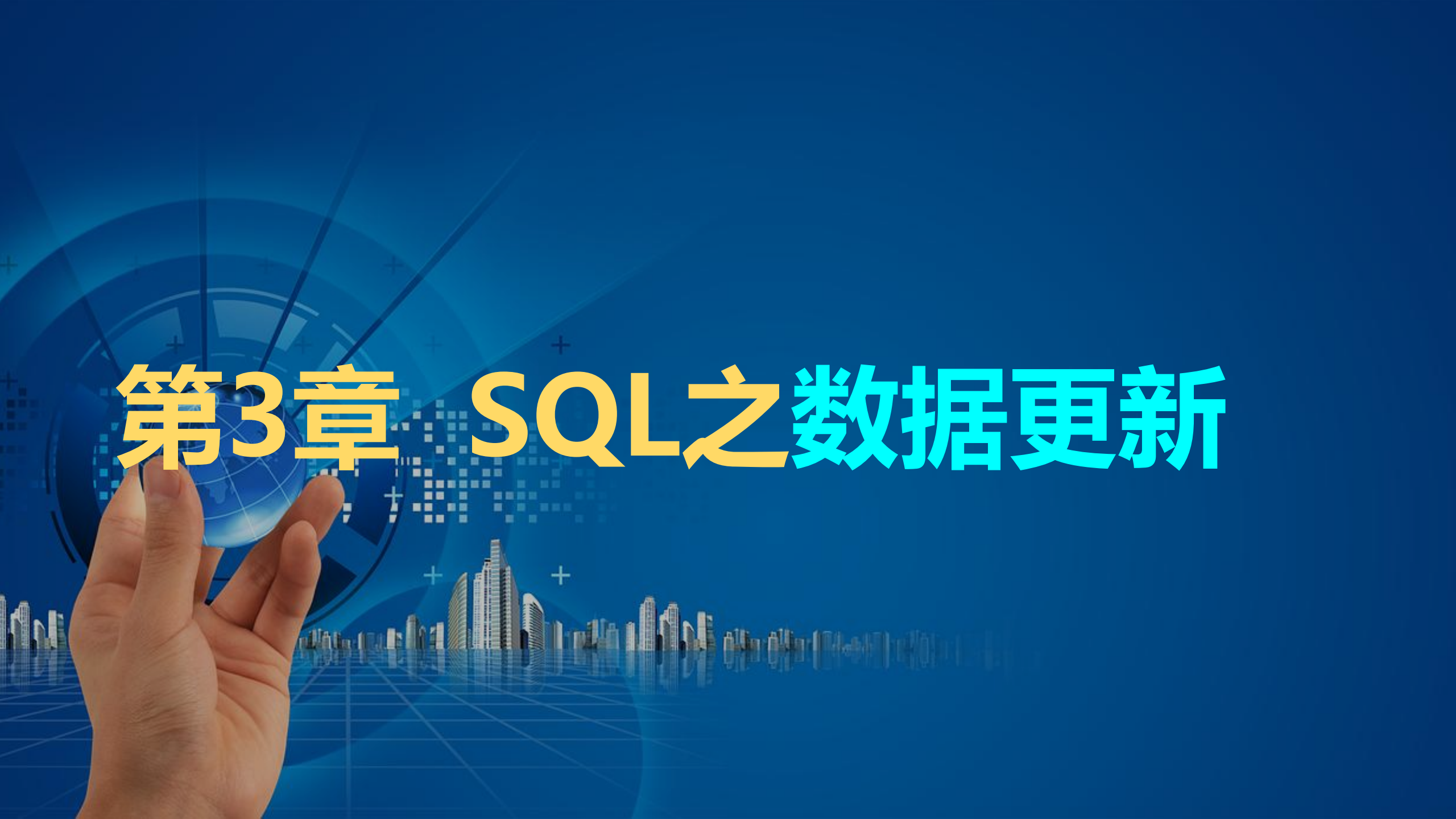

第3章 SQL之数据更新

7 数据更新

厦門大學 XIAMEN UNIVERSITY

- 本节主要内容
  - **插入数据**
  - **修改数据**
  - **删除数据**

7.1插入数据

- 语法格式：
  - **INSERT INTO <表名>[(<属性列1>[,<属性列2 >...)] VALUES (<常量1>[,<常量2>]...);**
  - 功能：将一个新元组插入到指定表中

