数据库系统概论

第3章 关系数据库标准查询语言

第3章 关系数据库标准语言SQL

- 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 空值的处理
- 第七节 视图

教学目标

- 掌握
  - 数据更新：增、删、改
  - 视图的定义、删除、查询、更新
- 了解
  - 视图的作用
- 重点
  - 数据更新、视图
- 难点
  - 视图更新

本讲内容

- 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 空值的处理
- 第七节 视图

数据更新

- 数据的插入
  - 插入元组
  - 插入子查询结果
- 数据的修改
- 数据的删除

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

- 将新的元组插入到指定表

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

VALUES子句

- 提供的值必须与INTO子句匹配
  - 值的个数
  - 值的类型

示例

[例1] 将一个新学生记录（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);
```

[例2] 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');
```

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

❖功能

- 将子查询结果插入指定表中

❖注意

- 子查询的结果必须包含和insert的字段列表一样多的字段，并且数据类型兼容

[例3] 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 Deptage
    (Sdept CHAR(15) ,        /* 系名*/
     Avgage SMALLINT);    /*学生平均年龄*/
```

第二步：插入数据

```
INSERT INTO Deptage(Sdept，Avgage)
   SELECT Sdept，AVG(Sage)  /**子查询*/
   FROM Student
   GROUP BY Sdept;
```

第五节 数据更新

- 数据的插入
- 数据的修改
  - 修改某元组的值
  - 修改多个元组的值
  - 带子查询的修改语句
- 数据的删除

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>];

- SET子句
  - 指定修改方式、要修改的列、修改后取值
- WHERE子句
  - 指定要修改的元组，缺省表示要修改表中的所有元组

功能

- 修改指定表中满足WHERE子句条件的元组

修改某一个元组的值

[例4] 将学生201215121的年龄改为22岁。

```
UPDATE  Student
SET     Sage = 22
WHERE   Sno=' 201215121 ';
```

修改多个元组的值

[例5] 将所有学生的年龄增加1岁。

```
UPDATE  Student
SET  Sage= Sage+1;
```

带子查询的修改语句

[例6] 将计算机科学系全体学生的成绩置零。

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

你知道吗?

- DBMS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

数据更新

- 数据的插入
- 数据的修改
- 数据的删除
  - 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

删除语句

定义

DELETE

FROM <表名>

[WHERE <条件>];

- WHERE子句
  - 指定要删除的元组，缺省表示要删除表中的所有元组

功能

- 删除指定表中满足WHERE子句条件的元组

删除某个元组的值

[例7] 删除学号为201215128的学生记录。

```
DELETE
FROM    Student
WHERE   Sno=' 201215128 ';
```

删除多个元组的值

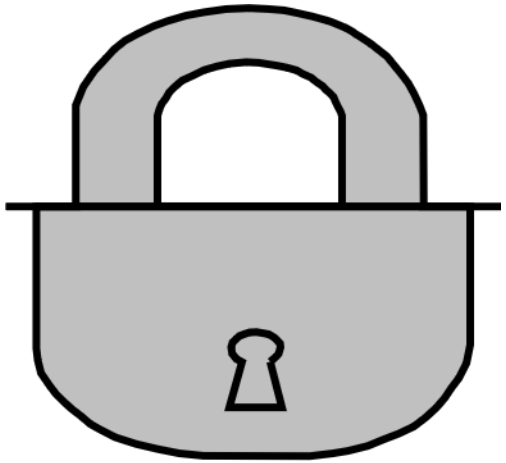
[例8]　删除所有学生的选课记录。

