实验二报告

一、 观察并回答问题

1. 关于视图

（1） sakila.mwb 模型图中共有几个 View？

答；7 个。分别是 actor_info,customer_list,film_list,nicer_but_slower_film_list, sales_by_film_category,sales_by_store,staff_list

（2） 分析以下 3 个视图，回答以下问题：

视图名	关联表	作用
actor_info	film,actor,film_actor,film_category,category	选出所有演员，得到其姓名及出演的不同电影
film_list	actor,category,film_category,film,film_actor	所有电影的基本信息，如出演者等
sales_by_store	payment,rental,inventory,store,address,city,country, staff	商店的基本信息，所在地，管理者以及总销售额

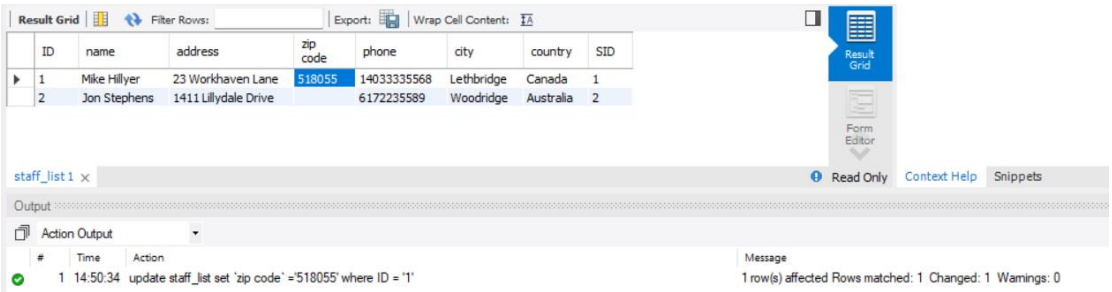
（3） 分别执行以下 2 句 SQL 语句：

```
update staff_list set `zip code` = '518055' where ID = '1';
```

```
update film_list set price = 1.99 where FID = '1';
```

截图执行结果，并分析一下视图在什么情况下可以进行 update 操作，什么情况下不能？

```
update staff_list set `zip code` = '518055' where ID = '1';
```



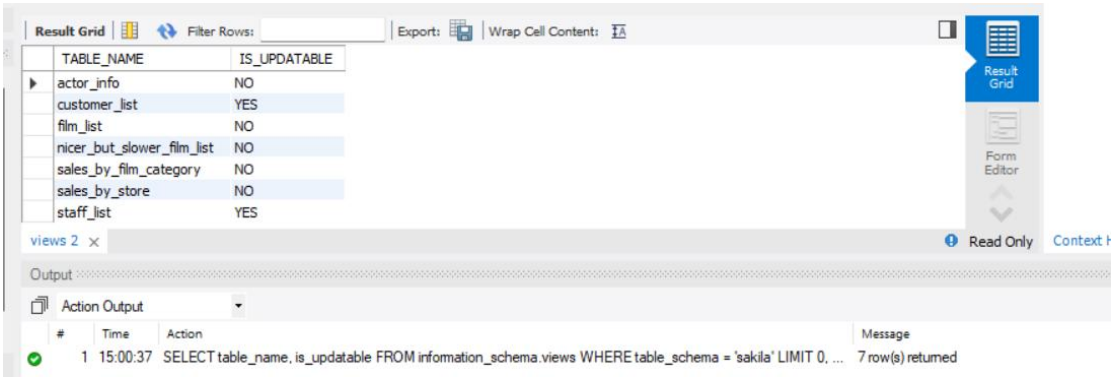
```
update film_list set price = 1.99 where FID = '1';
```

若视图的定义中有 GROUP BY 子句或聚集函数时，则此视图不允许更新
Film_list 中有 GROUP BY 定义语句

```
GROUP BY `film`.`film_id` , `category`.`name`
```

（4） 执行以下命令查询 sakila 数据库中的视图是否可更新，截图执行结果：

```
SELECT table_name, is_updatable FROM information_schema.views
WHERE table_schema = 'sakila';
```