INTO子句	VALUES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 • 没有指定属性列名：表示要插入的是一条完整的元组，且列属性与表定义中的顺序一致，新插入的元组必须在每个属性列上均有指定值 • 没有出现的属性列，新元组在这些列上将取空值 • 定义表时指定了相应属性列的缺省值，新元组在这些列上将取缺省值 • 在表定义时说明了NOT NULL的属性列不能取空值	• 提供的值必须与INTO子句匹配 – 值的个数 – 值的类型

7 示例

[例3.71] 将一个新学生元组（学号：**20180009**，姓名：陈冬，性别：男，出生日期：**2000-5-22**，主修专业：信息管理与信息系统）插入到**Student**表中

```
INSERT INTO Student (Sno, Sname, Ssex, Smajor, Sbirthdate)
VALUES ('20180009', '陈冬', '男','信息管理与信息系统', '2000-5-22');
```

- 字符串常数要用单引号（英文符号）括起来

[例3.72] 将学生张成民的信息插入到**Student**表中

```
INSERT INTO Student  VALUES ('20180008', '张成民', '男','2000-4-15','计算机科学与技术');
```

- 只有表名没有属性列名，要求表的所有属性列上都指定值，顺序与**CREATE TABLE**顺序相同

[例3.73] 插入一条选课记录

```
INSERT INTO SC(Sno,Cno,Semester,Teachingclass)  VALUES ('20180005','81004','20202','81004-01');
```

- **RDBMS**将在新插入记录的**Grade**列上自动地赋空值

```
INSERT INTO SC   VALUES ('20180005','81004',NULL, '20202','81004-01');
```

7.1插入数据(cont.)

- 插入子查询结果的语句格式

  INSERT INTO <表名>[(<属性列1> [,<属性列2>...)] 子查询;

  - 子查询中的SELECT子句的目标列必须与INTO子句后面属性列的值和类型都要匹配

[例3.74] 对每一个专业，求学生平均年龄，并把结果存入数据库

- 首先在数据库中建立一个新表，其中一列存放专业名，另一列存放学生的平均年龄

```
CREATE TABLE Smajor_age
                (Smajor  VARCHAR(20),
                 Avg_age SMALLINT);
```

- 对Student表按专业分组求平均年龄，再把专业名和平均年龄存入新表中

```
INSERT INTO Smajor_age(Smajor, Avg_age)
SELECT Smajor, AVG(extract(year from current_date)-extract(year from Sbirthdate))
FROM Student
GROUP BY Smajor;
```

7 openGauss的INSERT命令插入多行

厦門大學 XIAMEN UNIVERSITY

```
openGauss=# CREATE TABLE customer_t1
 (c_customer_sk integer,
  c_customer_id char(5),
  c_first_name char(6),
  c_last_name char(8)
  );
```

```
openGauss=# INSERT INTO customer_t1(c_customer_sk, c_customer_id,
c_first_name) VALUES (6885,'maps','Joes'), (4321,
'tpcds','Lily'),(9527,'world','James');
```

- 如果需要向表中插入多条数据，除了可以多次执行插入一行数据命令，也可以执行一次插入多条数据实现。
  - 使用此命令可以提升效率(建议)

7.1插入数据(cont.)

