第三章 关系数据库标准语言SQL

第三章 关系数据库标准语言SQL

- **重点：**

关系模型和关系数据库是《数据库系统概论》课程的重点，第3章又是重点中的重点。要熟练正确的使用SQL完成对数据库的查询、插入、删除、更新操作。在使用具体的SQL时，能有意识地和关系代数、关系演算等语言进行比较，了解他们各自的特点。

- **难点：**

用SQL语言正确完成复杂查询，掌握SQL语言强大的查询功能。因此在学习过程中一定要多练习，要在安装好的数据库系统上进行实际操作，检查你的答案，你查询的结果是否正确。只有通过大量练习才能真正达到举一反三的熟练程度。

3.1 SQL概述

- SQL（Structured Query Language）

  结构化查询语言，是关系数据库的标准语言

- SQL是一个通用的、功能极强的关系数据库语言

3.1 SQL概述

3.1.1 SQL 的产生与发展

3.1.2 SQL的特点

3.1.3 SQL的基本概念

SQL标准的进展过程

标准	大致页数	发布日期
SQL/86		1986.10
SQL/89（FIPS 127-1）	120页	1989年
SQL/92	622页	1992年
SQL99（SQL 3）	1700页	1999年
SQL2003	3600页	2003年
SQL2008	3777页	2006年
SQL2011		2010年
SQL 2016		2016年

目前，没有一个数据库系统能够支持SQL标准的所有概念和特性

3.1 SQL概述

3.1.1 SQL 的产生与发展

3.1.2 SQL的特点

3.1.3 SQL的基本概念

3.1.2 SQL的特点

1. 综合统一

- 集数据定义语言（DDL），数据操纵语言（DML），数据控制语言（DCL）功能于一体。
- 可以独立完成数据库生命周期中的全部活动：
  - 定义和修改、删除关系模式，定义和删除视图，插入数据，建立数据库;
  - 对数据库中的数据进行查询和更新;
  - 数据库重构和维护
  - 数据库安全性、完整性控制，以及事务控制
  - 嵌入式SQL和动态SQL定义
- 用户数据库投入运行后，可根据需要随时逐步修改模式，不影响数据库的运行。
- 数据操作符统一

2. 高度非过程化

- 非关系数据模型的数据操纵语言“面向过程”，必须指定存取路径。
- SQL只要提出“做什么”，无须了解存取路径。
- 存取路径的选择以及SQL的操作过程由系统自动完成。

3. 面向集合的操作方式

- 非关系数据模型采用面向记录的操作方式，操作对象是一条记录
- SQL采用集合操作方式
  - 操作对象、查找结果可以是元组的集合
  - 一次插入、删除、更新操作的对象可以是元组的集合

4. 以同一种语法结构提供多种使用方式

- SQL是独立的语言

  能够独立地用于联机交互的使用方式

- SQL又是嵌入式语言

  SQL能够嵌入到高级语言（例如C，C++，Java）程序中，供程序员设计程序时使用

5. 语言简洁，易学易用

SQL功能极强，完成核心功能只用了9个动词。

表 3.2　SQL 的动词

SQL 功　能	动词
数 据 查 询	SELECT
数 据 定 义	CREATE，DROP，ALTER
数 据 操 纵	INSERT，UPDATE，DELETE
数 据 控 制	GRANT，REVOKE

