**第1章 概论**

1.数据库系统的特点

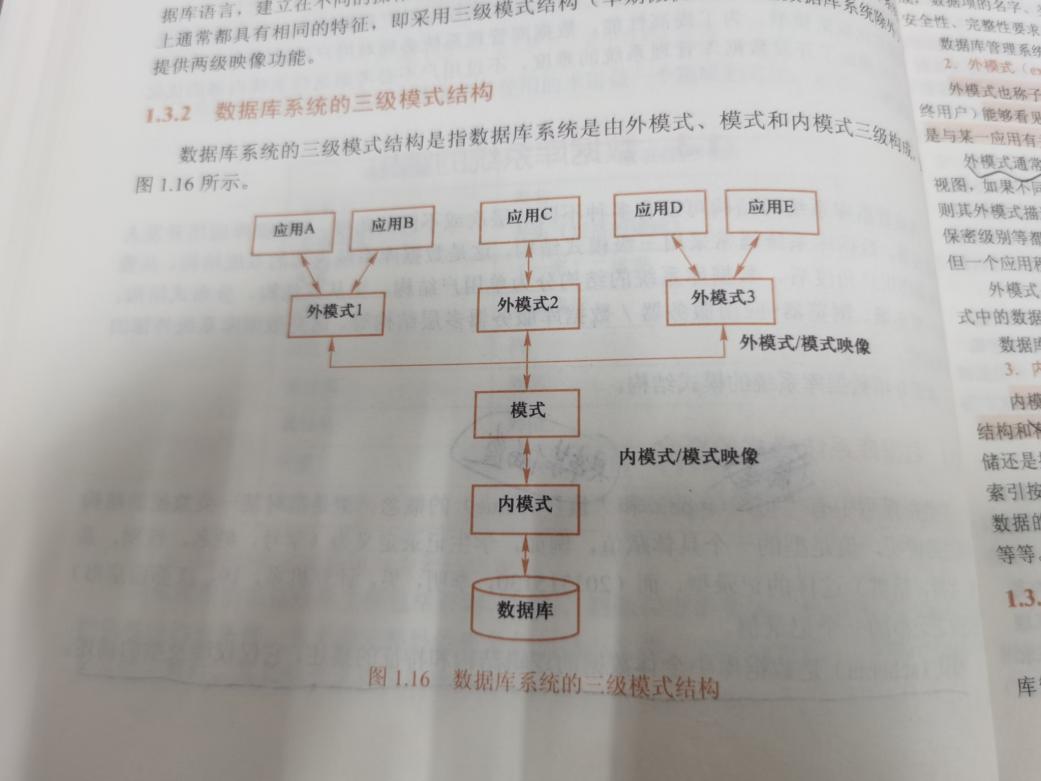
（1）数据结构化

（2）数据的共享性高、冗余度低且易扩充

（3）数据独立性高

（4）数据由数据库管理系统统一管理和控制

1. 数据库的三级模式结构



模式（表），也称逻辑模式，是数据库中全体数据的逻辑结构和特征的描述，是所有用户的公共数据视图。

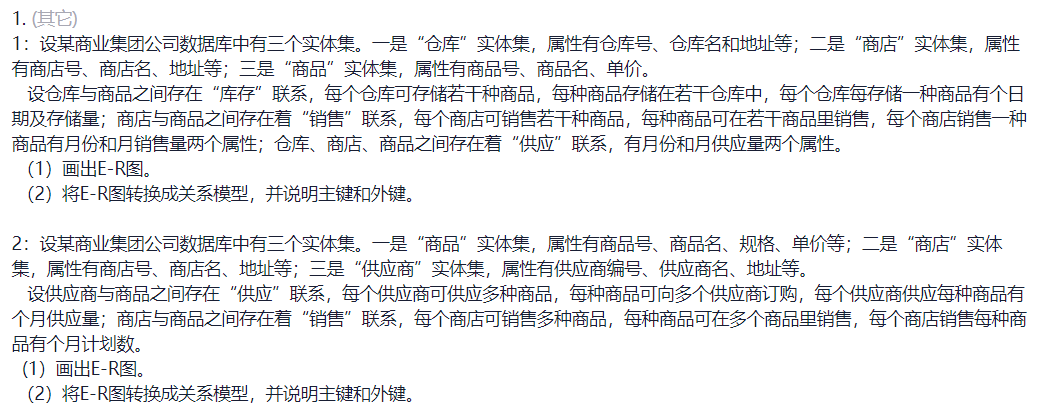
外模式（视图），也称子模式或用户模式，它是数据库用户能够看见和使用的局部数据的逻辑结构和特征的描述，是数据库用户的数据视图，是与某一应用有关的数据的逻辑表示。外模式通常是模式的子集，一个数据库可以有多个外模式。