- RDBMS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（check约束）

7.2修改数据

- 修改操作又称为更新操作，其语句的一般语法为：

  UPDATE <表名>

  SET<列名>=<表达式>[,<列名>=<表达式>] ... ---此处的=号是赋值号，不是等号

  [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值
  - 如果省略**WHERE**子句，则表示要修改表中的所有元组

- 修改数据的三种方式:
  - **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**

7 示例

- 修改某一个元组的值
  - [例3.75] 将学生20180001的出生日期改为2001年3月18日

```
UPDATE Student
SET Sbirthdate='2001-3-18'
WHERE Sno='20180001';
```

- 修改多个元组的值
  - [例3.76] 将2020年第1学期选修81002课程所有学生的成绩减少5分

```
UPDATE SC
SET Grade= Grade-5
WHERE Semester='20201' AND Cno='81002';
```

- 带子查询的修改语句
  - [例3.77] 将计算机科学与技术专业学生成绩置零

```
UPDATE SC
SET  Grade=0
WHERE Sno IN (SELECT Sno  FROM Student  WHERE Smajor='计算机科学与技术');
```

问：修改数据时什么场景必须使用带子查询实现？
提示：涉及表的个数

- RDBMS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
  - 不允许修改主码
  - 用户定义的完整性
    - NOT NULL约束
    - UNIQUE约束
    - 值域约束（check约束）

7.3删除数据

- 语法格式：

  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
    - 缺省表示要删除表中的全部元组，表的定义仍在字典中

- 删除数据的三种方式：
  - 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

7 示例

- 删除某一个元组的值
  - [例3.78] 删除学号为20180007的学生记录

```
DELETE FROM Student WHERE Sno='20180007';
```

- 删除多个元组的值
  - [例3.79] 删除所有的学生选课记录

```
DELETE FROM SC;    ---成功执行语句之后SC表变为空表
```

- 带子查询的删除语句
  - [例3.80] 删除计算机科学与技术专业所有学生的选课记录

```
DELETE FROM SC
WHERE Sno  IN
            (SELECT Sno FROM Student WHERE Smajor= '计算机科学与技术' );
```

  - 问：什么场景必须使用带子查询的删除?

openGauss之TRUNCATE命令

- 语法格式：

TRUNCATE TABLE table_name;

- 功能：清理表数据， **TRUNCATE**快速地从表中删除所有行

表：DELETE、DROP和TRUNCATE三者比较

DELETE	DROP	TRUNCATE
• 删除表中满足条件的所有行 • 逐行删除，每删除一行将在事务日志登记删除记录 • 删除内容，不删除定义，不释放空间	• 删除内容和定义，释放空间	• 效果同DELETE FROM table_name; • 不扫描表，按数据页删除，因而删除速度快，使用系统资源和事务日志少 • 删除内容，不删除定义，释放空间

– **[例3.79]可用Truncate改写为: TRUNCATE TABLE SC;**

7 数据更新小结

- 数据更新包括数据的插入、修改和删除
  - 三种操作都只能在单表上操作
  - 语法格式上易与多表查询混淆
- 当修改或删除操作涉及多张表时，只能通过子查询完成
- 一些**RDBMS**产品，如**openGauss**、**MySQL**支持一条**INSERT**语句实现插入多条记录，而不需要分别执行多条**insert**语句
- **openGauss**的**truncate**删除命令效率比**delete**更高，因其需要的事务日志资源更少

8 空值的处理

厦門大學 XIAMEN UNIVERSITY

- 空值(Null) 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
- 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
- 本节内容：
  - **空值的产生**
  - **空值的判断**
  - **空值约束**
  - **空值的算术运算、比较运算和逻辑运算**

8.1空值的产生

[例3.81] 向SC表中插入一个元组，学生号是“20180006”，课程号是“81004”，选课学期2021年第1学期，选课班没有确定，成绩还没有

```
INSERT INTO SC(Sno,Cno,Grade,Semester,Teachingclass)
VALUES('20180006', '81004',NULL, '20211',NULL);
```

或

```
INSERT INTO SC(Sno,Cno,Semester)  VALUES('20180006', '81004','20211');
```

[例3.82] 将Student表中学生号为“20180006”的学生主修专业改为空值

```
UPDATE Student
SET Smajor = NULL
WHERE Sno='20180006';
```