3.1 SQL概述

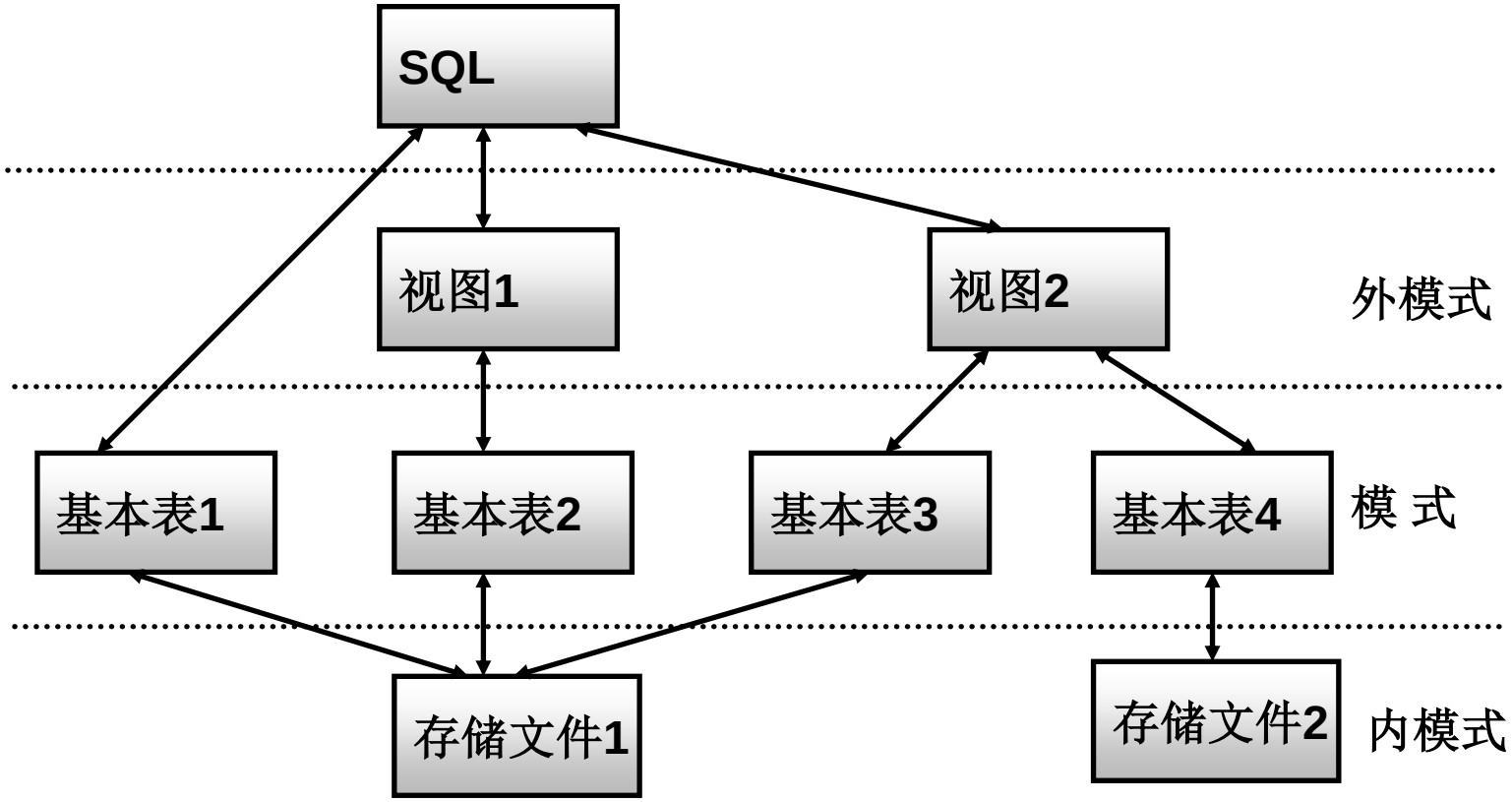
3.1.1 SQL 的产生与发展

3.1.2 SQL的特点

3.1.3 SQL的基本概念

SQL的基本概念（续）

SQL支持关系数据库三级模式结构



SQL的基本概念（续）

- 基本表——（模式）
  - 本身独立存在的表
  - SQL中一个关系就对应一个基本表
  - 一个（或多个）基本表对应一个存储文件
  - 一个表可以带若干索引

