

实验七 视图

实验目的

熟悉SQL语言支持的有关视图的操作，能够熟练使用SQL语句来创建需要的视图，对视图进行查询和取消视图。

实验内容

1. 定义常见的视图形式，包括：
 - 行列子集视图。
 - WITH CHECK OPTION的视图。
 - 基于多个基表的视图。
 - 基于视图的视图。
 - 带表达式的视图。
 - 分组视图。
2. 通过实验考察WITH CHECK OPTION这一语句在视图定义后产生的影响，包括对修改操作、删除操作、插入操作的影响。
3. 讨论视图的数据更新情况，对子行列视图进行数据更新。
4. 使用DROP语句删除一个视图，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。同样的原因，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。

注：以下是PostgreSQL包含check-option可更新视图简介，更多内容可参考[PostgreSQL官网](#)

一、创建可更新视图：

```
create or replace view usa_city as
    select city_id,city,country_id
        from city
        where country_id=103 order by city;
```

```
select * from usa_city;
```

更新视图插入数据：

```
insert into usa_city (city,country_id )values ('Birmingham', 102);
```

结果：插入成功，但是此插入的结果并不在视图usa_city中

插入的新行在视图中不可见。这可能会造成安全问题，为防止用户插入或更新通过视图不可见的行，在创建视图时可使用

WITH CHECK OPTION 子句。

二、创建有检查项的可更新视图

```
create or replace view usa_city as
    select city_id,city,country_id
        from city
        where country_id=103 order by city
        with check option;
```

更新视图插入数据：

```
insert into usa_city (city,country_id)values ('Birmingham', 102);
```

结果：> 错误： 新行违反了视图"usa_city"的检查选项

```
insert into usa_city (city,country_id)values ('Birenc', 103);
```

结果：插入成功，且只允许插入的数据满足原视图的where条件

三、检查项含local的可更新视图

(1)、创建一个可更新的基表视图

```
create or replace view usa_a as
    select city_id,city,country_id
    from city
    where city like 'A%';
```

(2)、创建检查项含local的可更新视图

```
create or replace view usa_a_city as
    select city_id,city,country_id
    from usa_a
    where country_id=103 order by city
    with local check option;
```

更新视图插入数据：

```
insert into usa_a_city(city,country_id)values('Mirmin', 103);
```

结果：插入成功，因为usa_a_city视图只需要检查自身的插入数据是否满足where条件即可

```
insert into usa_a_city(city,country_id)values('Mirmin', 102);
```

结果：插入失败，> 错误： 新行违反了视图"usa_a_city"的检查选项

(3)、创建检查项含cascaded的可更新视图

```
create or replace view usa_a_city as
    select city_id,city,country_id
    from usa_a
    where country_id=103 order by city
    with cascaded check option;
```

更新视图插入数据：

```
insert into usa_a_city(city,country_id)values('Meery', 103);
```

结果：插入失败，> 错误： 新行违反了视图"usa_a"的检查选项

因为usa_a_city视图使用了cascaded级联检查，即本身的where条件要满足同时也要满足基表视图的where条件

```
insert into usa_a_city(city,country_id)values('Aeery', 103);
```

结果：插入成功，同时满足了本身视图的where条件，也满足了基表视图usa_a的where条件

课内实验（遇到的问题为灰色字体）

要求：

以school数据库为例(与之前实验的数据同)，在该数据库中存在4张表格，分别为：

STUDENTS(sid,sname,email,grade)

TEACHERS(tid,tname,email,salary)

COURSES(cid,cname,hour)

CHOICES(no,sid,tid,cid,score)

1. 创建一个行列子集视图，给出选课成绩合格的学生的编号，所选课程号和该课程成绩

```
CREATE VIEW QualifiedStudents AS
SELECT sid, cid, score
FROM CHOICES
WHERE score >= 60;
```

The screenshot shows a database management tool interface. On the left, a tree view displays the database structure, including 'adjustedscores', 'qualifiedstudents', and 'Columns (4)' with columns 'no', 'sid', 'cid', and 'score'. The main panel shows the SQL query used to create the view: `CREATE VIEW QualifiedStudents AS SELECT no, sid, cid, score FROM CHOICES WHERE score >= 60;`. Below the query, the 'Messages' tab shows the confirmation: 'CREATE VIEW' and 'Query returned successfully in 114 msec.'