- 外连接也会产生空值，参见3.3.2小节 [例3.55]
- 空值的关系运算也会产生空值

8.2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3.83] 从Student表中找出漏填了数据的学生信息

```
SELECT *
FROM Student
WHERE Sname IS NULL OR Ssex IS NULL OR Sbirthdate IS NULL OR  Smajor IS NULL;
```

– Sno是主码，不允许取空值，不许漏填

8.3空值约束

- 以下几种情况不能取空值：
  - 在创建基本表时，如果属性定义(或者域定义)为**NOT NULL约束**，则该属性不能取空值
  - 主码的属性不能取空值，确切讲是主属性不能为空值
    - 如，**SC.Sno, SC.Cno, Student.Sno, Course.Cno**都不能取空值

- 具有**UNIQUE**约束的属性则可以取空值
  - 是否可以有多个记录在此属性上取空值?
    - **SQL**标准对此没有明确规定，各个**RDBMS**在实现时也不尽相同。**Oracle**，**KingbaseES**，**MySQL**和**PostgreSQL**允许，但**SQL Server**则不允许

- **DISTINCT**对**NULL**的作用
  - 在使用**distinct**查询时**SQL**标准认为**NULL**是相同的，即若查询结果出现多个**NULL**，则只保留一个
  - “Two null values are not distinct. A null value and a non-null value are distinct”

8.4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

表3.7 逻辑运算AND、OR、NOT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

[例3.84] 找出选修81001号课程且成绩不及格的学生

```
SELECT Sno
FROM SC
WHERE Grade < 60 AND Cno='81001';
```

[例3.85] 选出选修81001号课程且成绩不及格的学生以及缺考的学生

```
SELECT Sno
FROM SC  WHERE Cno='81001' AND (Grade < 60 OR Grade IS NULL);
```

或

```
SELECT Sno
FROM SC  WHERE Grade < 60  AND Cno='81001'
UNION
SELECT Sno
FROM SC  WHERE Grade IS NULL AND Cno='81001';
```

- 视图(view)是RDBMS提供给用户以多种角度观察数据库种数据的重要机制。
- 视图是从一个或几个基本表(或视图)导出的表，是虚表
  - 只存放视图的定义，不存放视图对应的数据
- 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

- 本节主要内容：
  - **定义视图**
  - **查询视图**
  - **更新视图**
  - **视图的作用**

9.1定义视图

- 定义视图的语句格式：

```
CREATE  VIEW <视图名>[(<列名>[,<列名>]...)]
AS <子查询>
[WITH  CHECK  OPTION];
```

视图字段全部省略：
- 由子查询中SELECT目标列中的诸字段组成

指明字段：
- 某个目标列是聚集函数或列表达式
- 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

– **WITH CHECK OPTION子句**

- 对视图进行**UPDATE**，**INSERT**和**DELETE**操作时要保证更新、插入或删除的行满足视图定义中的谓词条件（即子查询中的条件表达式）
- 子查询可以是任意的**SELECT**语句，是否可以含有**ORDER BY**子句和**DISTINCT**短语，则决定于具体系统的实现
- **RDBMS**执行**CREATE VIEW**语句时只是把视图定义存入数据字典，并不执行其中的**SELECT**语句
- 在对视图查询时，按视图的定义从基本表中将数据查出

9 示例

[例3.86] 建立信息管理与信息系统专业学生的视图

```
CREATE VIEW IS_Student
AS
SELECT Sno, Sname, Ssex, Sbirthdate, Smajor
FROM Student
WHERE Smajor='信息管理与信息系统' ;
```

[例3.87] 建立信息管理与信息系统专业学生的视图，并要求进行插入、修改和删除操作时仍需保证该视图只有信息管理与信息系统专业的学生

```
CREATE VIEW IS_Student
AS
SELECT Sno, Sname, Ssex, Sbirthdate, Smajor
FROM Student
WHERE Smajor='信息管理与信息系统'
WITH CHECK OPTION;
```

