

数据库原理与技术



An Introduction to Database System

第三章 关系数据库标准语言SQL

安徽农业大学计算机与信息学院



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 视图

3.7 小结



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

SQL概述（续）



❖ 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	1700页	1999年
■ SQL2003		2003年



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：
 - 定义关系模式，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、完整性控制等
- 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**”，必须制定存取路径
- ❖ 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 3.1 SQL 语言的动词

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



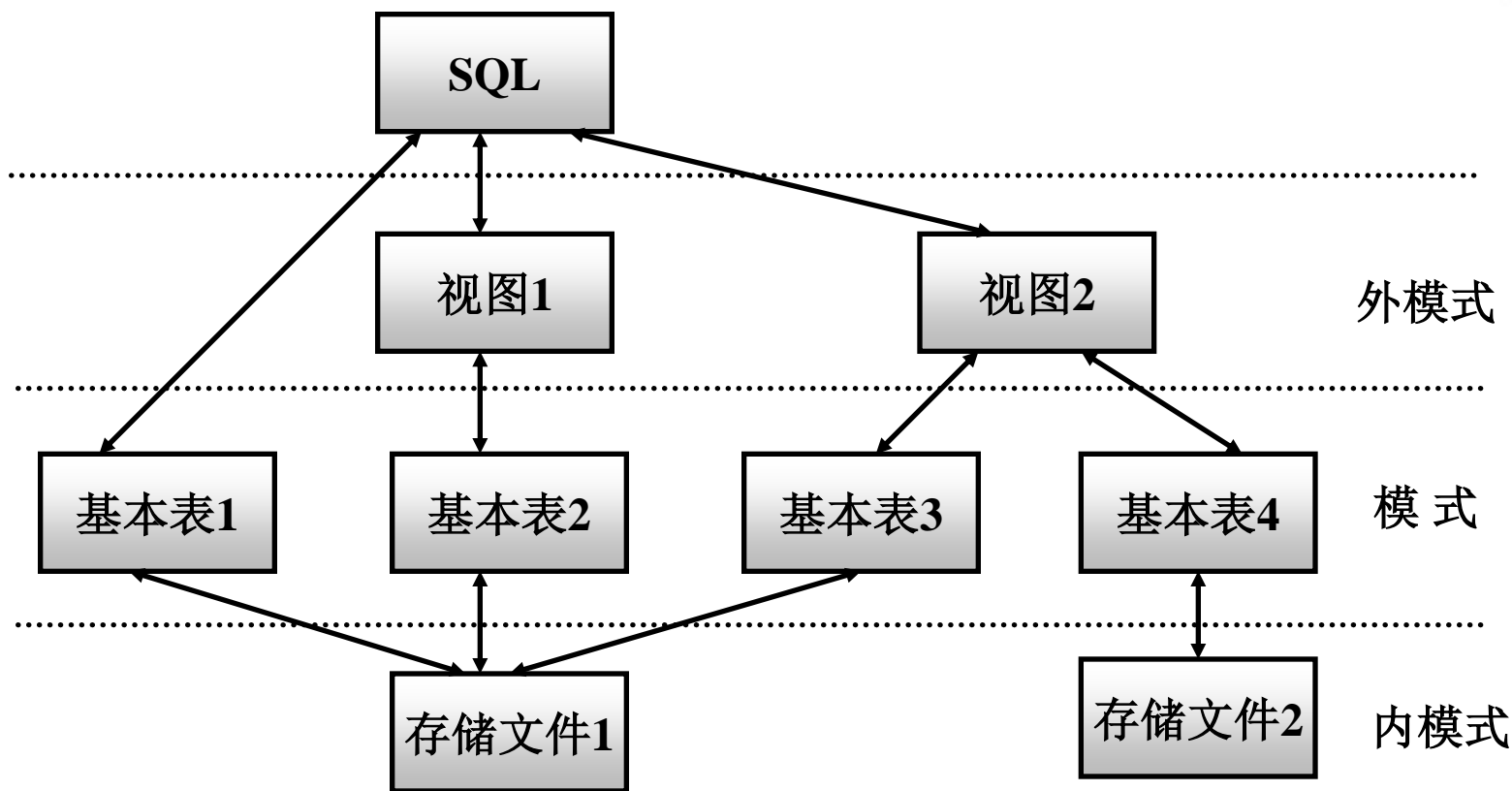
❖ 3.1.1 SQL 的产生与发展

❖ 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

- 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

- 逻辑结构组成了关系数据库的内模式
- 物理结构是任意的，对用户透明

❖ 视图

- 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



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 视图

3.7 小结



3.2 学生-课程 数据库

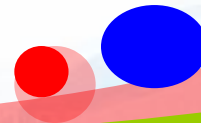


❖ 学生-课程模式 S-T :

