ER图设计/SQL

2.4　E-R模型设计实例

- 第一步　确定实体集
- 第二步　确定实体集之间的关联集
- 第三步　给实体集和关联集加上属性
- 第四步　把实体集和关联集用E-R图表示

一个实例

用E-R图表示某个工厂物资管理的概念模型

- 实体
  - 仓库： 仓库号、面积、电话号码
  - 零件 ：零件号、名称、规格、单价、描述
  - 供应商：供应商号、姓名、地址、电话号码、帐号
  - 项目：项目号、预算、开工日期
  - 职工：职工号、姓名、年龄、职称

一个实例

- 实体之间的关联如下：

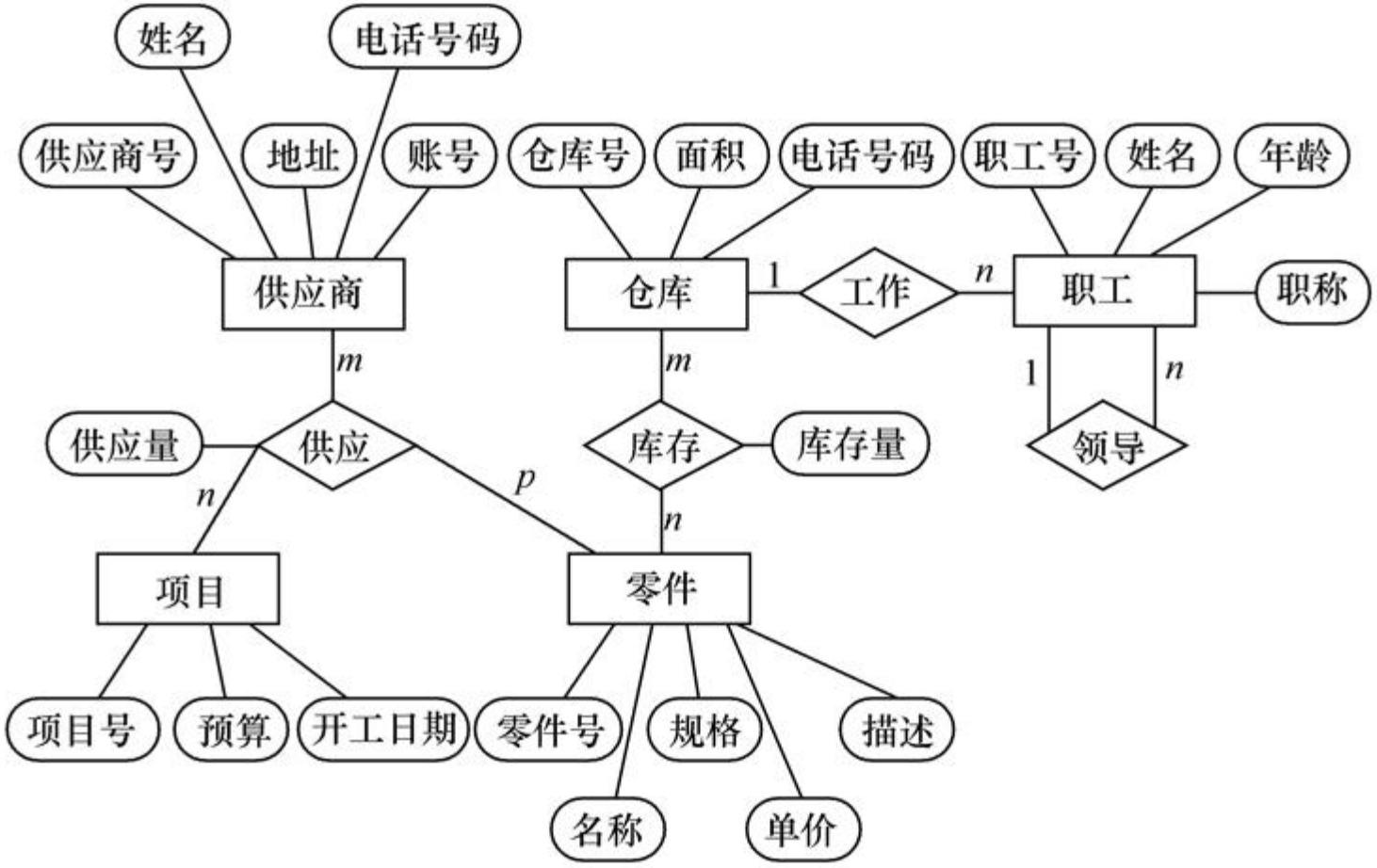
(1)一个仓库可以存放多种零件，一种零件可以存放在多个仓库中。仓库和零件具有多对多的关联。用库存量来表示某种零件在某个仓库中的数量。

(2)一个仓库有多个职工当仓库保管员，一个职工只能在一个仓库工作，仓库和职工之间是一对多的关联。职工实体型中具有一对多的关联

(3)职工之间具有领导-被领导关系。即仓库主任领导若干保管员。

(4)供应商、项目和零件三者之间具有多对多的关联

一个实例



(c) 完整的实体-联系图

逐一设计分E-R图（续）

［实例］销售管理子系统分E-R图的设计

- 销售管理子系统的主要功能：
  - 处理顾客和销售员送来的订单
  - 工厂是根据订货安排生产的
  - 交出货物同时开出发票
  - 收到顾客付款后，根据发票存根和信贷情况进行应收款处理