定义IS_Student视图时加上了WITH CHECK OPTION子句，对该视图进行插入、修改和删除操作时，RDBMS会自动加上Smajor='信息管理与信息系统' 的条件。

9.2视图类型

- 行列子集视图
  - 若一个视图是从单个基本表导出的，并且只是去掉了基本表的某些行和某些列，但保留了主码。如，**IS_Student**为行列子集视图

- 基于多个基表的视图

[例3.88] 建立信息管理与信息系统专业选修了**81001**号课程的学生的视图(包括学号、姓名、成绩属性).

```
CREATE VIEW IS_C1(Sno, Sname, Grade)
AS
SELECT Student.Sno, Sname, Grade
FROM Student, SC
WHERE Smajor='信息管理与信息系统' AND  Student.Sno=SC.Sno AND SC.Cno='81001';
```

- 基于视图或表的视图
  - [例3.89] 建立信息管理与信息系统专业选修了81001号课程且成绩在90分以上的学生的视图(包括学号、姓名、成绩属性)

```
CREATE VIEW IS_C2          --IS_C2建立在视图IS_C1上的
AS
SELECT Sno,Sname,Grade
FROM  IS_C1
WHERE Grade>=90;
```

- 带表达式的视图
  - [例3.90] 将学生的学号、姓名、年龄定义为一个视图.

```
CREATE VIEW S_AGE(Sno, Sname, Sage)    --Sage为虚拟列，其值由子查询计算得到
AS
SELECT Sno,Sname,(extract(year from current_date)-extract(year from Sbirthdate) )
FROM Student;
```

- 分组视图
  - [例3.91] 将学生的学号及平均成绩定义为一个视图

```
CREATE VIEW S_GradeAVG(Sno,Gavg)  --S_GradeAVG为分组视图
AS
SELECT Sno,AVG(Grade)
FROM SC
GROUP BY Sno;
```

  - [例3.92] 将Student表中所有女生的记录定义为一个视图.

```
CREATE VIEW F_Student(Fsno,Fname,Fsex,Fbirthdate,Fmajor)
AS
SELECT *
FROM Student
WHERE Ssex='女';
```

- 视图的属性列与Student表的属性列一一对应
- 修改基表Student的结构后，Student表与F_Student视图的映象关系被破坏，导致该视图不能正确工作

9.3删除视图

- 删除视图语句格式：

DROP VIEW <视图名> [CASCADE];

- 视图定义从数据字典中删除
- 若该视图上还导出了其他视图，使用**CASCADE**级联删除语句，把该视图和由它导出的所有视图一起删除
- 删除基表后，由该基表导出的所有视图均无法使用了，但视图的定义没有从字典中删除，所以必须显式地使用**DROP VIEW**语句删除这些无效视图定义

[例3.93] 删除视图**S_AGE**和视图**IS_C1**：

```
DROP VIEW S_AGE;      /*成功执行*/
DROP VIEW IS_C1;      /*报告错误，因视图IS_C2依赖于IS_C1*/
DROP VIEW IS_C1 CASCADE;   --使用CASCADE后IS_C2和IS_C1均被删除
```

9.4查询视图

- 用户角度：查询视图与查询基本表相同
- **RDBMS**实现视图查询的方法
  - 视图消解法(**View Resolution**)

  - 物化视图(**Materialized View**)
    - 物化视图用于预先计算并保存表连接或聚集等耗时较多的操作的结果，这样在执行查询时，就可以避免进行这些耗时的操作，快速得到结果，从而提高查询性能

      物化视图的特点：
    - 物理真实存在，故需要占用存储空间
    - 物化视图对应用透明，增加和删除物化视图不会影响应用程序中**SQL**语句的正确性和有效性
    - 当基本表发生变化时，物化视图也应刷新(**Refresh**)
- **oracle 物化视图参考：**https://blog.csdn.net/yangshangwei/article/details/53328605

