第3章 SQL之数据定义

本章目标

- 完成本章的学习，你应该能够
  - 了解SQL的发展历史、标准和特点
  - 熟练掌握使用SQL语句
    - 创建、更改和删除数据库、模式和基本表
    - 理解视图的作用，掌握视图的创建、使用和删除等基本功能
    - 理解并掌握索引的设计、创建、使用和维护等功能
    - 完成各类查询操作(单表查询、连接查询、嵌套查询和集合查询)
    - 完成更新操作(插入数据、修改数据，删除数据)
  - 理解和掌握常用系统函数的使用方法

大纲

- SQL概述
- 学生-课程数据库
- 数据定义
- 数据查询
- 数据更新
- 空值的处理
- 视图
- 本章小结

SQL概述

- SQL语言(Structured Query Language)
  - Pronounced /ɛskjuːˈ ɛl /; unofficially /ˈ siːkwəl/
  - 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
    - 不仅仅是查询，还包括数据库模式创建、数据库数据的插入与修改、数据库安全性和完整性定义与控制等功能
  - SQL作为共同的数据存取语言和标准接口，使不同数据库系统之间的互操作有了共同的基础

1.SQL的产生和发展

- 1974年，Boyce和Chamberlin提出 SQL 标准。
- 1975年～1979年IBM公司在System R原型系统上实现。

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
SQL2016		2016年
SQL2019		2019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。
- 许多软件厂商对SQL基本命令集进行不同程度的扩充和修改，使之可以支持标准以外的一些功能。

2. SQL的5个特点

① 综合统一

② 高度非过程化

③ 面向集合的操作方式

④ 以同一种语法结构提供两种使用方法

⑤ 语言简洁，易学易用

1.综合统一

- 集数据定义语言，数据操纵语言，数据控制语言功能于一体
- 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
- 大大减轻了用户负担，而且有利于提高数据独立性

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两种使用方式

- SQL是独立的语言
  - 能够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。
  - 通常使用JDBC或ODBC中间件实现访问
- 在上述两种不同的使用方式下，SQL的语法结构基本一致，这提供了极大的灵活性和方便性

