])∧w[1]=u[1]∧w[2]=v[2])}

在熟练后，可以直接写出上式。

③ 再转换成域表达式：

{w_1w_2|(∃u_1)(∃u_2)(∃u_3)(∃v_1)(∃v_2)(R($u_1u_2u_3$) ∧S(v_1v_2) ∧(u_2=v_1∨u_3=v_1) ∧w_1=u_1∧w_2=v_2)}

再化简（消去 u_1，v_2）可得：

{w_1w_2|(∃u_2)(∃u_3)(∃v_1)(R($w_1u_2u_3$) ∧S(v_1w_2) ∧(u_2=v_1∨u_3=v_1))}

3.9 假设 R 和 S 都是二元关系，试把元组表达式{t|R(t)∧(∃u)(S(u)∧u[1]≠t[2])}转换成等价的：

①汉语查询句子; ②域表达式; ③关系代数表达式。

答：①在关系 R 中选取第 2 列的值与关系 S 中某个元组的第 1 列值不相等的那些元组，组成新的关系。

②域表达式为：{ t_1t_2 | R(t_1t_2)∧(∃u_1) (∃u_2) (S(u_1u_2) ∧ u_1≠t_2)}

③关系代数表达式为：

$\pi_{1,2}(\sigma_{2\neq3}(R\times S))$ 或 $\pi_{1,2}(R \underset{2\neq3}{\bowtie} S)$

3.10 试把域表达式{ ab | R(ab) ∧ R(ba)}转换成等价的：⑴汉语查询句子；⑵关系代数表达式；⑶元组表达式。

答：（1）在关系 R 中选取属性值交换后仍是 R 中元组的那些元组，组成新的关系。

（2） 关系代数表达式为： $\pi_{1,2}(\sigma_{1=4\wedge 2=3}(R\times R))$

也可写成：$R\cap \pi_{2,1}(R)$

（3）元组表达式为：{ t |(∃u)(∃v)(R(u)∧R(v)∧u[1]=v[2]∧u[2]=v[1]∧t[1]=u[1]∧t[2]=u[2])}

或：{ t |(∃v)(R(t)∧R(v)∧t[1]=v[2]∧t[2]=v[1])}

3.11 有两个关系 R (A, B, C)和是 S（D, E, F）,试把下列关系代数表达式转换成等价的元组表达式:

①$\pi_A(R)$； ②$\sigma_{B='17'}(R)$；

③$R\times S$； ④$\pi_{A,F}(\sigma_{C=D}(R\times S)$

答：①πA（R）：{ t |(∃u)(R(u)∧t[1]=u[1])}

②σB='17'（R）：{ t|R(t)∧t[2]= '17'}

③ R×S：{ t|(∃u)(∃v)(R(u)∧S(v)∧t[1]=u[1]∧t[2]=u[2]∧t[3]=u[3]∧t[4]=v[1] ∧t[5]=v[2] ∧t[6]=v[3])}

④πA，F（σC=D（R×S））：{t|(∃u)(∃v)(R(u)∧ S(v)∧u[3]=v[1]∧t[1]=u[1]∧t[2]=v[3])}

3.12 设有三个关系

S (SNO,SNAME,AGE,SEX,SDEPT)

SC (SNO,CNO,GRADE)

C (CNO,CNAME,CDEPT,TNAME)

试用关系代数表达式表示下列查询语句：

（1）检索 LIU 老师所授课程的课程号、课程名。

答: $\pi_{CNO,CNAME}(\sigma_{TNAME='LIU'}(C))$

（2）检索年龄大于 23 岁的男学生的学号与姓名。

答: $\pi_{SNO,SNAME}(\sigma_{AGE>'23'\ \wedge SEX='男'}(S))$

(3)检索学号为 S3 学生所学课程的课程名与任课教师名。

答： $\pi_{CNAME,\ TNAME}(\sigma_{SNO='S3'}(SC \bowtie C))$

（4）检索至少选修 LIU 老师所授课程中一门课的女学生姓名。

答: $\pi_{SNAME}(\sigma_{SEX='女'\ \wedge TNAME='LIU'}(S \bowtie SC \bowtie C))$

（5）检索 WANG 同学不学的课程的课程号。

答: $\pi_{CNO}(C)-\pi_{CNO}(\sigma_{SNAME='WANG'}(S \bowtie SC))$

(6）检索至少选修两门课程的学生学号。

答： $\pi_1(\sigma_{2\neq5\wedge1=4}(SC\times SC))$

(7)检索全部学生都选修的课程的课程号与课程名。

答： $\pi_{CNO,\ CNAME}(C \bowtie \pi_{SNO,CNO}(SC)\div\pi_{SNO}(S))$

(8)检索选修课程包含 LIU 老师所授课程的学生学号。

答: $\pi_{SNO,CNO}(SC)\div\pi_{CNO}(\sigma_{TNAME='LIU'}(C))$

3.13 试用元组表达式表示 3.12 题中各个查询语句。

答：

⑴ $\{ t \mid (\exists u)\ (C(u)\wedge u[3]='LIU'\wedge\ t[1]=u[1]\wedge t[2]=u[2])\}$

⑵ $\{ t \mid (\exists u)\ (S(u)\ \wedge u[3]>'23'\wedge\ u[4]='男'\wedge t[1]=u[1]\wedge t[2]=u[2])\}$

⑶ $\{ t \mid (\exists u)\ (\exists v)\ (SC(u)\wedge C(v)\wedge u[1]='S3'\wedge\ u[2]=v[1]\ \wedge t[1]=v[2]\wedge t[2]=v[3])\}$

（此处自然联接条件 u[2]=v[1]不要遗漏）

⑷$\{t \mid (\exists u)\ (\exists v)\ (\exists w)\ (S(u)\wedge\ SC(v)\wedge C(w)\wedge w[3]='LIU'\wedge u[4]='F'\wedge u[1]=v[1]\wedge v[2]=w[1]\wedge t[1]=u[2])\}$

（此处自然联接条件 u[1]=v[1]和 v[2]=w[1]不要遗漏）

⑸$\{t \mid (\exists u)\ (\exists v)\ (\forall w)\ (C(u)\wedge S(v)\wedge SC(w)\wedge v[2]='WANG'\wedge(w[1]=v[1]\ \Rightarrow\ w[2]\neq u[1])\wedge t[1]=u[1])\}$

其意思是：在关系 C 中存在一门课程，在关系 S 中存在一个 WANG 同学，在关系 SC 中要求不存在 WANG 同学学这门课程的元组。也就是要求在关系 SC 中，WANG 同学的课程都不是这门课程（因此在元组表达式中要求全称量词∀）。

⑹$\{ t \mid (\exists u)\ (\exists v)\ (SC(u)\wedge SC(v)\wedge u[1]=v[1]\wedge u[2]\neq v[2]\wedge t[1]=u[1])\}$

⑺$\{ t \mid (\exists u)\ (\forall v)\ (\exists w)\ (C(u)\wedge S(v)\wedge SC(w)\wedge w[2]=u[1]\wedge w[1]=v[1]\wedge t[1]=u[1]\wedge t[2]=u[2])\}$

其意思是：在关系 C 中找一课程号，对于关系 S 中每一个学生，都应该学这门课（即在关系 SC 中存在这个学生选修这门课的元组）。

⑻$\{t|(\exists u)(SC(u)\wedge(\forall v)\ (C(v)\wedge(v[3]='LIU'\ \Rightarrow\ (\exists w)\ (SC(w)\wedge w[1]=u[1]\wedge w[2]=v[1])))\wedge t[1]=u[1])\}$

其意思是：在关系 SC 中找一个学号，对于关系 C 中 LIU 老师的每一门课，这个学生都学了（即在关系 SC 中存在这个学生选修这门课的元组）。

由于在括号中出现“=>”符号（包含有“∨”的语义），因此括号中的量词（∃w）就不能随意往左边提了。

3.14 试用域表达式表示第 3.12 题的各个查询语句。

答：

①$\{t_1\ t_2|(\exists u_1u_2u_3)(C(u_1u_2u_3)\wedge u_3='LIU'\wedge t_1=u_1\ \wedge t_2=u_2)\}$

再简化成：$\{t_1t_2\ |C(t_1t_2'LIU')\}$

此处$(\exists u_1u_2u_3)$是$(\exists u_1)(\exists u_2)(\exists u_3)$ 的简写，下同。

②$\{t_1t_2|(\exists u_1u_2u_3u_4)\ (S(u_1u_2u_3u_4)\wedge\ u_3>'23'\wedge\ u_4='男'\wedge t_1=u_1\wedge t_2=u_2)\}$

再简化成：$\{t_1t_2|(\exists u_3)(S(t_1t_2u_3'M')\wedge\ u_3>'23')\}$（以下各题的化简略）

③ $\{ t_1\ t_2 \mid (\exists u_1\ u_2\ u_3)\ (\exists v_1\ v_2\ v_3)\ (SC(u_1\ u_2\ u_3)\ \wedge\ C(v_1\ v_2\ v_3)\ \wedge\ u_1='s3'\ \wedge\ u_2=v_1\wedge\ t_1=v_2\ \wedge\ t_2=v_3)\}$

④ $\{ t_1 \mid (\exists u_1\ u_2\ u_3\ u_4)\ (\exists v_1\ v_2\ v_3)\ (\exists w_1\ w_2\ w_3)\ (S(u_1\ u_2\ u_3\ u_4)\ \wedge\ SC(v_1\ v_2\ v_3)\wedge\ C(w_1\ w_2\ w_3)\ \wedge\ w_3='LIU'\ \wedge\ u_4='F'\ \wedge\ u_1=v_1\ \wedge\ v_2=w_1\ \wedge\ t_2=u_2)\}$

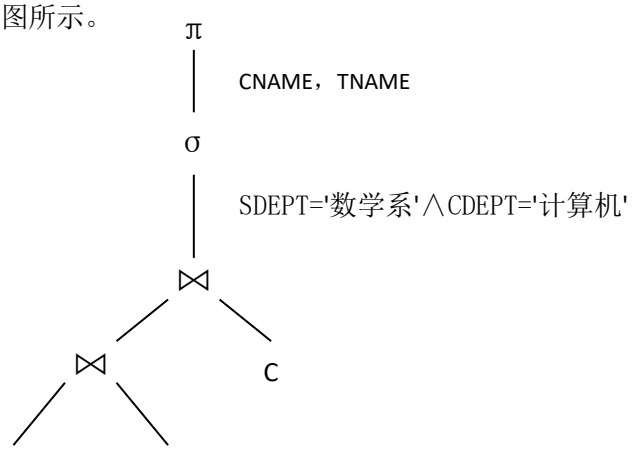
3.15 在 3.12 题的三个关系中，用户有一查询语句：检索数学系的学生选修计算机课程的课程

名和任课教师姓名。①试写出该查询的关系代数表达式；②画出该查询初始的关系代数表达式的语法树；③使用 3.4.4 节的优化算法，对语法树进行优化，试写出该查询优化的关系代数表达式；④画出优化后的语法树。

答：① 关系代数表达式为：检索数学系的学生选修计算机课程的课程名和任课教师姓名

$\pi_{CNAME,\ TNAME}(\sigma_{SDEPT='数学系' \wedge CDEPT='计算机'}(S \bowtie SC \bowtie C))$