[例3.94] 在信息管理与信息系统专业学生的视图中，找出年龄小于等于20岁的学生（包括学生的学号和出生日期）

```
SELECT Sno, Sbirthdate
FROM IS_Student
WHERE (extract(year from current_date)-extract(year from Sbirthdate) )<=20;
```

- 视图消解法的执行过程：
  - 先找到视图**IS_Student**的定义
  - 然后进行视图消解，转换后的查询语句为

```
CREATE VIEW IS_Student
AS
SELECT Sno, Sname, Ssex, Sbirthdate, Smajor
FROM Student
WHERE Smajor='信息管理与信息系统'
WITH CHECK OPTION;
```

```
SELECT Sno,Sbirthdate
FROM Student
WHERE Smajor='信息管理与信息系统' AND
      (extract(year from current_date)-extract(year from Sbirthdate) )<=20;
```

示例(cont.)

[例3.95] 查询选修了81001号课程的信息管理与信息系统专业学生

```
SELECT IS_Student.Sno,Sname
FROM IS_Student, SC
WHERE  IS_Student.Sno=SC.Sno AND SC.Cno='81001';
```

- 视图消解法的执行过程：
  - 先找到视图IS_Student的定义
  - 然后进行视图消解，转换后的查询语句为

```
SELECT S.Sno,Sname
FROM Student S, SC
WHERE  S.Sno=SC.Sno AND SC.Cno='81001' AND  Smajor='信息管理与信息系统';
```

9 ~视图消解法的局限性

- 有些情况下，视图消解法不能生成正确的查询

[例3.96] 在S_GradeAVG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_GradeAVG
WHERE Gavg>=90;
```

正确

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

消解错误

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

```
CREATE VIEW S_GradeAVG(Sno,Gavg)  --S_GradeAVG为分组视图
AS
SELECT Sno,AVG(Grade)
FROM SC
GROUP BY Sno;
```

- [例3.96]也可以用如下SQL语句完成：

```
SELECT *
FROM  (SELECT Sno, AVG(Grade)      /*子查询生成一个派生表S_GradeAVG*/
       FROM SC
       GROUP BY Sno) AS S_GradeAVG(Sno, Gavg)
WHERE Gavg>=90;
```

– 对视图查询与基于派生表查询的区别

  - 视图一旦定义，将保存在数据字典中，之后的所有查询都可以直接引用该视图
  - 派生表只是在语句执行时临时定义，语句执行后该派生表定义即被删除

9 oracle视图示例

```
1  CREATE VIEW salesman AS
2  SELECT
3      *
4  FROM
5      employees
6  WHERE
7      job_title = 'Sales Representative';
```

```
1  SELECT
2      *
3  FROM
4      salesman;
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE	HIRE_DATE	MANAGER_ID	JOB_TITLE
56	Evie	Harrison	evie.harrison@example.com	011.44.1344.486508	23-NOV-07	46	Sales Representative
57	Scarlett	Gibson	scarlett.gibson@example.com	011.44.1345.429268	30-JAN-04	47	Sales Representative
58	Ruby	Mcdonald	ruby.mcdonald@example.com	011.44.1345.929268	04-MAR-04	47	Sales Representative
59	Chloe	Cruz	chloe.cruz@example.com	011.44.1345.829268	01-AUG-04	47	Sales Representative
60	Isabelle	Marshall	isabelle.marshall@example.com	011.44.1345.729268	10-MAR-05	47	Sales Representative
61	Daisy	Ortiz	daisy.ortiz@example.com	011.44.1345.629268	15-DEC-05	47	Sales Representative
62	Freya	Gomez	freya.gomez@example.com	011.44.1345.529268	03-NOV-06	47	Sales Representative
80	Elizabeth	Dixon	elizabeth.dixon@example.com	011.44.1644.429262	04-JAN-08	50	Sales Representative
64	Florence	Freeman	florence.freeman@example.com	011.44.1346.229268	19-MAR-07	48	Sales Representative
65	Alice	Wells	alice.wells@example.com	011.44.1346.329268	24-JAN-08	48	Sales Representative

