

数据库应用上机实验报告

学号：15130110047

姓名：郑健

班级：1513011

西安电子科技大学软件学院

1. **实验目的**

熟悉oracle中约束和触发器，分别创建Oracle的6种约束，并设计实验用例来验证该约束；设计行级触发器、语句级触发器各1个，体会触发器的执行顺序对业务逻辑的影响，以及对DML语句性能的影响。

1. **实验环境**

操作系统：Windows7

开发工具： iSqlplus Oracle 10g

1. **实验内容与结果**

**1、约束**

1.创建约束NOT NULL

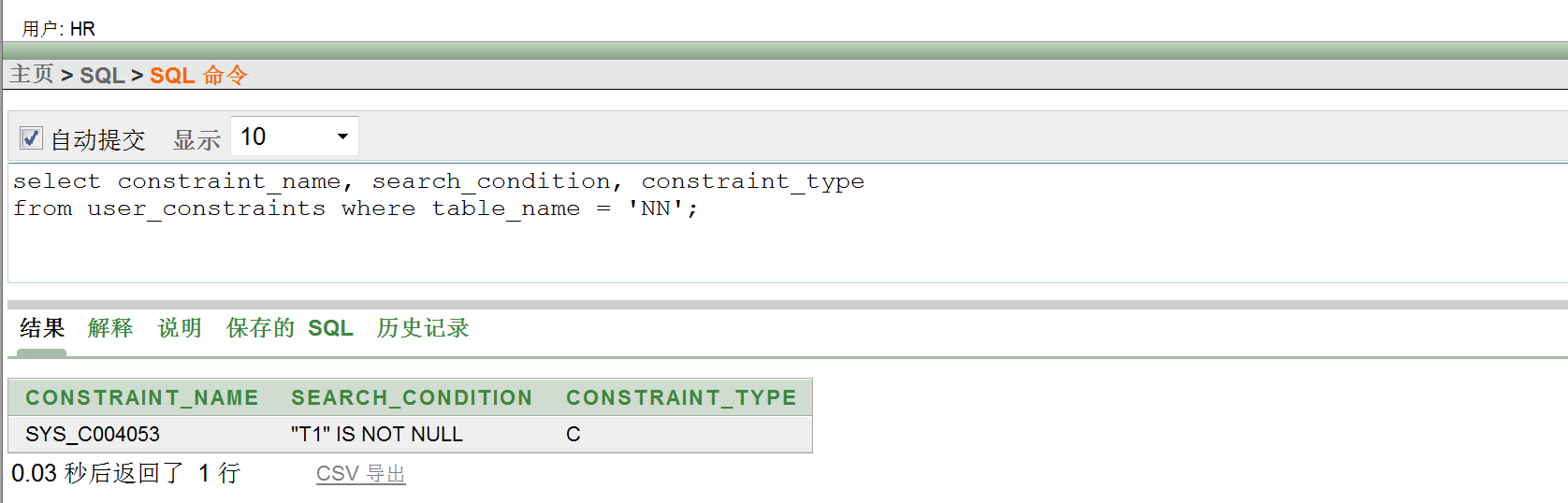
create table NN (

t1 char(10) not null,

t2 number(2) );

select constraint\_name, search\_condition, constraint\_type

from user\_constraints where table\_name = 'NN';



insert into NN values ( '123', 1 );

insert into NN values ( null, 2);

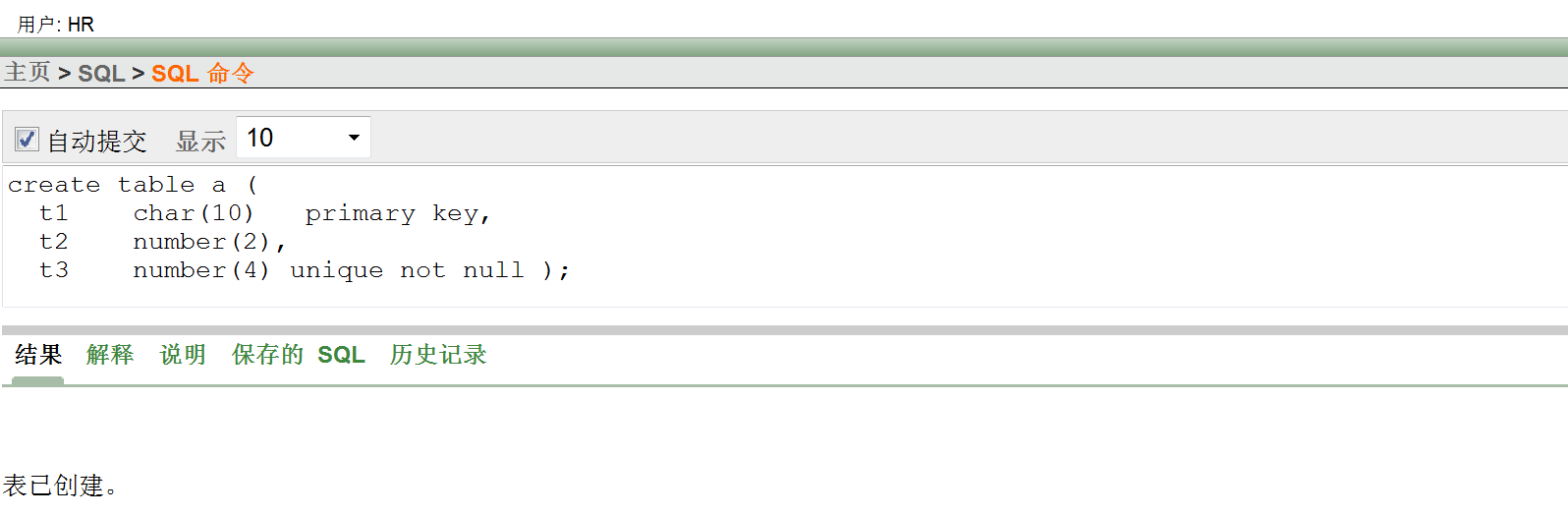
insert into NN values ( '234', null);