可以看到视图创建成功

2. 创建基于多个基表的视图，这个视图由学生姓名和其所选修的课程名及讲授该课程的教师姓名构成

```
CREATE VIEW StudentCoursesTeachers AS
SELECT s.sname, cr.cname, t.tname
FROM STUDENTS s
JOIN CHOICES ch ON s.sid = ch.sid
JOIN COURSES cr ON ch.cid = cr.cid
JOIN TEACHERS t ON ch.tid = t.tid;
```

Query Query History

```
1 CREATE VIEW StudentCoursesTeachers AS
2 SELECT s.sname, cr.cname, t.tname
3 FROM STUDENTS s
4 JOIN CHOICES ch ON s.sid = ch.sid
5 JOIN COURSES cr ON ch.cid = cr.cid
6 JOIN TEACHERS t ON ch.tid = t.tid;
```

Data Output Messages Notifications

CREATE VIEW

Query returned successfully in 95 msec.

可以看到此视图也创建成功

3. 创建带表达式的视图，由学生姓名、所选课程名和所有课程成绩都比原来多5分这几个属性组成

views (3)

fivemorescores

Columns (3)

sname

cname

?column?

Rules

Triggers

qualifiedstudents

Columns (3)

sid

cid

score

Rules

Triggers

studentcoursesteac

Query Query History

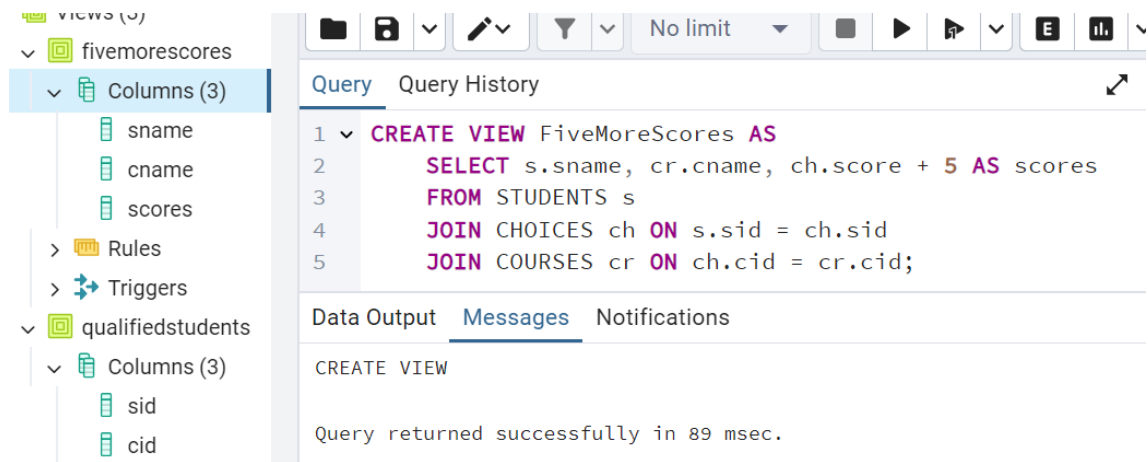
```
1 CREATE VIEW FiveMoreScores AS
2 SELECT s.sname, cr.cname, ch.score + 5
3 FROM STUDENTS s
4 JOIN CHOICES ch ON s.sid = ch.sid
5 JOIN COURSES cr ON ch.cid = cr.cid;
```

Data Output Messages Notifications

CREATE VIEW

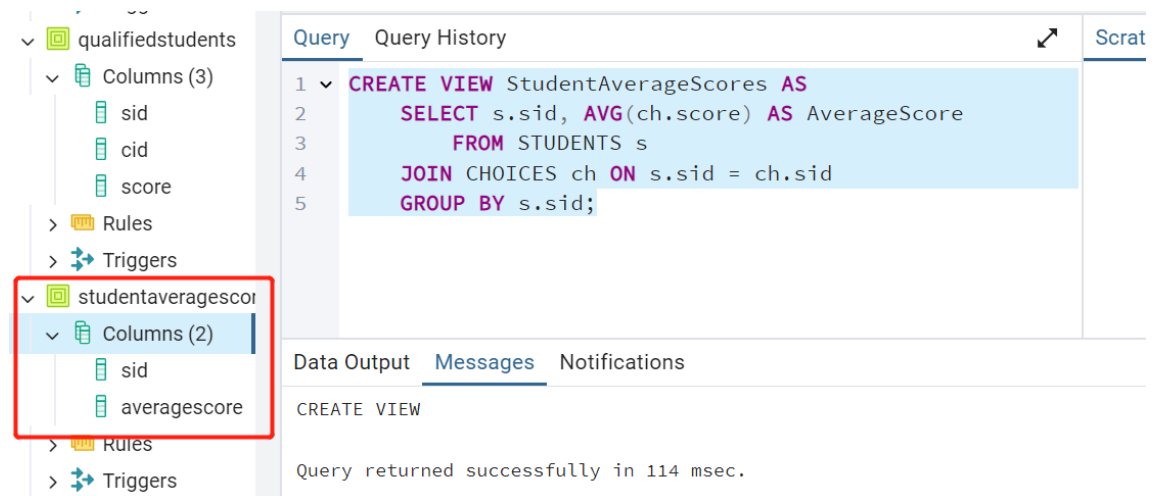
Query returned successfully in 87 msec.