9 oracle视图示例(cont.)

```sql
CREATE VIEW salesman_contacts AS
SELECT
    first_name,
    last_name,
    email,
    phone
FROM
    salesman;
```

```sql
SELECT
    *
FROM
    salesman_contacts;
```

FIRST_NAME	LAST_NAME	EMAIL	PHONE
Evie	Harrison	evie.harrison@example.com	011.44.1344.486508
Scarlett	Gibson	scarlett.gibson@example.com	011.44.1345.429268
Ruby	Mcdonald	ruby.mcdonald@example.com	011.44.1345.929268
Chloe	Cruz	chloe.cruz@example.com	011.44.1345.829268
Isabelle	Marshall	isabelle.marshall@example.com	011.44.1345.729268
Daisy	Ortiz	daisy.ortiz@example.com	011.44.1345.629268
Freya	Gomez	freya.gomez@example.com	011.44.1345.529268
Elizabeth	Dixon	elizabeth.dixon@example.com	011.44.1644.429262
Florence	Freeman	florence.freeman@example.com	011.44.1346.229268
Alice	Wells	alice.wells@example.com	011.44.1346.329268

oracle视图示例(cont.)


```
DROP VIEW salesman;
```

```
SELECT
    object_name,
    status
FROM
    user_objects
WHERE
    object_type = 'VIEW'
    AND object_name = 'SALESMAN_CONTACTS';
```

OBJECT_NAME	STATUS
SALESMAN_CONTACTS	INVALID

```
DROP VIEW salesman_contacts;
```

- 因为salesman_contacts视图依赖于salesman视图，所以saleman被删除后它就变得无效
- 通过user_objects视图可以查看salesman_contacts的状态，注意所有字母要全部大写
- 此时可以删除salesman_contacts视图

9 openGauss之视图与物化视图

厦門大學 XIAMEN UNIVERSITY

- 官网：
  - 视图

    https://www.opengauss.org/zh/docs/3.1.0/docs/BriefTutorial/%E8%A7%86%E5%9B%BE.html
  - 物化视图

    https://www.opengauss.org/zh/docs/3.1.0/docs/BriefTutorial/%E7%89%A9%E5%8C%96%E8%A7%86%E5%9B%BE.html

- 墨天轮：
  - **https://www.modb.pro/db/208337**
  - https://www.modb.pro/db/41233

9.5更新视图

- 更新视图是指通过视图来插入、删除和修改数据
  - 视图的更新操作通过视图消解，转换为对基本表的更新操作
  - 为防止用户有意无意地对不属于视图范围内的基本表数据进行操作，可在定义视图时加上**WITH CHECK OPTION**子句
    - 在视图上增、删、改数据时，**RDBMS**会检查视图定义中的条件，若不满足条件则拒绝执行该操作

[例3.97] 将信息管理与信息系统专业学生视图**IS_Student**中学号为“**20180005**”的学生姓名改为“刘新奇”

```
UPDATE IS_Student
SET Sname='刘新奇'
WHERE Sno='20180005';
```

⇒

```
UPDATE Student
SET Sname='刘新奇'
WHERE Sno='20180005'AND Smajor='信息管理与信息系统';
```

9 示例

[例3.98] 向信息管理与信息系统专业学生视图IS_Student中插入一个新的学生记录

(20180207, 赵新, 男, 2001-7-19)

```
INSERT INTO IS_Student   VALUES('20180207','赵新','男','2001-7-19','信息管理与信息系统');
```

```
INSERT INTO Student(Sno,Sname,Ssex,Sbirthdate,Smajor)
VALUES('20180207','赵新','男','2001-7-19,'信息管理与信息系统');
```

- 系统自动将'信息管理与信息系统'放入VALUES子句中

[例3.99] 删除信息管理与信息系统专业学生视图IS_Student中学号为 “20180207”的记录

```
DELETE FROM IS_Student  WHERE Sno='20180207';
```

```
DELETE FROM Student    WHERE Sno='20180207' AND Smajor='信息管理与信息系统';
```