- 模式：

```
8  create table Student (
9      SNo int primary key,
10     SName char(20),
11     SSex char(1),
12     SAge int,
13     SDept char(20)
14 );
```

```
15 create table Award (
16     SNo int,
17     AName char(40),
18     primary key(Sno, AName),
19     foreign key SNo references  Student(SNo)
20 );
21 **/
```

学号	姓名	性别	年龄	院系

学号	获得奖项

SQL的基本概念（续）

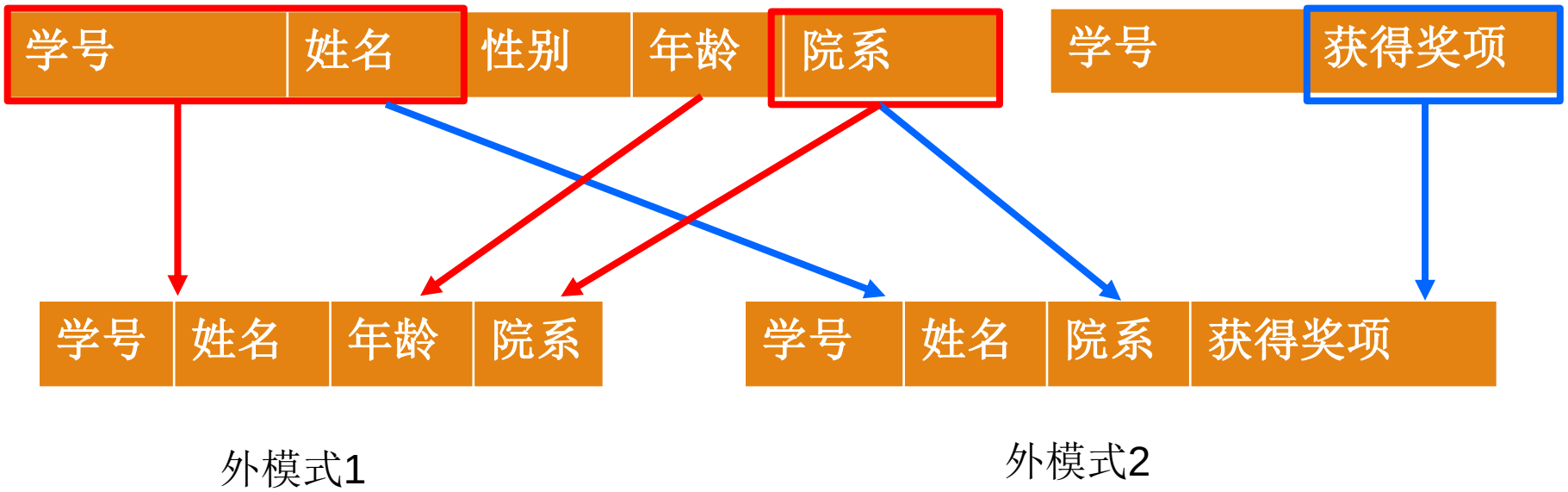
- 存储文件（内模式）
  - 存储文件的逻辑结构组成了关系数据库的内模式
  - 存储文件的物理结构对用户是隐蔽的
  - 索引也可存放在存储文件中

- 内模式：
- 记录的存储方式，比如某存储文件中
  - 按姓名顺序存放
  - 按院系顺序存放
  - 按学号顺序存放
- 索引的组织方式
- 数据是否压缩存储

SQL的基本概念（续）

- 视图（外模式）
  - 从一个或几个基本表导出的表
  - 数据库中只存放视图的定义而不存放视图对应的数据（数据仍存在基本表中）
  - 视图是一个虚表
  - 用户可以在视图上再定义视图

- 外模式：
- 从模式导出的一个子集



第三章 关系数据库标准语言SQL

3.1 SQL概述

3.2 学生-课程数据库

3.3 数据定义

3.4 数据查询

3.5 数据更新

3.6 空值的处理

3.7 视图

3.8 小结

3.2 学生-课程 数据库

- 学生-课程模式 S-T :

学生表：Student(Sno,Sname,Ssex,Sage,Sdept)