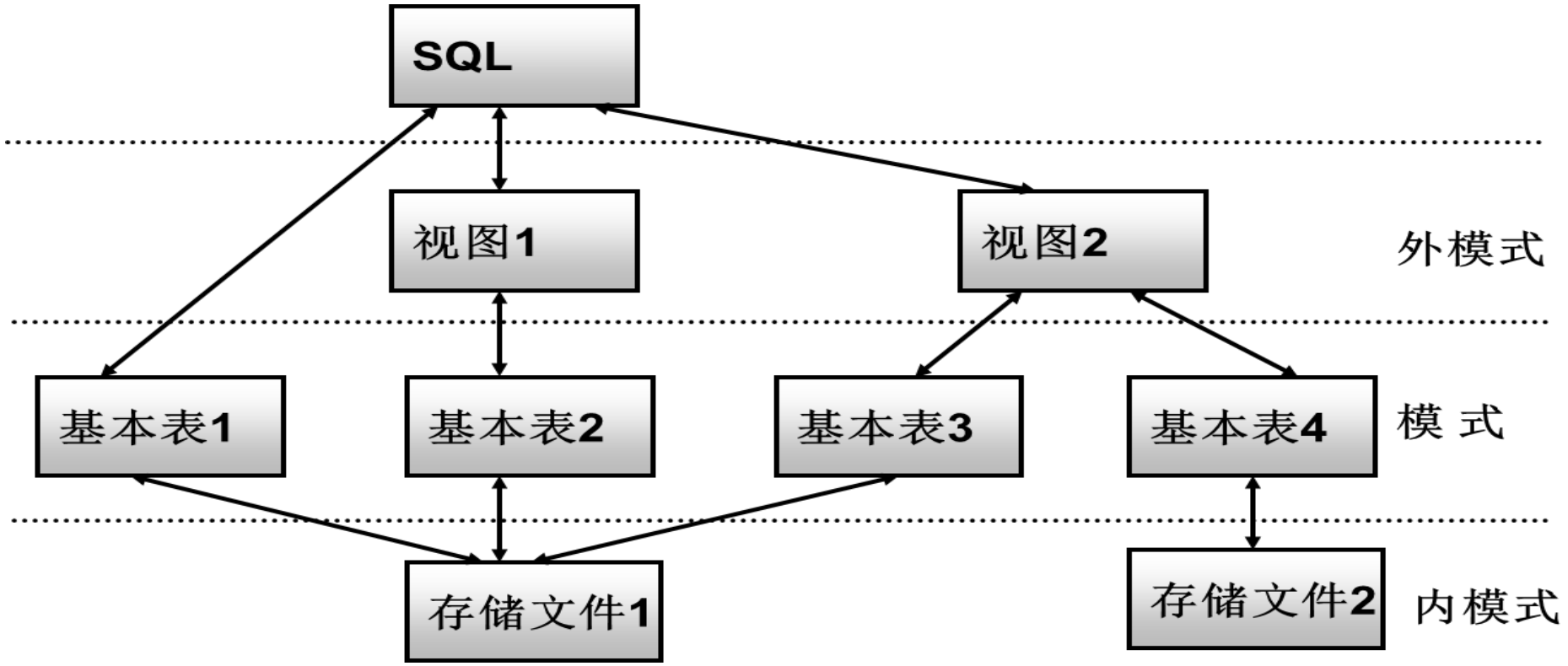
- 5.语言简洁，易学易用
  - SQL功能极强，完成核心功能只用了9个动词

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. SQL的基本概念

- SQL支持关系数据库的三级模式结构



- 基本表(base table)
  - 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
- 存储文件(stored file)
  - 物理结构组成了关系数据库的内模式
  - 物理结构对用户是隐蔽的
- 视图(view)
  - 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
  - 视图是一个虚表
  - 用户可以在视图上再定义视图

大纲

- SQL概述
- **学生-课程数据库**
- 数据定义
- 数据查询
- 数据更新
- 空值的处理
- 视图
- 本章小结

学生-课程S-T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)

课程表

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

学生表

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

选课表

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

大纲

- SQL概述
- 学生-课程数据库
- **数据定义**
- 数据查询
- 数据更新
- 空值的处理
- 视图
- 本章小结

数据定义

- SQL的数据定义功能:
  - 模式定义(Schema)
  - 表定义(Table)
  - 视图(View)和索引(Index)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

1.模式(Schema)

- 现代关系数据库管理系统提供了一个层次化的数据库对象命名机制。
  - 一个关系数据库管理系统的实例中可以建立多个数据库。
  - 一个数据库中可以建立多个模式。
- 定义模式实际上就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(基本表、视图、索引、触发器、存储过程、函数和包等)。
  - 简而言之，模式就是数据库对象的集合。

模式定义(创建)

CREATE SCHEMA <模式名> AUTHORIZATION <用户名>;

- 创建模式必须具有DBA权限，或获得了DBA授予的CREATE SCHEMA权限
- 若模式名缺失，则模式名默认为用户名
- CREATE SCHEMA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>]

示例：

[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;

