4.13

1. 不需要，因为被授予v上选择权限的用户可以只访问视图v中引用关系r的部分
2. 需要，因为视图v引用关系r，一个对视图v的有效更新必须更新引用的关系r才能满足更新要求
3. 举例，定义视图history_instructors

```
create view history_instructors as
select *
from instructor
where dept_name = 'history';
```

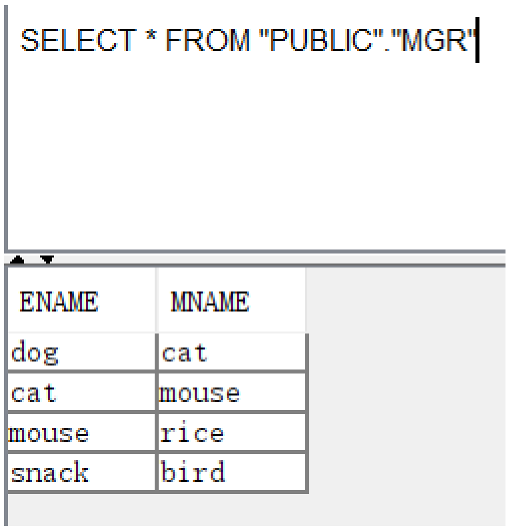
其中关系instructor(ID, name, dept_name, salary)，则向history_instructors插入('1', 'White', 'Comp. Sci.', 10000)，视图history_instructors在select时因不满足where条件而不会出现该元组。

5.1

通过mgr查询表内dog的manager，以及manager的manager，以此类推，直到不再有manager——查询的就是这样一个关于dog的manager链条

实验验证如下：

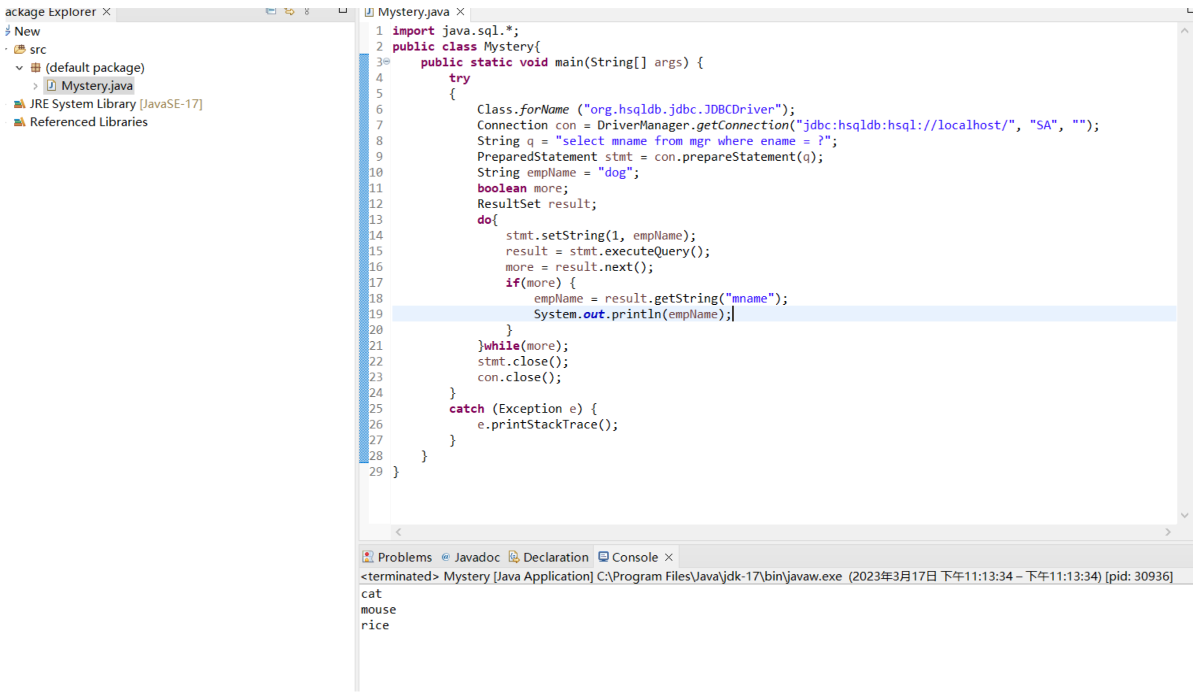
mgr表内数据为：



```
SELECT * FROM "PUBLIC"."MGR"
```

