

数据更新



讲授内容

- 1 插入元组
- 2 修改属性值
- 3 删除元组



实验数据库

学生选课数据库

学生 (学号, 姓名, 性别, 出生时间, 所在系)

课程 (课程编号, 课程名, 先修课程号)

选课 (学号, 课程编号, 成绩)

S (SNO, SN, SD, SB, SEX)

C (CNO, CN, PC)

SC (SNO, CNO, GRADE)



插入元组

- 插入单个元组的语句格式:

INSERT INTO <表名> [(<属性名1>[, <属性名2>, ...])]
VALUES (<常量1> [, <常量2>, ...])

- 常量值与相应的属性名值域相同、个数相同。
- 元组的某属性没在INTO后出现, 则这些属性上的值取空值NULL。
- INTO中没有指明任何属性, 则VALUES子句中新插入的元组在每个属性上必须有值, 且常量值的顺序要与表定义中属性的顺序一致。



插入元组

► 将一个新学生记录，插入学生关系表S中。

('S31','王浩','计算机','1999-10-15','男')

INSERT INTO S (SNO, SN, SD, SB, SEX)

VALUES ('S31' , '王浩' , '计算机' , '1999-10-15' , '男');



插入元组

► 将一个新学生记录，插入学生关系表S中。

('S31','王浩','计算机','1999-10-15','男')

INSERT INTO S

VALUES ('S31' , '王浩' , '计算机' , '1999-10-15' , '男');

A\SQLEXPRESS.学生选课 - dbo.S ×					
	SNO	SN	SD	SB	SEX
►	s01	王玲	计算机	2000-06-30	女
	s02	李渊	计算机	1995-03-23	男
	s03	罗军	计算机	1995-08-12	男
	s04	赵泽	计算机	1997-09-12	女
	s05	许若	自动化	1999-06-27	男
	s06	王仙华	自动化	1996-05-20	男
	s07	朱祝	自动化	1998-07-10	女
	s20	李国民	数学	1999-12-31	男
	s21	陈浩然	计算机	2000-10-15	男
	S31	王浩	计算机	1999-10-15	男



插入元组

► 向表SC中插入一条选课元组('S31' , 'C01')

```
INSERT INTO SC(SNO,CNO)  
VALUES ('S31','C01');
```

A\SQLEXPRESS.学生选课 - dbo.SC ×			
	SNO	CNO	GRADE
	s04	C06	90.0
	s04	C11	87.0
	s05	C03	79.5
	s05	C05	88.0
	s05	C07	90.0
	s06	C03	88.0
	s07	C11	88.0
	s21	C05	NULL
	S31	C01	NULL



插入元组

- 插入子查询结果的语句格式:

INSERT

INTO <表名> [(<属性名1> [, <属性名2>...])]

子查询



插入元组

► 插入“计算机”系学生选修“数据库”课程的选课记录。

```
INSERT INTO SC(SNO,CNO)
```

```
SELECT SNO,CNO
```

```
FROM S,C
```

```
WHERE SD='计算机' AND CN='数据库';
```

结果		消息	
	SNO	CNO	GRADE
1	s01	C01	93.0
2	s01	C02	98.0
3	s01	C03	85.0
4	s01	C04	NULL
5	s01	C05	80.0
6	s01	C07	89.0
7	s02	C04	NULL
8	s02	C05	90.0



修改属性值

- 修改属性值的语句格式为：

UPDATE <表名>

SET <属性名1> = <表达式1>[, <属性名2> = <表达式2>, ...]