可以看到虽然试图创建成功，但有一列名称很奇怪，所以我们修改一下，给它个名字



这时候就没问题了

4. 创建分组视图，将学生的学号及其平均成绩定义为一个视图



创建成功

5. 创建一个基于视图的视图，基于(1)中建立的视图，定义一个包括学生编号，学生所选课程数目和平均成绩的视图

```
CREATE VIEW StudentCourseInfo AS
  SELECT sid, COUNT(cid) AS CourseCount, AVG(score) AS AverageScore
  FROM QualifiedStudents
  GROUP BY sid;
```

Query Query History

```

1 CREATE VIEW StudentCourseInfo AS
2 SELECT sid, COUNT(cid) AS CourseCount, AVG(score) AS
3 FROM QualifiedStudents
4 GROUP BY sid;

```

Columns (2)

- sid
- averagescore

Rules

Triggers

studentcourseinfo

Columns (3)

- sid
- coursecount
- averagescore

Rules

Triggers

Data Output Messages Notifications

CREATE VIEW

Query returned successfully in 89 msec.

创建成功

6. 查询所有选修课程Software Engineering的学生姓名

Query Query History

```

1 SELECT s.sname, cr.cname
2 FROM STUDENTS s
3 JOIN CHOICES ch ON s.sid = ch.sid
4 JOIN COURSES cr ON ch.cid = cr.cid
5 WHERE cname = 'software engineering';

```

Data Output Messages Notifications

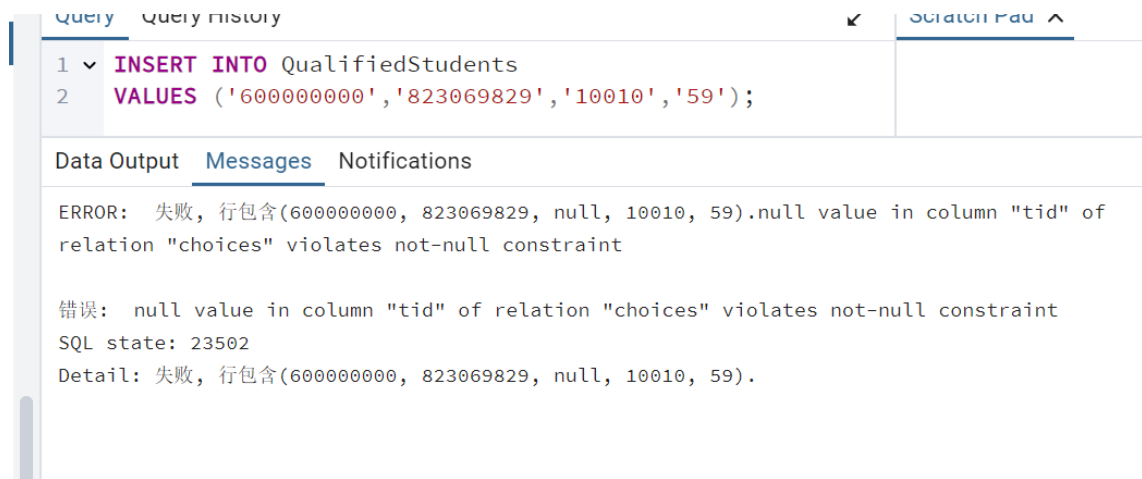
SQL

	sname character varying (30)	cname character varying (30)
1	qei ygk	software engineering
2	xizbbugq	software engineering
3	uyhrfe	software engineering
4	okfbo	software engineering
5	snldyau	software engineering
6	eyozzi	software engineering
7	yqeggyn	software engineering
8	xwjubxj	software engineering
9	yjydgypko	software engineering
10	lipkeic	software engineering
11	nuflun	software engineering
12	hgttbact	software engineering
13	nqlfd	software engineering

查询成功

7. 插入元组(600000000,823069829,10010,59)到视图CS中。若是在视图的定义中存在WITH CHECK OPTION子句对插入操作有什么影响?