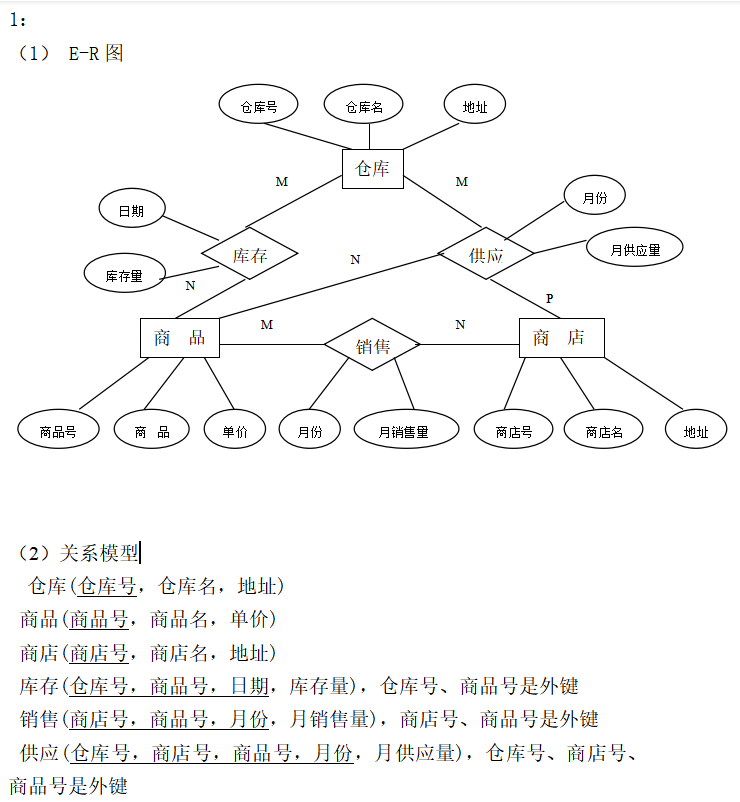
内模式（存储），也称存储模式，一个数据库只有一个内模式。它是数据物理结构和存储方式的描述，是数据在数据库内部的组织方式。

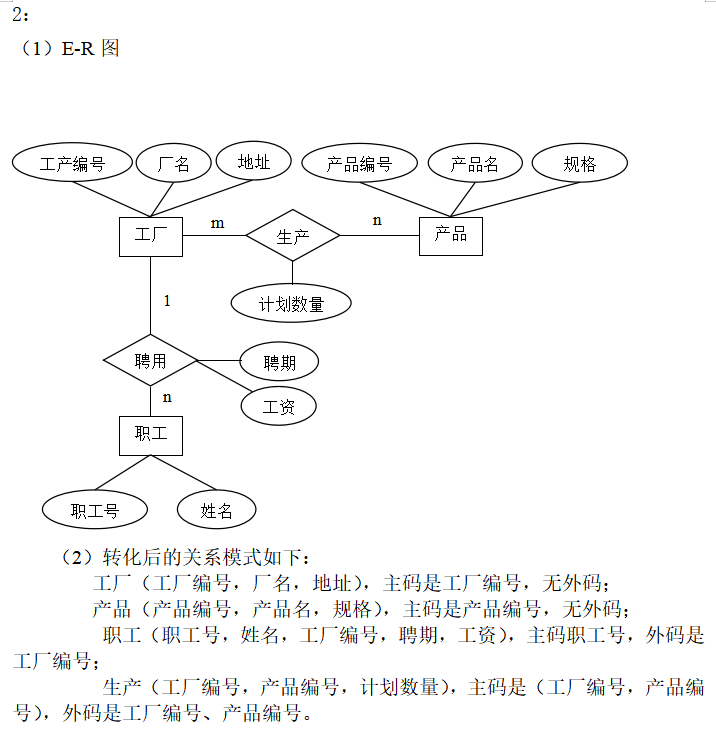
外模式/模式映像：当模式改变时（增加新的关系、属性或是改变属性的数据类型），由数据库管理员对各个外模式/模式的映像作相应改变，可以使外模式保持不变。应用程序是依据数据的外模式编写的，从而应用程序不必修改，保证了数据与程序的逻辑独立性，简称数据的逻辑独立性。