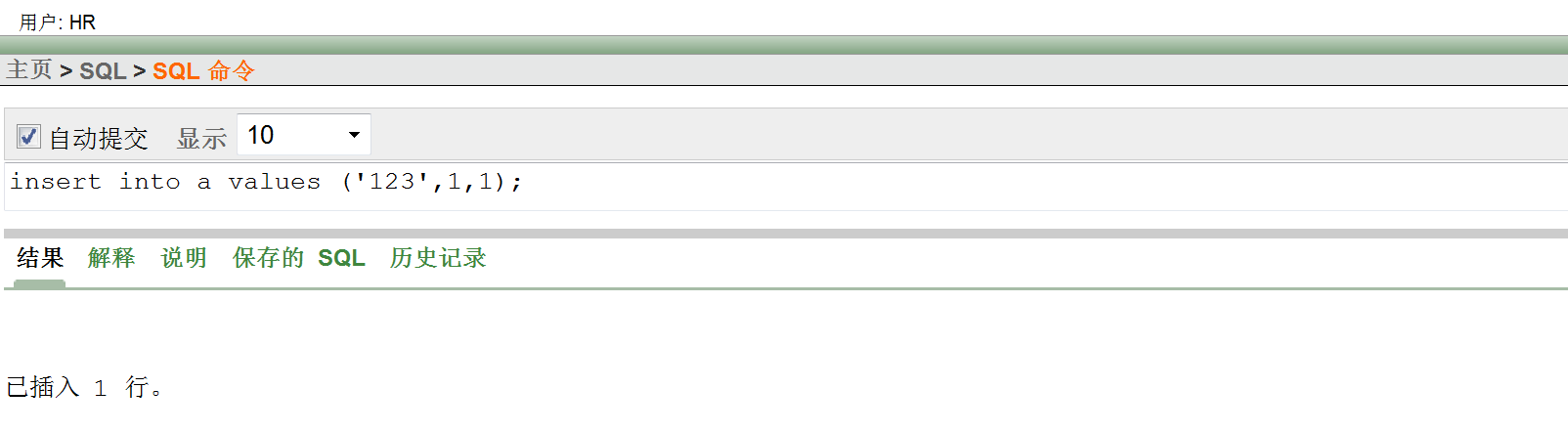
2.创建约束PRIMARY KEY和UNIQOE

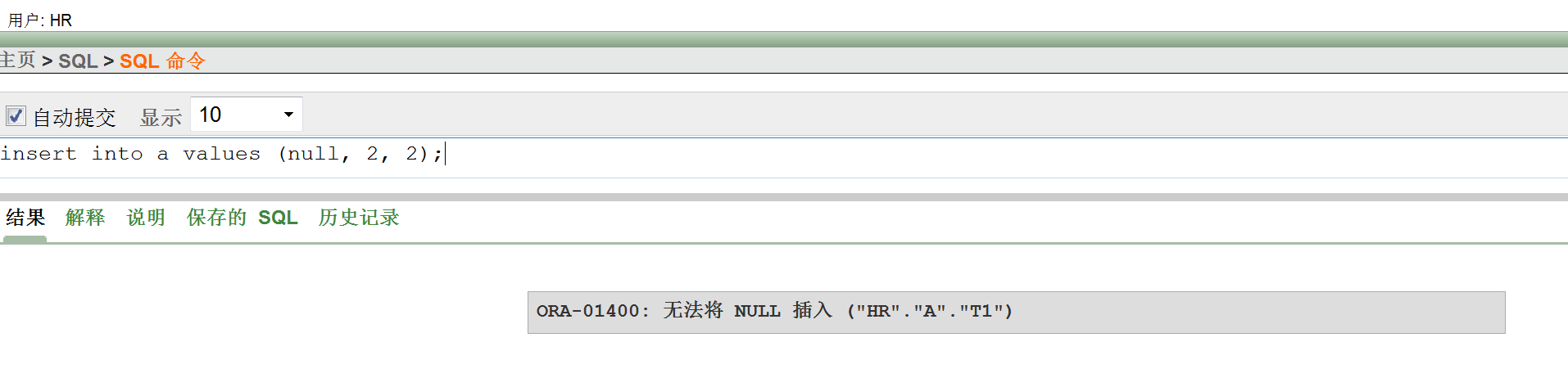
create table a (

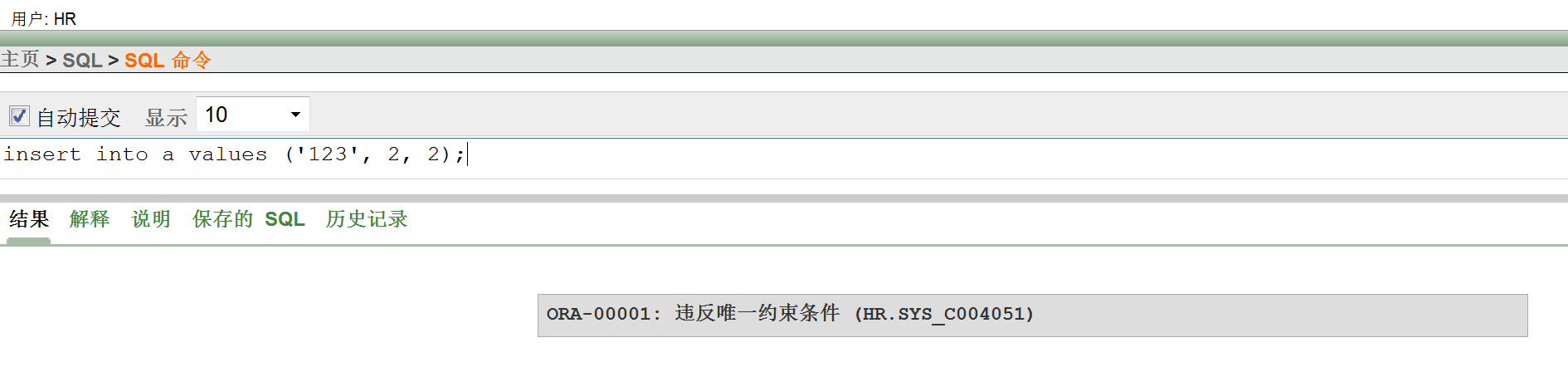
t1 char(10) primary key,

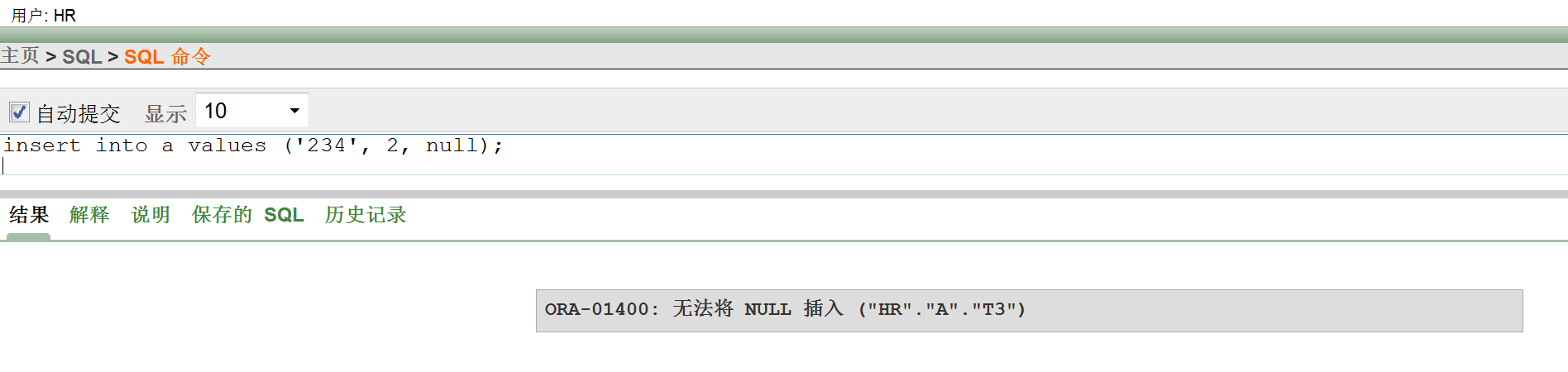
t2 number(2),

 t3 number(4) unique not null );

insert into a values ('123', 1, 1);

