数据库原理

辽东学院 鲁 琴

关系 (Relations)

三种类型关系

- ✓ 表(Table) —基表，存储关系(Base tables, Stored relations)
  - 用CREATE TABLE语句创建
  - 真实地存在于数据库中
  - 数据是持久的
- ✓ 视图(Views)— 虚拟关系(Virtual relations)
  - 不是物理存在的，是虚拟的
- ✓ 临时结果(Temporary results)—用于构建子查询的结果

视 图

示例数据库

teach数据库

- **学生表：**

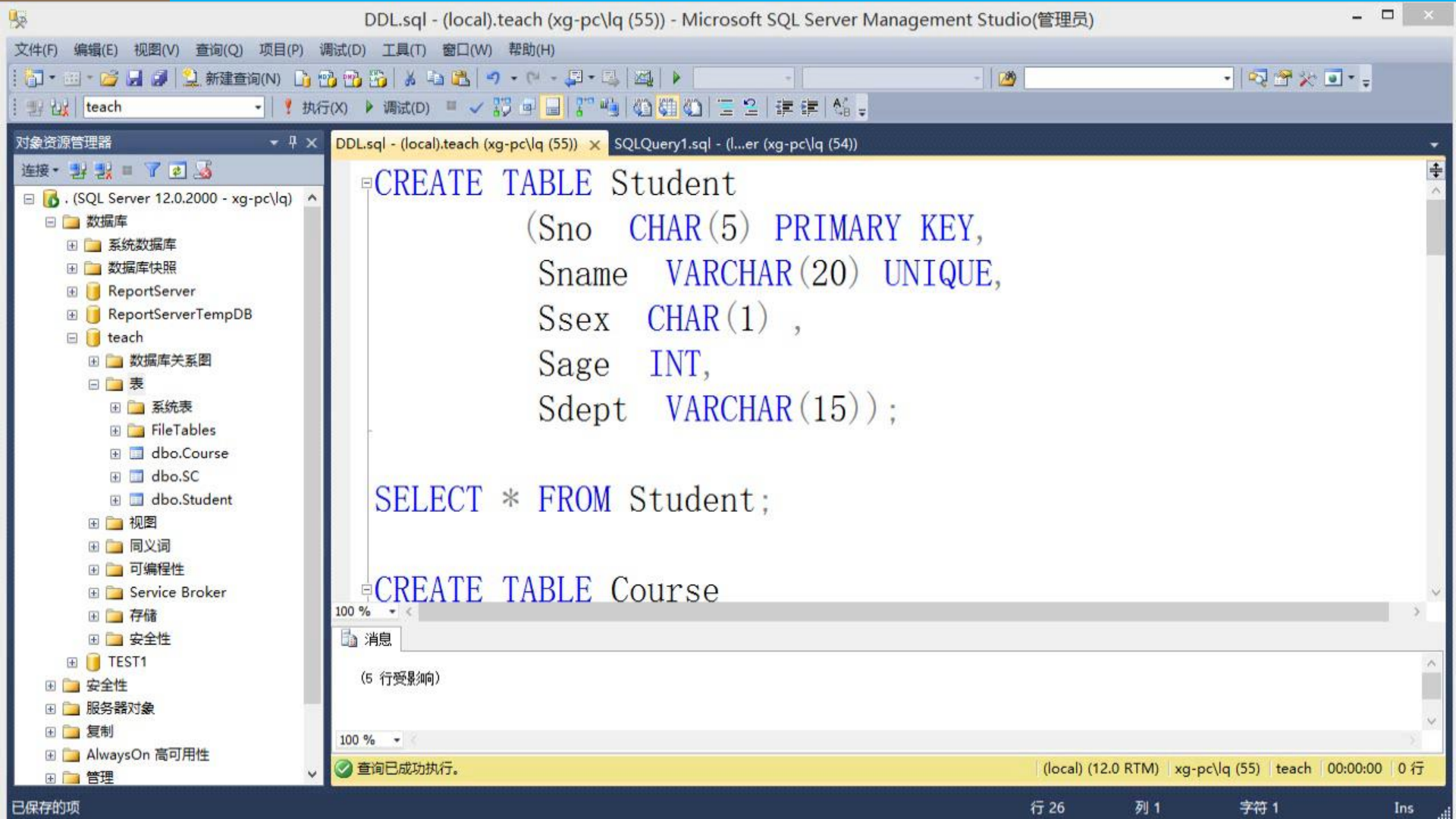
  Student(Sno，Sname，Ssex，Sage，Sdept)