ENAME	MNAME
dog	cat
cat	mouse
mouse	rice
snack	bird

运行程序后输出为：



```java
import java.sql.*;
public class Mystery{
    public static void main(String[] args) {
        try
        {
            Class.forName ("org.hsqldb.jdbc.JDBCDriver");
            Connection con = DriverManager.getConnection("jdbc:hsqldb:hsql://localhost/", "SA", "");
            String q = "select mname from mgr where ename = ?";
            PreparedStatement stmt = con.prepareStatement(q);
            String empName = "dog";
            boolean more;
            ResultSet result;
            do{
                stmt.setString(1, empName);
                result = stmt.executeQuery();
                more = result.next();
                if(more) {
                    empName = result.getString("mname");
                    System.out.println(empName);
                }
            }while(more);
            stmt.close();
            con.close();
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

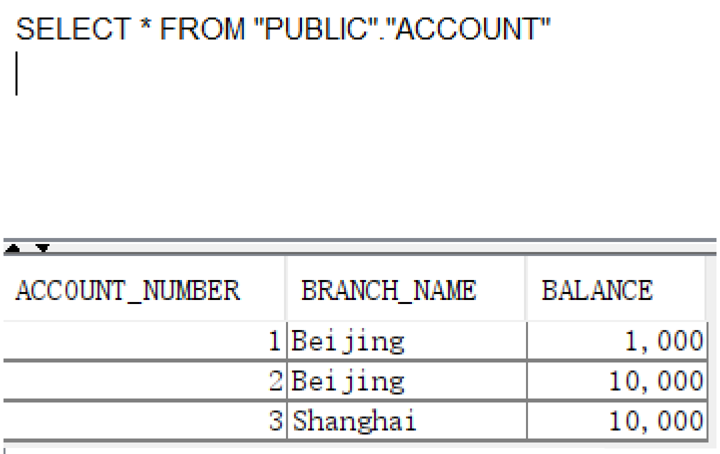
```
<terminated> Mystery [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe  (2023年3月17日 下午11:13:34 – 下午11:13:34) [pid: 30936]
cat
mouse
rice
```

输出满足期望。

5.7

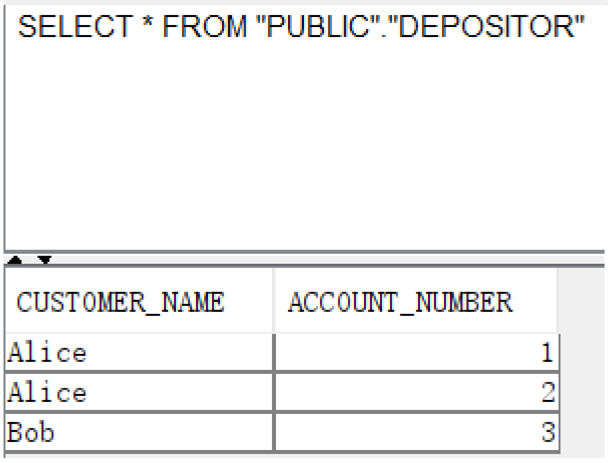
```sql
create trigger delete_check after delete on account
referencing old row as orow
for each row
delete from depositor
where depositor.customer_name not in (
    select customer_name
    from depositor
    where depositor.account_number <> orow.account_number
)
```

验证：删除前account和depositor表内容如下：



```
SELECT * FROM "PUBLIC"."ACCOUNT"
```

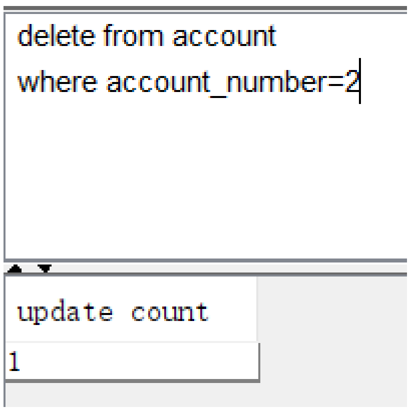
ACCOUNT_NUMBER	BRANCH_NAME	BALANCE
1	Beijing	1,000
2	Beijing	10,000
3	Shanghai	10,000