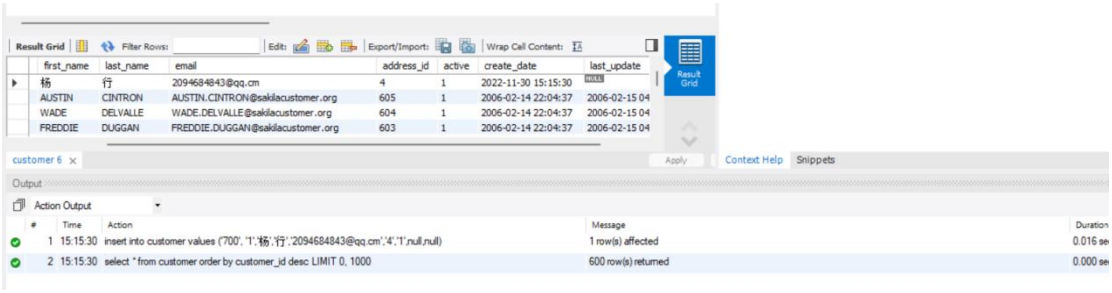


2. 关于触发器

（1） 触发器 customer_create_date 建在哪个表上？这个触发器实现什么功能？
答：建在 customer 表上；功能是在添加顾客信息时，记录创建时间 create_date.

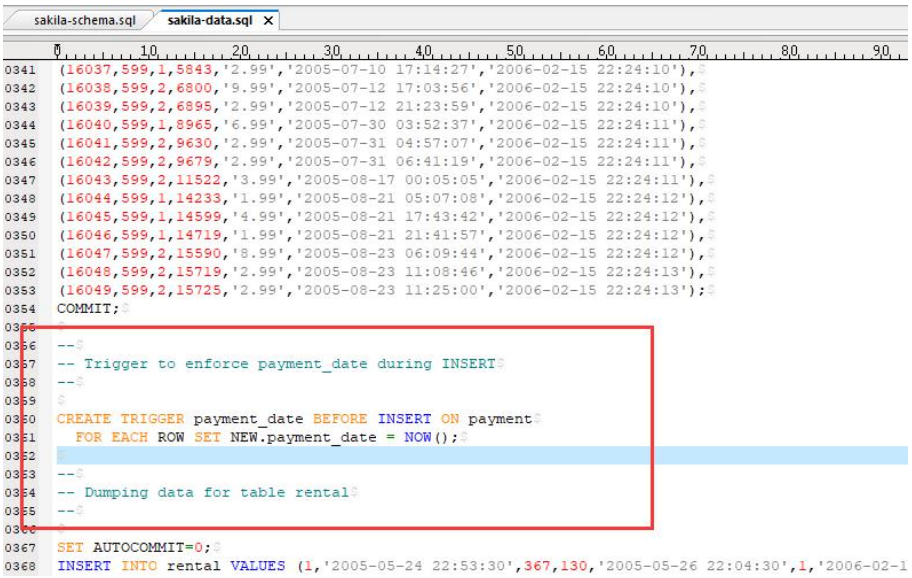
（2） 在这个表上新增一条数据，验证一下触发器是否生效。（截图语句和执行结果）

```
insert into customer
values ('700', '1','杨','行','2094684843@qq.cm','4','1',null,null);
select * from customer
order by customer_id desc;
```



可以看到由于触发器的原因，只有 create_date 得到时间值。

（3） 我们可以看到 sakila-schema.sql 里的语句是用于创建数据库的结构，包括表、视图、触发器等，而 sakila-data.sql 主要是用于往表写入数据。但 sakila-data.sql 里有这样一个建立触发器 payment_date 的语句，这个触发器是否可以移到 sakila-schema.sql 里去执行？为什么？



答：不可以，该触发器的功能为增加 payment 表中的数据时将 payment_date 修改为当前系统时间。而在创建数据库时的两个 SQL 脚本 sakila-schema.sql、sakila-data.sql 的执行存在先后顺序，先执行 sakila-schema.sql，后执行 sakila-data.sql，若将该触发器移到 sakila-schema.sql 中，则在之后执行 sakila-data.sql 给 payment 表写入数据时，所有的 payment_date 会被覆盖为写入时的系统时间，原始信息缺失。