课程表：Course(Cno,Cname,Cpno,Ccredit)

学生选课表：SC(Sno,Cno,Grade)

Student表

学号 Sno	姓名 Sname	性别 Ssex	年龄 Sage	所在系 Sdept
201215121	李勇	男	20	CS
201215122	刘晨	女	19	CS
201215123	王敏	女	18	MA
201215125	张立	男	19	IS

Course表

课程号 Cno	课程名 Cname	先行课 Cpno	学分 Ccredit
1	数据库	5	4
2	数学		2
3	信息系统	1	4
4	操作系统	6	3
5	数据结构	7	4
6	数据处理		2
7	PASCAL语言	6	4

SC表

学 号 Sno	课程号 Cno	成绩 Grade
201215121	1	92
201215121	2	85
201215121	3	88
201215122	2	90
201215122	3	80

第三章 关系数据库标准语言SQL

3.1 SQL概述

3.2 学生-课程数据库

3.3 数据定义

3.4 数据查询

3.5 数据更新

3.6 空值的处理

3.7 视图

3.8 小结

3.3 数据定义

- SQL的数据定义功能:
  - 模式定义
  - 表定义
  - 视图和索引的定义

表 3.3　SQL 的数据定义语句

操作对象	操作方式		
	创　建	删　除	修　改
模式	CREATE SCHEMA	DROP SCHEMA	
表	CREATE TABLE	DROP TABLE	ALTER TABLE
视图	CREATE VIEW	DROP VIEW	
索引	CREATE INDEX	DROP INDEX	ALTER INDEX

3.3 数据定义

3.3.1 模式的定义与删除

3.3.2 基本表的定义、删除与修改

3.3.3 索引的建立与删除

SQL语句格式的约定符号

- 语句格式中,<> 中的内容是必须的，是用户自定义语义;
- []为任选项
- 分隔符|表示必选项,即必选其中之一项
- [,...]表示前面的项可以重复多次

1. 定义模式

[例3.1] 为用户WANG定义一个学生-课程模式S_T

CREATE SCHEMA “S_T” AUTHORIZATION WANG;

[例3.2] CREATE SCHEMA AUTHORIZATION WANG;

该语句没有指定<模式名>，<模式名>隐含为<用户名>

定义模式（续）

- 定义模式实际上定义了一个命名空间。
- 在这个空间中可以定义该模式包含的数据库对象，例如基本表、视图、索引等。
- 在CREATE SCHEMA中可以接受CREATE TABLE，CREATE VIEW和GRANT子句。

CREATE SCHEMA <模式名> AUTHORIZATION <用户名>[<表定义子句>|<视图定义子句>|<授权定义子句>]

定义模式（续）

[例3.3]为用户ZHANG创建了一个模式TEST，并且在其中定义一个表TAB1

```
CREATE SCHEMA TEST AUTHORIZATION ZHANG
CREATE TABLE TAB1   ( COL1 SMALLINT,
                      COL2 INT,
                      COL3 CHAR(20),
                      COL4 NUMERIC(10,3),
                      COL5 DECIMAL(5,2)
                    );
```

2. 删除模式

- DROP SCHEMA <模式名> <CASCADE|RESTRICT>
  - CASCADE（级联）
    - ✓删除模式的同时把该模式中所有的数据库对象全部删除
  - RESTRICT（限制）
    - ✓如果该模式中定义了下属的数据库对象（如表、视图等），则拒绝该删除语句的执行。
    - ✓仅当该模式中没有任何下属的对象时才能执行。

请注意：本章介绍的是标准库的SQL的语法，但是在具体的DBMS中，有各自特有的语法要求，比如 SQL SERVER 中的DROP SCHEMA 没有CASCADE和RESTRICT的区分。
目前，没有一个数据库系统能够支持SQL标准的所有概念和特性。

删除模式（续）

[例3.4] DROP SCHEMA TEST CASCADE;