```
SELECT * FROM "PUBLIC"."DEPOSITOR"
```

CUSTOMER_NAME	ACCOUNT_NUMBER
Alice	1
Alice	2
Bob	3

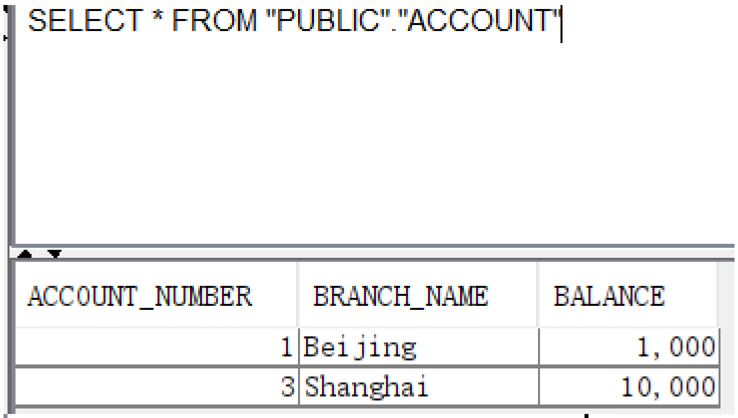
现在我们删除ACCOUNT_NUMBER为2的账户：



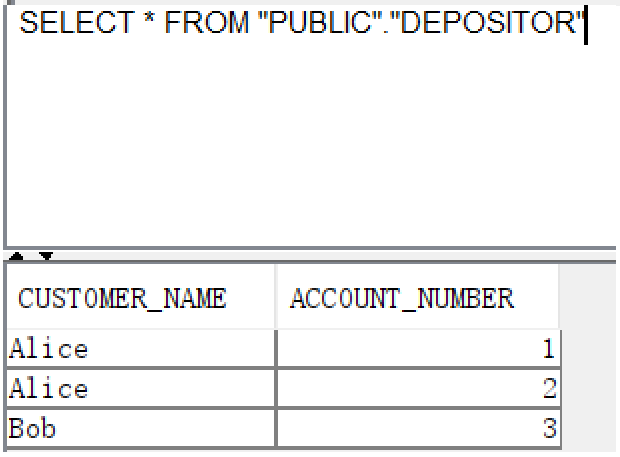
```
delete from account
where account_number=2
```

update count
1

删除后两张表如下，由于depositor表内Alice有1,2两个账户，则Alice不被删除：

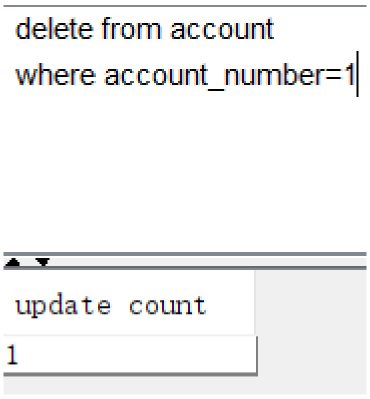

```
SELECT * FROM "PUBLIC"."ACCOUNT"
```

ACCOUNT_NUMBER	BRANCH_NAME	BALANCE
1	Beijing	1,000
3	Shanghai	10,000


```
SELECT * FROM "PUBLIC"."DEPOSITOR"
```

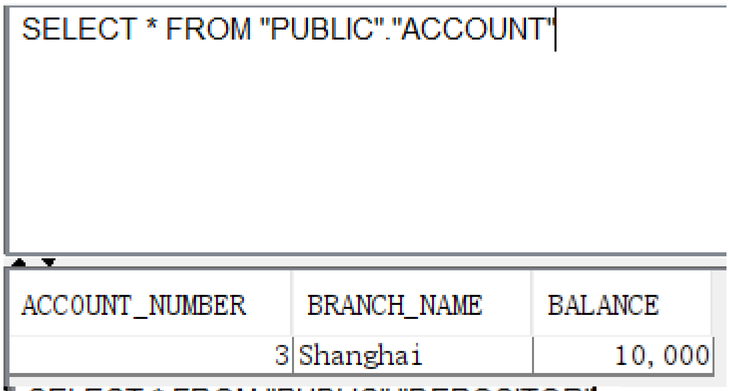
CUSTOMER_NAME	ACCOUNT_NUMBER
Alice	1
Alice	2
Bob	3

再删除ACCOUNT_NUMBER为1的账户：


```
delete from account
where account_number=1
```

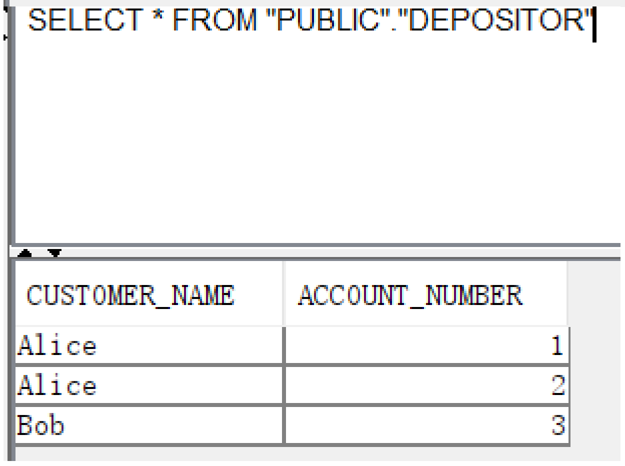
update count
1

删除后两张表如下，由于depositor表内Alice有1,2两个账户，则Alice不被删除：