```
DELETE
FROM    SC
```

带子查询的删除语句

[例9] 删除计算机科学系所有学生的选课记录。

```
DELETE
FROM    SC
WHERE  'CS' =
            (SELECT Sdept
             FROM student
             WHERE student.Sno = SC.Sno)
```



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 空值的处理**
- 第七节 视图

空值的处理

所谓空值就是“不知道”或“不存在”或无意义的值

- 空值的产生
- 空值的判断
- 空值的约束条件
- 空值的算术运算、比较运算和逻辑运算

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 空值的处理
- **第七节 视图**

视　图

- 定义视图
  - 定义视图
  - 删除视图
- 查询视图
- 更新视图
- 视图的作用

视　图

- 视图的特点
  - **虚表**，是从一个或几个基本表（或视图）导出的表
  - 只存放视图的定义，不会出现数据冗余
  - 基表中的数据发生变化，从视图中查询出的数据也随之改变
- 基于视图的操作
  - 查询
  - 删除
  - 受限更新
  - 定义基于该视图的新视图

定义视图

❖语句格式

CREATE VIEW <视图名> [(<列名> [，<列名>]…)]

AS <子查询>

[WITH CHECK OPTION];

- 子查询不允许含有ORDER BY子句和DISTINCT短语
- WITH CHECK OPTION
  - 透过视图进行增删改操作时，不得破坏视图定义中的谓词条件（即子查询中的条件表达式）

示例

[例10] 建立信息系学生的视图。

```
CREATE VIEW IS_Student
AS
SELECT  Sno, Sname, Sage
FROM    student
WHERE  Sdept= 'IS'
WITH CHECK OPTION
```

从单个基本表导出,只是去掉了基本表的某些行和某些列,保留了码————**行列子集视图**

[例11] 建立信息系选修了1号课程的学生视图。

```
CREATE VIEW IS_S1(Sno，Sname，Grade)
AS
SELECT Student.Sno，Sname，Grade
FROM Student，SC
WHERE Student.Sno=SC.Sno AND
      Sdept= 'IS' AND
      SC.Cno= '1';
```

[例12] 建立信息系选修了1号课程且成绩在90分以上的学生视图。

```
CREATE VIEW IS_S2
AS
SELECT Sno，Sname，Grade
FROM IS_S1
WHERE Grade>=90;
```

[例13]　将学生的学号及他的平均成绩定义为一个视图。

```
CREATE  VIEW S_G(Sno，Gavg)
AS
SELECT  Sno，AVG(Grade)
FROM  SC
GROUP BY Sno;
```

删除视图

❖语句格式

DROP VIEW <视图名> [CASCADE];

- 该语句从数据字典中删除指定的视图定义
- 由该视图导出的其他视图定义仍在数据字典中，但已不能使用，必须显式删除
- 删除基表时，由该基表导出的所有视图定义都必须显式删除
- 如果CASCADE选项，则删除该视图时会把由它导出的视图一块删除

视　图

- 定义视图
- 查询视图
- 更新视图
- 视图的作用

❖从用户角度：查询视图与查询基本表相同

❖DBMS实现视图查询的方法

- **实体化视图**（View Materialization）
  - 有效性检查：检查所查询的视图是否存在
  - 执行视图定义，将视图临时实体化，生成临时表
  - 查询视图转换为查询临时表
  - 查询完毕删除被实体化的视图(临时表)

视图消解法（View Resolution）

- 进行有效性检查，检查查询的表、视图等是否存在。如果存在，则从数据字典中取出视图的定义
- 把视图定义中的子查询与用户的查询结合起来，转换成等价的对基本表的查询
- 执行**修正**后的查询

[例14] 在信息系学生的视图中找出年龄小于20岁的学生。

```
SELECT  Sno，Sage
FROM    IS_Student
WHERE   Sage<20；
```

- **视图消解法**

转换后的查询语句为：

```
SELECT  Sno，Sage
FROM  Student
WHERE  Sdept= 'IS'  AND  Sage<20；
```

[例15] 在S_G视图中查询平均成绩在90分以上的学生学号和平均成绩。