②上述关系代数表达式的语法树如图所示。



③ 上述的关系代数表达式为：

$\pi_{CNAME,\ TNAME}(\sigma_{SDEPT='数学系' \wedge CDEPT='计算机'}(\pi_{L}(\sigma_{S.SNO=SC.SNO \ \wedge \ SC.CNO=C.CNO}((S \times SC) \times C))))$

此处 L 为 S、SC、C 中全部属性（公共属性只取一次）。

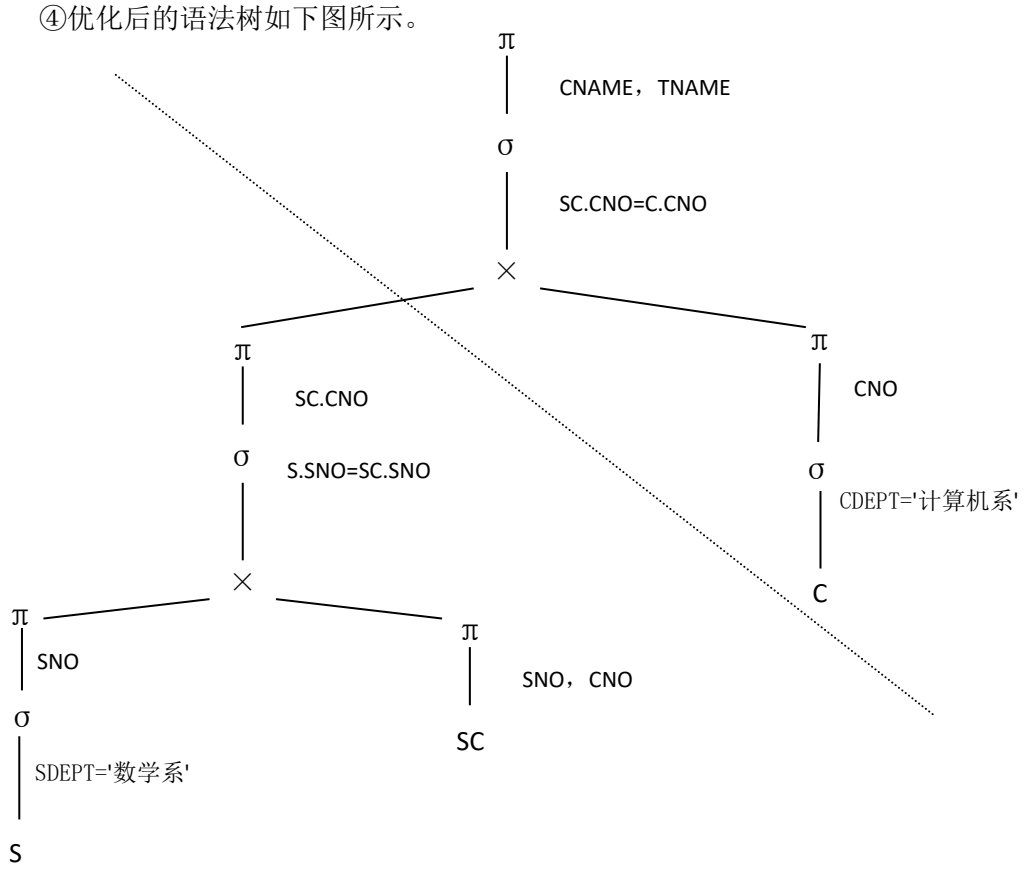
设 $L1=\pi_{SNO}(\sigma_{SDEPT='数学系'}(S))$

$L2=\pi_{SNO,CNO}(SC)$

则优化的关系代数表达式为：

$\pi_{CNAME,\ TNAME}(\sigma_{SC.CNO=C.CNO}(\pi_{SC.C\#}(\sigma_{S.SNO=SC.SNO}(L1 \times L2)) \times C))$

④优化后的语法树如下图所示。



3.16 为什么要对关系代数表达式进行优化？有哪三条启发式规则？对优化起什么作用？
答：关系代数表达式由关系代数操作组合而成。操作中，以笛卡尔积和联接操作最费时，并生成大量的中间结果。如果直接按表达式书写的顺序执行，必将花费很多时间，并生成大量的中间结果，效率较低。在执行前，由 DBMS 的查询子系统先对关系代数表达式进行优化，尽可能先执行选择和投影操作，以便减少中间结果，并节省时间。

优化工作是由 DBMS 做的，用户书写时不必关心优化一事，仍以简练的形式书写。
三条启发式规则是：尽可能早执行选择操作；尽可能早执行投影操作；把笛卡尔积与附近的一连串选择和投影合并起来做。

使用这三条规则，可以使计算时尽可能减少中间关系的数据量。

4.1 答：名词解释
• 基本表：实际存储在数据库中的表，称为基本表。
• 视图：是从基本表或其他视图中导出的表，它本身不独立存储在数据库中，也就是数据库中只存放视图的定义而不存放视图的数据。
• 实表：是对基本表的别称。
• 虚表：是对视图的别称。
• 相关子查询：SELECT 语句嵌套时，子查询中查询条件依赖于外层查询中的值，因此子查询要反复求值供外层查询使用。这种子查询称为相关子查询。
•联接查询：查询时要从多个基本表中提取数据，此时把多个基本表写在同一层的 FROM 子句中，这种查询形式称为联接查询。
• 嵌套查询：查询时要从多个基本表中提取数据，此时把多个基本表分别放在不同层次上的 FROM 子句中，这种查询形式称为嵌套查询。
• 交互式 SQL：在终端交互方式使用的 SQL 语言。
• 嵌入式 SQL：嵌入在高级语言的程序中使用的 SQL 语言。
• 共享变量：嵌入的 SQL 语句和主语言语句间传递信息的变量，称为共享变量。共享变量先由主语言程序定义，再用 SQL 的说明语句说明，然后 SQL 语句就可使用这些变量。
• 游标：游标是与某一查询相联系的符号名。游标有游标关系和游标指针两层含义。在游标打开时，游标（指针）指向查询结果的第一个记录之前。
• 卷游标：在游标推进时，可以进退自如的游标。

4.2 S （SNO,SNAME，AGE， SEX,SDEPT）
SC（SNO，CNO,GRADE）
C （CNO，CNAME,CDEPT,TNAME）
试用 SQL 的查询语句表达下列查询：
（1）检索 LIU 老师所授课程的课程号和课程名。
答：SELECT CNO,CNAME
FROM C
WHERE TNAME='LIU';
（2）检索年龄大于 23 岁的男学生的学号和姓名。
答：SELECT SNO,SNAME FROM S
WHERE AGE>23 AND SEX='男';
（3）检索学号为 S3 的学生所学课程的课程名和任课教师名。
答：SELECT CNAME,TNAME FROM SC,C
WHERE SC.CNO=C.CNO AND CNO='S3';

（4）检索至少选修 LIU 老师所授课程中一门课程的女学生姓名。

答：解法一：使用联接查询方式

```
SELECT SNAME FROM S,SC,C
WHERE S.SNO=SC.SNO
  AND SC.CNO=C.CNO
  AND SEX='女'
  AND TNAME='LIU';
```

解法二：使用嵌套查询方式

```
SELECT SNAME FROM S
WHERE SEX='女'
   AND SNO IN
    (SELECT SNO FROM SC
     WHERE CNO IN
       (SELECT CNO FROM C
        WHERE TNAME='LIU'));
```

解法三：使用存在量词查询方式

```
SELECT SNAME FROM S
WHERE SEX='女'
AND EXISTS
(SELECT * FROM SC
 WHERE S.SNO=SC.SNO
  AND  EXISTS
   (SELECT * FROM C
    WHERE SC.CNO=C.CNO
     AND TNAME='LIU'));
```

（5）检索 WANG 同学不学的课程的课程号。

答:

```
SELECT CNO FROM C
WHERE  EXISTS
  (SELECT * FROM S
   WHERE SNAME='WANG'
    AND NOT EXISTS
     (SELECT * FROM SC
      WHERE SC.SNO=S.SNO
       AND SC.CNO=C.CNO));
```

（6）检索至少选修两门课程的学生学号。

答：

```
SELECT SNO FROM SC
GROUP BY SNO
HAVING COUNT(*)>=2;
```

(7)检索全部学生都选修的课程的课程号与课程名。

答:

```
SELECT CNO,CNAME
FROM C
WHERE NOT EXISTS
 (SELECT * FROM S
```

```
      WHERE  NOT EXISTS
       (SELECT * FROM SC
        WHERE SC.SNO=S.SNO
        AND  SC.CNO=C.CNO));
```

（8）检索选修课程包含 LIU 老师所授课程的学生学号。

答：
```
SELECT DISTINCT SNO FROM SC X
    WHERE NOT EXISTS
     (SELECT * FROM C
      WHERE TNAME='LIU'
      AND NOT EXISTS
       (SELECT * FROM SC Y
        WHERE Y.SNO=X.SNO
        AND Y.CNO=C.CNO));
```

4．6 试用 SQL 查询语句表达下列对教学数据库中三个基本表 S、SC、C 的查询：

（1）统计有学生选修的课程门数。

答：
```
SELECT COUNT(DISTINCT CNO) AS 课程门数
    FROM SC;
```

（2）求选修 C4 课程的学生的平均年龄。

答：
```
SELECT AVG(AGE) AS 平均年龄                    ----AGE 为 INT 型
    FROM S,SC
    WHERE S.SNO=SC.SNO
    AND  CNO='C4';
```

或：
```
SELECT AVG(CAST(S.AGE AS INT))  AS 平均年龄  ----AGE 为字符型
    FROM S,SC
    WHERE S.SNO=SC.SNO
    AND  CNO='C4';
```

（3）求 LIU 老师所授课程的每门课程的学生平均成绩。

答：
```
SELECT SC.CNO,AVG(GRADE) AS 平均成绩
    FROM SC,C
    WHERE SC.CNO=C.CNO
    AND TNAME='LIU'
    GROUP BY SC.CNO;
```

或：
```
SELECT SC.CNO,CAST(AVG(GRADE) AS numeric(4,2)) AS 平均成绩
    FROM SC,C
    WHERE SC.CNO=C.CNO
    AND TNAME='LIU'
    GROUP BY SC.CNO;
```

（4）统计每门课程的学生选修人数（超 10 人的课程才统计）。要求输出课程号和选修人数，查询结果按人数降序排列，若人数相同，按课程号升序排列。

答：
```
SELECT SC.CNO,COUNT(SNO) AS 选修人数
    FROM SC,C
    WHERE SC.CNO=C.CNO
    GROUP BY SC.CNO HAVING COUNT(*)>2
    ORDER BY 2 DESC,SC.CNO ASC;
```