(视图CS即为第一题的视图QualifiedStudents)



The screenshot shows a SQL query editor with a query window and a messages window. The query window contains the following SQL statement:

```
1 INSERT INTO QualifiedStudents
2 VALUES ('600000000','823069829','10010','59');
```

The messages window shows the following error message:

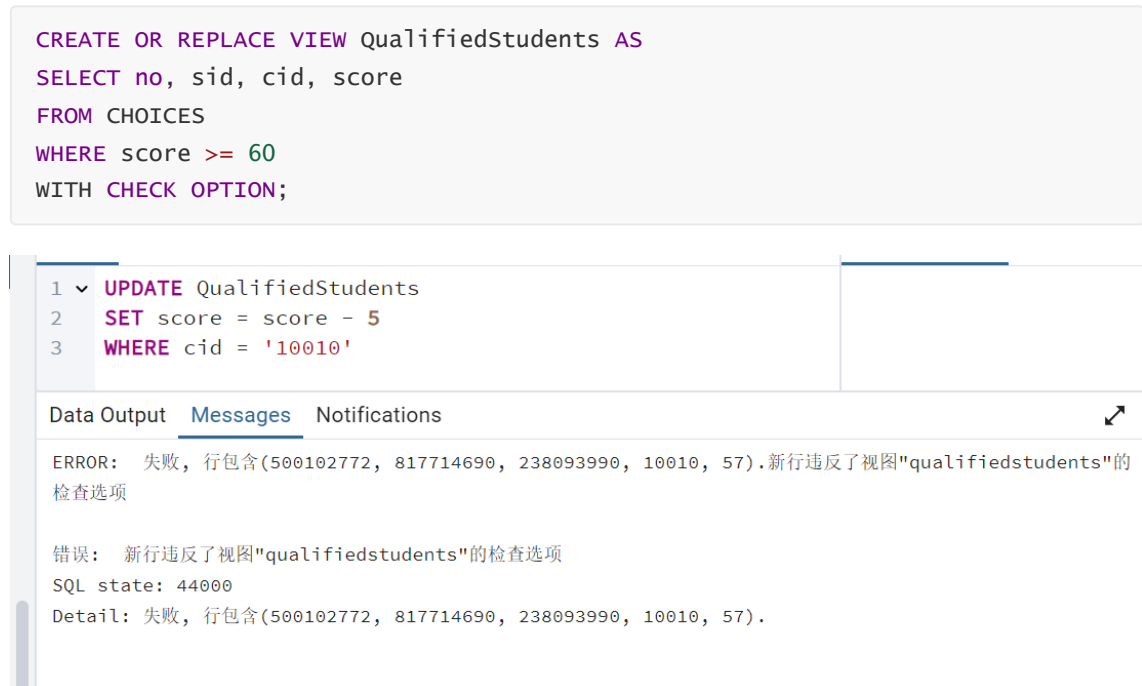
```
ERROR: 失败, 行包含(600000000, 823069829, null, 10010, 59).null value in column "tid" of
relation "choices" violates not-null constraint

错误: null value in column "tid" of relation "choices" violates not-null constraint
SQL state: 23502
Detail: 失败, 行包含(600000000, 823069829, null, 10010, 59).
```

如果视图QualifiedStudents中定义了WITH CHECK OPTION, 那么任何插入操作都必须满足视图的WHERE条件scores >= 60。插入的数据不满足这个条件, 所以插入操作失败。

8. 将视图CS (包含定义WITH CHECK OPTION)中, 所有课程编号为10010的课程的成绩都减去5分。这个操作数据库是否会正确执行, 为什么?如果加上5分(原来95分以上的不变)呢?