逐一设计分E-R图（续）

- 下图是第一层数据流图，虚线部分划出了系统边界

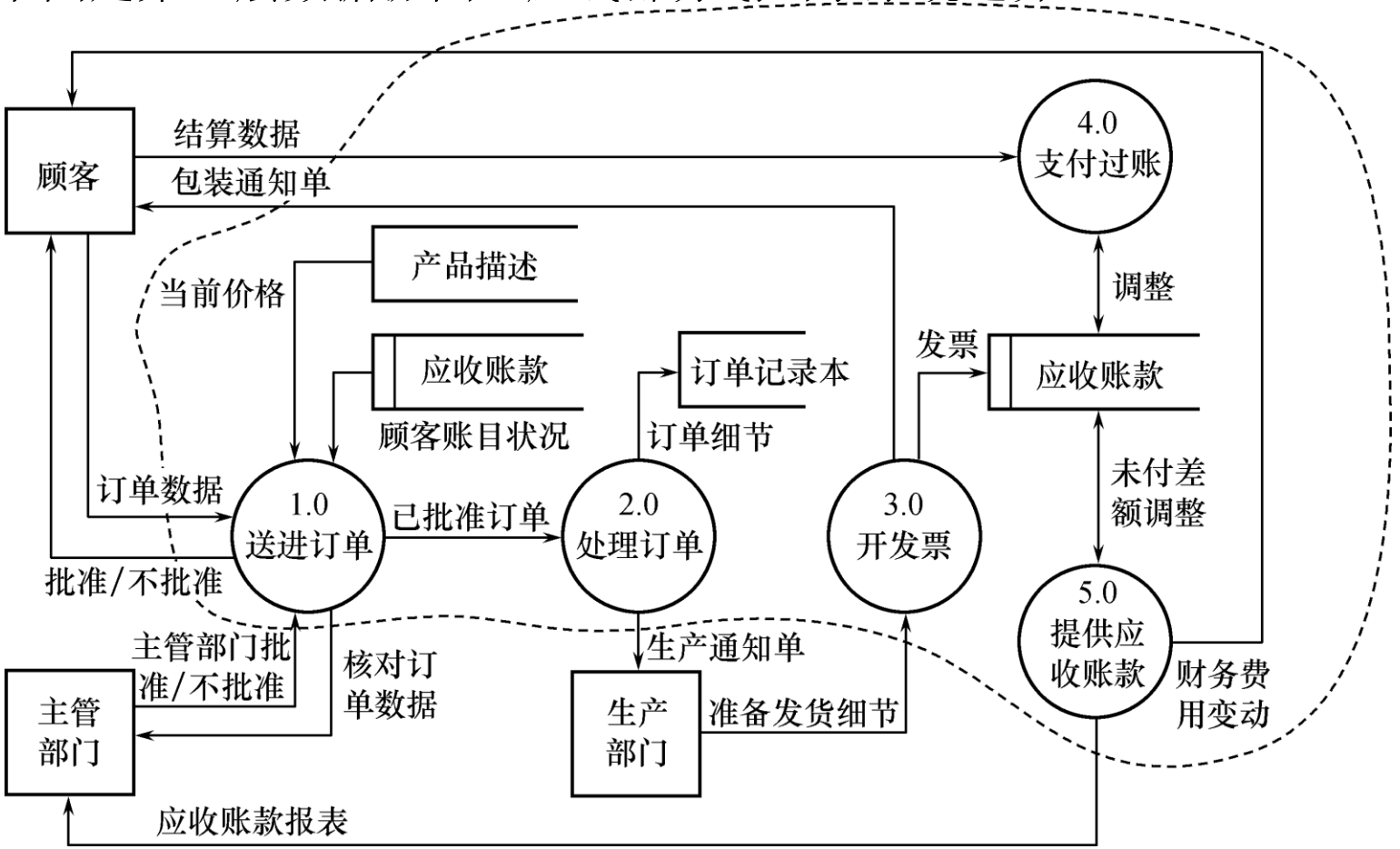


图7.18 销售管理子系统第一层数据流图

逐一设计分E-R图（续）

- 上图中把系统功能又分为4个子系统，下面四个图是第二层数据流图