[WHERE <元组选择条件>]



修改属性值

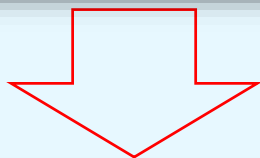
► 修改学生表S中“许若”的所在系为“计算机”系。

UPDATE S

SET SD= '计算机'

WHERE SN= '许若' ;

	s05	许若	自动化	1999-06-27	男
--	-----	----	-----	------------	---



	s05	许若	计算机	1999-06-27	男
--	-----	----	-----	------------	---



修改属性值

► 将学号为“S02”的学生所学“高等数学”的成绩改为93.0。

UPDATE SC

SET GRADE=93.0

WHERE SNO= 'S02' AND CNO IN

(SELECT CNO

FROM C

WHERE CN= '高等数学');



修改属性值

- ▶ 若课程成绩低于该课平均成绩，则将成绩提高5%。

```
UPDATE SC
SET GRADE=GRADE*1.05
WHERE GRADE<
  (SELECT AVG(GRADE)
   FROM SC SC1
   WHERE SC.CNO=SC1.CNO);
```



删除元组

- 删除元组的语句格式为：

DELETE

FROM <表名>

[WHERE <元组选择条件>]



删除元组

▶ 删除所有成绩为空值的选课记录。

DELETE

FROM SC

WHERE GRADE IS NULL;



删除元组

▶ 删除成绩低于所有课程平均成绩的选课元组。

DELETE

FROM SC

WHERE GRADE < (SELECT AVG(GRADE) FROM SC);



更新操作的完整性检查

```
SQLQuery1.sql - A...ministrator (52))* x
INSERT INTO SC
VALUES ('S35', 'C20', 78.5);
```

100 %

消息
消息 547, 级别 16, 状态 0, 第 142 行
INSERT 语句与 FOREIGN KEY 约束“FK__SC__SNO__173876EA”冲突。
语句已终止。

```
SQLQuery1.sql - A...ministrator (52))* x
UPDATE S
SET SNO='S29'
WHERE SNO='S02';
```

100 %

消息
消息 547, 级别 16, 状态 0, 第 145 行
UPDATE 语句与 REFERENCE 约束“FK__SC__SNO__173876EA”冲突。
语句已终止。

S (SNO, SN, SD, SB, SEX)

C (CNO, CN, PC)

SC (SNO, CNO, GRADE)

```
SQLQuery1.sql - A...ministrator (52))* x
DELETE
FROM S
WHERE SNO='S01';
```

100 %

消息
消息 547, 级别 16, 状态 0, 第 149 行
DELETE 语句与 REFERENCE 约束“FK__SC__SNO__173876EA”冲突。
语句已终止。



更新操作的完整性检查

- 更新操作不能满足参照完整性时，DBMS一般采取的处理策略：
 - 拒绝执行 (NO ACTION)
 - 产生级联操作 (CASCADE)
 - 设置为空值 (SET NULL)

A:\SQLEXPRESS. 学生选课 - dbo. SC

```
SELECT *  
FROM S  
WHERE SNO='S02' OR  
SNO='S29';
```

100 %

结果 消息

	SNO	SN	SEX	SB	SD
1	S29	李渊	男	1995-03-23	计算机

A:\SQLEXPRESS. 学生选课 - dbo. SC

```
SELECT *  
FROM SC  
WHERE SNO='S02' OR  
SNO='S29';
```

100 %

结果 消息

	SNO	CNO	GRADE
1	S29	C05	90.0
2	S29	C11	80.0






更新操作的完整性检查

- 在定义表SC时说明参照完整性的违约处理策略

```
CREATE TABLE SC
( SNO CHAR (6) ,
  CNO CHAR (6) ,
  GRADE DEC (4, 1) ,
  PRIMARY KEY (SNO, CNO) ,
  FOREIGN KEY (SNO) REFERENCES S (SNO)
    ON UPDATE CASCADE
    /当修改表S的SNO时, 级联修改SC中相应元组的SNO
  ON DELETE NO ACTION,
    /当删除表S的元组时, 拒绝执行
  FOREIGN KEY (CNO) REFERENCES C (CNO)
    ON UPDATE CASCADE
    ON DELETE NO ACTION,
  CHECK (GRADE BETWEEN 0.0 AND 100.0)
```



小结

-  数据更新不对关系表的模式结构进行改变。
-  更新操作往往是在查询的基础上进行的，在更新操作语句中可能会嵌套子查询。
-  更新操作时要考虑数据库上所定义的各类完整性约束，以及DBMS所采取的执行策略。