将QualifiedStudents定义WITH CHECK OPTION



The screenshot shows a SQL query editor with a query window and a messages window. The query window contains the following SQL statements:

```
CREATE OR REPLACE VIEW QualifiedStudents AS
SELECT no, sid, cid, score
FROM CHOICES
WHERE score >= 60
WITH CHECK OPTION;
```

The messages window shows the following error message:

```
ERROR: 失败, 行包含(500102772, 817714690, 238093990, 10010, 57).新行违反了视图"qualifiedstudents"的
检查选项

错误: 新行违反了视图"qualifiedstudents"的检查选项
SQL state: 44000
Detail: 失败, 行包含(500102772, 817714690, 238093990, 10010, 57).
```

这个操作不会正确执行, 因为WITH CHECK OPTION会阻止任何会使视图条件不满足的更新操作。有些成绩减去5分后低于60分, 所以这个更新操作会失败。

```
1  ▼ UPDATE QualifiedStudents
2    SET score = score + 5
3    WHERE cid = '10010' AND score < 95
```

Data Output Messages Notifications

UPDATE 4170

Query returned successfully in 168 msec.

此操作成功，因为视图条件仍然会被满足。

9. 在视图CS (包含定义WITH CHECK OPTION)删除编号为804529880学生的记录，会产生什么结果？

Query Query History

```
1  ▼ DELETE FROM QualifiedStudents
2    WHERE SID= '804529880'
```

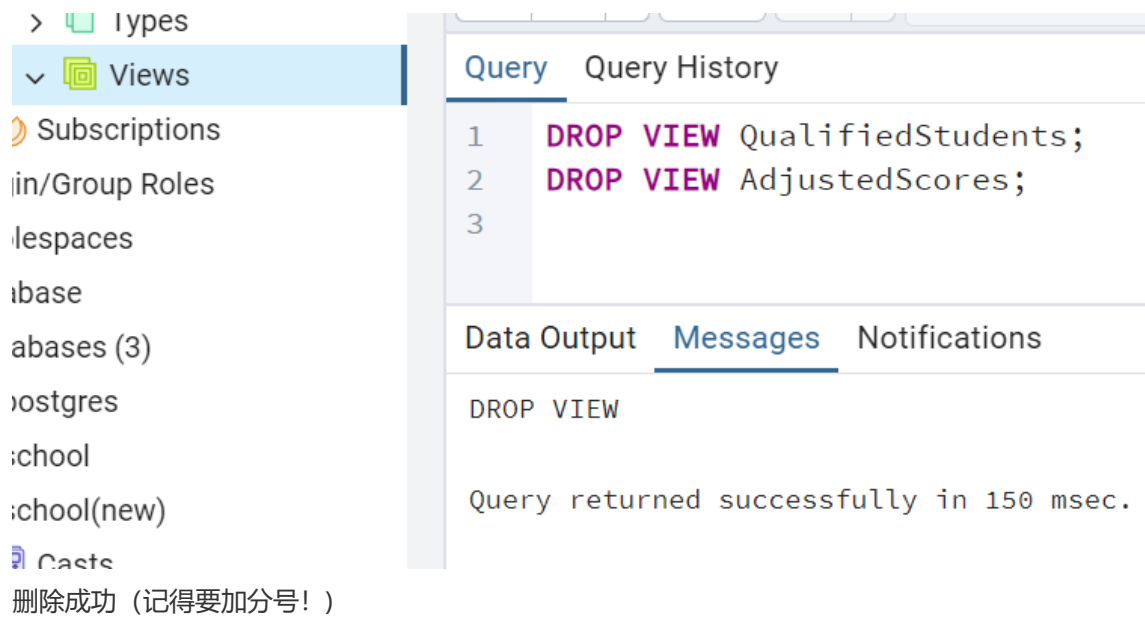
Data Output Messages Notifications

DELETE 5

Query returned successfully in 89 msec.

成功删除。因为删除它并不会与视图条件冲突

10. 取消视图SCT和视图CS



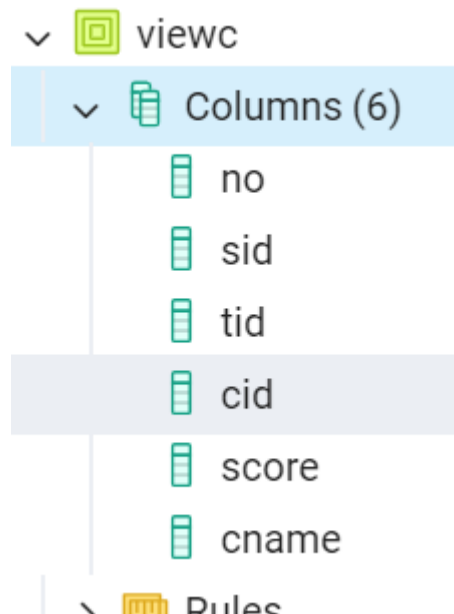
删除成功（记得要加分号！）

自我实践

1. 定义选课信息和课程名称的视图VIEWC

```
CREATE VIEW VIEWC AS
SELECT ch.*, cr.cname
FROM CHOICES ch
JOIN COURSES cr ON ch.cid = cr.cid;
```

