



SQL SERVER 2005

# 数据库系统概论

参考：第三章 关系数据库标准语言SQL P<sub>78</sub> - P<sub>127</sub>

# 本章内容

- 第一节 SQL概述
- 第二节 学生-课程数据库
- 第三节 数据定义
- 第四节 数据查询
- 第五节 数据更新
- 第六节 视图
- 第七节 小结



# 本节教学目标

## ❖ 掌握

- CREATE DATABASE    CREATE SCHEMA
- CREATE TABLE    CREATE INDEX

## ❖ 了解

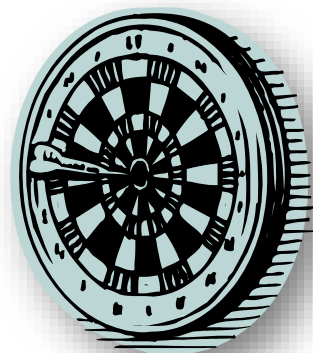
- SQL语言的历史、特点

## ❖ 重点

- 模式、表、索引的创建、删除

## ❖ 难点

- 模式



# 第一节 SQL概述

## ❖ SQL

- SQL语言原名SEQUEL（读作[si:kwəl]），是一个通用的、功能极强的关系数据库语言。同时也是一种介于关系代数与关系演算之间的结构化查询语言（Structured Query Language），其功能包括数据定义、数据查询、数据操纵和数据控制。

## ❖ 为什么学习SQL

- SQL已经成为关系数据库的**查询标准**；
- SQL也是现在和将来DBMS的标准；
- SQL促进了分布式数据库和客户/服务器数据库的开发。



# SQL的产生与发展

- ❖ 最早的SQL原型由IBM的研究人员在20世纪70年代开发的
- ❖ 20世纪80年代早期SQL开始成为国际标准的数据库语言：

标准	发布日期
SQL/86	1986.10
SQL/89(FIPS 127-1)	1989年
SQL/92	1992年
SQL99	1999年
SQL2003	2003年

# SQL的特点

## ❖ SQL的特点

- 综合统一
- 高度非过程化
- 面向集合的操作方式
- 两种使用方式，统一的语法结构
- 简洁易学



# 本章内容

- 第一节 SQL概述
- 第二节 学生-课程数据库
- 第三节 数据定义
- 第四节 数据查询
- 第五节 数据更新
- 第六节 视图
- 第七节 小结



## 第二节 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
200215121	李勇	男	20	CS
200215122	刘晨	女	19	CS
200215123	王敏	女	18	MA
200515125	张立	男	19	IS

# Course表

课程号 Cno	课程名 Cname	先修课 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
200215121	1	92
200215121	2	85
200215121	3	88
200215122	2	90
200215122	3	80

# 本章内容

- 第一节 SQL概述
- 第二节 学生-课程数据库
- 第三节 数据定义
- 第四节 数据查询
- 第五节 数据更新
- 第六节 视图
- 第七节 小结



## 第三节 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 3.2 SQL 的数据定义语句

操 作 对 象	操 作 方 式		
	创 建	删 除	修 改
数据库	<b>CREATE DATABASE</b>	<b>DROP DATABASE</b>	
模式	<b>CREATE SCHEMA</b>	<b>DROP SCHEMA</b>	
表	<b>CREATE TABLE</b>	<b>DROP TABLE</b>	<b>ALTER TABLE</b>
视 图	<b>CREATE VIEW</b>	<b>DROP VIEW</b>	
索 引	<b>CREATE INDEX</b>	<b>DROP INDEX</b>	

# 第三节 数据定义

❖ 数据库的创建和修改

❖ 模式的定义和删除

❖ 表的定义、修改和删除

❖ 索引的建立和删除

# 创建数据库

## ❖ 语法：

**CREATE DATABASE <database\_name>**

**例：CREATE DATABASE student**

## ❖ 使用数据库


**use <database\_name>**

**例：use student**

## ❖ 删除数据库

**drop database <database\_name>**

**例：drop database student**



不能删除当前数据库

# 第三节 数据定义

## ❖ 模式的定义和删除

- 模式定义
- 模式删除

## ❖ 表的定义、修改和删除

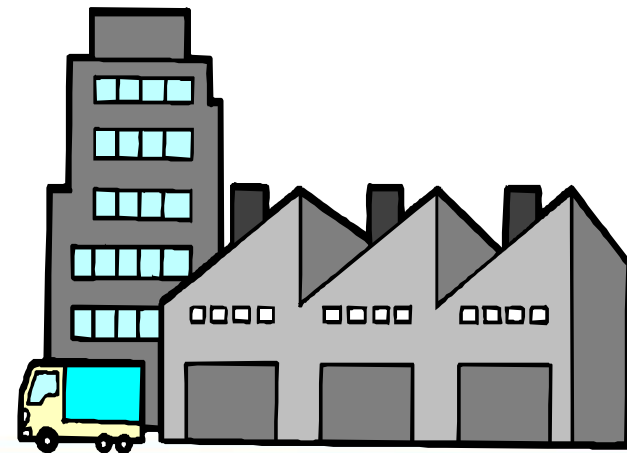
## ❖ 索引的建立和删除



# 模式的定义与删除

## ❖ 模式 (schema)

- 模式是一个独立于数据库用户的非重复命名空间，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。您可以将模式视为数据库对象的容器。
- 一个数据库可以有多个模式，模式隶属于数据库。