insert into a values (null, 2, 2);

insert into a values ('123', 2, 2);

insert into a values ('234', 2, null);

3.创建约束FOREIGN KEY

create table holo1(

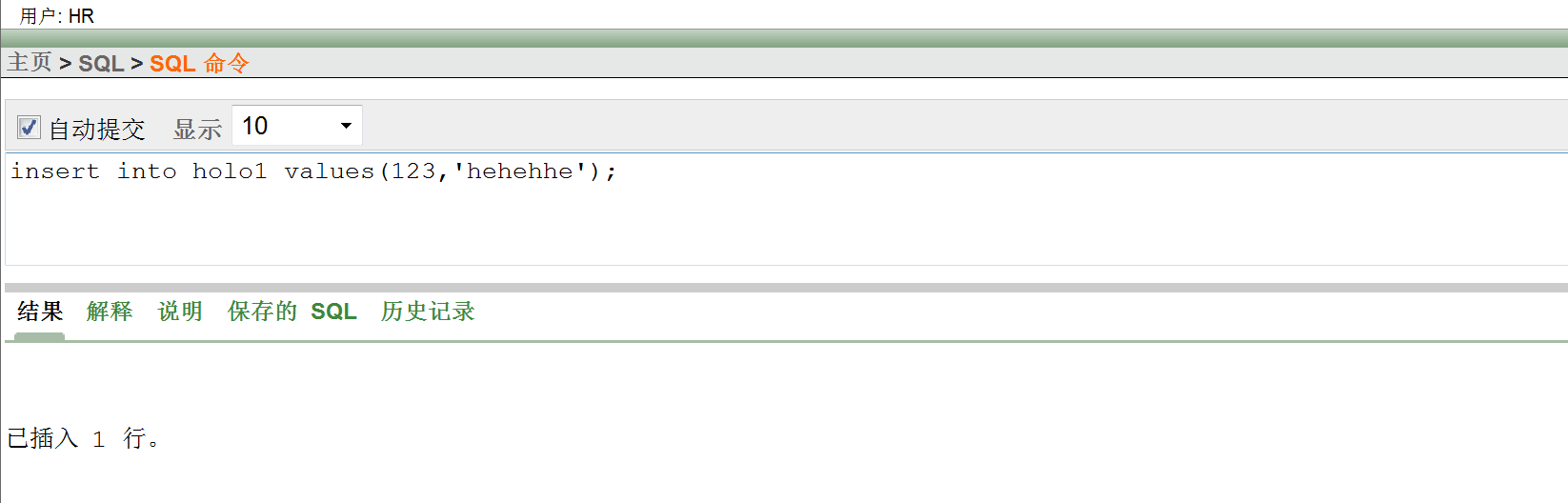
holo1\_id number,

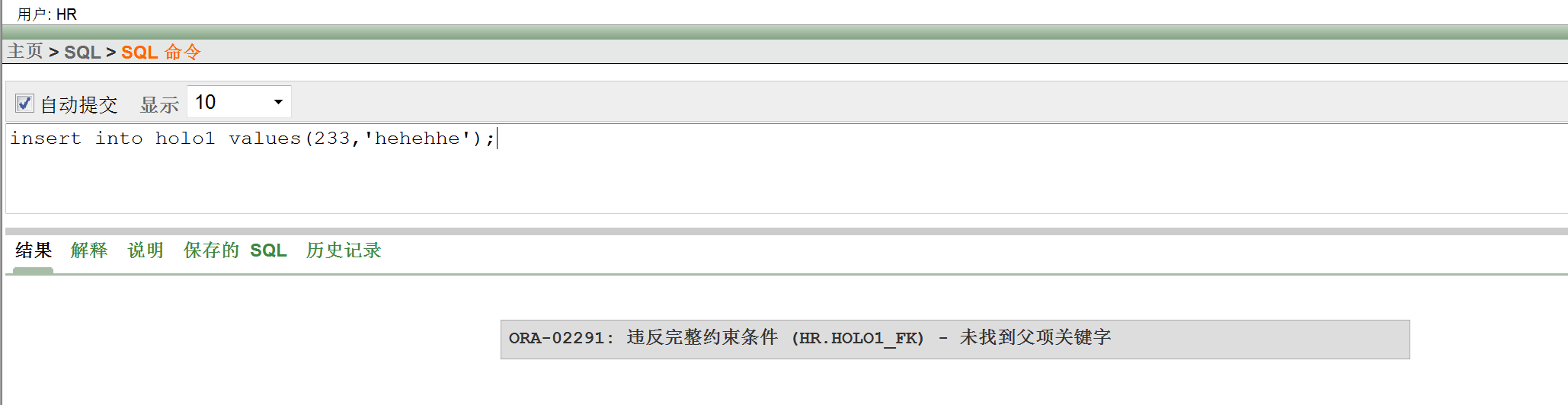
holo1\_name VARCHAR(10),

constraint holo1\_fk foreign key(holo1\_id)

references holo(holo\_id)