创建成功



2. 定义学生姓名与选课信息的视图VIEWS

The screenshot shows the SQL Server Enterprise Manager on the left and the Query Editor on the right. In the Enterprise Manager, the 'views' folder under the 'studentcoursesteac' database is selected, and its columns (sname, cid, score) are visible. The Query Editor shows the following SQL script:

```
1 CREATE VIEW VIEWS AS
2     SELECT sname, cid, score
3     FROM STUDENTS s
4     JOIN CHOICES ch ON s.sid = ch.sid;
```

The 'Messages' tab in the Query Editor shows the following output:

```
CREATE VIEW

Query returned successfully in 49 msec.
```

创建成功

3. 定义年级低于1998的学生的视图S1(SID,SNAME,GRADE)

The screenshot shows the SQL Server Enterprise Manager on the left and the Query Editor on the right. In the Enterprise Manager, the 's1' view under the 'adjustedscores' database is selected, and its columns (sid, sname, grade) are visible. The Query Editor shows the following SQL script:

```
1 CREATE VIEW S1 AS
2     SELECT sid,sname,grade
3     FROM STUDENTS
4     WHERE grade > 1998
```

The 'Messages' tab in the Query Editor shows the following output:

```
CREATE VIEW

Query returned successfully in 120 msec.
```

创建成功

4. 查询学生为“uxjof”的学生的选课信息

Query Query History

```

1 SELECT *
2   FROM VIEWS
3  WHERE sname = 'uxjof'

```

Data Output Messages Notifications

	sname character varying (30)	no integer	cid character (5)	score integer
1	uxjof	506978093	10046	94
2	uxjof	541221076	10018	84
3	uxjof	567316431	10037	98

5. 查询选修课程“UML”的学生的编号和成绩

Query Query History Scratch Pad X

```

1 SELECT *
2   FROM VIEWC
3  WHERE cname = 'uml'

```

Data Output Messages Notifications

	no integer	sid character (9)	tid character (9)	cid character (5)	score integer	cname character varying (30)
1	500006805	848035070	208952048	10007	88	uml
2	500010697	897664264	260797745	10007	62	uml
3	500041524	898453203	237947994	10007	91	uml
4	500056974	846108663	217840948	10007	86	uml
5	500070174	827984677	244871065	10007	86	uml
6	500082840	823352185	224956108	10007	[null]	uml
7	500087165	882778410	228002437	10007	69	uml
8	500091038	884993242	235035702	10007	74	uml
9	500099590	806427512	260989974	10007	96	uml
10	500126657	848803637	218987065	10007	85	uml
11	500154955	839666024	223844488	10007	92	uml
12	500156358	844745062	220113589	10007	96	uml
13	500188210	818493215	285423878	10007	88	uml
14	500195978	815954009	222720046	10007	62	uml
15	500198355	828224326	240882774	10007	80	uml

6. 向视图S1插入记录("60000001,Lily,2001")

Query

Query History

1

▼

INSERT INTO S1 (sid, sname, grade)

2

VALUES ('60000001', 'Lily', 2001);

Data Output

Messages

Notifications

INSERT 0 1

Query returned successfully in 47 msec.

插入成功，我们查找一下：

1

▼

SELECT *

2

FROM S1

3

WHERE sid = '60000001'

Data Output

Messages

Notifications

≡+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

SQL

	sid character (9) 🔒	sname character varying (30) 🔒	grade integer 🔒
1	60000001	Lily	2001

找到了，说明插入成功

7. 定义包括更新和插入约束的视图S1，尝试向视图插入记录("60000001,Lily,1997")，删除所有年级为1999的学生记录，讨论更新和插入约束带来的影响

先定义包括更新和插入约束的视图S1：

Query

Query History

1

2

3

4

5

▼

CREATE OR REPLACE VIEW S1 AS

SELECT sid, sname, grade

FROM STUDENTS

WHERE grade > 1998

WITH CHECK OPTION;

Data Output

Messages

Notifications

CREATE VIEW

Query returned successfully in 57 msec.