模式/内模式映像：当数据库的存储结构改变时（选用了另一种存储结构），由数据库管理员对模式/内模式映像作相应改变，可以使模式保持不变，从而应用程序也不必改变，保证了数据与程序的物理独立性，简称数据的物理独立性。

1. E-R图的设计及向关系模型的转化







**第2章 关系数据库**

1.数据模型的组成

数据模型通常由数据结构、数据操作和数据的完整性约束条件三部分组成。

1. 关系的定义

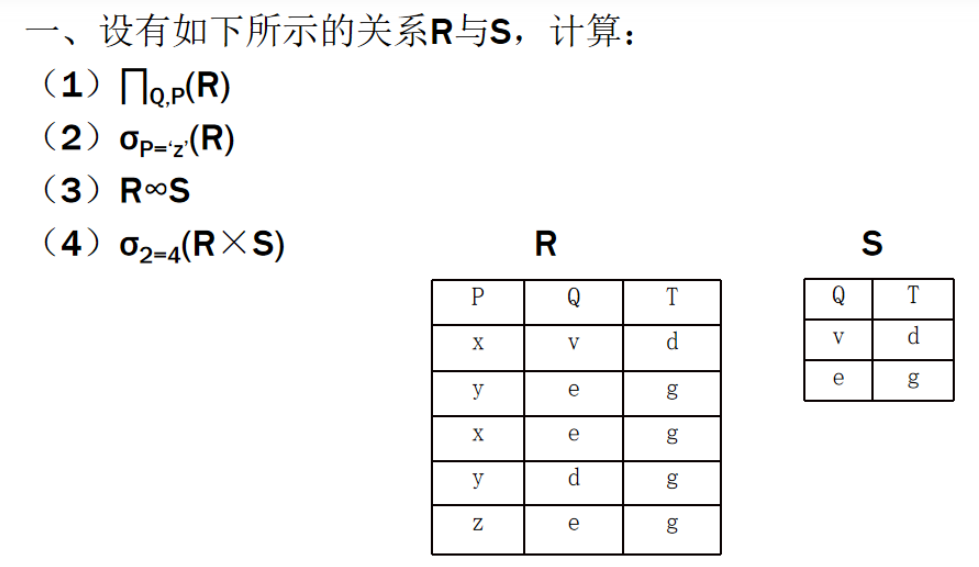
关系是笛卡尔积的有限子集。