9 视图更新的限制

- 并不是所有的视图都是可更新的
  - [例3.91]定义的视图S_GradeAVG是由学号和平均成绩两个属性列组成的，平均成绩是由Student表中对元组分组后计算平均值得来
  - 如果想把视图S_GradeAVG中学号为“20180001”学生的平均成绩改成90分

```
UPDATE S_GradeAVG  SET Gavg=90  WHERE Sno='20180001';
```

    - 对视图的更新是无法转换成对基本表SC的更新，因为系统无法修改各科成绩，以使20180001学生平均成绩成为90
    - 所以S_GradeAVG视图是不可更新的

- 一般地，行列子集视图是可更新的
  - 除行列子集视图外，有些视图理论上是可更新的，还有些视图从理论上就是不可更新的

- 不可更新的视图与不允许更新的视图不同
  - 前者指理论上已证明是不可更新视图
  - 后者指实际系统中不支持其更新，但它本身有可能是可更新的视图
  - **DB2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
    - 若视图的字段来自聚集函数，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
    - 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
  - **openGauss不支持基于视图的更新**

9.6视图的作用

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

9 作用1：视图能够对机密数据提供安全保护

- 对不同用户定义不同的视图，使机密数据不出现在不应看到这些数据的用户视图上
- 自动提供了对机密数据的安全保护功能
- 例如，**Student**表涉及全校**30**个院系的学生数据
  - 可以在其上定义**30**个视图
  - 每个视图只包含一个院系的学生数据
  - 只允许每个院系的主任查询和修改本院系的学生视图

9 作用2：视图对重构数据库提供了一定程度的逻辑独立性

厦門大學 XIAMEN UNIVERSITY

- 数据的逻辑独立性是指当数据库重构造时，如增加新的关系或对原有关系增加新的字段等，用户的应用程序不会受影响

- 数据库重构：
  - 例：学生关系**Student(Sno,Sname,Ssex,Sbirthdate,Smajor)**“垂直”地分成两个基本表：
    - **SX(Sno,Sname,Sbirthdate)**和**SY(Sno,Ssex,Smajor)**
  - 通过建立一个视图**Student**：

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

    - 使用户的外模式保持不变，用户的应用程序通过视图仍然能够查找数据
    - 视图只能在一定程度上提供数据的逻辑独立性
    - 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相应修改

9 作用3：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

- 适当的利用视图可以更清晰的表达查询，如，
  - 经常查询“对每个同学找出他获得最高成绩的课程号”
  - 可以先定义一个视图，求出每个同学获得的最高成绩

```
CREATE VIEW VMGrade
AS
SELECT Sno, MAX(Grade)  Mgrade
FROM  SC
GROUP BY Sno;
```

利用VMGrade

```
SELECT SC.Sno,Cno
FROM SC,VMGrade
WHERE SC.Sno=VMGrade.Sno AND
SC.Grade=VMGrade .Mgrade;
```

9 作用4：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

- 希望了解学生的平均成绩，学生的最高成绩和最低成绩，都可以在基本表SC上定义自己感兴趣的视图，直接对这些视图查询

- 由于上述优点，视图被广泛用于实际应用开发中

10 本章小结

- SQL可以分为数据定义、数据查询、数据更新、数据控制四大部分
  - 有时数据更新称为数据操纵，有时数据查询和数据更新合称为数据操纵
- SQL是关系数据库语言的工业标准。大部分数据库管理系统产品都能支持SQL 92，但是许多数据库系统只支持SQL 99、SQL2008和SQL2011的部分特征，至今尚没有一个数据库系统能够完全支持SQL99以上的标准
- SQL的数据查询功能非常丰富，也比较复杂

11 本章作业

- 教材第三章全部习题.
- 要求：作业在布置后一周内完成并提交到课程网站