```
SELECT * FROM "PUBLIC"."ACCOUNT"
```

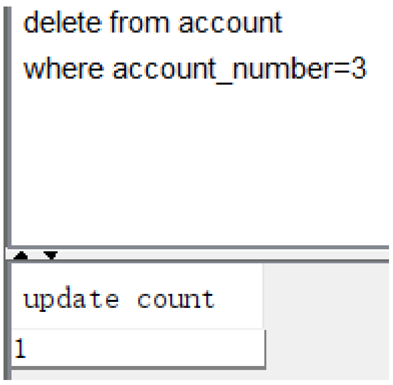
ACCOUNT_NUMBER	BRANCH_NAME	BALANCE
3	Shanghai	10,000



```
SELECT * FROM "PUBLIC"."DEPOSITOR"
```

CUSTOMER_NAME	ACCOUNT_NUMBER
Alice	1
Alice	2
Bob	3

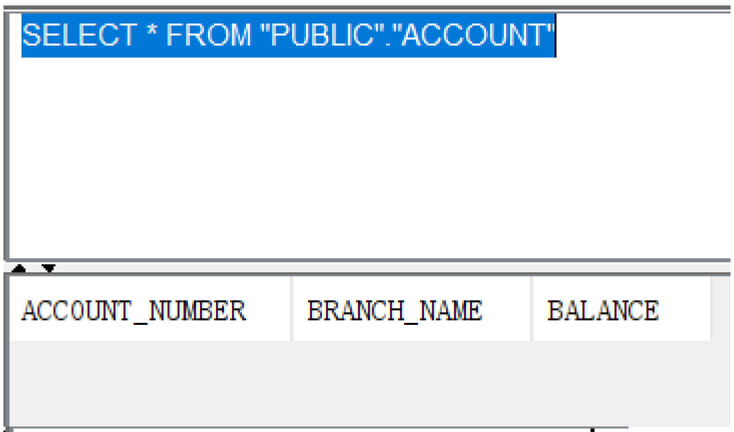
现在我们删除ACCOUNT_NUMBER为3的账户：



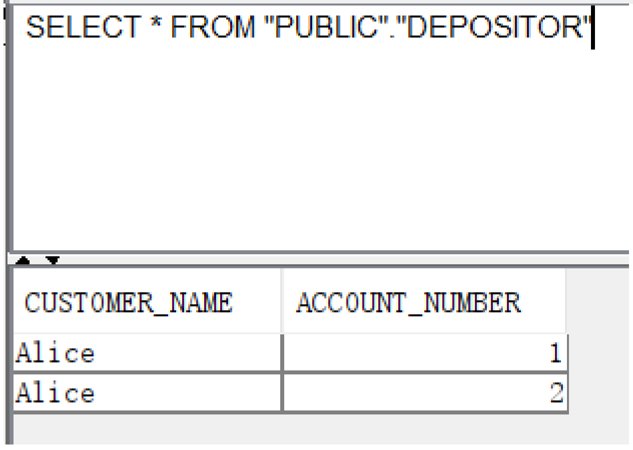
```
delete from account
where account_number=3
```

update count
1

删除后两张表如下，由于depositor表内Bob只有一个账户，则Bob不被删除：