- **课程表：**

  Course(Cno，Cname，Cpno，Ccredit)

- **学生选课表：**

  SC(Sno，Cno，Grade)


DDL.sql - (local).teach (xg-pc\lq (55)) - Microsoft SQL Server Management Studio(管理员)
文件(F) 编辑(E) 视图(V) 查询(Q) 项目(P) 调试(D) 工具(T) 窗口(W) 帮助(H)
新建查询(N)
teach
执行(X)
调试(D)
对象资源管理器
连接
. (SQL Server 12.0.2000 - xg-pc\lq)
数据库
系统数据库
数据库快照
ReportServer
ReportServerTempDB
teach
数据库关系图
表
系统表
FileTables
dbo.Course
dbo.SC
dbo.Student
视图
同义词
可编程性
Service Broker
存储
安全性
TEST1
安全性
服务器对象
复制
AlwaysOn 高可用性
管理
DDL.sql - (local).teach (xg-pc\lq (55))
SQLQuery1.sql - (l...er (xg-pc\lq (54))
CREATE TABLE Student
(Sno CHAR(5) PRIMARY KEY,
Sname VARCHAR(20) UNIQUE,
Ssex CHAR(1) ,
Sage INT,
Sdept VARCHAR(15));
SELECT * FROM Student;
CREATE TABLE Course
100 %
消息
(5 行受影响)
100 %
查询已成功执行。
(local) (12.0 RTM) xg-pc\lq (55) teach 00:00:00 0 行
已保存的项
行 26
列 1
字符 1
Ins

SQL支持数据库的三级模式结构

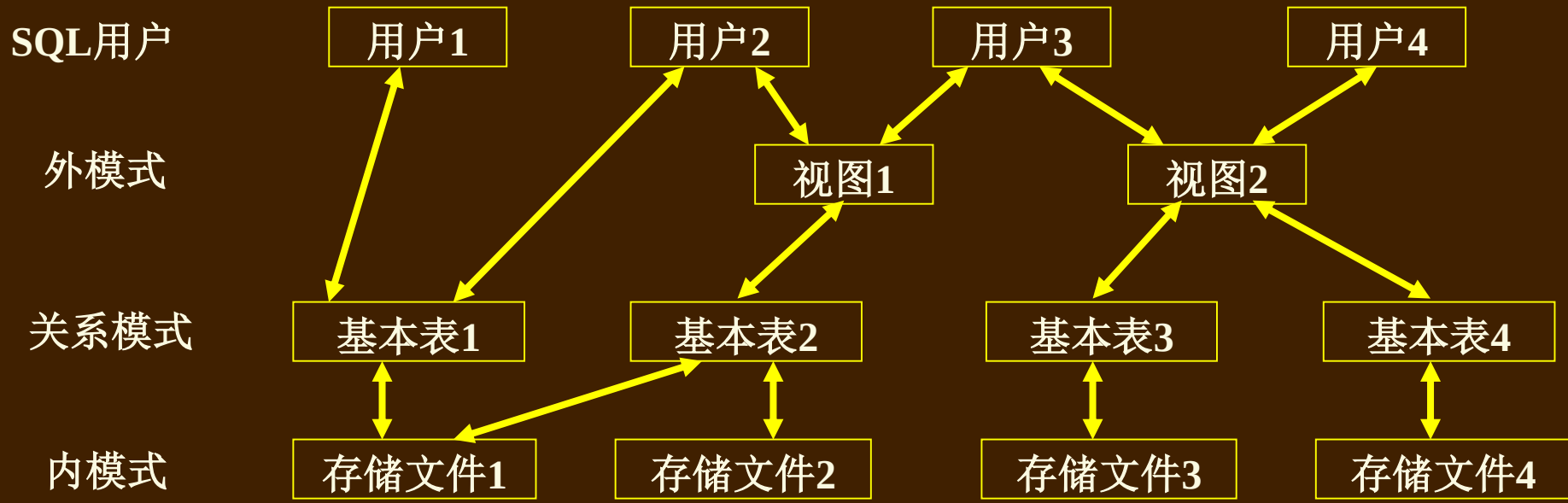


图　SQL数据库的体系结构

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

- 定义视图(**DDL**)
  - 建立
  - 定义基于该视图的新视图
  - 删除
- 查询视图(**DML**)
- 更新视图(**DML**)