（5）检索学号比 WANG 同学大，而年龄比他小的学生姓名。

答：
```
SELECT X.SNAME
FROM S X, S Y
WHERE  Y.SNAME = 'WANG'
AND  X.SNO>Y.SNO AND X.AGE<Y.AGE
```
或：
```
SELECT X.SNAME
FROM S X INNER JOIN
S Y ON X.SNO > Y.SNO AND X.AGE < Y.AGE
WHERE (Y.SNAME = 'WANG');
```
或：
```
SELECT SNAME FROM S
WHERE SNO>(SELECT SNO FROM S
            WHERE   SNAME = 'WANG')
 AND  AGE<(SELECT AGE FROM S
          WHERE   SNAME = 'WANG');
```
或：
```
SELECT X.SNAME FROM S X
WHERE X.SNO>SOME(SELECT SNO FROM S Y
                  WHERE  Y.SNAME = 'WANG'
                  AND  X.AGE<Y.AGE)
```
或：
```
SELECT SNAME FROM S X
WHERE (SNO>ANY(SELECT SNO  FROM S Y
        WHERE Y.SNAME='WANG' AND X.AGE<Y.AGE));
```
或：
```
SELECT SNAME FROM S
WHERE SNO>(SELECT SNO FROM S
            WHERE   SNAME = 'WANG')
AND  AGE<(SELECT AGE FROM S
            WHERE   SNAME = 'WANG')
```
(6)在表 SC 中检索成绩为空值的学生学号和课程号。

答：
```
SELECT SNO, CNO
FROM SC
WHERE GRADE IS NULL;
```

(7)检索姓名以 WANG 打头的所有学生的姓名和年龄。

答：
```
SELECT SNAME, AGE
FROM S
WHERE SNAME LIKE 'WANG%';
```
(8)求年龄大于女同学平均年龄的男学生姓名和年龄。

答：
```
SELECT SNAME, AGE  FROM S
WHERE SEX='男'
    AND AGE>(SELECT AVG(AGE)          -----AGE 为 int 型
    FROM S
    WHERE SEX='女 ');
```

或：

```
SELECT *
FROM S
WHERE SEX='男'
    AND AGE>(SELECT AVG(CAST(S.AGE AS INT))    ----AGE为字符型
             FROM S
             WHERE SEX='女');
```

(9)求年龄大于所有女同学年龄的男学生姓名和年龄。

答：

```
SELECT SNAME, AGE
 FROM S
 WHERE SEX='男'
  AND AGE>ALL(SELECT AGE
                FROM S
                WHERE SEX='女 ');
```

或：

```
  SELECT SNAME, AGE
  FROM S X
WHERE SEX='男'
  AND NOT EXISTS (SELECT *
                    FROM S Y
                    WHERE Y.SEX='女' AND Y.AGE>=X.AGE)
```

4．7 试用 SQL 更新语句表达对教学数据库中三个基本表 S、SC、C 的各个更新操作：

（1）往基本表 S 中插入一个学生元组（'S9'，'WU'，18）。

答：

```
INSERT INTO S(SNO,SNAME,AGE)
   VALUES('S9','WU',18);
```

（2）在基本表 S 中检索每一门课程成绩都大于等于 80 分的学生学号、姓名和性别，并把检索到的值送往另一个已存在的基本表 STUDENT（SNO，SNAME，SEX）。

答：解法一： 有两种情况：

1、假设每个学生都选课

```
INSERT   INTO    STUDENT
SELECT   SNO,SNAME,SEX
FROM     S
WHERE    80<= ALL (SELECT   GRADE
                   FROM    SC
                   WHERE SC.SNO=S.SNO);
```

2、不是每个学生都选课

```
INSERT   INTO    STUDENT
SELECT DISTINCT(S.SNO),SNAME,SEX
FROM S, SC X
WHERE S.SNO = X.SNO
   AND (80 <= ALL  (SELECT GRADE
                    FROM SC Y
                    WHERE Y.SNO = S.SNO));
```

解法二：

```
SELECT DISTINCT S.SNO, S.SNAME, S.SEX
 FROM S INNER JOIN
      SC ON S.SNO = SC.SNO
GROUP BY S.SNO, S.SNAME, S.SEX
HAVING (MIN(SC.GRADE) >= 80);
```

解法三:

```
SELECT DISTINCT S.SNO,SNAME,SEX
 FROM S,SC X
 WHERE S.SNO=X.SNO AND NOT EXISTS
 (SELECT *
   FROM SC Y
     WHERE GRADE<80
      AND S.SNO=Y.SNO);
```

解法四:有两种情况：

1、假设每个学生都选课

```
INSERT   INTO    STUDENT
SELECT SNO,SNAME,SEX
FROM S
WHERE NOT EXISTS
      (SELECT DISTINCT CNO
       FROM SC AS X
       WHERE S.SNO = X.SNO AND NOT EXISTS
            (SELECT *
             FROM SC AS Y
             WHERE S.SNO = Y.SNO
             AND   X.CNO = Y.CNO
             AND  Y.GRADE >= 80));
```

2、不是每个学生都选课：

```
INSERT   INTO    STUDENT
  SELECT DISTINCT S.SNO,S.SNAME,S.SEX
   FROM S,SC
   WHERE  S.SNO = SC.SNO
     AND  ( NOT  EXISTS
              (SELECT  DISTINCT CNO
               FROM  SC  AS  X
               WHERE S.SNO = X.SNO
                  AND NOT EXISTS
                    (SELECT *
                     FROM SC AS Y
                      WHERE S.SNO = Y.SNO
                       AND X.CNO = Y.CNO
                         AND Y.GRADE >= 80)));
```

(3）在基本表 SC 中删除尚无成绩的选课元组。

答：DELETE FROM SC
    WHERE GRADE IS NULL;

（4）把 WANG 同学的选课成绩全部删去。

答：DELETE FROM SC
    WHERE SNO IN
     (SELECT SNO FROM S
        WHERE SNAME='WANG');

（5）把选修 MATHS 课不及格的成绩 全改为空值。

答：UPDATE SC
     SET GRADE = NULL
     WHERE GRADE<60 AND
       CNO = (SELECT CNO FROM C
              WHERE CNAME='MATHS');

(6)把低于总平均成绩的女同学成绩提高 5%。

答：UPDATE SC
    SET GRADE=GRADE*1.05
    WHERE SNO IN(SELECT SNO FROM S WHERE SEX='女')
    AND GRADE<(SELECT AVG(GRADE) FROM SC);

(7)在基本表 SC 中修改 C4 课程的成绩，若成绩小于等于 75 分时提高 5%，若成绩大于 75 分时提高 4%（用两个 UPDATE 语句实现,顺序不能颠倒）。

答：用两个 UPDATE 语句实现：

```
UPDATE SC
  SET GRADE=GRADE*1.04
  WHERE CNO='C4' AND GRADE>70;
UPDATE SC
  SET GRADE=GRADE*1.05
  WHERE CNO='C4' AND GRADE<=70;
```

4.9 对于教学数据库中基本表 SC，建立一个视图：

```
CREATE VIEW S_GRADE(SNO,C_NUM,AVG_GRADE)
AS SELECT SNO,COUNT(CNO),AVG(GRADE)
    FROM SC
    GROUP BY SNO
```

试判断下列查询和更新操作是否允许执行。如允许，写出转换到基本表 SC 上的相应操作。

①SELECT * FROM S_GRADE;

答：允许查询。相应的操作如下：

```
SELECT SNO,COUNT(CNO) AS C_NUM,AVG(GRADE) AS AVG_GRADE
FROM SC
GROUP BY SNO;
```

②SELECT SNO,C_NUM
  FROM S_GRADE WHERE AVG_GRADE>80;

答：允许查询。相应的操作如下：

```
SELECT SNO,COUNT(CNO) AS C_NUM
FROM SC
```

```
GROUP BY SNO
HAVING AVG(GRADE)> 80;
③SELECT SNO,AVG_GRADE
   FROM S_GRADE
   WHERE C_NUM >(SELECT C_NUM
                 FROM S_GRADE
                 WHERE SNO='S4');
```
答：允许查询。相应的操作如下：
```
SELECT SNO,AVG(GRADE) AS AVG_GRADE
FROM SC
GROUP BY SNO
HAVING COUNT(CNO)>(SELECT COUNT(CNO)
                   FROM SC
                   GROUP BY SNO
                   HAVING SNO='S4');
```
等价于:
```
SELECT SNO,AVG(GRADE) AS AVG_GRADE
FROM SC
GROUP BY SNO
HAVING COUNT(CNO)>(SELECT COUNT(CNO)
                   FROM SC
                   WHERE SNO='S4'
                   GROUP BY SNO);
④UPDATE S_GRADE
  SET SNO='S3'
  WHERE SNO='S4';
```
答:不允许。C_NUM 是对 SC 中的学生选修门数进行统计,在未更改 SC 表时,要在视图 S_GRADE 中更改门数，是不可能的。
```
⑤DELETE FROM S_GRADE
WHERE C_NUM>4;
```
答：不允许。在视图 S_GRADE 中删除选修门数在 4 门 以上的学生元组，势必造成 SC 中这些学生学习元组的删除，这不一定是用户的原意，因此使用分组和聚合操作的视图，不允许用户执行更新操作。

4.10 预处理方式对于嵌入式 SQL 的实现有什么重要意义？

答：此时宿主语言的编译程序不必改动，只要提供一个 SQL 函数定义库，供编译时使用。预处理方式只是把源程序中的 SQL 语句处理成宿主语言的函数调用形式。

4.11 在宿主语言的程序中使用 SQL 语句有哪些规定？

答：有三条规定：

① 在程序中要区分 SQL 语句与宿主语言语句，所有 SQL 语句必须加前缀标识“EXEC SQL”以及结束标志“END_EXEC”;

② 允许嵌入的 SQL 语句引用宿主语言的程序变量,而主语句不能引用数据库中的字段变量;

③ SQL 的集合处理方式与宿主语言的单记录处理方式之间要用游标机制协调。

4.12 SQL 的集合处理方式与宿主语言单记录处理方式之间如何协调？

答：用游标机制协调。把 SELECT 语句查询结果定义成游标关系，以使用文件的方式来使用

游标关系。与游标有关的 SQL 语句有四个：游标定义，游标打开，游标推进，游标关闭。

4.13 嵌入式 SQL 的 DML 语句何时不必涉及到游标？何时必须涉及到游标？

答：不涉及游标的 DML 语句有下面两种情况：

① INSERT、DELETE、UPDATE 语句，只要加上前缀和结束标志，就能嵌入在宿主语言程序中使用；

② 对于 SELECT 语句，如果已知查询结果肯定是单元值，也可不必涉及游标操作。

涉及游标的 DML 语句有下面两种情况：

① 当 SELECT 语句查询结果是多个元组时，必须用游标机制把多个元组一次一个地传递给主程序处理；

② 对游标指向元组进行修改或删除操作时，也涉及到游标。

5.1 答：名词解释

• 函数依赖（FD）：在关系模式 R(U)中，FD 是形为 X→Y 的一个命题，只要 r 是 R 的当前关系，对 r 中任意两个元组 t 和 s，都有 t[X]=s[X]蕴涵 t[Y]=s[Y]，那么称 FD X→Y 在关系模式 R(U)中成立。

• 函数依赖 FD 的逻辑蕴涵：如果从已知的 FD 集 F 能推导出 X→Y 成立，那么称 F 逻辑蕴涵 X→Y，记为 $F \models X \rightarrow Y$。