[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));
```

模式删除

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。
- [例3.4]： DROP SCHEMA TEST CASCADE; --删除模式TEST及该模式中定义的表TAB1

openGauss之用户、角色和用户组

- openGauss使用用户USER和角色role来控制对数据库的访问。
- 根据角色自身的设置不同，一个角色可以看做是一个数据库用户，或者一组数据库用户。
- 在openGauss中角色和用户之间的区别只在于角色默认是没有LOGIN权限的。
- 在openGauss中一个用户唯一对应一个角色，不过可以使用角色叠加来更灵活地进行管理。

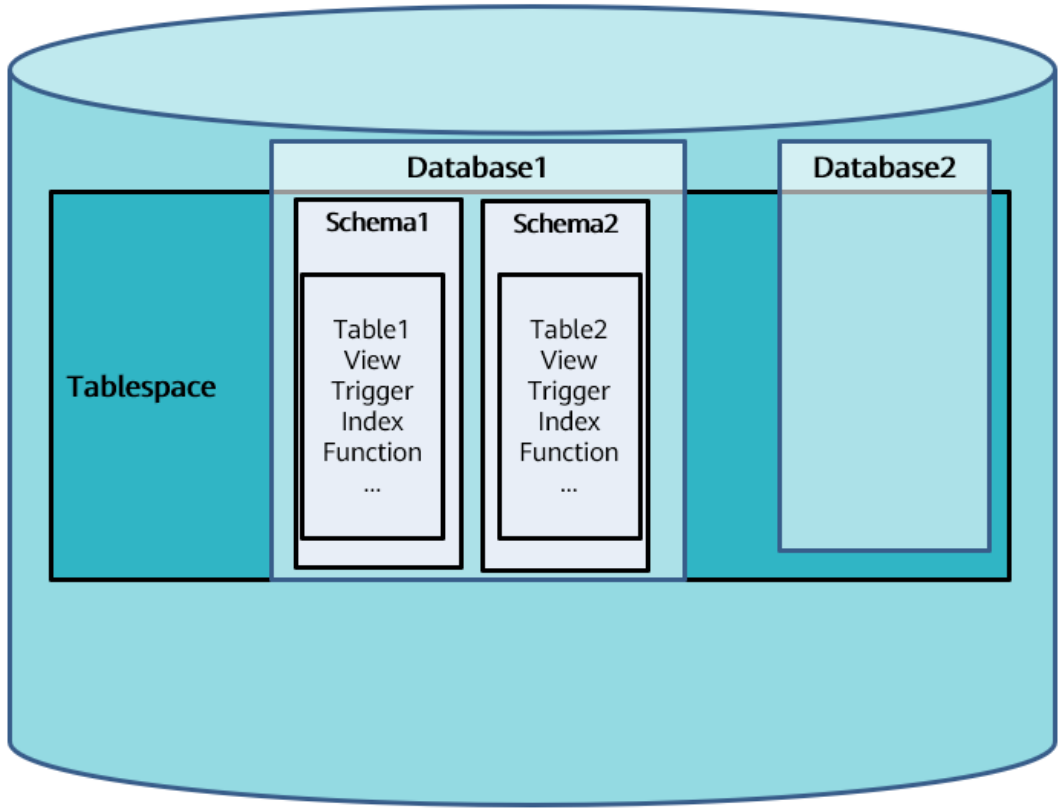
- 为了实现安装过程中安装帐户权限最小化及安装后openGauss的系统运行安全性。安装脚本在安装过程中会自动按照用户指定内容创建安装用户，并将此用户作为后续运行和维护openGauss的管理员帐户.

用户/组名	所属类型	规划建议
dbgrp	操作系统	建议规划单独的用户组，例如dbgrp。 初始化安装环境时，由-G参数所指定的安装用户所属的用户组。该用户组如果不存在，则会自动创建，也可提前创建好用户组。在执行gs_preinstall脚本时会检查权限。 gs_preinstall脚本会自动赋予此组中的用户对安装目录、数据目录的访问和执行权限。 创建dbgrp用户组命令：groupadd dbgrp
omm	操作系统	建议规划用户用于运行和维护openGauss，例如omm。 初始化安装环境时，由-U参数所指定和自动创建的操作系统用户，如果已经存在该用户，请清理该用户或更换初始化用户。从安全性考虑，对此用户的所属组规划如下：所属组： dbgrp

在安装openGauss过程中运行“ gs_install”时，会创建与安装用户同名的数据库用户，即数据库用户omm。此用户具备数据库的最高操作权限，此用户初始密码由用户指定。

- 关于omm的权限可参见：https://www.modb.pro/db/9151
- openGauss创建用户密码规则如下：
  - 密码默认不少于 8 个字符
  - 不能与用户名及用户名倒序相同
  - 至少包含大写字母(A-Z)，小写字母(a-z)，数字(0-9)，非字母数字字符(限定为~!@#$%^&*()-_=+\|[{}];:,<.>/?)四类字符中的三类字符
  - 创建用户时，应当使用双引号或单引号将用户密码括起来

openGauss之表空间、数据库与模式



- 表空间Tablespace对应磁盘上的一个目录，里面存储的是它所包含的数据库的各种物理文件。
- 可以存在多个表空间，每个表空间可以对应多个Database。
- 表空间仅是起到了物理隔离的作用，其管理功能依赖于文件系统。
- 两个默认表空间：pg_default、pg_global

- 数据库Database用于管理各类数据对象，各数据库间相互隔离。数据库管理的对象可分布在多个Tablespace上。
- 创建数据对象时可以指定对应的表空间，如果不指定相应的表空间，相关的对象会默认保存在PG_DEFAULT空间中。

openGauss之模式

- openGauss的模式Schema是对数据库做一个逻辑分割。所有的数据库对象都建立在模式下面。数据库对象包括：
  - 表、视图、索引、触发器、函数、包、存储过程等。
- openGauss的模式和用户是弱绑定的。
  - 所谓的弱绑定是指创建用户的同时会自动创建一个同名模式，但用户也可以单独创建模式，并且为用户指定其他的模式。
- 通过管理 Schema，允许多个用户使用同一数据库而不相互干扰
- 每个数据库包含一个或多个Schema

openGauss之SQL

- 支持标准的SQL92/SQL99/SQL2003/SQL2011规范，支持GBK和UTF-8字符集，支持SQL标准函数与分析函数，支持存储过程。
- openGauss之SQL学习网址：

  https://education.huaweicloud.com/courses/course-v1:HuaweiX+CBUCNXDR006+Self-paced/courseware/b86d3987a1294dc6b066a40b86e0032d/cad675e2e02e440587e1cbd6d518e187/

  – 请课后自行完成这部分内容的学习(非常重要！)。
- openGauss之SQL语句书写规范建议：
  – 因SQL语句大小写不敏感，关键字大写，其他小写
  – 在gsql中，为了提升操作效率，可以全部使用小写
  – 使用行缩进增强可读性

- openGauss的命名规范：
  - openGauss的命名规范遵循postgreSQL规定
    - 库名、表名限制命名长度，建议表名及字段名字符总长度不超过63;【强制】
    - 对象名（表名、列名、函数名、视图名、序列名等对象名称）规范，对象名务必只使用小写字母，下划线，数字。不要以pg开头，不要以数字开头，不要使用保留字;【强制】
    - query中的别名不要使用 "小写字母，下划线，数字" 以外的字符，例如中文.【强制】
  - 参考：https://www.cnblogs.com/panpanwelcome/p/12430122.html

2.基本表的定义、删除和修改

- 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> ] );
```