1　定义视图

（1）建立视图

CREATE VIEW <视图名> [(<列名> [，<列名>]…)]

AS <子查询>

[WITH CHECK OPTION];

组成视图的属性列名或全部省略或全部指定

- 省略视图的各个属性列名，则隐含该视图由子查询中SELECT子句目标列中的各字段组成。
- 必须明确指定组成视图的所有列名的情形
  - 某个目标列不是单纯的属性名，而是集函数或列表达式
  - 目标列为 *
  - 多表连接时选出了几个同名列作为视图的字段
  - 需要在视图中为某个列启用新的更合适的名字

建立视图（续）

- 视图定义的**SELECT**语句
  - 不能包含**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）

建立视图（续）

DBMS执行CREATE VIEW语句时只是把视图的定义存入数据字典，并不执行其中的SELECT语句。

只是在对视图查询时，才按视图的定义从基本表中将数据查出。

常见的视图形式

①行列子集视图

②**WITH CHECK OPTION**的视图

③基于多个基表的视图

④基于视图的视图

⑤带表达式的视图

⑥分组视图

①行列子集视图

- 从单个基本表导出
- 只是去掉了基本表的某些行和某些列，但保留了码

[例1] 建立信息系学生的视图。

```
CREATE VIEW V_StuIS   AS
    SELECT Sno,Sname,Sage
    FROM  Student
    WHERE  Sdept= 'IS';
```

行列子集视图V_StuIS由Sno,Sname,Sage三列组成

②WITH CHECK OPTION的视图

[例2] 建立信息系学生的视图，并要求透过该视图进行的更新操作只涉及信息系学生。

```
CREATE VIEW V_StuIS
    AS SELECT Sno,Sname,Sage FROM  Student
    WHERE  Sdept= 'IS'  WITH CHECK OPTION;
```

- 对V_StuIS视图的更新操作
  - 修改操作：DBMS自动加上Sdept= 'IS'的条件
  - 删除操作：DBMS自动加上Sdept= 'IS'的条件
  - 插入操作：DBMS自动检查Sdept属性值是否为'IS'
    - 如果不是，则拒绝该插入操作
    - 如果没有提供Sdept属性值，则自动定义Sdept为'IS'

［例3］建立1号课程的选课视图，并要求透过该视图进行的更新操作只涉及1号课程。

```
CREATE VIEW V_SCCno
        AS
        SELECT Sno,Cno,Grade
        FROM  SC
        WHERE  Cno= 1
       WITH CHECK OPTION;
```

③基于多个基表的视图

[例4] 建立信息系统选修了1号课程的学生视图。

```
CREATE VIEW V_S1(Sno,Sname,Grade)
    AS  SELECT Student.Sno,Sname,Grade
    FROM  Student,SC
    WHERE  Sdept= 'IS' AND
        Student.Sno=SC.Sno AND
        SC.Cno= 1;
```

- 由于视图V_S1的属性列中包含了Student表与SC表的同名列Sno，所以必须在视图名后面明确说明视图的各个属性列名。

④基于视图的视图

[例5] 建立信息系统选修了1号课程且成绩在90分以上的学生的视图

```
CREATE VIEW V_S2
    AS
    SELECT Sno,Sname,Grade
    FROM  V_S1
    WHERE  Grade>=90;
```

- 视图V_S2建立在视图V_S1之上

④带表达式的视图

- 在设计数据库时，为了减少数据冗余，基本表中只存放基本数据，由基本数据经过各种计算派生出的数据一般是不存储的。
- 视图中的数据并不实际存储，所以定义视图时可以根据应用的需要，设置一些派生属性列，以方便应用程序的编制。
- 派生属性称为虚拟列。带虚拟列的视图称为带表达式的视图。
- 带表达式的视图必须明确定义组成视图的各个属性列名

建立视图（续）

[例6] 定义一个反映学生出生年份的视图。

```
CREATE  VIEW V_BTS(Sno,Sname,Sbirth)
    AS
        SELECT Sno,Sname,2018-Sage
        FROM  Student;
```