# 具有架构的数据库整体结构



# 一、模式定义

## ❖ 模式定义

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

- 如果没有指定模式名，则模式名隐含为用户名
- 权限：使用该命令，用户必须具有DBA权限，或获得了DBA授权CREATE SCHEMA 的权限

## ❖ 例：

```
CREATE SCHEMA Test AUTHORIZATION ZHANG
CREATE TABLE student (Sno char(9) PRIMARY KEY,
                        Sname char(20),
                        Sage int )
GRANT SELECT TO Zhao
DENY SELECT TO Li;
```

## 二、删除模式

### ❖ 定义

**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。
- 当该模式中没有任何下属的对象时 才能执行。

# 第三节 数据定义

## ❖ 模式的定义和删除

## ❖ 表的定义、修改和删除

- 数据类型
- 表的定义
- 表的修改
- 表的删除

## ❖ 索引的建立和删除

# 基本表的定义、删除与修改

## ❖ 创建表时，需要搞清楚的问题

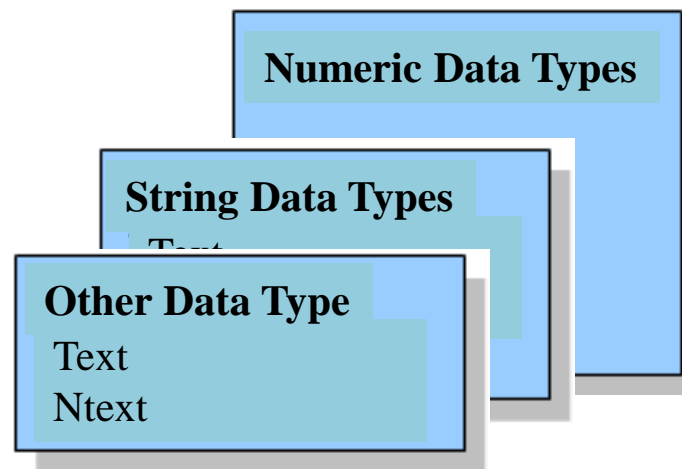
- 表名是什么？
- 此表包括那些**列**？
- 各列名是什么？
- 各列的**长度**和**数据类型**是什么？
- 列是否允许取**空值**？
- 列是否取**唯一值**？
- 哪些列组成表的**主键**？
- **外键**及**被参照的关系**是什么？



# 一、数据类型

## ❖ 数据类型

- SQL中域的概念用数据类型来实现
- 定义列时 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



# 基本数据类型

数据类型	含义
<b>CHAR(n)</b>	长度为n的定长字符串
<b>VARCHAR(n)</b>	最大长度为n的变长字符串
<b>INT</b>	长整数（也可以写作INTEGER）
<b>SMALLINT</b>	短整数
<b>NUMERIC(p, d)</b>	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
<b>REAL</b>	取决于机器精度的浮点数
<b>Double Precision</b>	取决于机器精度的双精度浮点数
<b>FLOAT(n)</b>	浮点数，精度至少为n位数字
<b>DATE</b>	日期，包含年、月、日，格式为YYYY-MM-DD
<b>TIME</b>	时间，包含一日的时、分、秒，格式为HH:MM:SS



# 二、定义基本表

## ❖ 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> ] ) ;

- <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

# 学生表Student

❖ [例5] 建立“学生”表Student, 学号是主码, 姓名取值唯一。

```
CREATE TABLE Student
```

```
(Sno CHAR(9) PRIMARY KEY, /*主键 列级完整性约束条件*/
```

```
Sname VARCHAR(20) UNIQUE, /* Sname取唯一值*/
```

```
Ssex CHAR(2),
```

```
Sage SMALLINT,
```

```
Sdept CHAR(20)
```

```
);
```

# 课程表Course

❖ [例6] 建立一个“课程”表Course

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

Cpno是外码，被参照表是Course，被参照列是Cno

# 学生选课表SC

❖ [例7] 建立一个“学生选课”表SC，它由学号Sno、课程号Cno，修课成绩Grade组成，其中(Sno, Cno)为主码。

```
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)
);
```