• 平凡的 FD：如果 X→Y，且 Y⊆X，则称 X→Y 是一个“平凡的 FD”。

• FD 集 F 的闭包 F^+：被 F 逻辑蕴涵的函数依赖全体构成的集合，称为 F 的闭包，记为 F^+，即 $F^+=\{ X \rightarrow Y \mid F \models X \rightarrow Y\}$。

• 属性集 X 的闭包 X^+：从已知的 FD 集 F 使用 FD 推理规则推出的所有满足 X→A 的属性 A 的集合，称为 X 的闭包，记为 X^+，即 $X^+=$ { 属性 A | X→A 在 F^+中 }。

• FD 集的等价：对于两个 FD 集 F 和 G，有 $F^+=G^+$，则称 F 和 G 是等价的依赖集。

• 最小依赖集：设 F 是属性集 U 上的 FD 集，F_{min} 是 F 的最小依赖集，那么 F_{min} 应满足下列四个条件：$F_{min}^+=F^+$；每个 FD 的右边都是单属性；F_{min} 中没有冗余的 FD；每个 FD 的左边没有冗余的属性。

• 无损分解：设关系模式 R，F 是 R 上的 FD 集，$\rho = \{ R_1, \cdots, R_k \}$ 是 R 的一个分解。如果对 R 中满足 F 的每一关系 r，都有 $r = \bowtie_{i=1}^{k} \pi_{R_i}(r)$，那么称分解 ρ 相对 F 是“无损分解”。

• 泛关系假设：指数据库中每一个关系都是全部属性构成的关系的投影，此时，由全部属性构成的关系称为泛关系。

• chase 过程：根据已知 FD 集，对 R 分解成 ρ 构造的初始表格的值进行修改，使之符合 FD 集，这个过程称为 chase 过程。

• 保持 FD：设关系模式 R，F 是 R 上的 FD 分解，$\rho = \{ R_1, \cdots, R_k \}$ 是 R 的一个分解，如果有 $\bigcup_{i=1}^{k} \pi_{R_i}(F) \models F$，那么称分解 ρ 保持 FD 集 F。

• 1NF：如果关系模式 R 的每个关系 r 的属性值都是不可分的原子值，那么称 R 是 1NF 的模式。

• 2NF：如果 R 是 1NF 的模式，且每个非主属性完全函数依赖于 R 的候选键，那么称 R 是 2NF 的模式。

• 3NF：如果 R 是 1NF 的模式，且每个非主属性都不传递依赖于 R 的候选键，那么称 R 是 3NF 的模式。

• BCNF：如果 R 是 1NF 的模式，且每个属性都不传递依赖于 R 的候选键，那么称 R 是 BCNF 的模式。

• 4NF：设 D 是关系模式 R 上成立的 FD 和 MVD 集合。如果 D 中每个非平凡的 MVD X→→Y 的左部 X 都是 R 的超键，那么称 R 是 4NF 模式。

• 5NF：如果关系模式 R 的每个 JD 均由 R 的候选键蕴涵，那么称 R 是 5NF 的模式。

• 多值依赖（MVD）：设关系模式 R(U)，X 和 Y 是 U 的子集，Z=U-X-Y。对于 R 的关系 r，若在 r 中存在元组（x，y_1，z_1）和（x，y_2，z_2），就也应存在元组（x，y_2，z_1）和（x，y_1，z_2），那么称 MVD X→→Y 在模式 R 上成立。

• 联接依赖（JD）：设关系模式 R(U)，R_1、…、R_n 是 U 的子集，并满足 $U=R_1 \cup \cdots \cup R_n$，$\rho = \{ R_1, \cdots, R_n \}$ 是 R 的一个分解。如果对于 R 的每个关系 r 都有 $m_\rho(r)=r$，那么称 JD *（R_1，…，R_n）在模式 R 上成立。

5.2 设关系模式 R 有 n 个属性，在模式 R 上可能成立的函数依赖有多少个？其中平凡的 FD 有多少个？非平凡的 FD 有多少个？

答：这个问题是排列组合问题。FD 形为 X→Y，从 n 个属性值中选择属性组成 X 共有 $C_n^0+C_n^1+\cdots+C_n^n=2^n$ 种方法；同理，组成 Y 也有 2^n 种方法。因此组成 X→Y 形式应该有 $2^n \cdot 2^n=4^n$ 种方法。即可能成立的 FD 有 4^n 个。

平凡的 FD 要求 Y⊆X，组合 X→Y 形式的选择有：

$$C_n^0 \cdot C_0^0+C_n^1 \cdot (C_1^0+C_1^1)+C_n^2 \cdot (C_2^0+C_2^1+C_2^2)+\cdots+C_n^n (C_n^0+C_n^1+\cdots C_n^n)$$

$$=C_n^0 \cdot 2^0+C_n^1 \cdot 2^1+C_n^2 \cdot 2^2+\cdots+C_n^n \cdot 2^n=(1+2)^n=3^n$$

即平凡的 FD 有 3^n。因而非平凡的 FD 有 4^n-3^n 个。

5.3 对函数依赖 X→Y 的定义加以扩充，X 和 Y 可以为空属性集，用 Φ 表示，那么 X→Φ，Φ→Y，Φ→Φ 的含义是什么？

答：据推理规则的自反律可知，X→Φ 和 Φ→Φ 是平凡的 FD，总是成立的。

而 Φ→Y 表示在当前关系中，任意两个元组的 Y 值相等，也就是当前关系的 Y 值都相等。

5.4 已知关系模式 R（ABC），F 是 R 上成立的 FD 集，F= { A→B，B→C }，试写出 F 的闭包 F^+。

答：据已知条件和推理规则，可知 F^+有 43 个 FD：

A→Φ	AB→Φ	AC→Φ	ABC→Φ	B→Φ	C→Φ
A→A	AB→A	AC→A	ABC→A	B→B	C→C
A→B	AB→B	AC→B	ABC→B	B→C	Φ→Φ
A→C	AB→C	AC→C	ABC→C	B→BC	
A→AB	AB→AB	AC→AB	ABC→AB	BC→Φ	
A→AC	AB→AC	AC→AC	ABC→AC	BC→B	
A→BC	AB→BC	AC→BC	ABC→BC	BC→C	
A→ABC	AB→ABC	AC→ABC	ABC→ABC	BC→BC	

5.5 设关系模式 R（ABCD），如果规定，关系中 B 值与 D 值之间是一对多联系，A 值与 C 值之间是一对一联系。试写出相应的函数依赖。

答：从 B 值与 D 值之间有一对多联系，可写出函数依赖 D→B,从 A 值与 C 值之间是一对一联系。可写出函数依赖 A→C 和 C→A。

5.6 试举出反例说明下列规则不成立：

① { A→B } ⊨ { B→A }

② { AB→C，A→C } ⊨ { B→C }

③ { AB→C } ⊨ { A→C }

答：设有三个关系：

r_1

A	B
1	1
2	1

r_2

A	B	C
2	1	2
2	2	2
3	2	3[1]

r_3

A	B	C
1	2	3
1	3	4

（1）在关系 r_1 中，A→B 成立，但 B→A 不成立。

（2）在关系 r_2 中，AB→C 和 A→C 成立，但 B→C 不成立

（3）在关系 r_3 中，AB→C 成立，但 A→C 不成立。

5.7 设关系模式 R（ABCD），F 是 R 上成立的 FD 集，F= { A→B，C→B }，则相对于 F，试写出关系模式 R 的关键码。并说明理由。

答：R 的关键码为 ACD。因为从已知的 F，只能推出 ACD→ABCD。

5.8 设关系模式 R（ABCD），F 是 R 上成立的 FD 集，F= {A→B，B→C}，

① 试写出属性集 BD 的闭包 $(BD)^+$。

② 试写出所有左部是 B 的函数依赖（即形为“B→？”）。

答：①从已知的 F，可推出 BD→BCD，所以 $(BD)^+=BCD$。

②由于 $B^+=BC$，因此左部是 B 的 FD 有四个：

B→φ，B→B，B→C，B→BC。

5.9 设关系模式 R（ABC）分解成 ρ = { AB，BC }，如果 R 上的 FD 集 F= { A→B }，那么这个分解是损失分解。试举出 R 的一个关系 r，不满足 $m_\rho(r)=r$。

答：这个反例 r 可以举测试时的初始表格：

	A	B	C
AB	a_1	a_2	b_{13}
BC	b_{21}	a_2	a_3

$\pi_{AB}(r) \bowtie \pi_{BC}(r)$ 有四个元组：

A	B	C
a_1	a_2	b_{13}
a_1	a_2	a_3
b_{21}	a_2	b_{13}
b_{21}	a_2	a_3

即 $m_\rho(r) \neq r$。

5.10 试解释数据库“丢失信息”与“未丢失信息”两个概念。“丢失信息”与“丢失数据”有什么区别？

答：数据库中丢失信息是指 $r \neq m_\rho(r)$，未丢失信息是指 $r=m_\rho(r)$。

丢失信息是指不能辨别元组的真伪，而丢失数据是指丢失元组。

5.11 设关系模式 R（ABC），F 是 R 上成立的 FD 集，F= { A→C，B→C }，试分别求 F 在模式 AB 和 AC 上的投影。

答： $\pi_{AB}(F) = \phi$（即不存在非平凡的 FD）

[1]

π_{AC}（F）={ A→C }

5.12 设关系模式 R（ABC），F 是 R 上成立的 FD 集，F= { B→A，C→A }， ρ = { AB，BC } 是 R 上的一个分解，那么分解 ρ 是否保持 FD 集 F？并说明理由。

答：已知 F={ B→A，C→A }，而 π_{AB}（F）={ B→A }， π_{BC}（F）= ϕ，

显然，分解 ρ 丢失了 FD C→A。

5.13 设关系模式 R（ABC），F 是 R 上成立的 FD 集，F= { B→C，C→A }，那么分解 ρ = { AB，AC } 相对于 F，是否无损分解和保持 FD？并说明理由。

答：①已知 F={ B→C，C→A }，

而 π_{AB}（F）= ϕ， π_{AC}（F）={ C→A }

显然，这个分解丢失了 FD B→C

② 用测试过程可以知道， ρ 相对于 F 是损失分解。

5.14 设关系模式 R（ABCD），F 是 R 上成立的 FD 集，F= { A→B，B→C，A→D，D→C }， ρ = { AB，AC，BD } 是 R 的一个分解。

① 相对于 F， ρ 是无损分解吗？为什么？

② 试求 F 在 ρ 的每个模式上的投影。

③ ρ 保持 F 吗？为什么？

答：①用测试过程可以知道， ρ 相对于 F 是损失分解。

② π_{AB}（F）={ A→B }， π_{AC}（F）={ A→C }， π_{BD}（F）= ϕ。

③显然，分解 ρ 不保持 FD 集 F，丢失了 B→C、A→D 和 D→C 等三个 FD。

5.15 设关系模式 R（ABCD），R 上的 FD 集 F= { A→C，D→C，BD→A}，试说明 ρ = { AB，ACD，BCD } 相对于 F 是损失分解的理由。