);

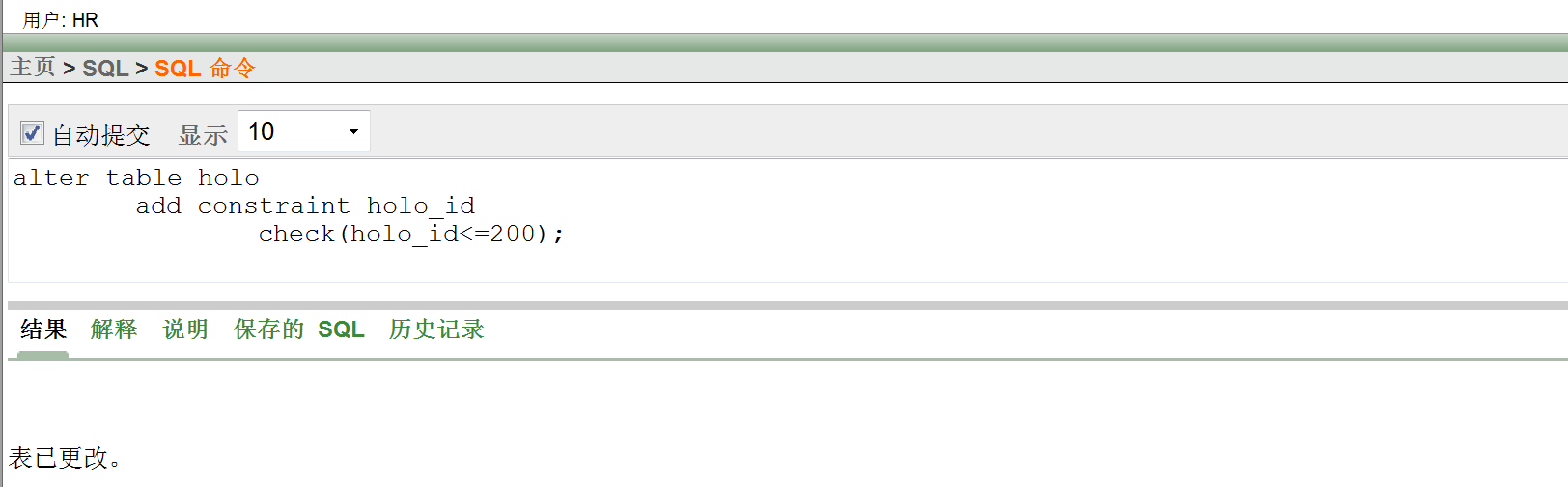
insert into holo1 values(123,'hehehhe');

insert into holo1 values(233,'hehehhe');

4.创建约束CHECK

alter table holo

add constraint holo\_id

 check(holo\_id<=200);

insert into holo values(233,'mmp');