删除模式TEST

同时该模式中定义的表TAB1也被删除

3.3 数据定义

3.3.1 模式的定义与删除

3.3.2 基本表的定义、删除与修改

3.3.3 索引的建立与删除

3.3.2 基本表的定义、删除与修改

- 定义基本表

CREATE TABLE <表名>
(<列名> <数据类型>[<列级完整性约束条件>]
[,<列名> <数据类型>[<列级完整性约束条件>]]
…
[,<表级完整性约束条件>]);

- <表名>：所要定义的基本表的名字
- <列名>：组成该表的各个属性（列）
- <列级完整性约束条件>：涉及相应属性列的完整性约束条件
- <表级完整性约束条件>：涉及一个或多个属性列的完整性约束条件
- 如果完整性约束条件涉及到该表的多个属性列，则必须定义在表级上，否则既可以定义在列级也可以定义在表级。

学生表Student

[例3.5] 建立“学生”表Student。学号是主码，姓名取值唯一。

主码

```
CREATE TABLE Student
    (Sno   CHAR(9) PRIMARY KEY,
                          /* 列级完整性约束条件,Sno是主码*/
     Sname CHAR(20) UNIQUE,          /* Sname取唯一值*/
     Ssex    CHAR(2),
     Sage   SMALLINT,
     Sdept  CHAR(20)
    );
```

UNIQUE 约束

课程表Course

[例3.6] 建立一个“课程”表Course

```
CREATE TABLE  Course
        (Cno       CHAR(4) PRIMARY KEY,
         Cname  CHAR(40),
         Cpno     CHAR(4),
         Ccredit  SMALLINT，
         FOREIGN KEY (Cpno) REFERENCES
Course(Cno)
        );
```

先修课

表级完整性约束条件，Cpno是外码
被参照表是Course
被参照列是Cno

参照表和被参照表可以是同一个表！

学生选课表SC

[例3.7] 建立一个学生选课表SC

```
CREATE TABLE  SC
        (Sno  CHAR(9),
         Cno  CHAR(4),
         Grade  SMALLINT，
         PRIMARY KEY (Sno,Cno),
                  /* 主码由两个属性构成，必须作为表级完整性进行定义*/
         FOREIGN KEY (Sno) REFERENCES Student(Sno),
                  /* 表级完整性约束条件，Sno是外码，被参照表是Student */
         FOREIGN KEY (Cno)REFERENCES Course(Cno)
                  /* 表级完整性约束条件， Cno是外码，被参照表是Course*/
       );
```

2. 数据类型

- SQL中域的概念用数据类型来实现
- 定义表的属性时需要指明其数据类型及长度
- 选用哪种数据类型
  - 取值范围
  - 要做哪些运算 （比如年龄运算不使用CHAR类型）

数据类型（续）

数据类型	含义
CHAR(*n*),CHARACTER(*n*)	长度为n的定长字符串
VARCHAR(*n*), CHARACTERVARYING(*n*)	最大长度为n的变长字符串
CLOB	字符串大对象
BLOB	二进制大对象
INT，INTEGER	长整数（4字节）
SMALLINT	短整数（2字节）
BIGINT	大整数（8字节）
NUMERIC(*p*，*d*)	定点数，由*p*位数字（不包括符号、小数点）组成，小数后面有*d*位数字
DECIMAL(*p*, *d*), DEC(*p*, *d*)	同NUMERIC
REAL	取决于机器精度的单精度浮点数
DOUBLE PRECISION	取决于机器精度的双精度浮点数
FLOAT(*n*)	可选精度的浮点数，精度至少为*n*位数字
BOOLEAN	逻辑布尔量
DATE	日期，包含年、月、日，格式为YYYY-MM-DD
TIME	时间，包含一日的时、分、秒，格式为HH:MM:SS
TIMESTAMP	时间戳类型
INTERVAL	时间间隔类型