答：据已知的 F 集，不可能把初始表格修改为有一个全 a 行的表格，因此 ρ 相对于 F 是损失分解。

5.16 设关系模式 R（ABCD），F 是 R 上成立的 FD 集，F= { AB→CD，A→D }。

① 试说明 R 不是 2NF 模式的理由。

② 试把 R 分解成 2NF 模式集。

答：①从已知 FD 集 F，可知 R 的候选键是 AB。

另外，AB→D 是一个局部依赖，因此 R 不是 2NF 模式。

③ 此时 R 应分解成 ρ ={ AD，ABC }， ρ 是 2NF 模式集。

5.17 设关系模式 R（ABC），F 是 R 上成立的 FD 集，F= { C→B，B→A }。

① 试说明 R 不是 3NF 模式的理由。

② 试把 R 分解成 3NF 模式集。

答：①从已知 FD 集 F，可知 R 的候选键是 C。

从 C→B 和 B→A，可知 C→A 是一个传递依赖，因此 R 不是 3NF 模式。

③ 此时 R 应分解成 ρ ={ CB，BA }， ρ 是 3NF 模式集。

5．18 设有一个记录各个球队队员每场比赛进球数的关系模式

R（队员编号，比赛场次，进球数，球队名，队长名）

如果规定每个队员只能属于一个球队，每个球队只有一个队长。

① 试写出关系模式 R 的基本 FD 和关键码。

② 说明 R 不是 2NF 模式的理由，并把 R 分解成 2NF 模式集。

③ 进而把 R 分解成 3NF 模式集，并说明理由。

答：（1）根据每个队员只能属于一个球队，可写出 FD：队员编号→球队名

根据每个球队只有一个队长，可写出 FD：球队名→队长名

“每个队员每场比赛只有一个进球数”，这条规则也是成立的。因此还可写出 FD：

（队员编号，比赛场次）→进球数

R 的关键码为（队员编号，比赛场次）。

（2）R 中存在这样的 FD：

（队员编号，比赛场次）→（球队名，队长名）

队员编号 →（球队名，队长名）

可见前一个 FD 是局部依赖，所以 R 不是 2NF 模式。

R 应分解成 R1（队员编号，球队名，队长名）

R2（队员编号，比赛场次，进球数）

此处，R1 和 R2 都是 2NF 模式。

（3）R2 已是 3NF 模式。

在 R1（队员编号，球队名，队长名）中，存在两个 FD：

队员编号 → 球队名

球队名 → 队长名

关键码为队员编号，存在传递依赖，因此 R1 不是 3NF 模式。

R1 应分解成 R11（队员编号，球队名）

R12（球队名，队长名）

这两个模式都是 3NF 模式。

因此，R 分解成 3NF 模式集时，ρ = { R11，R12，R2 }。

5.19 设有关系模式 R（职工编号，日期，日营业额，部门名，部门经理），该模式统计商店里每个职工的日营业额，以及职工所在的部门和经理信息。

如果规定：每个职工每天只有一个营业额；每个职工只在一个部门工作；每个部门只有一个经理。

试回答下列问题：

（1）根据上述规定，写出模式 R 的基本 FD 和关键码；

（2）说明 R 不是 2NF 的理由，并把 R 分解成 2NF 模式集；

（3）进而分解成 3NF 模式集。

答：（1）基本的 FD 有三个：

（职工编号，日期）→ 日营业额

职工编号 → 部门名

部门名 → 部门经理

R 的关键码为（职工编号，日期）。

（2）R 中有两个这样的 FD：

（职工编号，日期）→（部门名，部门经理）

职工编号 →（部门名，部门经理）

可见前一个 FD 是局部依赖，所以 R 不是 2NF 模式。

R 应分解成 R1（职工编号，部门名，部门经理）

R2（职工编号，日期，日营业额）

此处，R1 和 R2 都是 2NF 模式。

（3）R2 已是 3NF 模式。

在 R1 中，存在两个 FD：职工编号 → 部门名

部门名 → 部门经理

因此，“职工编号 → 部门经理”是一个传递依赖，R1 不是 3NF 模式。

R1 应分解成 R11（职工编号，部门名）

R12（部门名，部门经理）

这样，ρ = { R11，R12，R2 }是一个 3NF 模式集。

5.20 设关系模式 R（ABC）上有一个 MVD A→→B。如果已知 R 的当前关系存在三个元组（ab_1c_1）、（ab_2c_2）和（ab_3c_3），那么这个关系中至少还应该存在哪些元组？

答：这个关系中至少还应存在下面 6 个元组：（ab_1c_2），（ab_2c_1），（ab_1c_3），（ab_3c_1），（ab_2c_3），（ab_3c_2）。

5.21 试撰写 2000 字短文，论述泛关系假设、无损联接和保持依赖间的联系。

答：这篇短文的要点如下：

（1）“泛关系假设”是在谈论数据库时必须存在泛关系情况下再讨论分解。

（2）谈论无损分解的先决条件是泛关系假设。

（3）谈论保持 FD 时，不提泛关系假设。

（4）无损分解与保持 FD 之间，没有必然的联系。

（5）满足无损分解的数据库，有 $r=m_\rho(r)$ 性质。

（6）满足保持 FD 的数据库，数据的语义值肯定满足 FD。

6.2 设某商业集团数据库中有三个实体集。一是“商店”实体集，属性有商店编号、商店名、地址等；二是“商品”实体集，属性有商品号、商品名、规格、单价等；三是“职工”实体集，属性有职工编号、姓名、性别、业绩等。

商店与商品间存在“销售”联系，每个商店可销售多种商品，每种商品也可放在多个商店销售，每个商店销售一种商品，有月销售量；商店与职工间存在着“聘用”联系，每个商店有许多职工，每个职工只能在一个商店工作，商店聘用职工有聘期和月薪。

（1） 试画出 ER 图，并在图上注明属性、联系的类型。

（2） 将 ER 图转换成关系模型，并注明主键和外键。

答：(1) ER 图如图 6.1 所示。

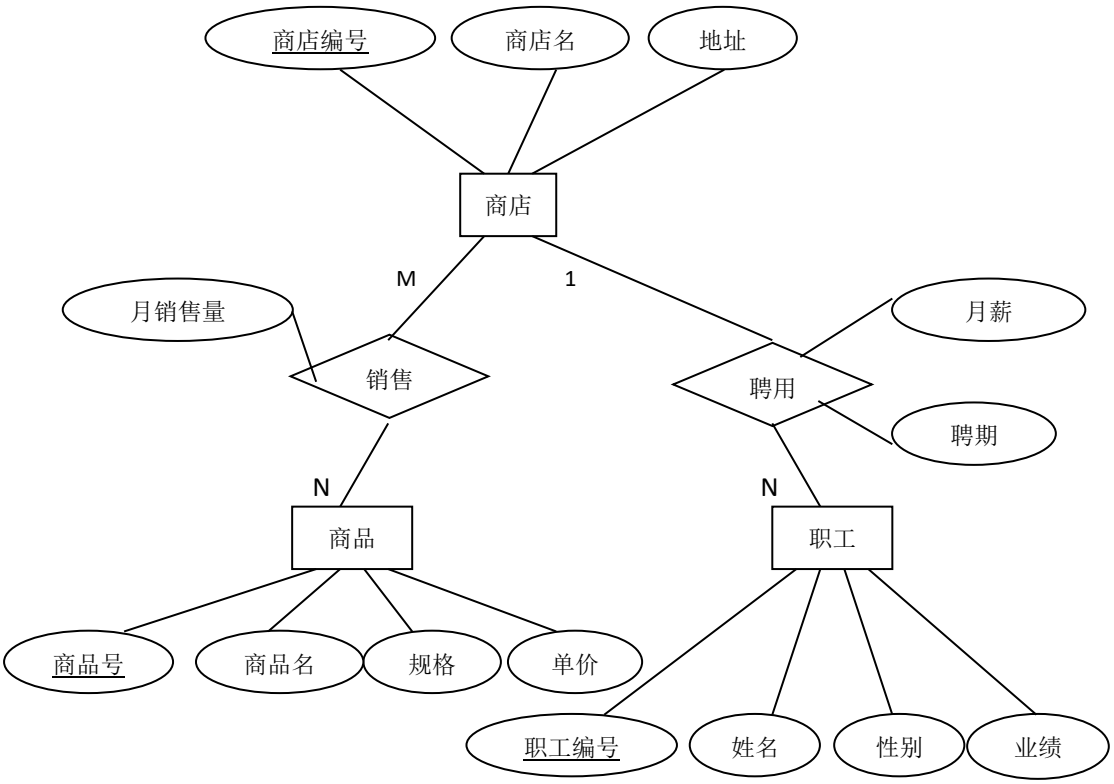


图 6.1

（2）这个 ER 图可转换 4 个关系模式：

商店（商店编号，商店名，地址）

职工（职工编号，姓名，性别，业绩，商店编号，聘期，月薪）

商品（商品号，商品名，规格，单价）

销售（商店编号，商品号，月销售量）

6.3 设某商业集团数据库中有三个实体集。一是“公司”实体集，属性有公司编号、公司名、地址等；二是“仓库”实体集，属性有仓库编号、仓库名、地址等；三是“职工”实体集，属性有职工编号、姓名、性别等。

公司与仓库间存在“隶属”联系，每个公司管辖若干仓库，每个仓库只能属于一个公司管辖；仓库与职工间存在“聘用”联系，每个仓库可聘用多个职工，每个职工只能在一个仓库工作，仓库聘用职工有聘期和工资。