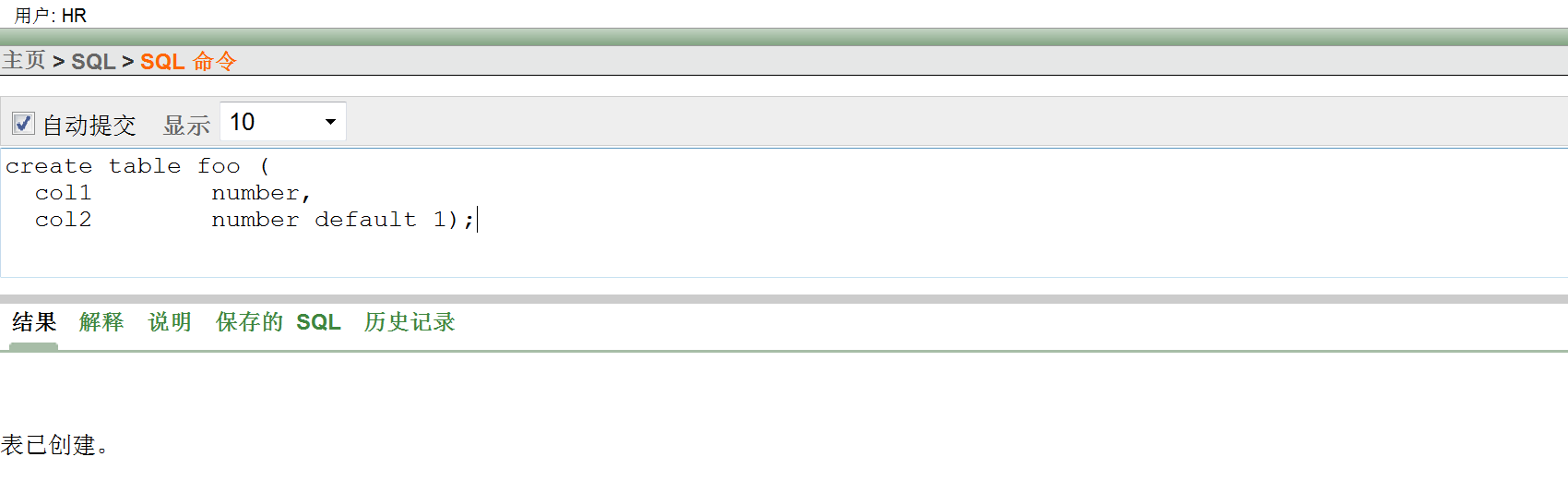


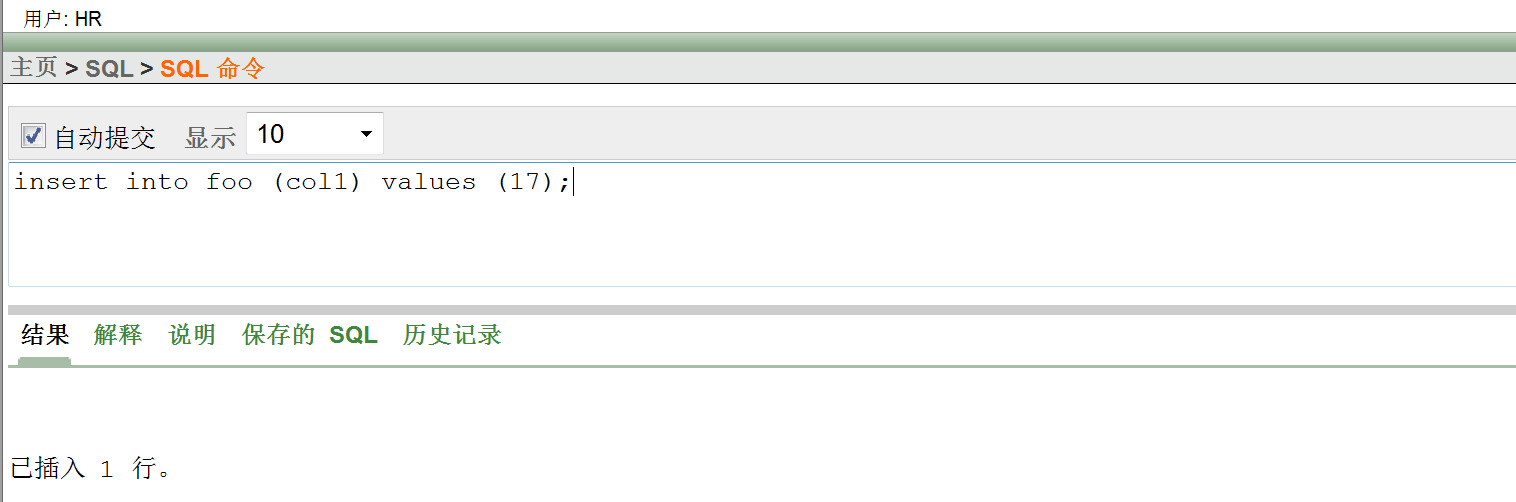
5.创建约束DEFAULT

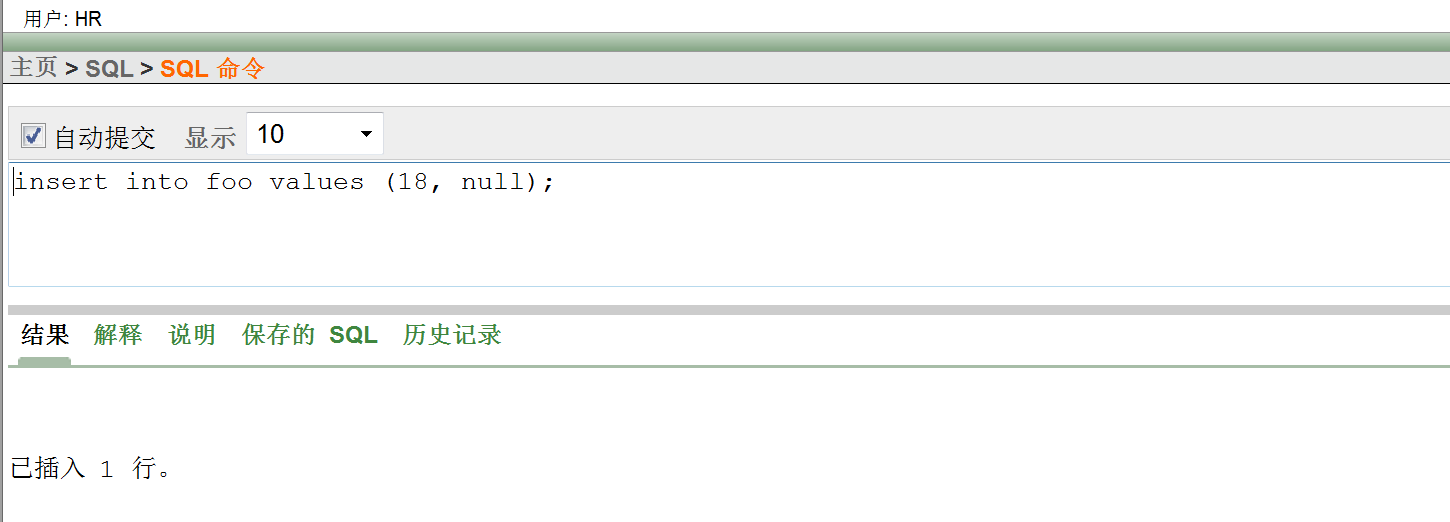
create table foo (

col1 number,

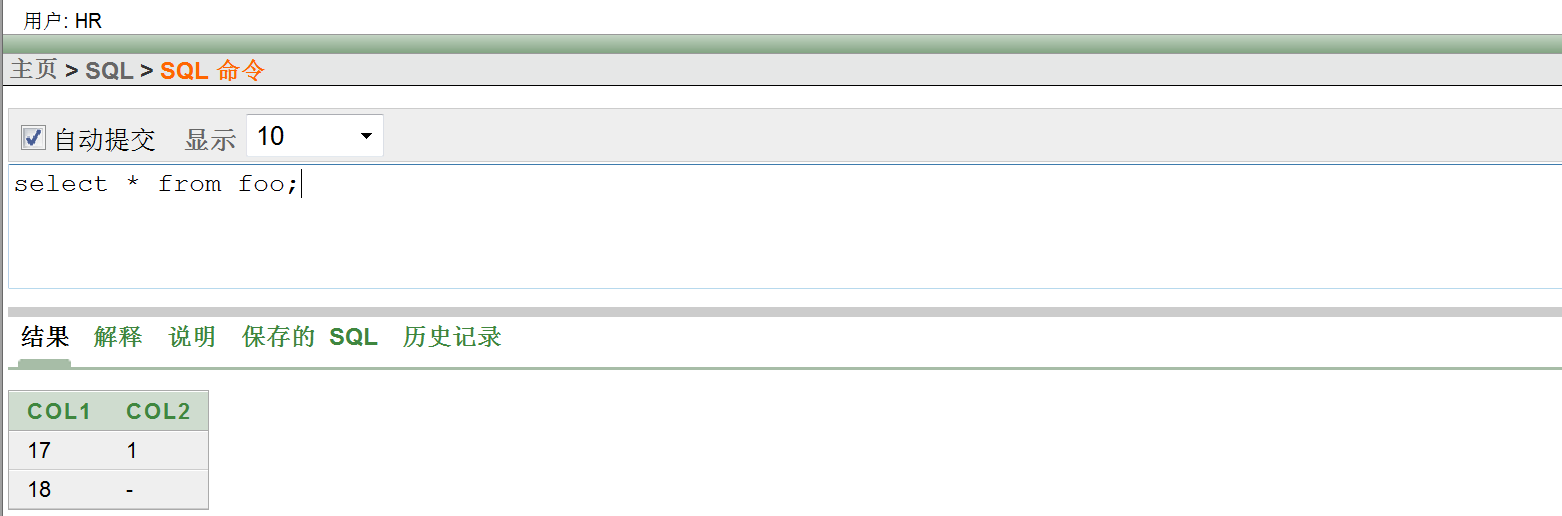
col2 number default 1);



insert into foo (col1) values (17);

insert into foo values (18, null);

select \* from foo;