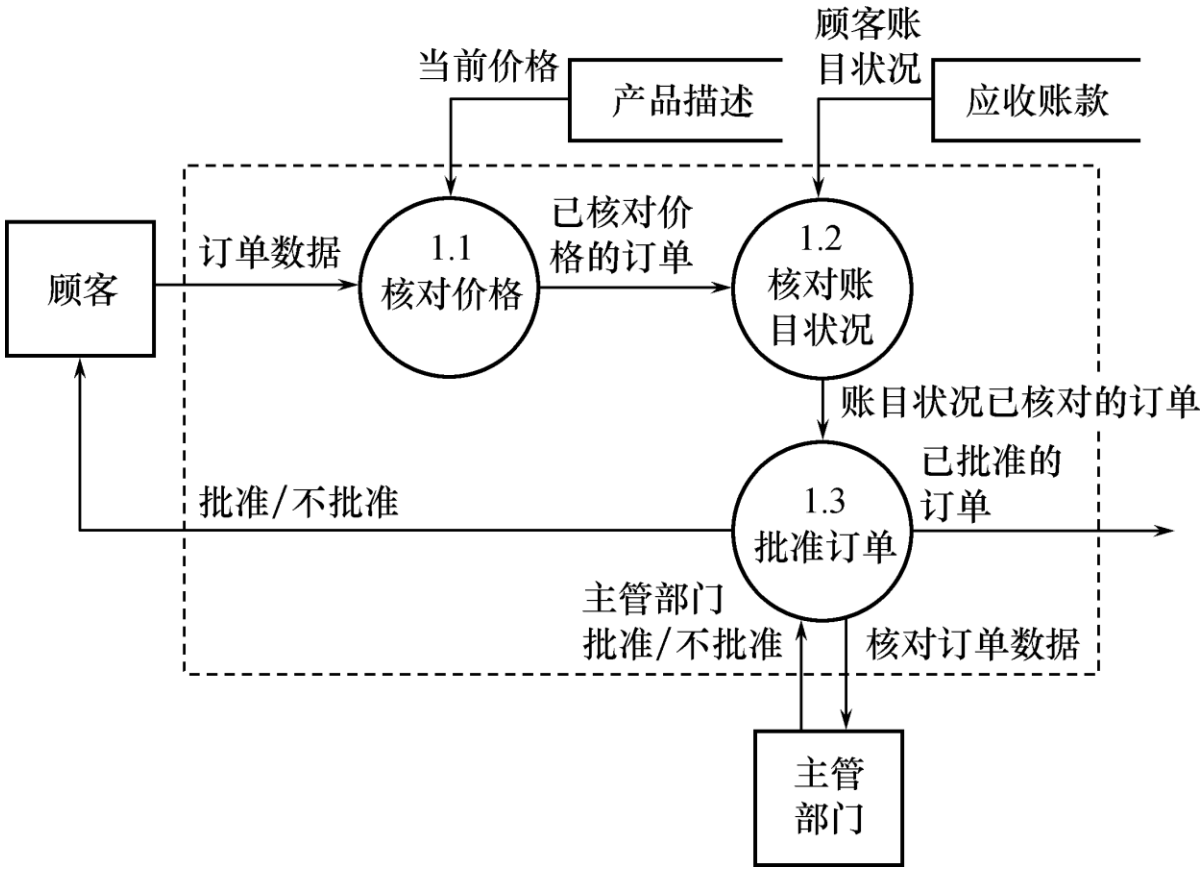


图7.19 接收订单

逐一设计分E-R图（续）

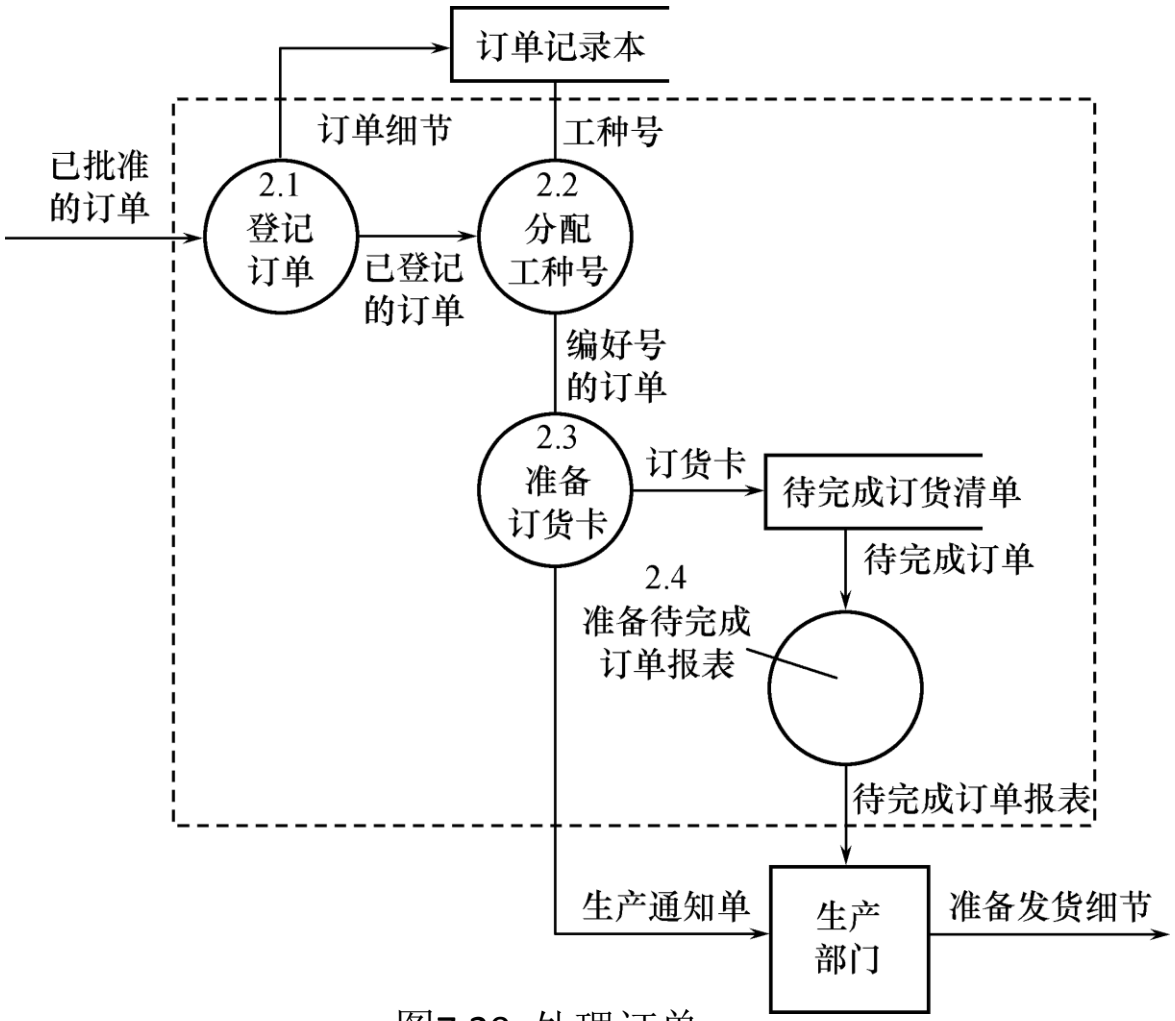


图7.20 处理订单

逐一设计分E-R图（续）