3. 模式与表

- 每一个基本表都属于某一个模式
- 一个模式包含多个基本表
- 定义基本表所属模式

  - 可在创建模式语句中同时创建表 如例3.3

4. 修改基本表

ALTER TABLE <表名>

[ADD[COLUMN] <新列名> <数据类型> [完整性约束]]

[ADD <表级完整性约束>]

[DROP [COLUMN] <列名> [CASCADE| RESTRICT]]

[DROP CONSTRAINT<完整性约束名>[RESTRICT | CASCADE]]

[ALTER COLUMN <列名><数据类型>] ;

修改基本表（续）

- <表名>是要修改的基本表
- ADD子句用于增加新列、新的列级完整性约束条件和新的表级完整性约束条件
- DROP COLUMN子句用于删除表中的列
  - 如果指定了CASCADE短语，则自动删除引用了该列的其他对象
  - 如果指定了RESTRICT短语，则如果该列被其他对象引用，关系数据库管理系统将拒绝删除该列
- DROP CONSTRAINT子句用于删除指定的完整性约束条件
- ALTER COLUMN子句用于修改原有的列定义，包括修改列名和数据类型

修改基本表（续）

[例3.8] 向Student表增加“入学时间”列，其数据类型为日期型

```
ALTER TABLE Student ADD S_entrance DATE;
```

不管基本表中原来是否已有数据，新增加的列一律为空值

修改基本表（续）

[例3.9] 将年龄的数据类型由字符型（假设原来的数据类型是字符型）改为整数。

```
ALTER TABLE Student ALTER COLUMN Sage INT;
```

[例3.10] 增加课程名称必须取唯一值的约束条件。

```
ALTER TABLE Course ADD UNIQUE(Cname);
```

5. 删除基本表

DROP TABLE <表名>［RESTRICT| CASCADE］;

- RESTRICT：删除表是有限制的。
  - 欲删除的基本表不能被其他表的约束所引用
  - 如果存在依赖该表的对象，则此表不能被删除
- CASCADE：删除该表没有限制。
  - 在删除基本表的同时，相关的依赖对象一起删除

删除基本表（续）

[例3.11] 删除Student表

```
DROP TABLE Student CASCADE;
```

- 基本表定义被删除，数据被删除
- 表上建立的索引、视图、触发器等一般也将被删除

删除基本表（续）

[例3.12]若表上建有视图，选择RESTRICT时表不能删除;选择CASCADE时可以删除表，视图也自动删除。

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

/*标准的SQL语言执行上面的语句后，再执行下面这句话，因为在该表上建有视图，若RESTRICT删除表会出现ERROR*/

```
 DROP TABLE Student RESTRICT;
--ERROR: cannot drop table Student because other objects depend on it
```

删除基本表（续）

[例3.12续]如果选择CASCADE时可以删除表，视图也自动被删除

```
DROP TABLE Student CASCADE;

 --NOTICE: drop cascades to view IS_Student
```

/*标准的SQL语言执行上面的语句后，再执行下面这句话，会出现ERROR,因为已经级联删除*/

```
SELECT * FROM IS_Student;

--ERROR: relation " IS_Student " does not exist
```

- 在DROP TABLE时，关于删除策略，RESTRICT 、CASCADE具体执行时，标准库的SQL语言与某些具体的DBMS是不一定一致的，并且不同的DBMS处理策略也可能不一样，具体如下表。
- 下表中可见，SQL SERVER的drop table 命令不分RESTRICT与CASCADE命令。

删除基本表（续）

DROP TABLE时，SQL2011 （2011版标准库文件）与3个DBMS的处理策略比较

序号	标准及主流数据库的处理方式 / 依赖基本表的对象	SQL2011		Kingbase ES		Oracle 12c		MS SQL Server 2012
		R	C	R	C		C	
1	索引	无规定		√	√	√	√	√
2	视图	×	√	×	√	√ 保留	√ 保留	√ 保留
3	DEFAULT，PRIMARY KEY，CHECK（只含该表的列）NOT NULL 等约束	√	√	√	√	√	√	√
4	外码FOREIGN KEY	×	√	×	√	×	√	×
5	触发器TRIGGER	×	√	×	√	√	√	√
6	函数或存储过程	×	√	√ 保留	√ 保留	√ 保留	√ 保留	√ 保留