```
SELECT *
FROM   S_G
WHERE  Gavg>=90;
```

查询转换：

```
SELECT Sno，AVG(Grade)
FROM   SC
WHERE  AVG(Grade)>=90
GROUP BY Sno;
```

```
SELECT  Sno，AVG(Grade)
FROM    SC
GROUP BY  Sno;
HAVING  AVG(Grade)>=90;
```

- 定义视图
- 查询视图
- 更新视图
- 视图的作用

更新视图

- **用户角度：更新视图与更新基本表相同**
- **DBMS实现视图更新的方法**
  - 视图实体化法（View Materialization）
  - 视图消解法（View Resolution）
- **指定WITH CHECK OPTION子句后**

DBMS在更新视图时会进行检查，防止用户通过视图对**不属于视图范围内**的基本表数据进行更新

示例

[例16] 将信息系学生视图IS_Student中学号201215122的学生姓名改为“刘辰” 。

```
UPDATE  IS_Student
SET  Sname= '刘辰'
WHERE  Sno= ' 201215122 ';
```

转换后的语句:

```
UPDATE  student
SET  Sname= '刘辰'
WHERE  Sno= ' 201215122 '  AND  Sdept =  ‘IS’ ;
```

- 一些视图是不可更新的，因为对这些视图的更新不能唯一地有意义地转换成对相应基本表的更新（对两类方法均如此）

例：视图S_G为不可更新视图。

```
CREATE VIEW  S_G (Sno，Gavg)
AS
SELECT  Sno，AVG(Grade)
FROM    SC
GROUP  BY  Sno；
```

对于如下更新语句：