尝试插入记录("60000001,Lily,1997"):

Query

Query History

1

2

▼

INSERT INTO S1 (sid, sname, grade)

VALUES ('60000001', 'Lily', 1997);

Data Output

Messages

Notifications

ERROR: 键值"(sid)=(60000001)" 已经存在重复键违反唯一约束"students_pkey"

错误: 重复键违反唯一约束"students_pkey"

SQL state: 23505

Detail: 键值"(sid)=(60000001)" 已经存在

由于在上一题中STUDENTS已经存有sid = 60000001的信息，所以这时候插入失败
即使我们把那条信息删除，这条也无法插入成功，因为条件并不满足视图

Query	Query History	Scratchpad
<pre>1 INSERT INTO S1 (sid, sname, grade) 2 VALUES ('600000001', 'Lily', 1997);</pre>		
<p>Data Output Messages Notifications</p> <p>ERROR: 失败，行包含(600000001 , Lily, null, 1997). 新行违反了视图"s1"的检查选项</p> <p>错误: 新行违反了视图"s1"的检查选项</p> <p>SQL state: 44000</p> <p>Detail: 失败，行包含(600000001 , Lily, null, 1997).</p>		

删除所有年级为1999的学生记录

Query	Query History	Scratchpad
<pre>1 DELETE FROM S1 WHERE grade = 1999;</pre>		
<p>Data Output Messages Notifications</p> <p>ERROR: 键值对(sid)=(800008565)仍然是从表"choices"引用的. 在 "students" 上的更新或删除操作违反了在 "choices" 上的外键约束 "fk_choices_students"</p> <p>错误: 在 "students" 上的更新或删除操作违反了在 "choices" 上的外键约束 "fk_choices_students"</p> <p>SQL state: 23503</p> <p>Detail: 键值对(sid)=(800008565)仍然是从表"choices"引用的.</p>		

这时候会受到约束，把约束删除即可正常操作：

```
--删除级联
ALTER TABLE CHOICES
DROP CONSTRAINT fk_CHOICES_STUDENTS;
```

Query	Query History	Scratchpad
<pre>1 DELETE FROM S1 WHERE grade = 1999;</pre>		
<p>Data Output Messages Notifications</p> <p>DELETE 6664</p> <p>Query returned successfully in 96 msec.</p>		

8. 在视图VIEWS中将姓名为“uxjof”的学生的选课成绩都加上5分

```
1  v UPDATE VIEWS
2  SET score = score + 5
3  WHERE sname = 'uxjof';
```

Data Output Messages Notifications

ERROR: 不来自单表或单视图的视图不能自动更新,无法更新视图"views"

错误: 无法更新视图"views"

SQL state: 55000

Detail: 不来自单表或单视图的视图不能自动更新。

Hint: 启用对视图的更新操作, 需要提供INSTEAD OF UPDATE触发器或者一个无条件的 ON UPDATE DO INSTEAD 规则。

由于我们定义VIEWS来自多个表, 因此无法更新

要解决这个问题, 需要直接对基表执行更新操作, 而不是对视图执行。在这个例子中, 我更新了CHOICES表, 因为score列是CHOICES表的一部分。

Query Query History

```
1  v UPDATE CHOICES
2  SET score = score + 5
3  WHERE sid IN (
4      SELECT sid
5      FROM STUDENTS
6      WHERE sname = 'uxjof'
7  );
```

Data Output Messages Notifications

UPDATE 3

Query returned successfully in 107 msec.

Query Query History

```

1 SELECT *
2 FROM VIEWS
3 WHERE sname = 'uxjof'

```

Data Output Messages Notifications

	sname character varying (30)	no integer	cid character (5)	score integer
1	uxjof	506978093	10046	99
2	uxjof	541221076	10018	89
3	uxjof	567316431	10037	103

加分成功

9. 取消以上建立的所有视图

> 1.3 Sequences
> Tables
> Trigger Functions
> Types
▼ Views (1)
 > adjustedscores

Subscriptions
Login/Group Roles (15)
pg_checkpoint
pg_create_subscription
pg_database_owner

Query Query History

```

1 DROP VIEW VIEWC;
2 DROP VIEW VIEWS;
3 DROP VIEW S1;

```

Data Output Messages Notifications

DROP VIEW

Query returned successfully in 93 msec.

删除成功