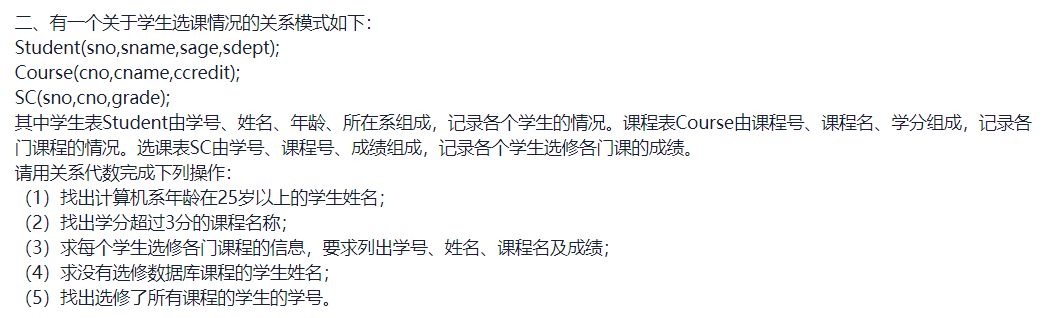
1. 主键与外键

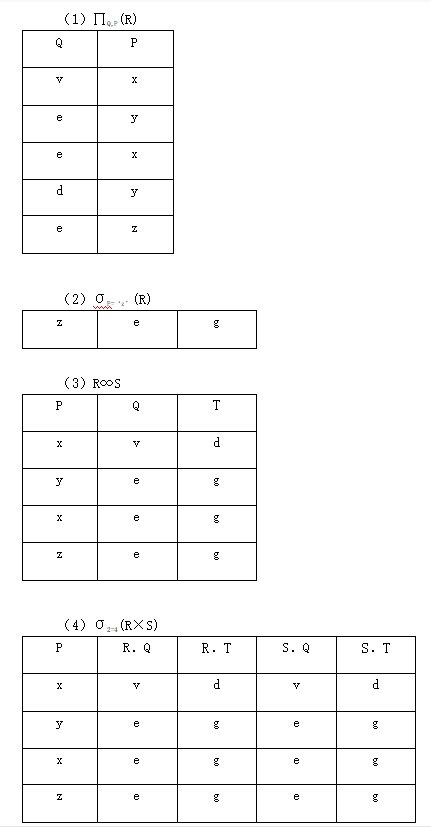
主键：主键就是数据行的唯一标识。不会重复的列才会当主键，一个表中只能有一个主键。

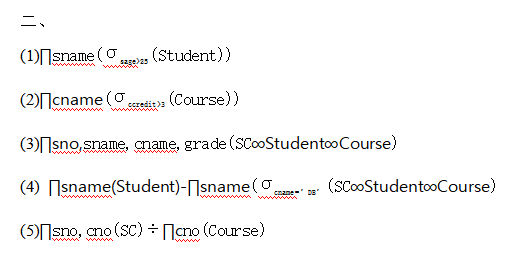
外键：把一张表中的主键添加到另一张表中，另一张表中的列就是数据的外键列，有外键列的表就叫做“外键表”，相对来说，主键被引用的那张表就叫做“主键表”。