分析：实验通过创建6种约束让我体会到各自的功能，认识到约束是将大量业务规则的声明作为数据定义的组成部分，约束能实现简单的数据验证逻辑，更复杂的则需要用触发器实现。

1. **触发器**
2. 语句级触发器

//创建语句级触发器，用于判定是否在正确的时间插入数据到time表中

（若系统时间为星期六、星期天或不在8:00-18:00这一时间段，则插入数据到time表失败）

CREATE OR REPLACE TRIGGER secure\_holo

BEFORE INSERT ON holo

BEGIN

IF ( TO\_CHAR(SYSDATE,'DY') in ('SAT','SUN'))

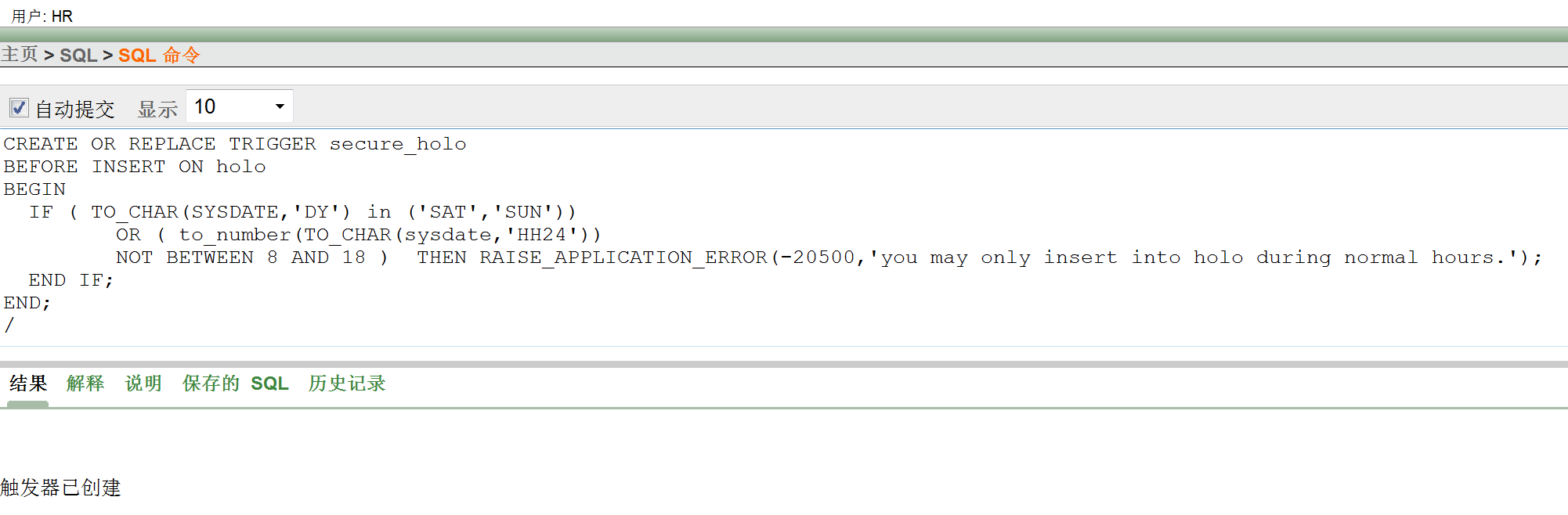
OR ( to\_number(TO\_CHAR(sysdate,'HH24'))

NOT BETWEEN 8 AND 18 ) THEN RAISE\_APPLICATION\_ERROR(-20500,'you may only insert into holo during normal hours.');

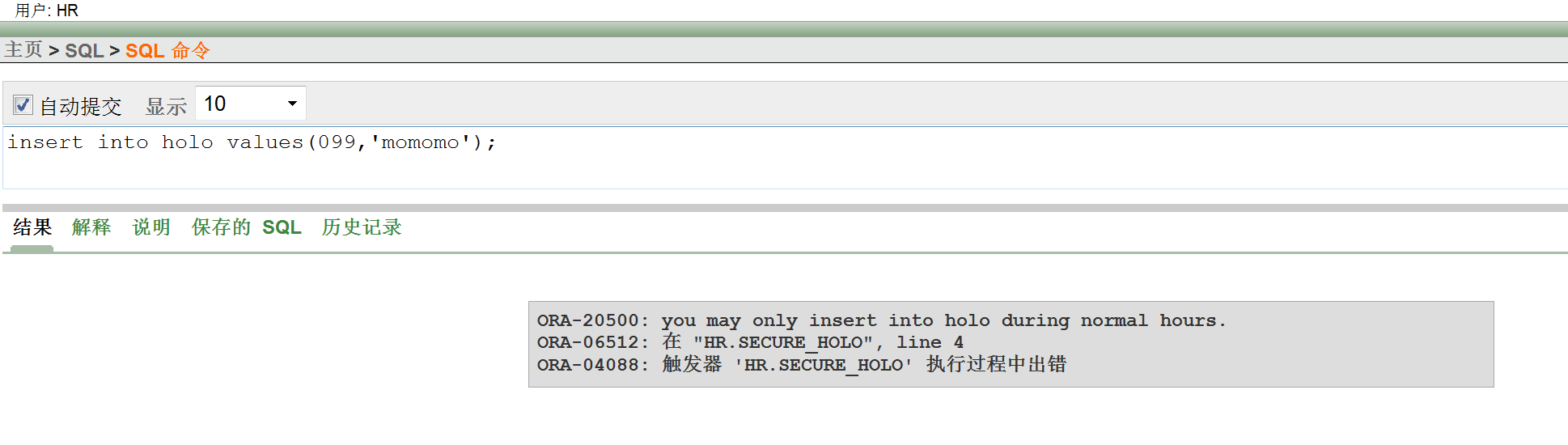
END IF;

END;

/



insert into holo values(099,'momomo');