(1) 试画出 ER 图，并在图上注明属性、联系的类型。

(2) 将 ER 图转换成关系模型，并注明主键和外键。

答：(1) ER 图如图 6.2 所示。

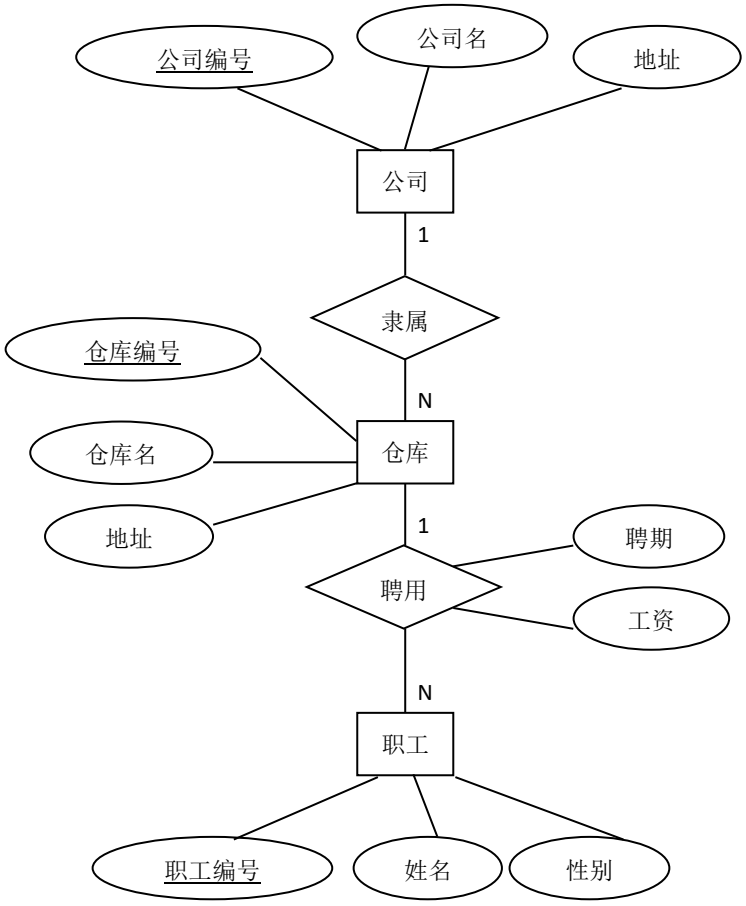


图 6.2

（2）这个 ER 图可转换 3 个关系模式：

公司（公司编号，公司名，地址）

仓库（仓库编号，仓库名，地址，公司编号）

职工（职工编号，姓名，性别，仓库编号，聘期，工资）

6.4 假设要为银行的储蓄业务设计一个数据库，其中涉及到储户、存款、取款等信息。试设计 ER 模型。

答：储蓄业务主要是存款、取款业务，可设计如图 6.3 所示的 ER 图。

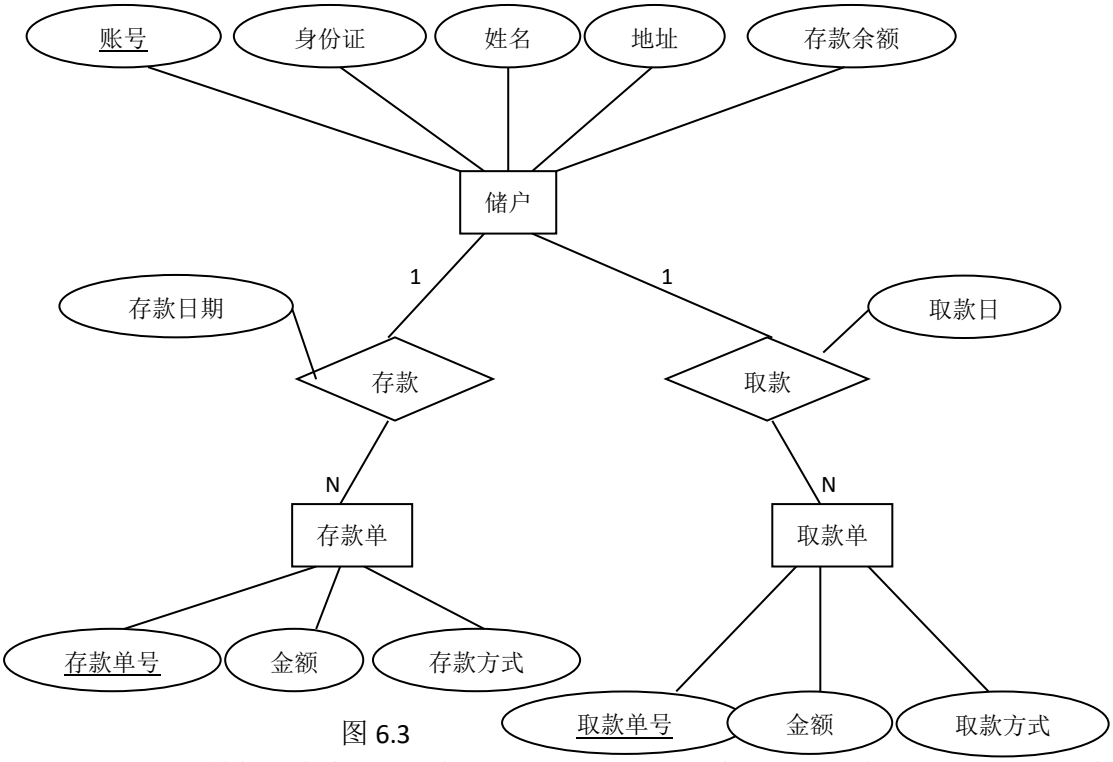


图 6.3

6.5 某体育运动锦标赛有来自世界各国运动员组成的体育代表团参赛各类比赛项目。试为该锦标赛各个代表团、运动员、比赛项目、比赛情况设计一个 ER 模型。

答：图 6.4 是 ER 图的一种设计方案。

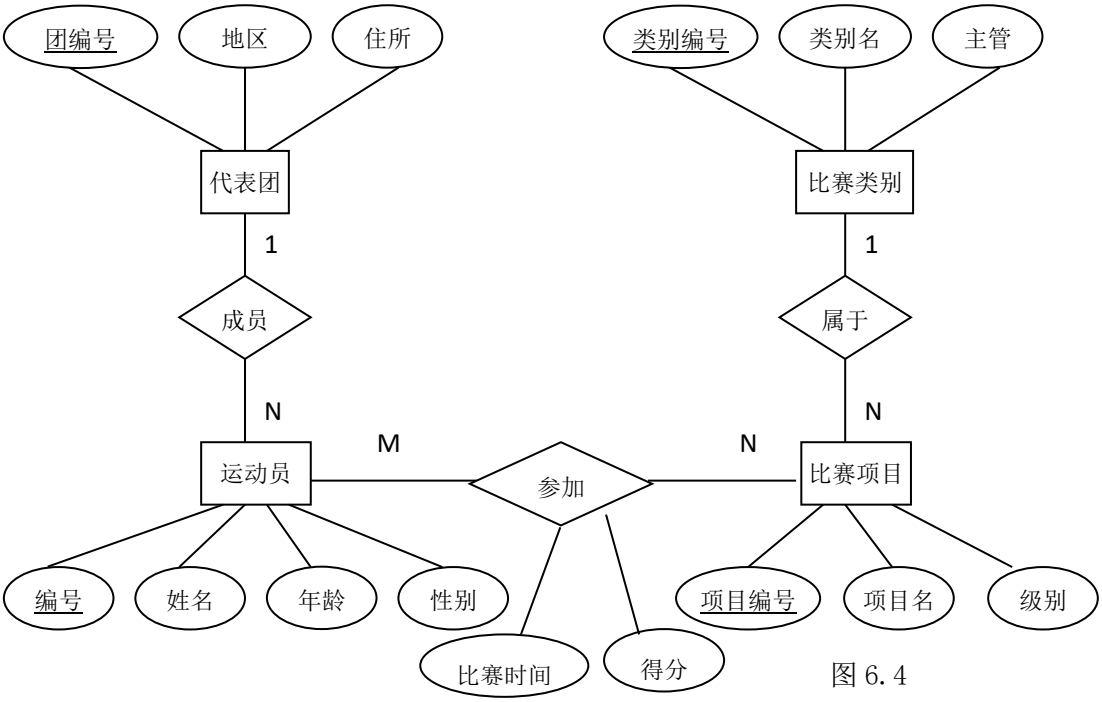


图 6.4

6.6 假设某超市公司要设计一个数据库系统来管理该公司的业务信息。该超市公司的业务管理规则如下:

⑴该超市公司有若干仓库，若干连锁商店，供应若干商品。

⑵每个商店有一个经理和若干收银员，每个收银员只在一个商店工作。

⑶每个商店销售多种商品，每种商品可在不同的商店销售。

⑷每个商品编号只有一个商品名称，但不同的商品编号可以有相同的商品名称。每种商品可以有多种销售价格。

⑸超市公司的业务员负责商品的进货业务。

试按上述规则设计 ER 模型

答：图 6.5 是 ER 图的一种设计方案。

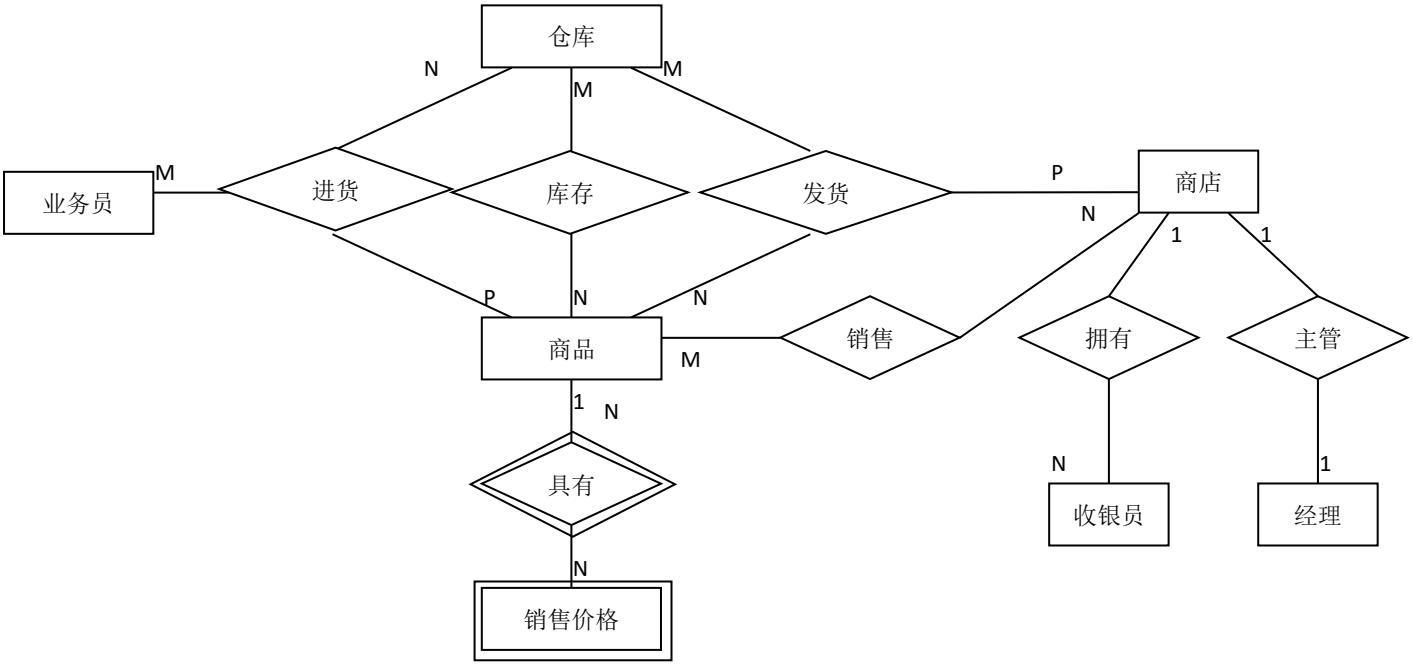


图 6.5

6.7 假设要根据某大学的系、学生、班级、学会等信息建立一个数据库，一个系有若干专业，每个专业每年只招一个班，每个班有若干学生。一个系的学生住在同一宿舍区。每个学生可以参加多个学会，每个学会有若干学生，学生参加某学会有个入会年份。试为该大学的系、学生、班级、学会等信息设计一个 ER 模型。

