

数据库系统概论

An Introduction to Database System

第三章 关系数据库标准语言SQL

中国人民大学信息学院

数据更新



3.5 数据更新

3.5.1 插入数据

3.5.2 修改数据

3.5.3 删除数据



3.5.1 插入数据

❖ 两种插入数据方式

- 插入元组

- 插入子查询结果

 - 可以一次插入多个元组



1. 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

- 指定要插入数据的表名及属性列
- 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：表示要插入的是一条完整的元组，且属性列属性与表定义中的顺序一致
- 指定部分属性列：插入的元组在其余属性列上取空值



插入元组（续）

❖ 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3.69]将一个新学生元组（学号：**201215128**;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 ('201215128','陈冬','男','IS',18);



插入元组（续）

[例3.71] 插入一条选课记录（ '200215128','1 '）。

```
INSERT INTO SC(Sno,Cno)
```

```
VALUES ('201215128 ','1 ');
```

关系数据库管理系统将在新插入记录的**Grade**列上自动地赋空值。

或者：

```
INSERT INTO SC
```

```
VALUES (' 201215128 ','1 ',NULL);
```



插入元组（续）

[例3.70]将学生张成民的信息插入到**Student**表中。

```
INSERT INTO Student  
VALUES ('201215126','张成民','男',18,'CS');
```



2. 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;

■ INTO子句

■ 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3.72] 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*/
```

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）

❖ 关系数据库管理系统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



3.5 数据更新

3.5.1 插入数据

3.5.2 修改数据

3.5.3 删除数据



3.5.2 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
- **SET**子句给出<表达式>的值用于取代相应的属性列
- 如果省略**WHERE**子句，表示要修改表中的所有元组

修改数据（续）

❖ 三种修改方式

- 修改某一个元组的值
- 修改多个元组的值
- 带子查询的修改语句



1. 修改某一个元组的值

[例3.73] 将学生201215121的年龄改为22岁

UPDATE Student

SET Sage=22

WHERE Sno=' 201215121 ';



2. 修改多个元组的值

[例3.74] 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3.75] 将计算机科学系全体学生的成绩置零。

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



修改数据（续）

❖ 关系数据库管理系统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



3.5 数据更新

3.5.1 插入数据

3.5.2 修改数据

3.5.3 删除数据



3.5.3 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
- 无该子句将会删除表中的全部元组



删除数据（续）

❖ 三种删除方式

- 删除某一个元组的值
- 删除多个元组的值
- 带子查询的删除语句



1. 删除某一个元组的值

[例3.76] 删除学号为201215128的学生记录。

```
DELETE FROM Student
```

```
WHERE Sno= 201215128 ';
```



2. 删除多个元组的值

[例3.77] 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3.78] 删除计算机科学系所有学生的选课记录。

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





空值处理



3.6 空值的处理

- ❖ 空值就是“不知道”或“不存在”或“无意义”的值。
- ❖ 一般有以下几种情况：
 - 该属性应该有一个值，但目前不知道它的具体值
 - 该属性不应该有值
 - 由于某种原因不便于填写



1. 空值的产生

- ❖ 空值是一个很特殊的值，含有不确定性。对关系运算带来特殊的问题，需要做特殊的处理。
- ❖ 空值的产生有其实际需求
学生在选课后，产生选课表，但是还没有成绩。这时候成绩部分就为空值，它和0不一样（不是0分）



2. 空值的判断

❖ 判断一个属性的值是否为空值，用**IS NULL**或**IS NOT NULL**来表示。

[例 3.81] 找出漏填了性别或者年龄信息的记录

```
SELECT *  
FROM Student  
WHERE Ssex IS NULL OR Sage IS NULL ;
```



3. 空值的约束条件

❖ 属性定义（或者域定义）中

- 有**NOT NULL**约束条件的不能取空值
- 加了**UNIQUE**限制的属性不能取空值
- 码属性不能取空值



4. 空值的算术运算、比较运算和逻辑运算

- 空值与另一个值（包括另一个空值）的算术运算的结果为空值
- 空值与另一个值（包括另一个空值）的比较运算的结果为 **UNKNOWN**。
- 有**UNKNOWN**后，传统二值（**TRUE**，**FALSE**）逻辑就扩展成了三值逻辑



空值的算术运算、比较运算和逻辑运算(续)

表3.8 逻辑运算符真值表

x	y	x AND y	x OR y	NOT x
T	T	T	T	F
T	U	U	T	F
T	F	F	T	F
U	T	U	T	U
U	U	U	U	U
U	F	F	U	U
F	T	F	T	T
F	U	F	U	T
F	F	F	F	T

T表示TRUE，F表示FALSE，U表示UNKNOWN

空值的算术运算、比较运算和逻辑运算（续）

[例3.82] 找出选修1号课程的不及格的学生。

```
SELECT Sno  
FROM SC  
WHERE Grade < 60 AND Cno='1';
```

查询结果不包括缺考的学生，因为他们的**Grade**值为
null。



空值的算术运算、比较运算和逻辑运算（续）

[例 3.83] 选出选修1号课程的不及格的学生以及缺考的学生。

```
SELECT Sno  
FROM SC  
WHERE Cno='1' AND (Grade<60 OR Grade IS NULL);
```