//当系统时间设为星期三21:15，尝试插入数据到time表，结果失败

分析：给表设置语句级触发器后，触发器只执行一次，本示例中插入数据到time表时，

依据触发器的限制得出插入结果的成功与否。

2.行级触发器

CREATE OR REPLACE TRIGGER holo1\_cascade\_update

AFTER UPDATE OR DELETE ON holo

FOR EACH ROW

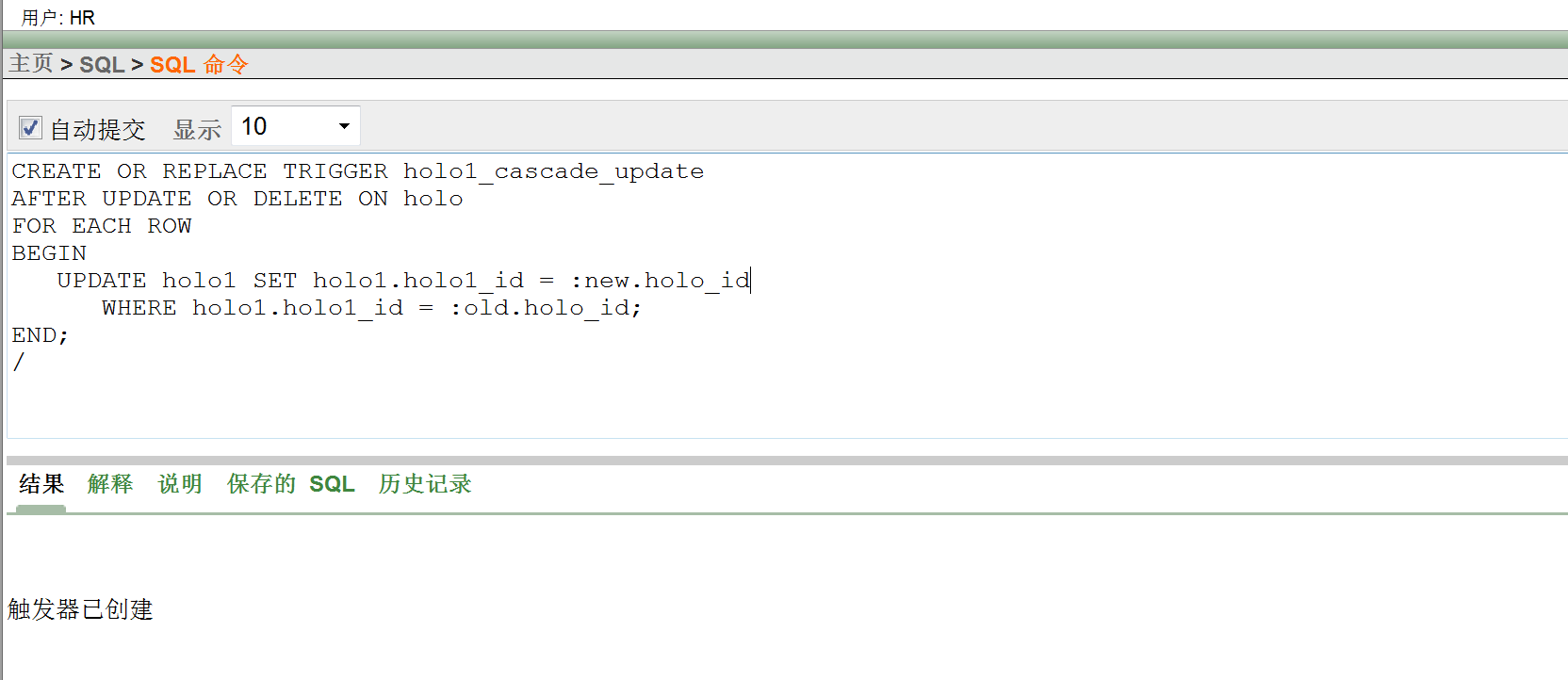
BEGIN

UPDATE holo1 SET holo1.holo1\_id = :new.holo\_id

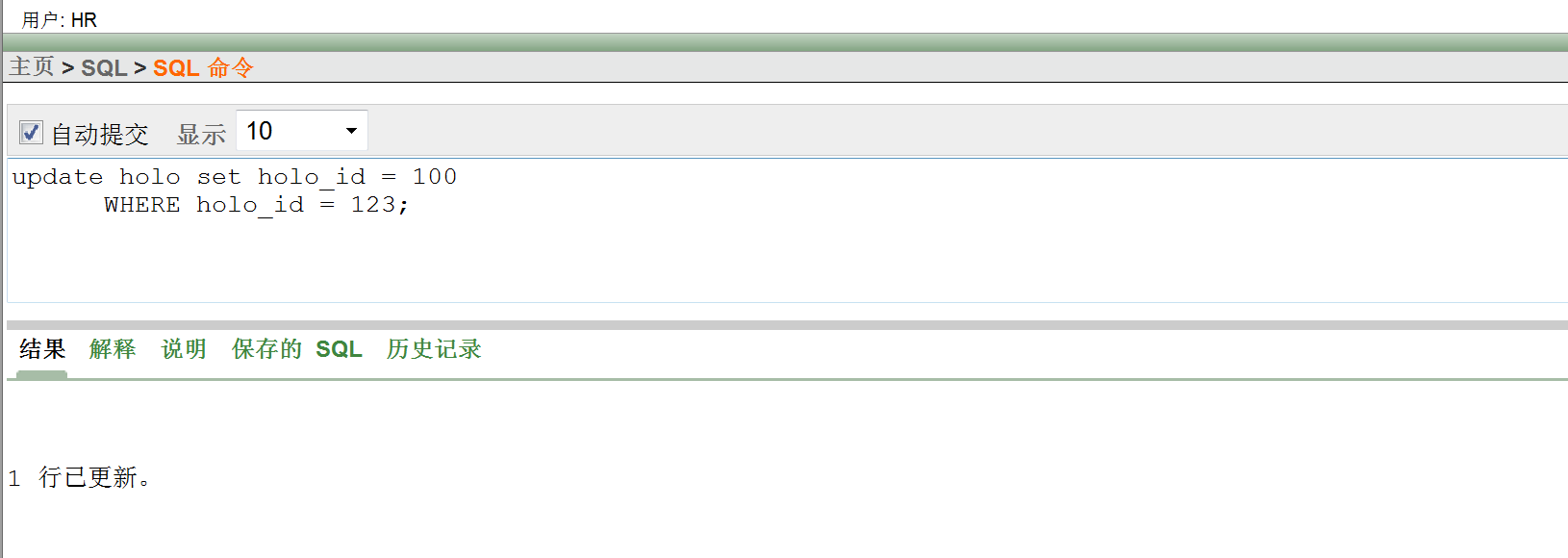
WHERE holo1.holo1\_id = :old.holo\_id;