  - 完整性约束条件在定义后被存入系统的数据字典中，并在对相关数据进行操作时被系统用于自动检查是否满足这些约束条件.

示例：

[例3.5] 建立一张“学生(Student)表”，学号是主码，姓名取值唯一。

```
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)
        );
```

[例3.6] 建立一张“课程Course表”。

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

[例3.7] 建立一张学生“选课SC表”。

```
CREATE TABLE SC
       (Sno      CHAR(9),
        Cno      CHAR(4),
        Grade    SMALLINT,
        PRIMARY KEY(Sno,Cno),
        FOREIGN KEY(Sno)REFERENCES Student(Sno),
        FOREIGN KEY(Cno)REFERENCES Course(Cno)
        );
```

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
- 注意！
  - 不同的关系数据库管理系统中支持的数据类型不完全相同。
  - openGauss的内建数据类型可参考华为openGauss文档

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

模式与表

- 一个模式包含多个基本表，每个基本表都属于某一个模式。
- 定义基本表所属模式的三种方法：
  - 方法一：在表名中显式地给出模式名
    - Create table "S-T".Student(.....); /*模式名为 S-T*/
    - Create table "S-T".Course(.....);
    - Create table "S-T".SC(.....); /*模式名与表名之间用圆点连接，无空格*/
  - 方法二：在创建模式语句中同时创建表
  - 方法三：设置所属的模式