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

课程表: Course(Cno, Cname, Cpno, Ccredit)

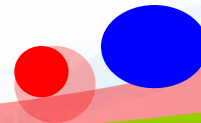
学生选课表: SC(Sno, Cno, Grade)



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 视图

3.7 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

定义模式（续）



[例1]定义一个学生-课程模式S-T

```
CREATE SCHEMA "S-T" AUTHORIZATION
```

```
WANG;
```

为用户WANG定义了一个模式S-T

[例2]CREATE SCHEMA AUTHORIZATION WANG;

<模式名>隐含为用户名WANG

- 如果没有指定<模式名>，那么<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]

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

为用户ZHANG创建了一个模式TEST，并在其中定义了一个表TAB1。



二、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。

当该模式中没有任何下属的对象时 才能执行。

删除模式（续）

[例4] DROP SCHEMA ZHANG CASCADE;

删除模式ZHANG

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

一、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>]) ;

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, /* 列级完整性约束条件*/  
  Sname CHAR(20) UNIQUE, /* Sname取唯一值*/  
  Ssex CHAR(2),  
  Sage SMALLINT,  
  Sdept CHAR(20)  
);
```

主码

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) ,
```

```
  Ccredit SMALLINT,
```

```
  FOREIGN KEY (Cpno) REFERENCES
```

```
Course(Cno)
```

```
);
```

先修课

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

```
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*/

```
);
```



二、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

二、数据类型



数据类型	含义
CHAR(n)	长度为n的定长字符串
VARCHAR(n)	最大长度为n的变长字符串
INT	长整数（也可以写作 INTEGER ）
SMALLINT	短整数
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
REAL	取决于机器精度的浮点数
Double Precision	取决于机器精度的双精度浮点数
FLOAT(n)	浮点数，精度至少为n位数字
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

三、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

- 方法一：在表名中明显地给出模式名

Create table “S-T”.Student (.....) ; /*模式名为 S-T*/

Create table “S-T”.Course (.....) ;

Create table “S-T”.SC (.....) ;

- 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

模式与表（续）

- ❖ 创建基本表（其他数据库对象也一样）时，若没有指定模式，系统根据**搜索路径**来确定该对象所属的模式
- ❖ RDBMS会使用模式列表中**第一个存在的模式**作为数据库对象的模式名
- ❖ 若搜索路径中的模式名都不存在，系统将给出错误
- ❖ 显示当前的搜索路径：`SHOW search_path;`
- ❖ 搜索路径的当前默认值是：`$user, PUBLIC`



模式与表（续）

❖ DBA用户可以设置搜索路径，然后定义基本表

```
SET search_path TO "S-T", PUBLIC;
```

```
Create table Student (.....);
```

结果建立了S-T.Student基本表。

RDBMS发现搜索路径中第一个模式名S-T存在，就把该模式作为基本表Student所属的模式。



四、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);
```


五、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12] 若表上建有视图，选择RESTRICT时表不能删除



```
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
```

删除基本表（续）



[例12]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

删除基本表（续）

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

3.3 数据定义



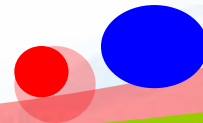
❖ 3.3.1 模式的定义与删除

❖ 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
- ❖ 谁可以建立索引
 - DBA 或 表的属主（即建立表的人）
 - DBMS一般会自动建立以下列上的索引
 - PRIMARY KEY
 - UNIQUE
- ❖ 谁 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

索引



- ❖ RDBMS中索引一般采用B+树、HASH索引来实现
 - 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
- ❖ 采用B+树，还是HASH索引 则由具体的RDBMS来决定
- ❖ 索引是关系数据库的内部实现技术，属于内模式的范畴
- ❖ CREATE INDEX语句定义索引时，可以定义索引是**唯一索引、非唯一索引或聚簇索引**