1. 关系代数的基本运算
2. 关系代数表达式

、







1. **SQL语言**

1.SQL语言具有的功能

数据定义、数据操纵、数据控制

2.表定义、表修改

表定义：create table 表名

表修改：alter table 表名

表删除：drop table 表名

3.插入、删除、查询、修改、授权、收权语句

插入：insert into表名（列名） values（属性值）

修改：update 表名 set 列名=属性值

查询：select 列名 from 表名

删除：delete from表名

授权：grant 权限名 on 表名 to 用户名

收权：revoke 权限名 on 表名 from 用户名

4.索引的创建

Create index 索引名 on 表名（列名，次序）

5.视图的定义及查询

定义：Create view 视图名 as 子查询

查询：与查询表一样

1. **关系数据理论**

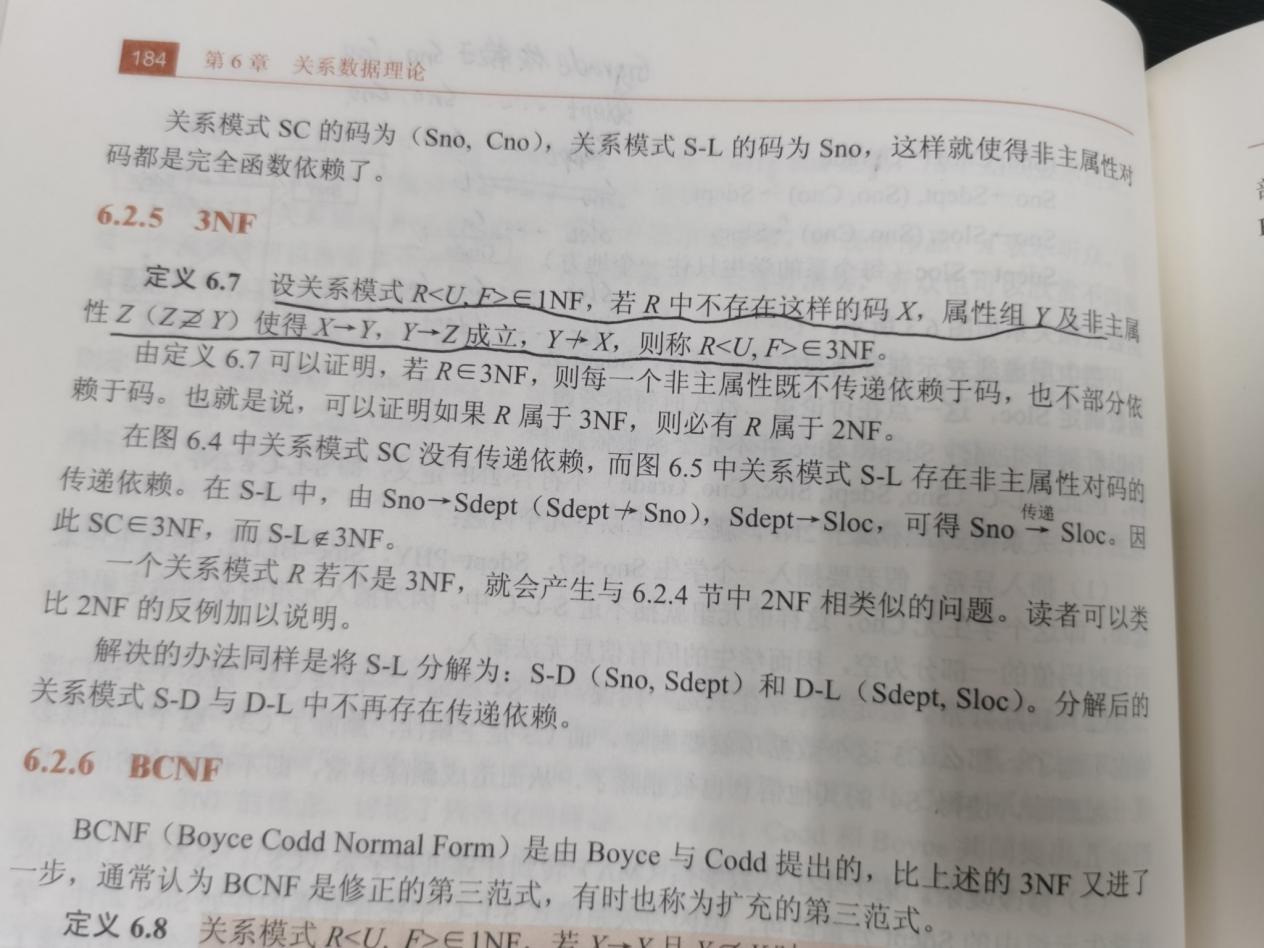
1.基本概念：如函数依赖、部分函数依赖、传递函数依赖等

见书P180

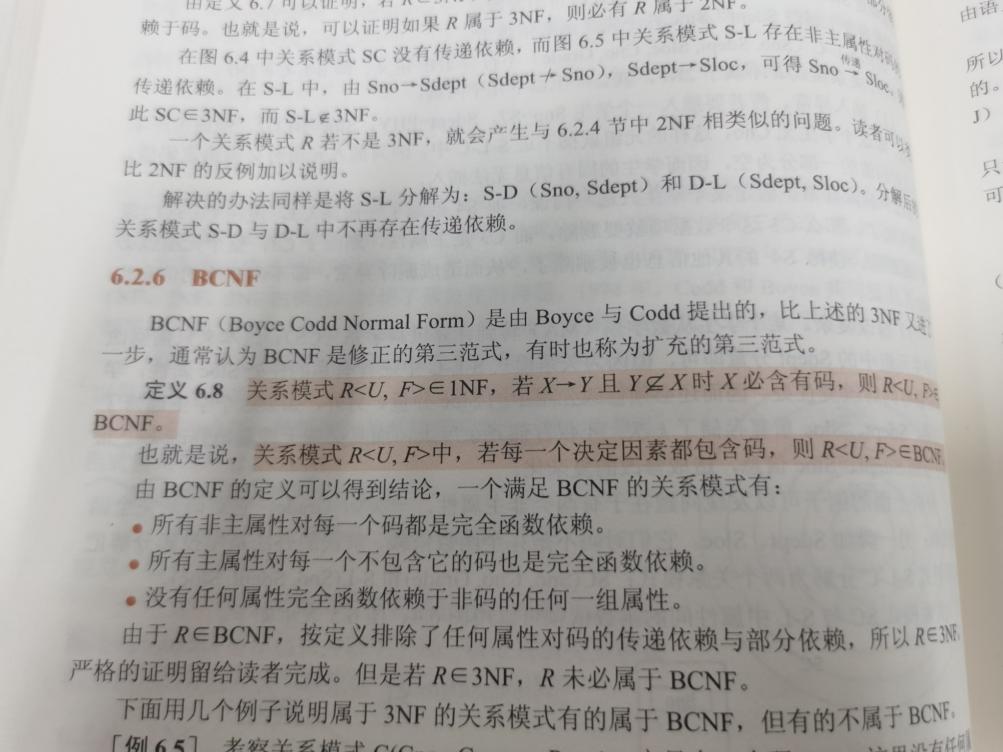
2.各种范式的定义（4NF不要求）

2NF：若R满足1NF，且每一个非主属性完全函数依赖于任何一个候选码，则R满足2NF。（存在部分函数依赖的不满足2NF）