答：图 6.6 是 ER 图的一种设计方案。

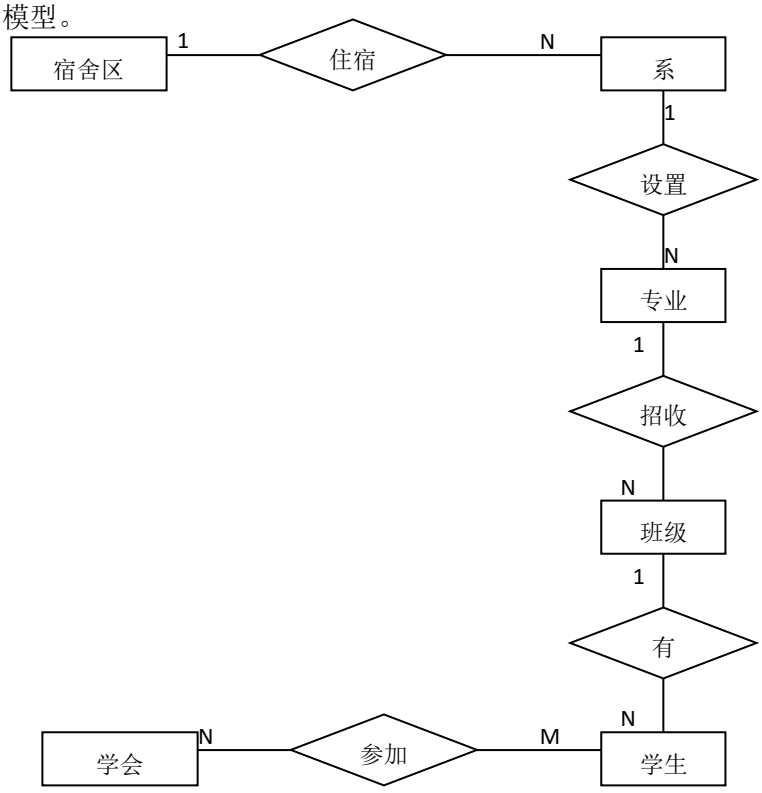


图 6.6

7.2 数据库系统的生存期分成哪几个阶段？数据库结构的设计在生存期中的地位如何？
答：对 DBS 生存期的划分，一般分为七个阶段，即规划、需求分析、概念设计、逻辑设计、物理设计、实现和运行维护。

DB 结构设计的任务就是把概念设计阶段设计好的基本 ER 图转换成与选用的具体机器上的 DBMS 所支持的数据模型相符合的逻辑结构。

7.4 基于数据库系统生存期的数据库设计分成哪几个阶段？
答：基于 DBS 生存期的 DBD 分成以下五个阶段：

规划；需求描述和分析；概念设计；逻辑设计；物理设计。

7.5 数据库设计的规划阶段应做哪些事情？
答：DBD 中规划阶段的主要任务是进行建立 DB 的必要性及可行性分析，确定 DBS 在组织中和信息系统中的地位，以及各个 DB 之间的联系。
7.6 数据库设计的需求分析阶段是如何实现的？目标是什么？
答：需求分析阶段的工作由下面四步组成：

- 分析用户活动，产生用户活动图；
- 确定系统范围，产生系统范围图；
- 分析用户活动所涉及的数据，产生数据流图；
- 分析系统数据，产生数据字典。

需求分析阶段的目标是对系统的整个应用情况作全面的、详细的调查，确定企业组织的目标，收集支持系统总的设计目标的基础数据和对这些数据的要求，确定用户的需求；并把这些要求写成用户和数据库设计者都能接受的文档。
7.10 概念设计的具体步骤是什么？
答：概念设计的主要步走可分为三步：
（1） 进行数据抽象，设计局部概念模式；
（2） 将局部概念模式综合成全局概念模式；
（3） 评审。
7.13 逻辑设计的目的是什么？
答：逻辑设计的目的是把概念设计阶段设计好的基本 ER 图转换成与选用的具体机器上的 DBMS 所支持的数据模型相符合的逻辑结构（包括数据库模式和外模式）。这些模式在功能、性能、完整性和一致性约束及数据库的可扩充性等方面均应满足用户的各种要求。
7.14 试述逻辑设计阶段的主要步骤及内容。
答：逻辑设计阶段主要有五步：形成初始模式，设计子模式，设计应用程序梗概，评价模式和修改模式。
7.15 规范化理论对数据库设计有什么指导意义？
答：规范化理论是数据库设计的指南和工具.具体地讲可在以下三个方面起重要作用:
(1) 在数据库分析阶段用数据依赖的概念来分析和表示各数据项之间的联系;
(2) 在概念设计阶段,用规范化理论消除初步 ER 图中冗余的联系;
(3)在 ER 图向关系模型转换过程中,用模式分解的概念和算法指导设计.
7.16 什么是数据库结构的物理设计？试述其具体步骤。
答：对于给定的基本数据模型选取一个最适合应用环境的物理结构的过程，称为 DB 的物理设计。

物理设计有五步：

确定 DB 的存储记录结构；确定数据存储按排；存取方法的设计；完整性和安全性的设计；应用程序设计。

7.17 数据库实现阶段主要做哪几件事情？

答：数据库实现阶段主要有以下三项工作：

建立实际 DB 结构；装入试验数据调试应用程序；装入实际数据进入试运行状态。

7.18 数据库系统投入运行后，有哪些维护工作？

答：DBS 投入运行以后，就进入运行维护阶段。其主要工作有四项：

维护 DB 的安全性与完整性及系统的转储和恢复；

DB 性能的监督、分析与改进；

增加 DB 新功能；

改正运行中发现的系统错误。

8．2 试叙述事务的四个性质，并解释每一个性质由 DBMS 的哪个子系统实现？每一个性质对 DBS 有什么益处？

答：① 事务的原子性，是指一个事务对 DB 的所有操作，是一个不可分割的工作单元。原子性是由 DBMS 的事务管理子系统实现的。事务的原子性保证了 DBS 的完整性。

② 事务的一致性，是指数据不会因事务的执行而遭受破坏。事务的一致性是由 DBMS 的完整性子系统实现的。事务的一致性保证数据库的完整性。

③ 事务的隔离性，是指事务的并发执行与这些事务单独执行时结果一样。事务的隔离性是由 DBMS 的并发控制子系统实现的。隔离性使并发执行的事务不必关心其他事务，如同在单用户环境下执行一样。

④ 事务的持久性，是指事务对 DB 的更新应永久地反映在 DB 中。持久性是由 DBMS 的恢复管理子系统实现的。持久性能保证 DB 具有可恢复性。

8．3 事务的 COMMIT 语句和 ROLLBACK 语句各做什么事情？

答：COMMIT 语句表示事务执行成功地结束（提交），此时告诉系统，DB 要进入一个新的正确状态，该事务对 DB 的所有更新都已交付实施（写入磁盘）。

ROLLBACK 语句表示事务执行不成功地结束（应该“回退”），此时告诉系统，已发生错误，DB 可能处在不正确的状态，该事务对 DB 的所有更新必须被撤消，DB 应恢复该事务到初始状态。

8．5“检查点机制”的主要思想是什么？COMMIT 语句与检查点时刻的操作如何协调？

答：“检查点机制”的主要思想是在检查点时刻才真正做到把对 DB 的修改写到磁盘。在 DB 恢复时，只有那些在最后一个检查点到故障点之间还在执行的事务才需要恢复。

事务在 COMMIT 时，事务对 DB 的更新已提交，但对 DB 的更新可能还留在内存的缓冲区，在检查点时刻才真正写到磁盘。因此事务的真正结束是在 COMMIT 后还要加上遇到检查点时刻。

8．6 什么是 UNDO 操作和 REDO 操作？为什么要这样设置？

答：UNDO 和 REDO 是系统内部命令。

在 DB 恢复时，对于已经 COMMIT 但更新仍停留在缓冲区的事务要执行 REDO（重做）操作，即根据日志内容把该事务对 DB 修改重做一遍。

对于还未结束的事务要执行 UNDO（撤消）操作，即据日志内容把该事务对 DB 已作的修改撤消掉。

设置 UNDO 和 REDO 操作，是为了使数据库具有可恢复性。

8．7 什么是“运行记录优先原则”？其作用是什么？

答：写一个修改到 DB 中和写一个表示这个修改的登记记录到日志文件中是两个不同的操作，后者比前者重要，后者应先做。这就是运行记录优先原则。其作用是保证 DBS 具有可恢复

性。

8．8 数据库恢复的基本原则是什么？具体实现方法是什么？

答：恢复的基本原则是“冗余”，即数据重复存储。

为了做好恢复工作，在平时应做好两件事：定时对 DB 进行备份；建立日志文件，记录事务对 DB 的更新操作。

8．9 数据库的并发操作会带来哪些问题？如何解决？

答：如果不加控制，数据库的并发操作会带来三个问题：丢失更新问题、依赖于未提交更新的问题和不一致分析问题。

解决并发操作带来的问题，可以使用封锁技术和时标技术。

8．10 为什么 DML 可以单独提供解除 S 封锁的命令，而不单独提供解除 X 封锁的命令?

答：为防止由事务的 ROLLBACK 引起丢失更新操作，X 封锁必须保留到事务终点，因此 DML 不提供专门的解除 X 锁的操作，即解除 X 锁的操作合并到事务的终点去做。而在未到事务终点时，执行解除 S 锁的操作，可以增加事务并发操作的程度，但对 DB 不会产生什么错误的影响，因此 DML 可以提供专门的解除 S 锁的的操作，让用户使用。

8．12 死锁的发生是坏事还是好事？试说明理由。如何解除死锁状态？

答：在 DBS 运行时，死锁状态是我们不希望发生的，因此死锁的发生本身是一件坏事。但是坏事可以转换为好事。如果我们不让死锁发生，让事务任意并发做下去，那么有可能破坏中的数据，或使用户读了错误的数据。从这个意义上讲，死锁的发生是一件好事，能防止错误的发生。在发生死锁后，系统的死锁处理机制和恢复程序就能起作用，抽取某个事务作为牺牲品，把它撤消，做 ROLLBACK 操作，使系统有可能摆脱死锁状态，继续运行下去。