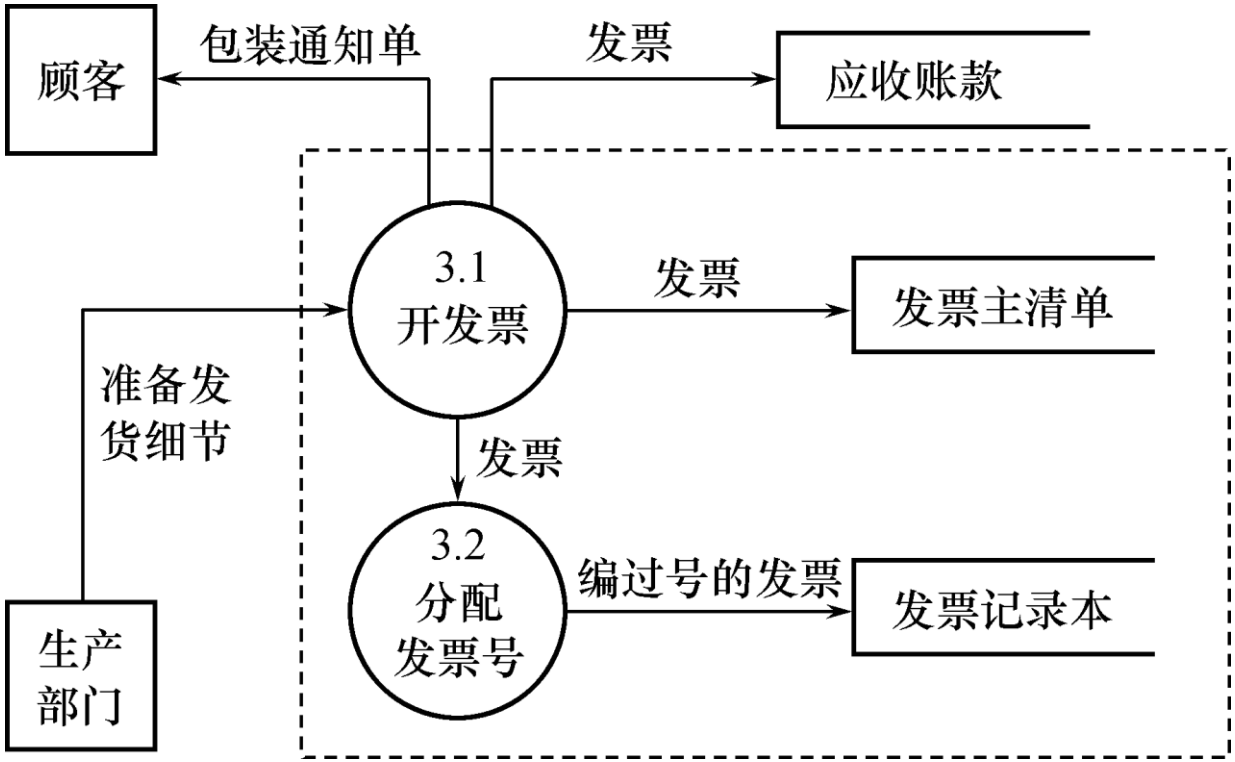


图7.21 开发票

逐一设计分E-R图（续）

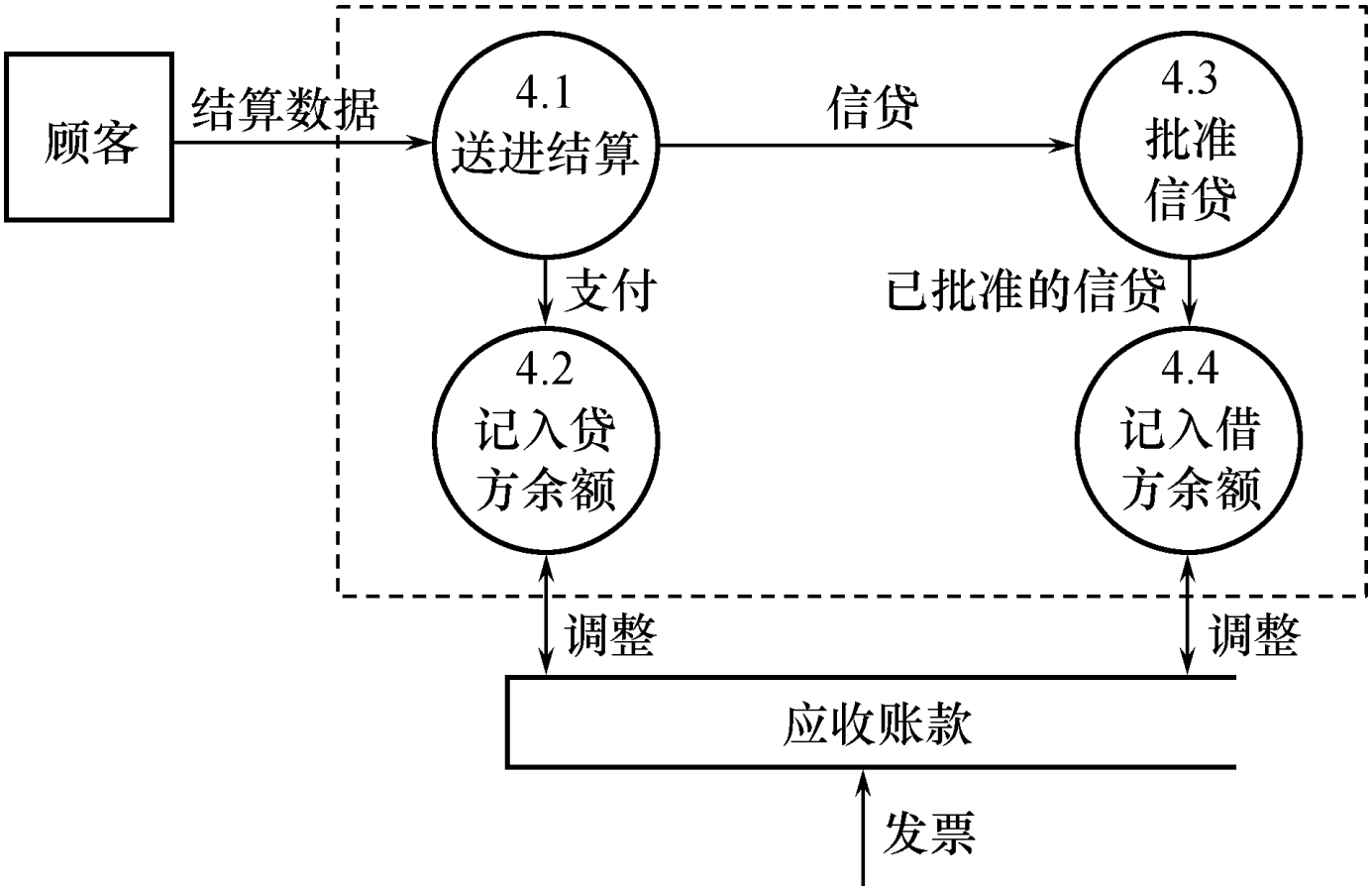
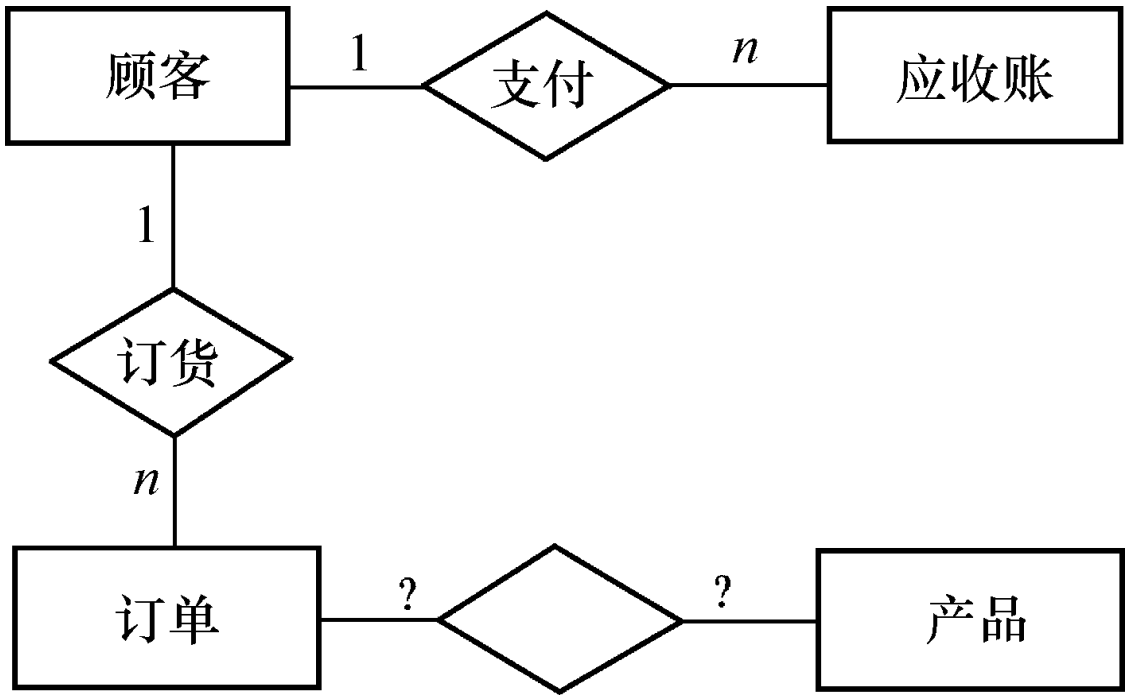