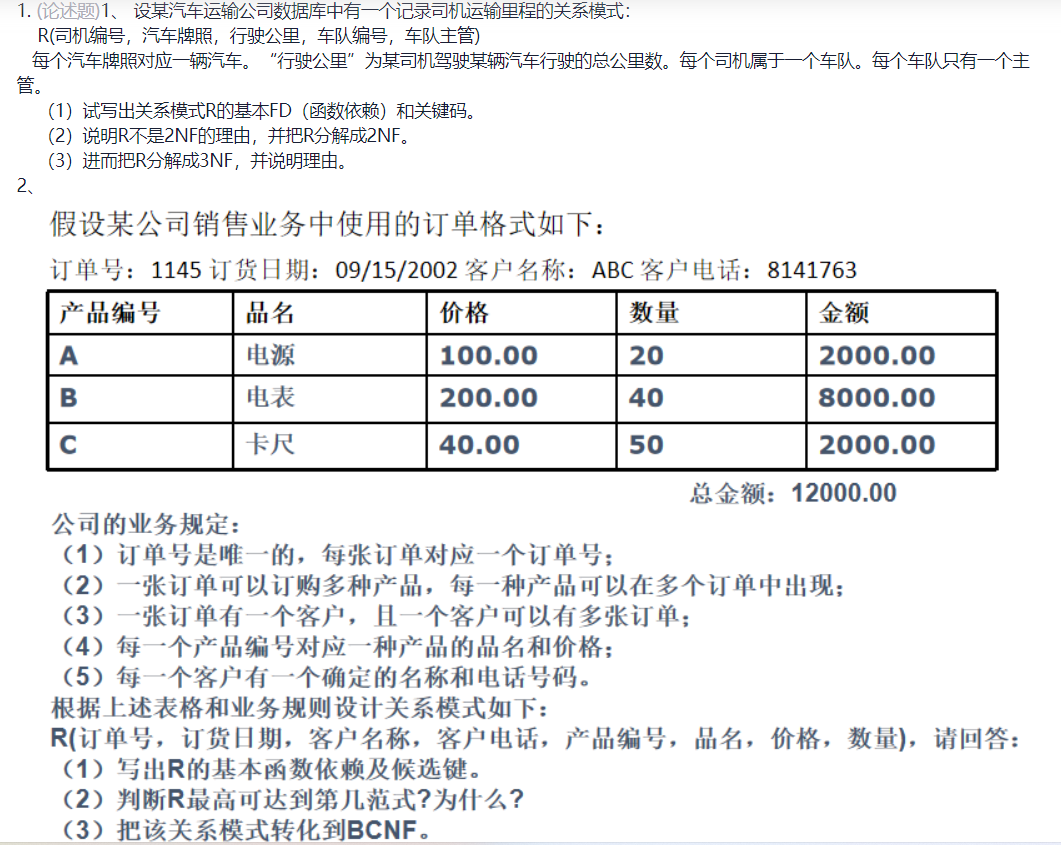
3NF：

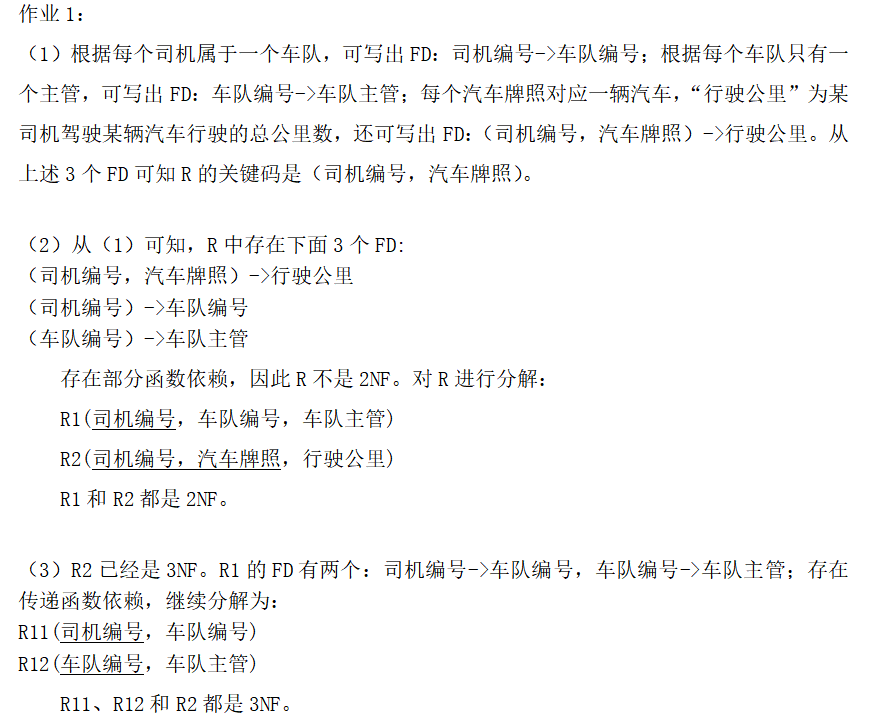


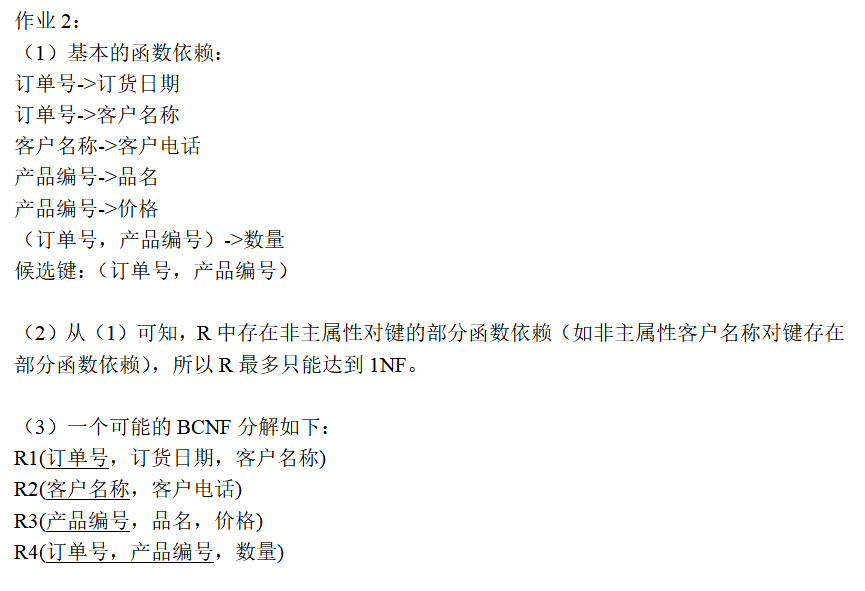
BCNF：



3.范式的判断及优化







1. **数据库设计**

1.六大基本步骤及每一步骤包含的基本工作

需求分析

概念结构设计

逻辑结构设计

物理结构设计

数据库实施

数据库运行与维护

1. **恢复**

1.什么是事务

事务是用户定义的一个数据库操作序列，这些操作要么全做，要么全不做，是一个不可分割的工作单位。

2.事务的四大性质

原子性、一致性、隔离性、持续性。这四大性质简称为ACID特性。

3.故障的类型

事务内部的故障、系统故障、介质故障、计算机病毒

4.故障的恢复方法

数据转储、登记日志文件

1. **并发控制**

1.并发控制的潜在问题

丢失修改、读“脏”数据、不可重复读、“幽灵”问题

2.锁的类型

排他式封锁（X锁、写锁），共享性封锁（S锁、读锁）

3.封锁技术

一级封锁协议：事务T在修改数据R之前必须先对其加X锁，直到事务结束才释放。

二级封锁协议：在一级封锁协议基础上，增加事务T在读取数据R之前必须先对其加S锁，读完后即可释放S锁。