```
SELECT * FROM "PUBLIC"."ACCOUNT"
```

ACCOUNT_NUMBER	BRANCH_NAME	BALANCE


```
SELECT * FROM "PUBLIC"."DEPOSITOR"
```

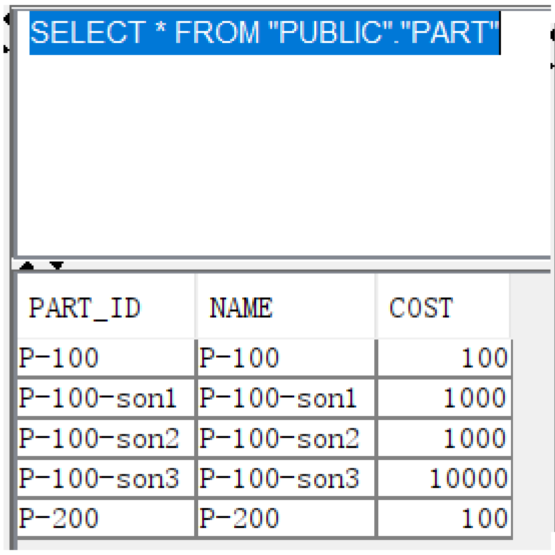
CUSTOMER_NAME	ACCOUNT_NUMBER
Alice	1
Alice	2

可以看到查询结果正确

5.16

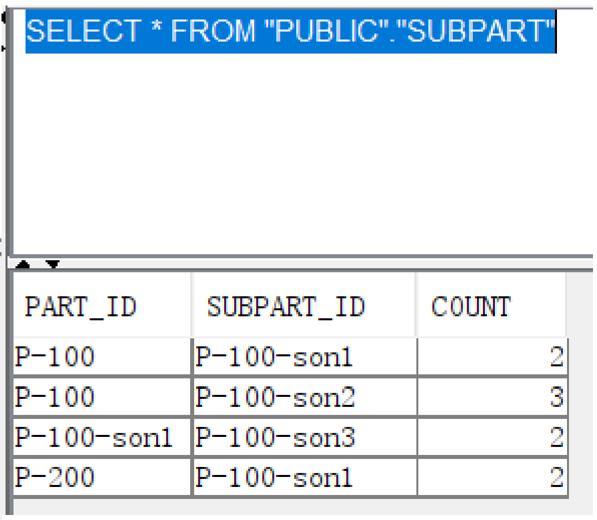
```sql
with recursive rec_subpart(part_id, subpart_id) as (
        select part_id, subpart_id
        from subpart
    union
        select rec_subpart.part_id, subpart.subpart_id
        from rec_subpart, subpart
        where rec_subpart.subpart_id = subpart.part_id
)

select part.name
from part, rec_subpart
where rec_subpart.part_id = 'P-100' and part.part_id = rec_subpart.subpart_id
```

我们的part和subpart表为：



```
SELECT * FROM "PUBLIC"."PART"
```

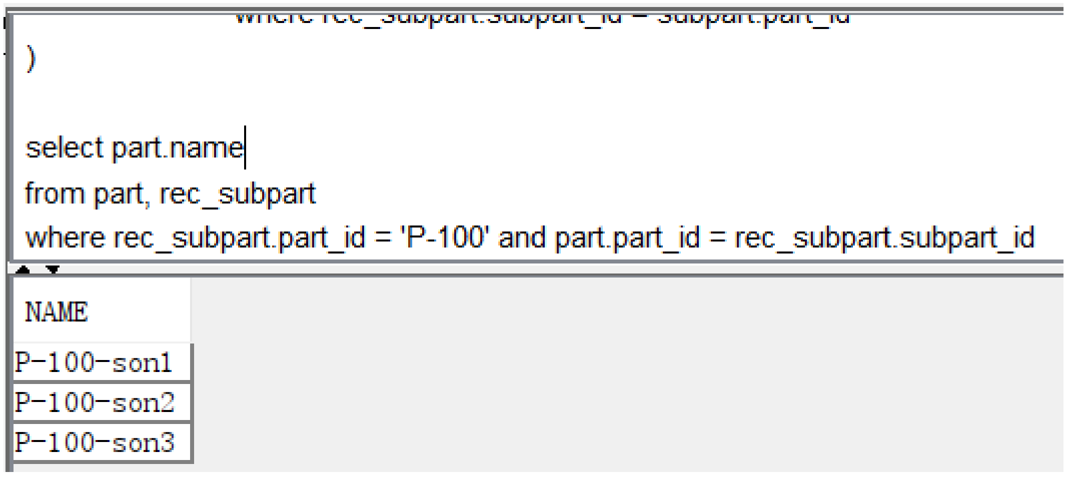
PART_ID	NAME	COST
P-100	P-100	100
P-100-son1	P-100-son1	1000
P-100-son2	P-100-son2	1000
P-100-son3	P-100-son3	10000
P-200	P-200	100



```
SELECT * FROM "PUBLIC"."SUBPART"
```

PART_ID	SUBPART_ID	COUNT
P-100	P-100-son1	2
P-100	P-100-son2	3
P-100-son1	P-100-son3	2
P-200	P-100-son1	2

查询结果为：



```
)

select part.name
from part, rec_subpart
where rec_subpart.part_id = 'P-100' and part.part_id = rec_subpart.subpart_id
```

NAME
P-100-son1
P-100-son2
P-100-son3

可以看到查询结果正确