图7.22 支付过账

逐一设计分E-R图（续）



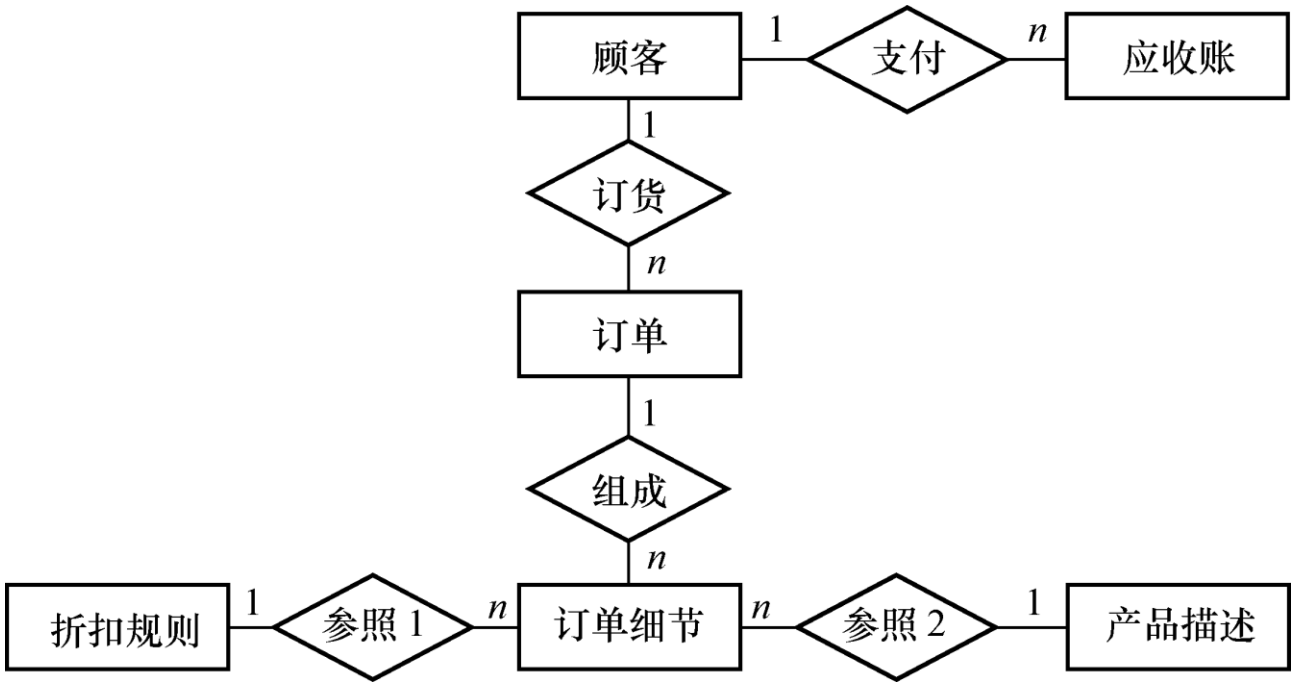
分E-R图的框架

逐一设计分E-R图（续）

- 参照第二层数据流图和数据字典，遵循两个准则，进行如下调整：
  - (1) 订单与订单细节是1：*n*的联系
  - (2) 原订单和产品的联系实际上是订单细节和产品的联系。
  - (3) 图7.21中“发票主清单”是一个数据存储，不必作为实体加入分E-R图
  - (4) 工厂对大宗订货给予优惠

逐一设计分E-R图（续）

- 得到分E-R图如下图所示



销售管理子系统的分E-R图

逐一设计分E-R图（续）

对每个实体定义的属性如下：

- 顾客：{顾客号，顾客名，地址，电话，信贷状况，账目余额}
- 订单：{订单号，顾客号，订货项数，订货日期，交货日期，工种号，生产地点}
- 订单细则：{订单号，细则号，零件号，订货数，金额}
- 应收账款：{顾客号，订单号，发票号，应收金额，支付日期，支付金额，当前余额，货款限额}
- 产品描述：{产品号，产品名，单价，重量}
- 折扣规则：{产品号，订货量，折扣}

4.4 数据查询

- **4.4.1 单表查询**
- **4.4.2 连接查询**
- **4.4.3 嵌套查询**
- **4.4.4 集合查询**
- **4.4.5 Select语句的一般形式**