# 定义基本表（续）

## ❖ 常用完整性约束

- 主码约束： PRIMARY KEY
- 唯一性约束： UNIQUE
- 非空值约束： NOT NULL
- 参照完整性约束： FOREIGN KEY



**PRIMARY KEY与 UNIQUE的区别？**



# 数工原题

❖ 某工厂的仓库管理数据库的部分关系模式如下所示：

仓库（仓库号，面积，负责人，电话）

原材料（编号，名称，数量，储备量，仓库号）

要求一种原材料只能存放在同一仓库中。“仓库”和“原材料”的关系实例分别如表2-1和表2-2所示。

表 2-1 “仓库”关系

仓库号	面积	负责人	电话
01	500	李劲松	87654121
02	300	陈东明	87654122
03	300	郑爽	87654123
04	400	刘春来	87654125

表 2-2 “原材料”关系

编号	名称	数量	储备量	仓库号
1001	小麦	100	50	01
2001	玉米	50	30	01
1002	大豆	20	10	02
2002	花生	30	50	02
3001	菜油	60	20	03



# QUESTION

❖ 根据上述说明，用SQL定义“原材料”和“仓库”的关系模式如下，请在空缺处填入正确的内容。（4分）

```
CREATE TABLE 仓库  
（仓库号 CHAR（4），  
  面积 INT，  
  负责人 CHAR（8），  
  电话 CHAR（8），  
  _____(a)_____）； //主键定义
```

```
CREATE TABLE 原材料  
（编号 CHAR（4） _____(b)_____, //主键定义  
  名称 CHAR（16），  
  数量 INT _____(c)_____, //数量大于0  
  储备量 INT，  
  仓库号 _____(d)_____,  
  _____(e)_____）； //外键定义
```

# 三、修改基本表

**ALTER TABLE <表名>**

[ **ADD** <新列名> <数据类型> [ 完整性约束 ] ]

[ **DROP** <完整性约束名> ]

[ **ALTER COLUMN** <列名> <数据类型> ];

- **<表名>**: 要修改的基本表
- **ADD子句**: 增加新列和新的完整性约束条件
- **DROP子句**: 删除指定的完整性约束条件
- **ALTER COLUMN子句**: 用于修改列名和数据类型



❖ [例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;
- 不论基本表中原来是否已有数据，新增加的列一律为空值。

❖ [例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10]增加课程名称必须取唯一值的约束条件。

- ALTER TABLE Course ADD UNIQUE(Cname);

## 四、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]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
```

```
SELECT * FROM IS_Student;
```

```
--ERROR: relation “ IS_Student ” does not exist
```

# 第三节 数据定义

❖ 模式的定义和删除

❖ 表的定义、修改和删除

❖ 索引的建立和删除

- 建立索引

- 删除索引

# 索引的建立与删除

## ❖ 建立索引是加快查询速度的有效手段

### ❖ 建立索引

- DBA或表的属主（即建立表的人）根据需要建立
- 有些DBMS自动建立以下列上的索引
  - PRIMARY KEY
  - UNIQUE

### ❖ 维护索引

- DBMS自动完成

### ❖ 使用索引

- DBMS自动选择是否使用索引以及使用哪些索引

# 一、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>]]...);

- 用<表名>指定要建索引的基本表名字
- 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
- 用<次序>指定索引值的排列次序，升序：ASC，降序：DESC。缺省值：ASC
- UNIQUE表明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**

❖ [例6] 为学生-课程数据库中的Student, Course, SC三个表建立索引。其中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 SChno ON SC(Sno ASC, Cno  
DESC);
```

# 建立索引（续）

## ❖ 唯一值索引

- 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



# 建立索引（续）


## ❖ 聚簇索引

- 建立聚簇索引后，基表中数据也需要按指定的聚簇属性值的升序或降序存放。也即聚簇索引的索引项顺序与表中记录的物理顺序一致

例：

```
CREATE CLUSTER INDEX Stusname ON Student(Sname);
```

在Student表的Sname（姓名）列上建立一个聚簇索引，而且Student表中的记录将按照Sname值的升序存放

- 
- 在一个基本表上最多只能建立一个聚簇索引
  - 聚簇索引的用途：对于某些类型的查询，可以提高查询效率
  - 聚簇索引的适用范围
    - 很少对基表进行增删操作
    - 很少对其中的变长列进行修改操作

## 二、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7] 删除Student表的Stusname索引。

- DROP INDEX Stusname;

- ❖ 常见的数据库对象有哪些？SCHEMA和数据库对象之间关系是怎样的？
- ❖ 一般来说，建立索引可以提高查询效率，那么，索引建得越多越好吗？
- ❖ 那些情况不适合给表建立索引？



- 
- ❖ SQL是一个非过程化语言，使用者只需要说明“做什么”而不需要说明“怎么做”；
  - ❖ SQL是一个集定义、操作、查询和控制为一体的语言；
  - ❖ 如何使用Create Schema、Create Table语句和Create Index语句创建模式、基本表和索引。

# 作业安排

## ❖ 理论题作业

- 第三章理论练习一
- 时间:

## ❖ 实践

- 实验三