三级封锁协议：在一级封锁协议基础上，增加事务T在读取数据R之前必须先对其加S锁，直到事务结束才释放。

1. 两段锁协议

实现并发调度的可串行性，从而保证调度的正确性。所谓的两段，指的是事务分为两个阶段。第一阶段是获得封锁，也称为扩展阶段；第二阶段是释放封锁，也称为收缩阶段。

**编程题**

1.函数设计

CREATE OR REPLACE FUNCTION f\_sum\_credits

(p\_sno IN TB\_Grade.StuID%type)

return TB\_Course.CourseGrade%type

as

p\_credits TB\_Course.CourseGrade%type;

begin

p\_credits:=0;

for sc in(select TotalScore,CourseGrade

from TB\_Grade natural join TB\_Course where StuID = p\_sno)loop

if sc.TotalScore> 60 then

p\_credits:= p\_credits + sc.CourseGrade;

end if;

end loop;

return p\_credits;

end f\_sum\_credits;

/

DECLARE

sno TB\_Grade.StuID%type:=&v\_sno;

BEGIN

DBMS\_OUTPUT.PUT\_LINE(';学号为';||sno||';的同学已获得学分：';||f\_sum\_credits(sno));

END;

2.触发器设计

D13-1源程序:在TB\_Student中加一字段TotalCredit，设计一触发器，要求在TB\_Grade中插入成绩时，如果成绩大于等于60分则在该学生的TotalCredit中自动加上相应课程的学分（TB\_Course表中的CourseGrade）

create or replace Trigger SumCredit

after insert

on TB\_Grade

for each row

declare

credit TB\_Course.CourseGrade%type;

begin

if (:new.TotalScore>=60) then

select CourseGrade into credit from TB\_Course where CourseID=:new.CourseID;

update TB\_Student set TotalCredit=TotalCredit+credit where StuID=:new.StuID;

end if;

end;