4.4.1 单表查询

- 查询仅涉及一个表：
  - 一、 选择表中的若干列
  - 二、 选择表中的若干元组
  - 三、 ORDER BY子句
  - 四、 聚集函数
  - 五、 GROUP BY子句

一、选择表中的若干列

- 查询指定列

  [例1] 查询全体学生的学号与姓名。

  ```
  SELECT Sno，Sname
  FROM Student；
  ```

  [例2] 查询全体学生的姓名、学号、所在系。

  ```
  SELECT Sname，Sno，Sdept
  FROM Student；
  ```

2. 查询全部列

- 选出所有属性列：
  - 在SELECT关键字后面列出所有列名
  - 将<目标列表达式>指定为 *

[例3] 查询全体学生的详细记录。

```
SELECT Sno，Sname，Ssex，Sage，Sdept
FROM Student；
  或
SELECT *
FROM Student；
```

3. 查询经过计算的值

- SELECT子句的<目标列表达式>可以为：
  - 算术表达式
  - 字符串常量
  - 函数
  - 列别名

查询经过计算的值（续）

[例4] 查全体学生的姓名及其出生年份。

```
SELECT Sname，2004-Sage   /*假定当年的年份为2004年*/
FROM Student；
```

输出结果：

Sname	2004-Sage
李勇	1984
刘晨	1985
王敏	1986
张立	1985

查询经过计算的值（续）

[例5] 查询全体学生的姓名、出生年份和所有系，要求用小写字母表示所有系名

```
SELECT Sname，  ‘Year of Birth: '，2004-Sage，
               ISLOWER(Sdept)
FROM Student；
```

输出结果：

Sname	'Year of Birth:'	2004-Sage	ISLOWER(Sdept)
李勇	Year of Birth:	1984	cs
刘晨	Year of Birth:	1985	is
王敏	Year of Birth:	1986	ma
张立	Year of Birth:	1985	is

查询经过计算的值（续）

- 使用列别名改变查询结果的列标题:

```
SELECT Sname NAME，'Year of Birth: ’ BIRTH，
       2004-Sage  BIRTHDAY，LOWER(Sdept)  DEPARTMENT
FROM Student；
```

输出结果：

NAME	BIRTH	BIRTHDAY	DEPARTMENT
李勇	Year of Birth:	1984	cs
刘晨	Year of Birth:	1985	is
王敏	Year of Birth:	1986	ma
张立	Year of Birth:	1985	is

4.4.1 单表查询

- 查询仅涉及一个表：
  - 一、 选择表中的若干列
  - 二、 选择表中的若干元组
  - 三、 ORDER BY子句
  - 四、 聚集函数
  - 五、 GROUP BY子句

二、选择表中的若干元组

- 1. 消除取值重复的行

  如果没有指定DISTINCT关键词，则缺省为ALL

  [例6] 查询选修了课程的学生学号。

  SELECT Sno FROM SC;

  等价于：

  SELECT ALL Sno FROM SC;

  执行上面的SELECT语句后，结果为：

Sno
200215121
200215121
200215121
200215122
200215122

消除取值重复的行（续）

- 指定DISTINCT关键词，去掉表中重复的行

```
SELECT DISTINCT Sno
FROM SC;
```

执行结果：

Sno
200215121
200215122

2.查询满足条件的元组

表3.4 常用的查询条件

查 询 条 件	谓　词
比　较	=，>，<，>=，<=，!=，<>，!>，!<；NOT+上述比较运算符
确定范围	BETWEEN AND，NOT BETWEEN AND
确定集合	IN，NOT IN
字符匹配	LIKE，NOT LIKE
空　值	IS NULL，IS NOT NULL
多重条件（逻辑运算）	AND，OR，NOT

(1) 比较大小

［例7］ 查询计算机科学系全体学生的名单。

```
SELECT Sname
FROM Student
WHERE Sdept='CS';
```

[例8] 查询所有年龄在20岁以下的学生姓名及其年龄。

```
SELECT Sname，Sage
  FROM   Student
  WHERE Sage < 20;
```

［例9］ 查询考试成绩有不及格的学生的学号。

```
SELECT DISTINCT Sno
FROM  SC
WHERE Grade<60;
```

（2）确定范围

- **谓词：** BETWEEN … AND …

  NOT BETWEEN … AND …

[例10] 查询年龄在20~23岁（包括20岁和23岁）之间的学生的姓名、系别和年龄

```
SELECT Sname，Sdept，Sage
    FROM    Student
    WHERE   Sage BETWEEN 20 AND 23；
```

[例11] 查询年龄不在20~23岁之间的学生姓名、系别和年龄

```
SELECT Sname，Sdept，Sage
FROM    Student
WHERE Sage NOT BETWEEN 20 AND 23；
```

(3) 确定集合

- **谓词：IN <值表>，NOT IN <值表>**

[例12]查询信息系（IS）、数学系（MA）和计算机科学系（CS）学生的姓名和性别。

```
SELECT Sname，Ssex
FROM  Student
WHERE Sdept IN ( 'IS'，'MA'，'CS' );
```

[例13]查询既不是信息系、数学系，也不是计算机科学系的学生的姓名和性别。

```
SELECT Sname，Ssex
FROM Student
WHERE Sdept NOT IN ( 'IS'，'MA'，'CS' );
```

(4)字符匹配

- **谓词：** [NOT] LIKE ‘<匹配串>’ [ESCAPE ‘ <换码字符>’]

1) 匹配串为固定字符串

[例14] 查询学号为200215121的学生的详细情况。

```
SELECT *
 FROM  Student
 WHERE  Sno LIKE ‘200215121';
```

等价于：

```
SELECT  *
 FROM  Student
 WHERE Sno = ' 200215121 ';
```

字符匹配（续）

2) 匹配串为含通配符的字符串

[例15] 查询所有姓刘学生的姓名、学号和性别。

```
SELECT Sname，Sno，Ssex
FROM Student
WHERE  Sname LIKE ‘刘%’;
```

[例16] 查询姓"欧阳"且全名为三个汉字的学生的姓名。

```
SELECT Sname
FROM   Student
WHERE  Sname LIKE '欧阳__';
```

字符匹配（续）

[例17] 查询名字中第2个字为"阳"字的学生的姓名和学号。

```
SELECT Sname，Sno
FROM Student
WHERE Sname LIKE ‘__阳%’;
```