8．13 试叙述“串行调度”与“可串行化调度”的区别。

答：如果多个事务依次执行，则称事务串行调度。

如果利用分时的方法，同时处理多个事务，则称为事务的并发调度。如果一个并发调度的结果与某一串行调度执行结果等价，则称这个并发调度是可串行化调度。

8．15 什么是数据库的完整性？DBMS 的完整性子系统的主要功能是什么？

答：DB 中完整性是指数据的正确性、有效性和相容性，防止错误的数据进入 DB。

DBMS 完整性子系统的主要功能有两点：监督事务的执行，并测试是否违反完整性规则；若有违反现象，则采取恰当的操作。

8．16 完整性规则由哪几个部分组成？SQL 中的完整性约束有哪些？

答：完整性规则由三部分组成：触发条件，约束条件和 ELSE 子句。

SQL 中把完整性约束分成三大类：域约束。基本表约束和断言。

8．17 参照完整性规则在 SQL 中可以用哪几种方法实现？删除参照关系的元组时，对依赖关系有哪些影响？修改参照关系的主键值时，对依赖关系有哪些影响？

答：参照完整性规则，在 SQL 中可以用外键子句、检查子句、断言等三种方式实现。

删除参照关系的元组时，对依赖关系的影响可以采取下列三种做法之一：

RESTRICT 方式、CASCADE 方式和 SET NULL 方式。

修改参照关系的主键值时，对依赖关系的影响也可以采取与上述类似的三种做法之一。

8．18 试对 SQL 中检查约束（CHECK 子句）和断言两种完整性约束进行比较，各说明什么对象？何时激活？能保证数据库的一致性吗？

答：检查子句主要用于对属性值、元组值加以限制和约束。断言实际上是一种涉及面广的检查子句，用 CREATE 语句来定义。

这两种约束都是在进行插入或修改时激活，进行检查。

检查子句只在定义它的基本表中有效，而对其他基本表无约束力，因此在与检查子句有关的其他基本表进行修改时，就不能保证这个基本表中检查子句的语义了。

而断言能保证完整性约束彻底实现。

8．19 设教学数据库的关系如下：

S（SNO，SNAME，AGE，SEX）

SC（SNO，CNO，GRADE）

C（CNO，CNAME，TEACHER）

试用多种方法定义下列完整性约束：

⑴ 在关系 S 中插入的学生年龄值应在 16~25 岁之间。

⑵ 在关系 SC 中插入元组时，其 SNO 值和 CNO 值必须分别在 S 和 C 中出现。

⑶ 在关系 C 中删除一个元组时，首先要把关系 SC 中具有同样 CNO 值的元组全部删去。

⑷ 在关系 S 中把某个 SNO 值修改为新值时，必须同时把关系 SC 中那些同样的 SNO 值也修改为新值。

答：这里每个约束用一种方式定义。

（1）用检查子句定义：

```
CHECK（AGE  BETWEEN  16  AND  25）;
```

（2）在关系 SC 的定义中，用外键子句定义：

```
FOREIGN  KEY（SNO） REFERENCES  S（SNO）;
FOREIGN  KEY（CNO） REFERENCES  C（CNO）;
```

（3）在关系 SC 的定义中，用外键子句定义：

```
FOREIGN  KEY（CNO） REFERENCES  C（CNO）
    ON  DELETE  CASCADE;
```

（4）在关系 SC 的定义中，用外键子句定义：

```
FOREIGN  KEY（SNO） REFERENCES  S（SNO）
    ON  UPDATE  CASCADE;
```

8．20 在教学数据库中的关系 S、SC、C 中，试用 SQL 的断言机制定义下列两个完整性约束：

⑴ 每位教师开设的课程不能超过 10 门。

⑵不允许男同学选修 WU 老师的课程。

⑶每门课程最多 50 名男同学选修

⑷学生必须在选修 Maths 课后，才能选修其他课程。

⑸每个男学生最多选修 20 门课程。

答：⑴ 每位教师开设的课程不能超过 10 门。

```
CREATE  ASSERTION  ASSE5  CHECK
   （10>=ALL（SELECT  COUNT（CNO）
              FROM  C
              GROUP  BY TNAME））
```

⑵不允许男同学选修 WU 老师的课程。

```
CREATE  ASSERTION  ASSE2  CHECK
    （NOT  EXISTS ( SELECT  *
                    FROM  SC
                    WHERE  CNO IN (SELECT  CNO
                                   FROM  C
                                   WHERE TNAME=’WU’
                    AND    SNO IN (SELECT  SNO
                                   FROM  C
                                   WHERE SEX=’男’ ）））;
```

(3)每门课程最多 50 名男同学选修

```
CREATE  ASSERTION  ASSE3  CHECK
  (50>=ALL(SELECT  COUNT(SC.SNO)
           FROM  S, SC
           WHERE  S.SNO=SC.SNO  AND  SEX=' 男'
           GROUP  BY  CNO));
```

(4) 学生必须在选修 Maths 课后，才能选修其他课程。

这个约束可用下列形式表达：

“不存在一个学生的选课，这个学生没学过 Maths 课”。

这样就能很容易地写出断言：

```
CREATE  ASSERTION  ASSE4  CHECK
  (NOT  EXISTS( SELECT  SNO
                FROM  SC  X
                WHERE  NOT  EXISTS
                      (SELECT  *
                       FROM  SC  Y, C
                       WHERE  Y.CNO=C.CNO
                        AND   Y.SNO=X.SNO
                        AND   CNAME=' Maths')));
```

(5)每个男学生最多选修 20 门课程。

```
CREATE  ASSERTION  ASSE5  CHECK
  (20>=ALL(SELECT  COUNT(CNO)
           FROM  S, SC
           WHERE  S.SNO=SC.SNO  AND  SEX=' 男'
           GROUP  BY  S.SNO));
```

8．21 什么是数据库的安全性？有哪些级别的安全措施？

答：DB 的安全性是指保护 DB，防止不合法的使用，以免数据的泄密、更新或破坏。为了保护 DB，防止恶意的滥用，可以在从低到高五个级别上设置各种安全措施：环境级、职员级、OS 级、网络级、DBS 级。（解释略）

8．24 SQL 的视图机制有哪些优点？

答：SQL 的视图机制使系统具有三个优点：数据安全性，逻辑独立性和操作简便性。

8．25 SQL2 中的用户权限有哪几类？并做必要的解释。

答：SQL2 中的用户权限有六类：SELECT、INSE

10.3 什么是对象联系图？图中，椭圆、小圆圈、单箭头（→）、双箭头（→→）、双线箭头（⇒）、双向箭头（←→）这些结构各表示什么含义？

答:描述类型定义间嵌套和递归联系的图称为对象联系图。图中,每个对象可以有若干属性，属性的类型可以是基本数据类型、元组类型或集合类型，而元组或集合是以指针形式（引用类型）实现。

对象联系图中椭圆表示对象类型（相当于实体类型）；小圆圈表示属性是基本数据类型，单箭头（→）表示属性值是单值；双箭头（→→）表示属性值是多值；双线箭头（=>）表示对象类型之间的子类与超类联系（从子类指向超类）；双向箭头（↔）表示两个属性之间值的联系为逆联系。

10.4 面向对象的类型系统有哪三部分组成？每一部分又有哪些数据类型？

答：面向对象的类型系统基本数据类型、复合类型和引用类型三部分组成。

基本数据类型有整型、浮点型、字符、字符串型、布尔型和枚举型等五种。

复合类型有行类型、数组类型、列表类型、包类型和集合类型等五种。

引用类型只要一种。

10.5 在 ORDB 中有哪些基本数据类型？有哪些复合数据类型？

答：基本数据类型有整型、浮点型、字符串型和日期型等。

复合类型有结构类型、数组类型、多集类型和集合类型等四种。

10.6 ORDB 中，子表和超表应满足哪两个一致性要求？

答：（1）超表中每个元组最多可以与每个子表中的一个元组对应。

（2）子表中每个元组在超表中恰有一个元组对应。

10.7 图 13.11 是有关教师（Faculty）、系（Department）和系主任（Director）信息的对象联系图。

（1）试用 ORDB 的定义语言，定义这个数据库。

（2）试用 ORDB 的查询语言，分别写出下列查询的 SELECT 语句：

① 检索精通俄语（Russian）的教师工号和姓名。

② 检索复旦大学出访过瑞士（Switzerland）并且精通日语（Japanese）的系主任。

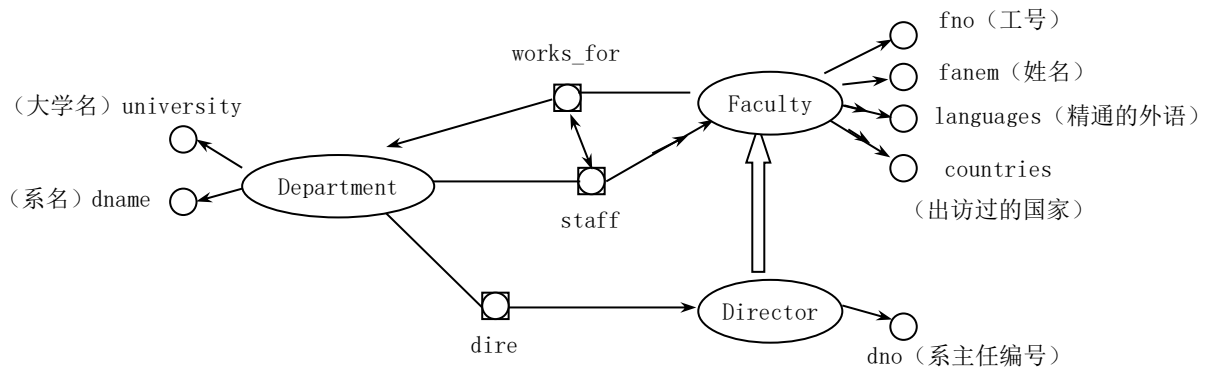


图 13.11　对象联系图

答：（1）CREATE　TYPE　MyString　char　varying；

```
CREATE　TABLE　department(university　MyString,
                          dname　MyString,
                          staff　setof(ref(faculty)),
                          dire　ref(director));
CREATE　TABLE　faculty(fno　integer,
                       fname　MyString
                       languages　setof(MyString),
                       countries　setof(MyString),
                       works_for　ref(department));
CREATE　TABLE　director(dno　integer)
                    Under　faculty;
```

（2）① SELECT　fno, fname

```
FROM　faculty
WHERE　'Russian'　in　languages;
```

② SELECT D.dno, D.fname
FROM director as D
WHERE D.works_for.university='Fudan University'
AND 'Switzerland' in D.countries
AND 'Japanese' in D.languages;

10.8 图 13.12 是有关学生（student）和学习（study）信息的对象联系图。

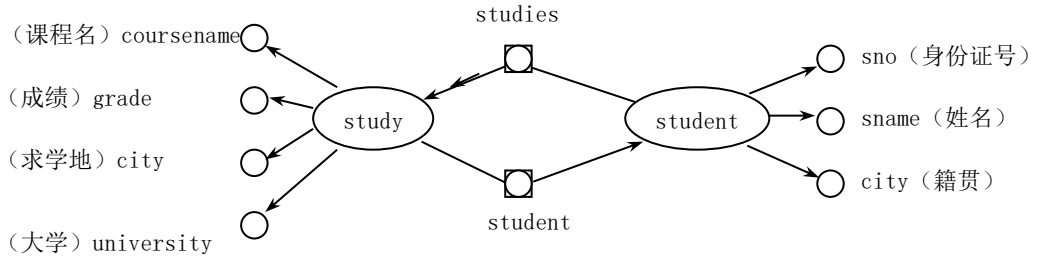


图 13.12 对象联系图

（1）试解释这个对象联系图。

（2）试用 ORDB 的定义语言，定义这个数据库。

（3）试用 ORDB 的查询语言，分别写出下列查询的 SELECT 语句：

① 检索每个学生的学习课程和成绩。

② 检索至少有一门课程的求学地与籍贯在同一城市的学生学号和姓名。

答：（1）对象 student 包含身份证号、姓名、籍贯和学习（studies）等属性，对象 study 包含课程名、成绩、求学地、大学以及学生（student）等属性。对象 student 和 study 之间联系为 1:N。

（2）CREATE TYPE MyString char varying;
CREATE TABLE student(sno integer,
sname MyString,
city MyString,
studies setoff(ref(study)));
CREATE TABLE study(coursename MyString,
grade integer,
city MyString,
university MyString,
student ref(student));

（3） ①SELECT A.sname, B.coursename, B.grade
FROM student as A, A.studies as B;
② SELECT A.sno, A.sname
FROM student as A, A.student as B
WHERE A.city=B.city;