⑥分组视图

- 用带集函数和GROUP BY子句的查询来定义的视图称为分组视图
- 分组视图必须明确定义组成视图的各个属性列名

建立视图（续）

[例7] 将学生的学号及他的平均成绩定义为一个视图

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

以 SELECT * 方式创建的视图

– 可扩充性差，应尽可能避免

[例8] 将Student表中所有女生记录定义为一个视图

```
CREATE VIEW V_Student1(stdnum,name,sex,age,dept)
    AS  SELECT *
        FROM  Student
        WHERE Ssex='女';
```

- 修改基表Student的结构后，Student表与V_Student1视图的映象关系被破坏，导致该视图不能正确工作。

上例可以这样写

```
CREATE VIEW V_Student2(stdnum,name,sex,age,dept)
  AS  SELECT Sno,Sname,Ssex,Sage,Sdept
      FROM  Student
      WHERE Ssex='女';
```

为基表Student增加属性列不会破坏Student表与V_Student2视图的映象关系。

(2) 删除视图

语句格式

DROP VIEW <视图名>;

- 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

删除视图(续)

[例9] 删除视图V_S1

```
DROP VIEW V_S1;
```

执行此语句后，V_S1视图的定义将从数据字典中删除。由V_S1视图导出V_S2视图已无法使用，但其定义虽然仍在数据字典中。

2　查询视图

从用户角度而言，查询视图与查询基本表的方法相同

DBMS实现视图查询的方法(1)

– 视图实体化法（View Materialization）

- 进行有效性检查，检查所查询的视图是否存在。如果存在，则从数据字典中取出视图的定义
- 执行视图定义，将视图临时实体化，生成临时表
- 将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)

DBMS实现视图查询的方法(2)

– 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

查询视图（续）

[例1] 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  V_StuIS
WHERE  Sage<20;
```

V_StuIS视图的定义(视图定义例1)：

```
CREATE VIEW V_StuIS
    AS
    SELECT Sno,Sname,Sage
    FROM  Student
    WHERE  Sdept= 'IS';
```

- 视图实体化法
- 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;
```

查询视图（续）

[例2] 查询信息系统选修了1号课程的学生学号和姓名

```
SELECT  Sno,Sname
FROM  V_StuIS,SC
WHERE V_StuIS.Sno=SC.Sno AND  SC.Cno= 1;
```

V_StuIS视图的定义(视图定义例1)：

```
CREATE VIEW V_StuIS
    AS
    SELECT Sno,Sname,Sage
    FROM  Student
    WHERE  Sdept= 'IS';
```

- 视图实体化法
- 视图消解法

转换后的查询语句为：

```
SELECT  SC.Sno,Sname
FROM  Student,SC
WHERE  Student.Sno=SC.Sno  AND
                Sdept= 'IS'  AND  SC.Cno= 1;
```

视图消解法的局限

- 有些情况下，视图消解法不能生成正确查询。
- 采用视图消解法的DBMS会限制这类查询。

查询视图（续）

[例3] 在V_SG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  V_SG
WHERE  Gavg>=90;
```

转换后的查询：

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

V_SG视图定义：

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

查询视图（续）

正确转换：

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

3 更新视图

从用户角度而言，更新视图与更新基本表的方法相同

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**）

更新视图（续）

定义视图时指定**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数据进行增加、删除、修改时，操作不属于视图范围内的基本表数据

更新视图（续）

[例1] 将信息系学生视图V_StuIS中学号为95002的学生姓名改为刘辰

```
UPDATE  V_StuIS
SET  Sname= '刘辰'
WHERE  Sno= '95002';
```

更新视图（续）

- 视图实体化法
- 视图消解法

转换后的语句为：

```
UPDATE Student
SET Sname= '刘辰'
WHERE Sno= '95002' AND Sdept= 'IS';
```

更新视图（续）

[例2] 向信息系学生视图V_StuIS中插入一个新的学生记录，其中学号为95029，姓名为赵新，年龄为20岁

```
INSERT
INTO V_StuIS
 VALUES('95029','赵新',20);
```

转换为对基本表的更新：

```
INSERT
INTO  Student(Sno,Sname,Sage,Sdept)
VALUES('95029','赵新',20,'IS' );
```

更新视图（续）

[例3] 删除计算机系学生视图

V_StuIS中学号为95029的记录