[例18] 查询所有不姓刘的学生姓名。

```
SELECT Sname，Sno，Ssex
FROM Student
WHERE Sname NOT LIKE '刘%';
```

字符匹配（续）

3) 使用换码字符将通配符转义为普通字符

[例19] 查询DB_Design课程的课程号和学分。

```
SELECT Cno，Ccredit
FROM Course
WHERE Cname LIKE 'DB\_Design' ESCAPE '\‘；
```

[例20] 查询以"DB_"开头，且倒数第3个字符为 i的课程的详细情况。

```
SELECT  *
FROM   Course
WHERE  Cname LIKE  'DB\_%i_ _' ESCAPE ' \ ‘;
```

ESCAPE ' \ ' 表示“ \ ” 为换码字符

(5) 涉及空值的查询

- **谓词：** IS NULL 或 IS NOT NULL
- “IS” 不能用 “=” 代替

[例21] 某些学生选修课程后没有参加考试，所以有选课记录，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
```

[例22] 查所有有成绩的学生学号和课程号。

```
SELECT Sno，Cno
FROM  SC
WHERE  Grade IS NOT NULL;
```

(6) 多重条件查询

- 逻辑运算符：AND和 OR来联结多个查询条件
  - AND的优先级高于OR
  - 可以用括号改变优先级
- 可用来实现多种其他谓词
  - [NOT] IN
  - [NOT] BETWEEN … AND …

多重条件查询（续）

[例23] 查询计算机系年龄在20岁以下的学生姓名。

```
SELECT Sname
FROM  Student
WHERE Sdept= 'CS' AND Sage<20；
```

多重条件查询（续）

- **改写[例12]**

[例12] 查询信息系（IS）、数学系（MA）和计算机科学系（CS）学生的姓名和性别。

```
SELECT Sname，Ssex
FROM Student
WHERE Sdept IN ( 'IS'，'MA'，'CS' )
```

可改写为：

```
SELECT Sname，Ssex
FROM   Student
WHERE  Sdept= ' IS ' OR Sdept= ' MA' OR Sdept= ' CS ';
```

4.4.1 单表查询

- 查询仅涉及一个表：
  - 一、 选择表中的若干列
  - 二、 选择表中的若干元组
  - 三、 ORDER BY子句
  - 四、 聚集函数
  - 五、 GROUP BY子句

三、ORDER BY子句

- ORDER BY子句
  - ■可以按一个或多个属性列排序
  - ■升序：ASC；降序：DESC；缺省值为升序
- 当排序列含空值时
  - ■ASC：排序列为空值的元组最后显示
  - ■DESC：排序列为空值的元组最先显示

ORDER BY子句（续）

[例24] 查询选修了3号课程的学生的学号及其成绩，查询结果按分数降序排列。

```
SELECT Sno，Grade
FROM  SC
WHERE  Cno='3'
ORDER BY Grade DESC;
```

［例25］ 查询全体学生情况，查询结果按所在系的系号升序排列，同一系中的学生按年龄降序排列。

```
SELECT  *
FROM  Student
ORDER BY Sdept，Sage DESC;
```

4.4.1 单表查询

- 查询仅涉及一个表：
  - 一、 选择表中的若干列
  - 二、 选择表中的若干元组
  - 三、 ORDER BY子句
  - 四、 聚集函数
  - 五、 GROUP BY子句

四、聚集函数

- 聚集函数：
  - 计数

    COUNT（[DISTINCT|ALL] *）

    COUNT（[DISTINCT|ALL] <列名>）
  - 计算总和

    SUM（[DISTINCT|ALL] <列名>）
  - 计算平均值

    AVG（[DISTINCT|ALL] <列名>）
  - 最大最小值

    MAX（[DISTINCT|ALL] <列名>）

    MIN（[DISTINCT|ALL] <列名>）

聚集函数（续）

[例26] 查询学生总人数。

```
SELECT COUNT(*)
FROM  Student；
```

[例27] 查询选修了课程的学生人数。

```
SELECT COUNT(DISTINCT Sno)
FROM SC；
```

[例28] 计算1号课程的学生平均成绩。

```
SELECT AVG(Grade)
FROM SC
WHERE Cno= ' 1 '；
```

聚集函数（续）

[例29] 查询选修1号课程的学生最高分数。

```
SELECT MAX(Grade)
FROM SC
WHER Cno=‘1’;
```

［例30］查询学生200215012选修课程的总学分数。

```
SELECT SUM(Ccredit)
FROM  SC， Course
WHER Sno='200215012' AND SC.Cno=Course.Cno;
```

4.4.1 单表查询

- 查询仅涉及一个表：
  - 一、 选择表中的若干列
  - 二、 选择表中的若干元组
  - 三、 ORDER BY子句
  - 四、 聚集函数
  - 五、 GROUP BY子句

五、GROUP BY子句

- GROUP BY子句分组：

  细化聚集函数的作用对象

  - 未对查询结果分组，聚集函数将作用于整个查询结果
  - 对查询结果分组后，聚集函数将分别作用于每个组
  - 作用对象是查询的中间结果表
  - 按指定的一列或多列值分组，值相等的为一组

GROUP BY子句（续）

[例31] 求各个课程号及相应的选课人数。

```
SELECT Cno，COUNT(Sno)
FROM   SC
GROUP BY Cno；
```

查询结果：

Cno	COUNT(Sno)
1	22
2	34
3	44
4	33
5	48

GROUP BY子句（续）

[例32] 查询选修了3门以上课程的学生学号。

```
SELECT Sno
FROM  SC
GROUP BY Sno
HAVING  COUNT(*) >3；
```

GROUP BY子句（续）

- HAVING短语与WHERE子句的区别：
  - 作用对象不同
  - WHERE子句作用于基表或视图，从中选择满足条件的元组
  - HAVING短语作用于组，从中选择满足条件的组。

4.5 数 据 更 新

4.5.1 插入数据

4.5.2 修改数据

4.5.3 删除数据

4.5.1 插入数据

- 两种插入数据方式

  1. 插入元组

  2. 插入子查询结果

  - 可以一次插入多个元组

一、插入元组

- 语句格式

  INSERT

  INTO <表名> [(<属性列1>[，<属性列2 >…)]

  VALUES (<常量1> [，<常量2>] …)

- 功能
  - 将新元组插入指定表中

插入元组（续）

- INTO子句
  - 属性列的顺序可与表定义中的顺序不一致
  - 没有指定属性列
  - 指定部分属性列
- VALUES子句
  - 提供的值必须与INTO子句匹配
    - 值的个数
    - 值的类型

插入元组（续）

［例1］ 将一个新学生元组（学号：200215128；姓名：陈冬；性别：男；所在系：IS；年龄：18岁）插入到Student表中。