R表示RESTRICT , C表示CASCADE

'×'表示不能删除基本表，'√'表示能删除基本表，‘保留’表示删除基本表后，还保留依赖对象

“×”表示不能删除基本表，“√”表示能删除基本表，“保留”表示删除基本表后，还保留依赖对象。从比较表中可以知道：

（1）对于索引，删除基本表后，这三个关系数据库管理系统都自动删除该基本表上已经建立的所有索引。

（2）对于视图，Oracle 12c 与 SQL Server 2012 是删除基本表后，还保留此基本表上的视图定义，但是已经失效。Kingbase ES 分两种情况，若删除基本表时带 RESTRICT 选项，则不可以删除基本表；若删除基本表时带 CASCADE 选项，则可以删除基本表，同时也删除视图。Kingbase ES 的这种策略符合 SQL 2011 标准。

（3）对于存储过程和函数，删除基本表后，这三个数据库产品都不自动删除建立在此基本表上的存储过程和函数，但是已经失效。

（4）如果欲删除的基本表上有触发器，或者被其他基本表的约束所引用（CHECK，FOREIGN KEY 等），读者可以从比较表中得到这三个系统的处理策略，这里就不一一说明了。

同样，对于其他的 SQL 语句，不同的数据库产品在处理策略上会与标准有所差别。因此，如果发现本书中个别例子在某个数据库产品上不能通过时，请读者参见有关产品的用户手册，适当修改即可。

3.3 数据定义

3.3.1 模式的定义与删除

3.3.2 基本表的定义、删除与修改

3.3.3 索引的建立与删除

3.3.3 索引的建立与删除

- 建立索引的目的：加快查询速度
- 关系数据库管理系统中常见索引：
  - 顺序文件上的索引
  - B+树索引
  - 散列（hash）索引
  - 位图索引
- 特点：
  - B+树索引具有动态平衡的优点
  - HASH索引具有查找速度快的特点

索　引

- 谁可以建立索引
  - 数据库管理员 或 表的属主（即建立表的人）
- 谁维护索引
  - 关系数据库管理系统自动完成
- 使用索引
  - 关系数据库管理系统自动选择合适的索引作为存取路径，用户不必也不能显式地选择索引

1. 建立索引

- 语句格式

CREATE [UNIQUE] [CLUSTER] INDEX <索引名>

ON <表名>(<列名>[<次序>][,<列名>[<次序>]]…);

- <表名>：要建索引的基本表的名字
- 索引：可以建立在该表的一列或多列上，各列名之间用逗号分隔
- <次序>：指定索引值的排列次序，升序：ASC，降序：DESC。缺省值：ASC
- UNIQUE：此索引的每一个索引值只对应唯一的数据记录
- CLUSTER：表示要建立的索引是聚簇索引

建立索引（续）

[例3.13] 为学生-课程数据库中的Student，Course，SC三个表建立索引。Student表按学号升序建唯一索引，Course表按课程号升序建唯一索引，SC表按学号升序和课程号降序建唯一索引

```
CREATE UNIQUE INDEX  Stusno ON Student(Sno);

CREATE UNIQUE INDEX  Coucno ON Course(Cno);

CREATE UNIQUE INDEX  SCno ON SC(Sno ASC,Cno DESC);
```

2. 修改索引

- ALTER INDEX <旧索引名> RENAME TO <新索引名>

  - [例3.14] 将SC表的SCno索引名改为SCSno

    ALTER INDEX SCno RENAME TO SCSno;

3. 删除索引

- DROP INDEX <索引名>;

  删除索引时，系统会从数据字典中删去有关该索引的描述。

  [例3.15] 删除Student表的Stusname索引

  ```
  DROP INDEX Stusname;
  ```