```
DELETE
FROM V_StuIS
WHERE Sno= '95029';
```

转换为对基本表的更新：

```
DELETE
FROM Student
WHERE Sno= '95029' AND Sdept= 'IS';
```

DBMS对视图更新的限制

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V_SG为不可更新视图。

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

- ✓ V_SG “平均成绩” Gavg属性列为导出列
- ✓ 对于如下更新语句：

```
UPDATE V_SG
SET Gavg=90
WHERE Sno= '95001';
```

- ✓ 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

视图的可更新性

- 行列子集视图是可更新的。
- 除行列子集视图外，还有些视图理论上是可更新的，但它们的确切特征还是尚待研究的课题。
- 还有些视图从理论上是不可更新的。

不可更新的视图与不允许更新

- 不可更新的视图与不允许更新的视图是两个不同的概念
- 实际系统对视图更新的限制
  - 允许对行列子集视图进行更新
  - 对其他类型视图的更新不同系统有不同限制

SQL Server 对视图更新的限制

- 若视图是由两个以上基本表导出的，则此视图不允许更新。
- 若视图的字段来自字段表达式或常数，则不允许对此视图执行INSERT和UPDATE操作，但允许执行DELETE操作。
- 若视图的字段来自集函数，则此视图不允许更新。
- 若视图定义中含有GROUP BY子句，则此视图不允许更新。
- 若视图定义中含有DISTINCT短语，则此视图不允许更新。
- 一个不允许更新的视图上定义的视图也不允许更新。

SQL Server 对视图更新的限制

- 若视图定义中有嵌套查询，并且内层查询的FROM子句中涉及的表也是导出该视图的基本表，则此视图不允许更新。

例：视图GOOD_SC(修课成绩在平均成绩之上的元组)

```
CREATE VIEW GOOD_SC
    AS SELECT  Sno，Cno，Grade
      FROM SC
      WHERE Grade >
        (SELECT AVG(Grade)
         FROM  SC)；
```

4　视图的作用

视图最终是定义在基本表之上的，对视图的一切操作最终也要转换为对基本表的操作。

对于非行列子集视图进行查询或更新时还有可能出现问题。

视图的作用（续）

合理使用视图能够带来许多好处

- 1. 视图能够简化用户的操作
- 2. 视图使用户能以多种角度看待同一数据
- 3. 视图对重构数据库提供了一定程度的逻辑独立性
- 4. 视图能够对机密数据提供安全保护

（1）视图能够简化用户的操作

当视图中数据不是直接来自基本表时，定义视图能够简化用户的操作

- 基于多张表连接形成的视图
- 基于复杂嵌套查询的视图
- 含导出属性的视图

（2）视图使用户能以多种角度看待同一数据

视图机制能使不同用户以不同方式看待同一数据，适应数据库共享的需要

（3）视图对重构数据库提供了一定程度的逻辑独立性

物理独立性与逻辑独立性的概念

视图在一定程度上保证了数据的逻辑独立性

视图对重构数据库提供了一定程度的逻辑独立性

例：数据库逻辑结构发生改变

将学生关系

Student(Sno，Sname，Ssex，Sage，Sdept)

“垂直”地分成两个基本表：

SX(Sno，Sname，Sage)

SY(Sno，Ssex，Sdept)

视图对重构数据库提供了一定程度的逻辑独立性

通过建立一个视图Student：

```
CREATE VIEW  Student(Sno，Sname，Ssex，Sage，Sdept)
AS
SELECT  SX.Sno，SX.Sname，SY.Ssex，SX.Sage，SY.Sdept
FROM  SX，SY
WHERE  SX.Sno=SY.Sno；
```

使用户的外模式保持不变，从而对原Student表的查询程序不必修改

视图对重构数据库提供了一定程度的逻辑独立性

视图只能在一定程度上提供数据的逻辑独立性

- 由于对视图的更新是有条件的，因此应用程序中修改数据的语句可能仍会因基本表结构的改变而改变。

（4）视图能够对机密数据提供安全保护

- 对不同用户定义不同视图，使每个用户只能看到他有权看到的数据
- 通过**WITH CHECK OPTION**对关键数据定义操作时限制