Tips：在openGauss中，指明表时应把该表所属的模式也加上去

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>] ;
```

示例：

[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

[例3.9] 将年龄的数据类型由字符型(假设这是原数据类型)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);
```

删除基本表

DROP TABLE <表名> [RESTRICT | CASCADE];

- RESTRICT：删除表有限制，为默认值。
  - 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：删除表没有限制
  - 在删除基本表的同时，相关的依赖对象一起被删除

示例：

[例3.11] 删除Student表.

```
DROP TABLE Student CASCADE;
```

[例3.12] 若表上建有视图，选择RESTRICT时表不能删除（下图一）；
选择CASCADE时可以删除表，视图也自动删除（下图二）.

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

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

例图一

```
DROP TABLE Student CASCADE;
 --NOTICE: drop cascades to view IS_Student

SELECT * FROM IS_Student;
--ERROR: relation " IS_Student " does not exist
```

例图二

DROP TABLE时，SQL2011 与 3个RDBMS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

3.索引的建立与修改

- 索引(Index)在数据库的三级模式结构中属于内模式的范畴，它能够帮助快速找到满足条件的数据。
- 索引的结构：

search-key	pointer

- 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(具有动态平衡的优点)
  - 散列(hash)索引(具有查找速度快的特点)
  - 位图(Bitmap)索引

- 谁可以建立索引?
  - 数据库管理员(DBA)或表的属主(即建立表的人Owner)
  - DBMS一般会自动建立以下列上的索引：PRIMARY KEY，UNIQUE
- 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。

建立索引

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
  - 对于已含重复值的属性列不能建UNIQUE索引。
  - 对某个列建立UNIQUE索引后，插入新记录时DBMS会自动检查新记录在该列上是否取了重复值，这相当于增加了一个UNIQUE约束。
- CLUSTER：表示要建立的索引是聚簇索引

示例：

[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);
```

```
CREATE UNIQUE INDEX pg_job_id_index ON pg_job USING btree (job_id) TABLESPACE pg_global
CREATE INDEX gs_asp_sampletime_index ON gs_asp USING btree (sample_time) TABLESPACE pg_default
```

openGauss索引示例

索引修改

ALTER INDEX <旧索引名> RENAME TO <新索引名>;

- 上述命令将索引改名。

[例3.14] 将SC表的SCno索引名改为SCSno。

```
ALTER INDEX SCno RENAME TO SCSno;
```

索引删除

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;

- 可以通过对索引进行先删后建的方式实现索引的修改。

4.数据字典

- 数据字典(Data dictionary)是关系数据库管理系统内部的一组系统表，它记录了数据库中所有定义信息：
  - 关系模式定义
  - 视图定义
  - 索引定义
  - 完整性约束定义
  - 各类用户对数据库的操作权限
  - 统计信息等
- 关系数据库管理系统在执行SQL的数据定义语句时，实际上就是在更新数据字典表中的相应信息

openGauss的系统表

- 除了创建的表以外，数据库还包含很多系统表。这些系统表包含openGauss安装信息以及openGauss上运行的各种查询和进程的信息。可以通过查询系统表来收集有关数据库的信息。

- openGauss提供了以下类型的系统表和视图：
  - 继承自PG的系统表和视图
    - 这类系统表和视图具有PG前缀
  - openGaussl新增的系统表和视图
    - 这类系统表和视图具有GS前缀

openGauss系统表的查看

- 通过系统表的查看可以掌握openGauss数据库的相关信息。

[例]：在PG_TABLES系统表中查看public schema中包含的所有表。

```
SELECT distinct(tablename) FROM pg_tables WHERE SCHEMANAME = 'public';
```

[例]：通过PG_USER可以查看数据库中所有用户的列表，还可以查看用户ID(USESYSID)和用户权限。

```
SELECT * FROM pg_user;
```

本章作业

- 教材第三章习题：1-9题(全部)