```
INSERT
INTO  Student (Sno，Sname，Ssex，Sdept，Sage)
VALUES ('200215128'，'陈冬'，'男'，'IS'，18)；
```

插入元组（续）

［例2］ 将学生张成民的信息插入到Student表中。

```
INSERT
        INTO  Student
        VALUES (‘200215126’，  ‘张成民’，  ‘男’，18，'CS');
```

插入元组（续）

［例3］ 插入一条选课记录('200215128'，'1 ')。

```
INSERT
INTO SC(Sno，Cno)
VALUES (‘ 200215128 ’，  ‘ 1 ’);
```

RDBMS将在新插入记录的Grade列上自动地赋空值。

或者：

```
INSERT
INTO SC
VALUES (' 200215128 '，' 1 '，NULL);
```

二、插入子查询结果

- 语句格式

  INSERT

  INTO <表名> [(<属性列1> [，<属性列2>…)]

  子查询；
- 功能

  将子查询结果插入指定表中

插入子查询结果（续）

- INTO子句(与插入元组类似)

- 子查询
  - SELECT子句目标列必须与INTO子句匹配
    - 值的个数
    - 值的类型

插入子查询结果（续）

[例4] 对每一个系，求学生的平均年龄，并把结果存入数据库。

第一步：建表

```
CREATE  TABLE  Dept_age
   (Sdept  CHAR(15)          /* 系名*/
    Avg_age SMALLINT);              /*学生平均年龄*/
```

插入子查询结果（续）

第二步：插入数据

```
INSERT
INTO  Dept_age(Sdept，Avg_age)
    SELECT  Sdept，AVG(Sage)
    FROM  Student
    GROUP BY Sdept；
```

插入子查询结果（续）

RDBMS在执行插入语句时会检查所插元组是否破坏表上已定义的完整性规则

- 实体完整性
- 参照完整性
- 用户定义的完整性
  - NOT NULL约束
  - UNIQUE约束
  - 值域约束

4.5 数 据 更 新

4.5.1 插入数据

4.5.2 修改数据

4.5.3 删除数据

4.4.2 修改数据

- 语句格式

  UPDATE <表名>
  SET <列名>=<表达式>[，<列名>=<表达式>]…
  [WHERE <条件>];

- 功能
  - 修改指定表中满足WHERE子句条件的元组

修改数据（续）

- SET子句
  - 指定修改方式
  - 要修改的列
  - 修改后取值
- WHERE子句
  - 指定要修改的元组
  - 缺省表示要修改表中的所有元组

修改数据（续）

- 三种修改方式
  1. 修改某一个元组的值
  2. 修改多个元组的值
  3. 带子查询的修改语句

1. 修改某一个元组的值

[例5] 将学生200215121的年龄改为22岁

```
UPDATE  Student

SET Sage=22

WHERE  Sno=' 200215121 ';
```

2. 修改多个元组的值

[例6] 将所有学生的年龄增加1岁

```
UPDATE Student

SET Sage= Sage+1；
```

3. 带子查询的修改语句

[例7] 将计算机科学系全体学生的成绩置零。

```
UPDATE SC
SET  Grade=0
WHERE  'CS'=
          (SELETE Sdept
           FROM  Student
           WHERE  Student.Sno = SC.Sno)；
```

修改数据（续）

RDBMS在执行修改语句时会检查修改操作是否破坏表上已定义的完整性规则

- 实体完整性
- 主码不允许修改
- 用户定义的完整性
  - NOT NULL约束
  - UNIQUE约束
  - 值域约束

4.5 数 据 更 新

4.5.1 插入数据

4.5.2 修改数据

4.5.3 删除数据

4.5.3 删除数据

- 语句格式

  DELETE

  FROM　<表名>

  [WHERE <条件>];

- 功能
  - 删除指定表中满足WHERE子句条件的元组
- WHERE子句
  - 指定要删除的元组
  - 缺省表示要删除表中的全部元组，表的定义仍在字典中

删除数据（续）

- 三种删除方式

  1. 删除某一个元组的值

  2. 删除多个元组的值

  3. 带子查询的删除语句

1. 删除某一个元组的值

[例8] 删除学号为200215128的学生记录。

```
DELETE
FROM Student
WHERE Sno= 200215128 ';
```

2. 删除多个元组的值

[例9] 删除所有的学生选课记录。

```
DELETE
FROM SC;
```

3. 带子查询的删除语句

[例10] 删除计算机科学系所有学生的选课记录。

```
DELETE
FROM SC
WHERE  'CS'=
            (SELETE Sdept
             FROM Student
             WHERE Student.Sno=SC.Sno)；
```

一、建立视图

- 语句格式

  CREATE VIEW

  <视图名> [(<列名> [，<列名>]…)]

  AS <子查询>

  [WITH CHECK OPTION]；

- 组成视图的属性列名：全部省略或全部指定
- 子查询不允许含有ORDER BY子句和DISTINCT短语

建立视图（续）

- RDBMS执行CREATE VIEW语句时只是把视图定义存入数据字典，并不执行其中的SELECT语句。
- 在对视图查询时，按视图的定义从基本表中将数据查出。

建立视图（续）

[例1] 建立信息系学生的视图。

```
CREATE VIEW IS_Student
AS
SELECT Sno，Sname，Sage
FROM    Student
WHERE  Sdept= 'IS'；
```

建立视图（续）

[例2]建立信息系学生的视图，并要求进行修改和插入操作时仍需保证该视图只有信息系的学生。

```
CREATE VIEW IS_Student
AS
SELECT Sno，Sname，Sage
FROM  Student
WHERE  Sdept= 'IS'
WITH CHECK OPTION；
```