```
UPDATE  S_G
SET     Gavg=90
WHERE   Sno= '201215121';
```

无论实体化法还是消解法都无法将其转换成对基本表SC的更新

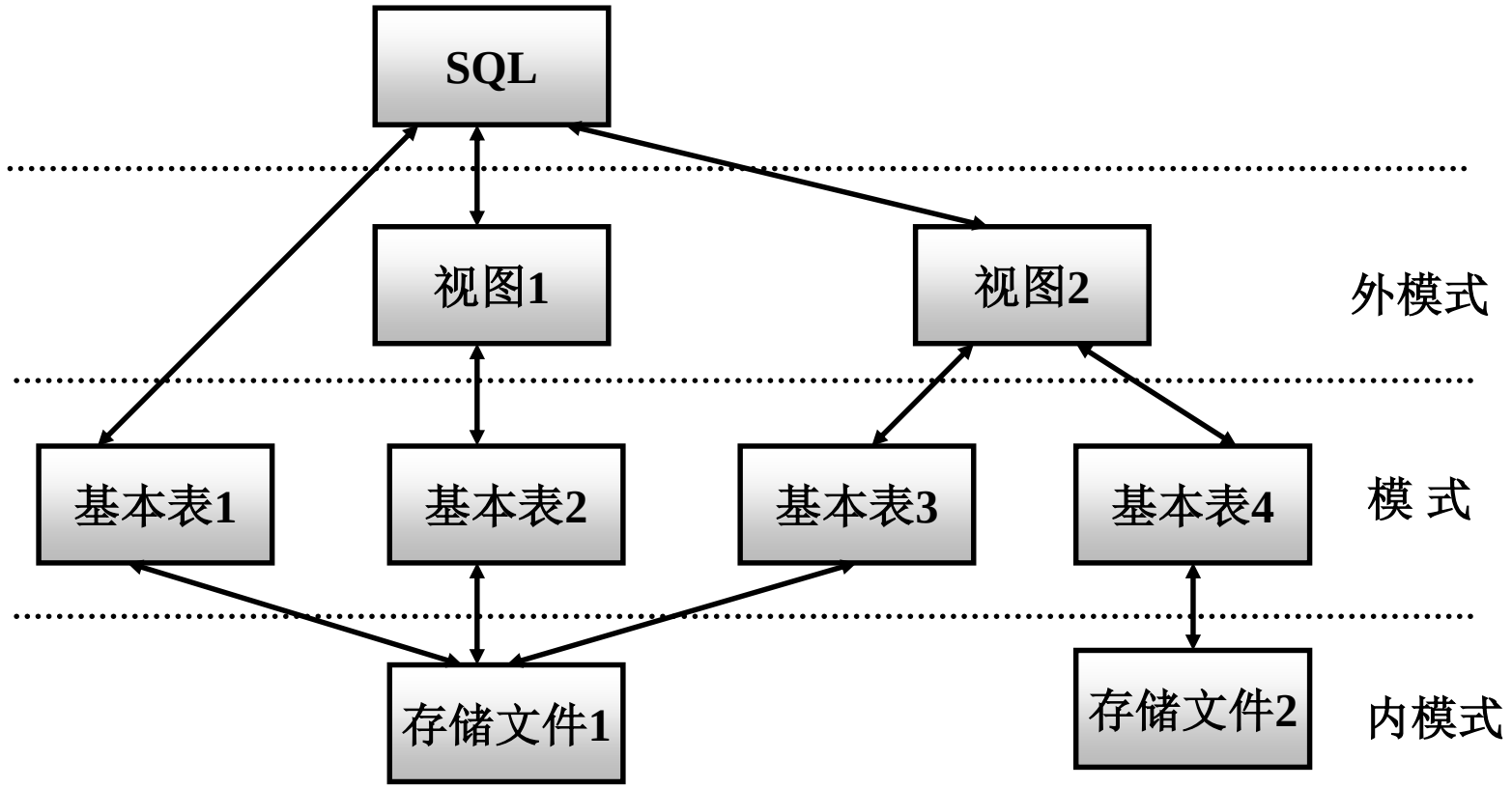
第七节 视图

- 定义视图
- 查询视图
- 更新视图
- 视图的作用

视图的作用

- 视图能够简化用户的操作
- 视图使用户能以多种角度看待同一数据
- 视图对重构数据库提供了一定程度的逻辑独立性
- 视图能够对机密数据提供安全保护
- 适当的利用视图可以更清晰的表达查询

SQL支持关系数据库三级模式结构

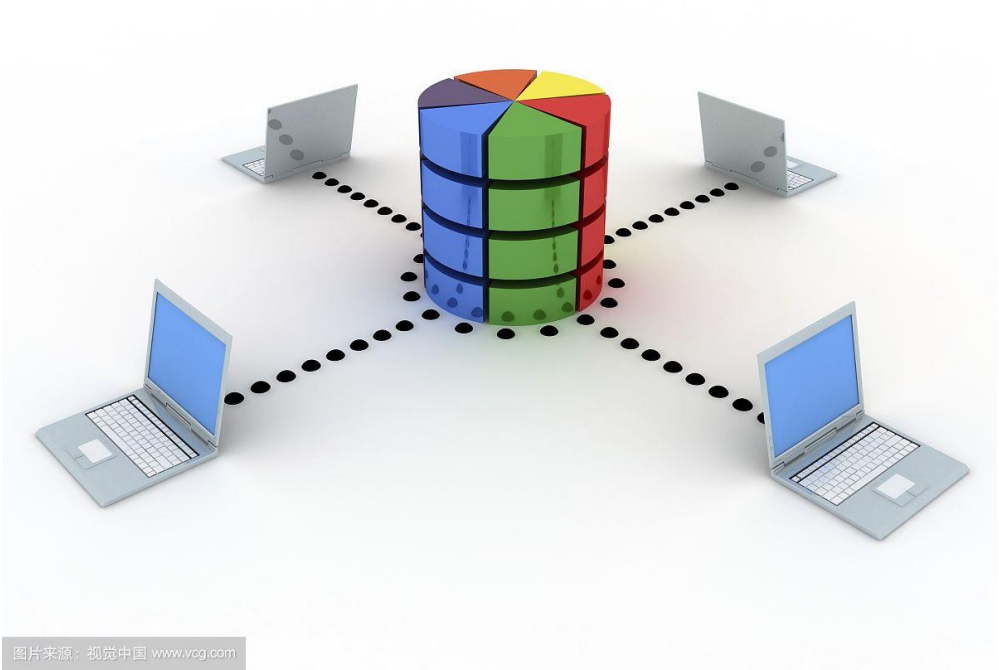


Q & A

- 视图的作用是什么?

这次课我们学到了...

- INSERT、UPDATE、DELETE
- 视图的定义、更新、删除



子曰：“见贤思齐焉，见不贤而内自省也。”