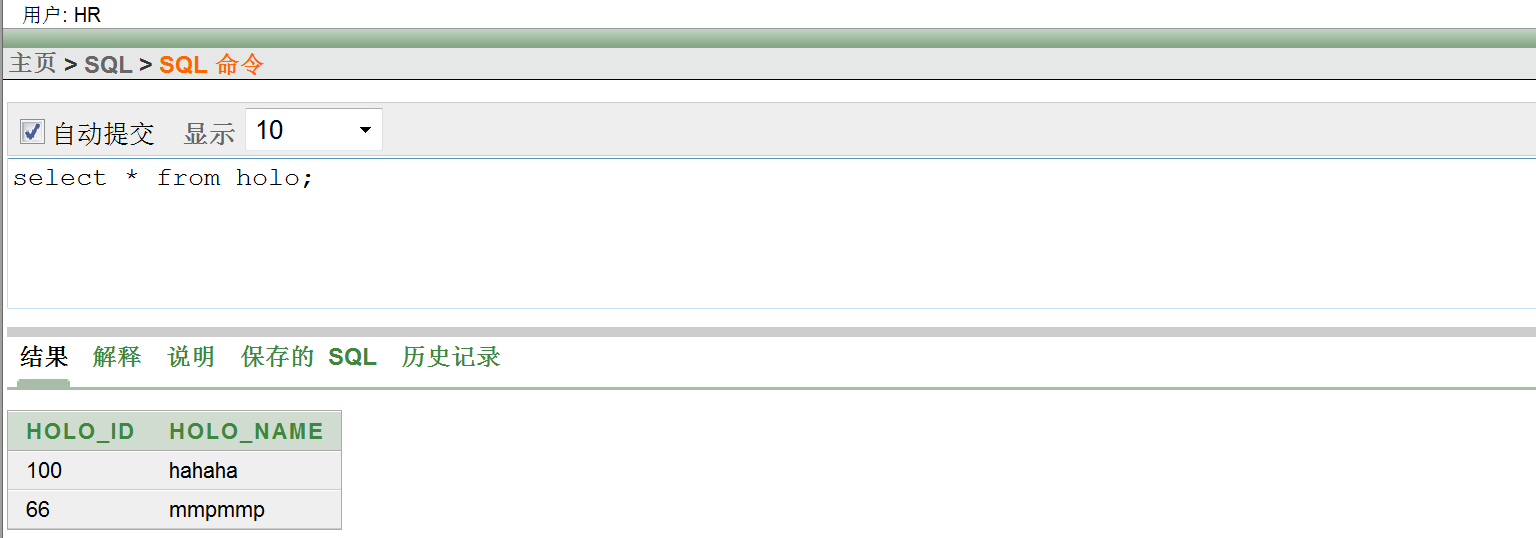
END;

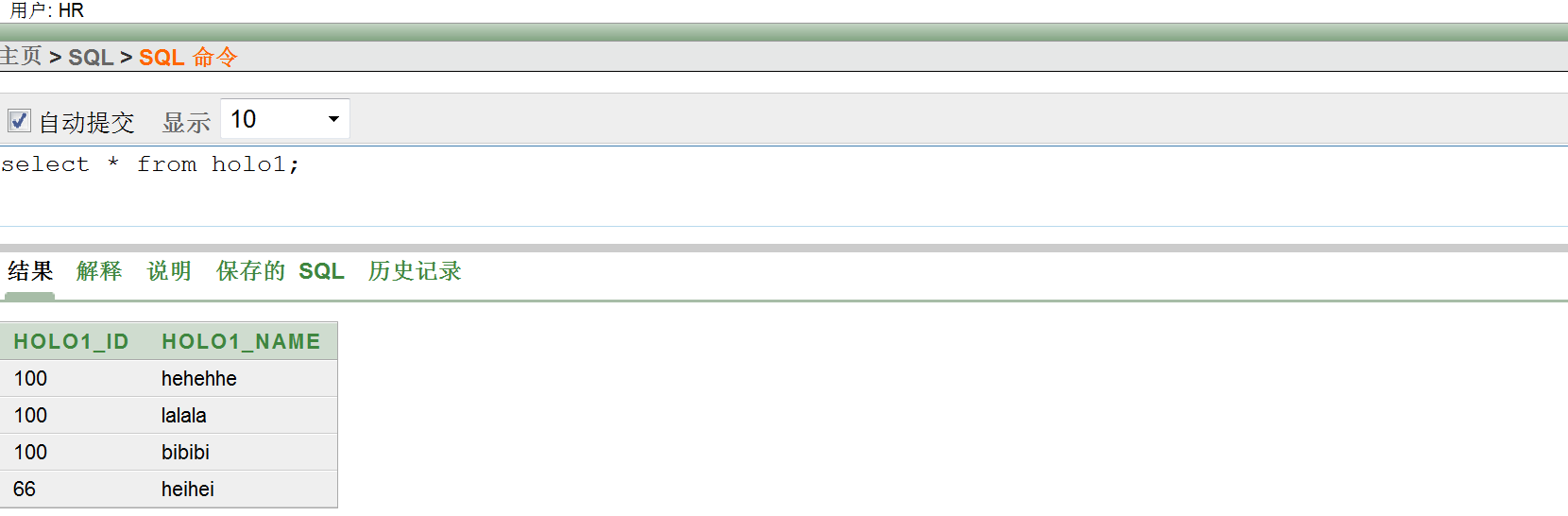
/



update holo set holo\_id = 100

 WHERE holo\_id = 123;

select \* from holo;

select \* from holo1;

**实验分析与总结：**

本次实验我学会了在Oracle中设置用户来访问数据库，对用户赋予不同的资源使得用户具有操作数据库的不同权限，学会了创建Oracle的6种约束，并设计实验用例来验证该约束，尤其对外键约束有了深刻的印象。同时对触发器有了一定认识，通过创建语句级触发器和行级触发器，我体会到触发器的执行顺序对业务逻辑的影响，以及对DML语句性能的影响。其中语句级触发器激发时该语句只执行一次，适用于一次更新或删除等操作；而行级触发器是按触发语句所处理的行激发的，该语句每处理一行，则触发器被执行一次，适用多次更新或删除等操作，并且认识到在BEFORE触发器中修改“:new”,不能修改“:old”,在AFTER触发器中不能修改“:new”。