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>]]...);

- **<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13] CREATE CLUSTER INDEX Stusname
ON Student(Sname);



- 在Student表的Sname（姓名）列上建立一个聚簇索引
- ❖ 在最经常查询的列上建立聚簇索引以提高查询效率
- ❖ 一个基本表上最多只能建立一个聚簇索引
- ❖ 经常更新的列不宜建立聚簇索引

建立索引（续）

[例14]为学生-课程数据库中的Student，Course，SC三个表建立索引。

```
CREATE UNIQUE INDEX Stusno ON Student(Sno);  
CREATE UNIQUE INDEX Coucno ON Course(Cno);  
CREATE UNIQUE INDEX SCno ON SC(Sno ASC, Cno  
DESC);
```

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

二、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;



三、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15] 删除Student表的Stusname索引

```
DROP INDEX Stusname;
```



四、数据字典

- ❖ 数据字典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。



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 视图

3.7 小结



数据查询



❖ 语句格式

SELECT [ALL|DISTINCT] <目标列表达式>
[, <目标列表达式>] ...
FROM <表名或视图名>[, <表名或视图名>] ...
[**WHERE** <条件表达式>]
[**GROUP BY** <列名1> [**HAVING** <条件表达式>]]
[**ORDER BY** <列名2> [ASC|DESC]];

数据查询

- **SELECT**子句：指定要显示的属性列
- **FROM**子句：指定查询对象（基本表或视图）
- **WHERE**子句：指定查询条件
- **GROUP BY**子句：对查询结果按指定列的值分组，该属性列值相等的元组为一个组。通常会在每组中作用聚集函数。
- **HAVING**短语：只有满足指定条件的组才予以输出
- **ORDER BY**子句：对查询结果表按指定列值的升序或降序排序



3.4 数据查询

3.4.1 单表查询

3.4.2 连接查询

3.4.3 嵌套查询

3.4.4 集合查询

3.4.5 基于派生表的查询

3.4.6 Select语句的一般形式



3.4.1 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, 2014-Sage /*假定当年的年份为2014年*/  
FROM Student;
```

输出结果：

Sname	2014-Sage
-------	-----------

李勇	1994
刘晨	1995
王敏	1996
张立	1995



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: ', 2014-Sage,  
       ISLOWER(Sdept)  
FROM Student;
```

输出结果：

```
Sname 'Year of Birth:' 2014-Sage ISLOWER(Sdept)
```

李勇	Year of Birth:	1994	cs
刘晨	Year of Birth:	1995	is
王敏	Year of Birth:	1996	ma
张立	Year of Birth:	1995	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,  
       2000-Sage BIRTHDAY, LOWER(Sdept) DEPARTMENT  
FROM Student;
```

输出结果:

NAME	BIRTH	BIRTHDAY	DEPARTMENT
李勇	Year of Birth:	1994	cs
刘晨	Year of Birth:	1995	is
王敏	Year of Birth:	1996	ma
张立	Year of Birth:	1995	is

3.4.1 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

201215121
201215121
201215121
201215122
201215122



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：

<u>Sno</u>
201215121
201215122



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201215121的学生的详细情况。

```
SELECT *  
FROM Student  
WHERE Sno LIKE '201215121';
```

等价于:

```
SELECT *  
FROM Student  
WHERE Sno = '201215121';
```

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 的课程的具体情况。

```
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 ';
```



3.4.1 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

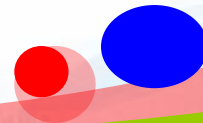
❖ 当排序列含空值时

- 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;
```

3.4.1 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
WHERE Cno= ' 1 ' ;
```

[例30] 查询学生201215012选修课程的总学分数。

```
SELECT SUM(Ccredit)
FROM SC, Course
WHERE Sno='201215012' AND SC.Cno=Course.Cno;
```

3.4.1 单表查询

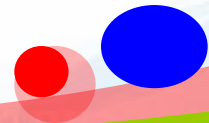
❖ 查询仅涉及一个表：

- 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
- 按指定的一系列或多列值分组，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;
```

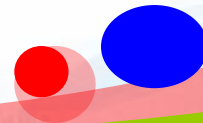
查询结果：

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。