3. 关于约束

（1） store 表上建了哪几种约束？这些约束分别实现什么功能？（至少写 3 个）

答：建了 5 种约束。

约束类型	功能
主键	确定该数据的唯一性
非空约束	来约束该列的取值不能为空，确保不存在无效值
唯一约束	保证所有记录中字段的值不能重复出现

（2） 图中第 343 行的 ON DELETE RESTRICT 和 ON UPDATE CASCADE 是什么意思？

答：ON DELETE RESTRICT 表示对删除存在约束当在父表(即外键的来源表)中删除对应记录时，

首先检查该记录是否有对应外键，如果有则不允许删除。

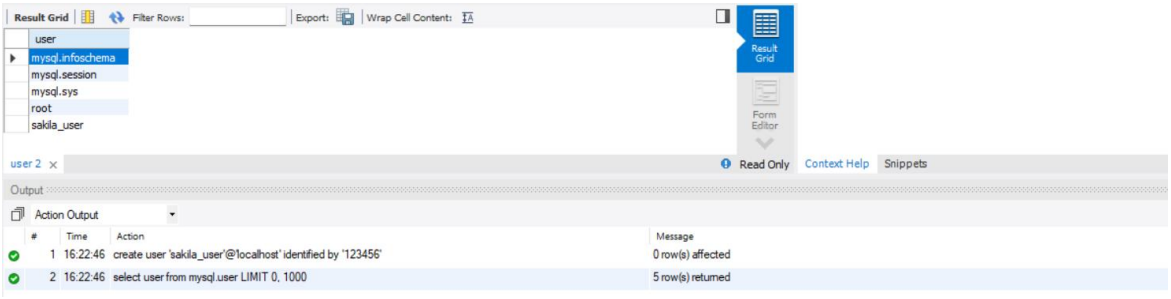
ON UPDATE CASCADE 表示级联更新当在父表(即外键的来源表)中更新对应记录时，首先检查该记录是否有对应外键，如果有则也更新外键在子表(即包含外键的表)中的记录。

二、 创建新用户并分配权限

（截图语句和执行结果）

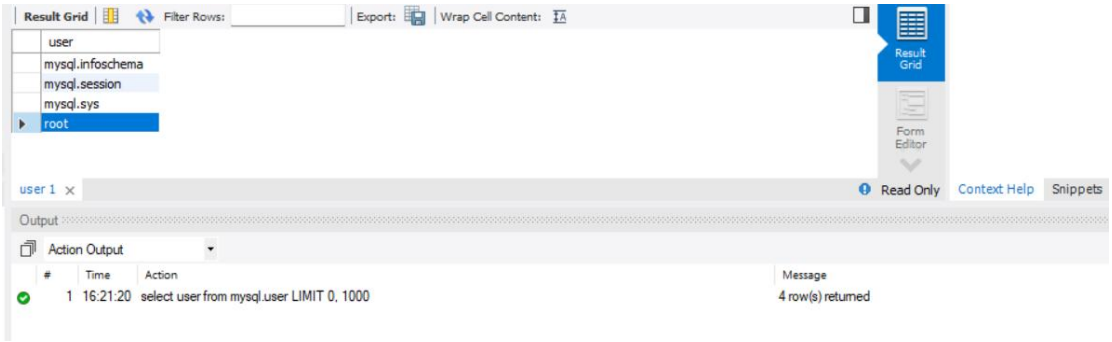
（1） 执行命令新建 sakila_user 用户（密码 123456）；

```
create user 'sakila_user'@'localhost' identified by '123456';
select user from mysql.user;
```



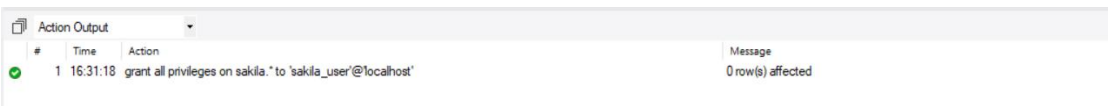
（2） 执行命令查看当前已有用户；

```
select user from mysql.user;
```

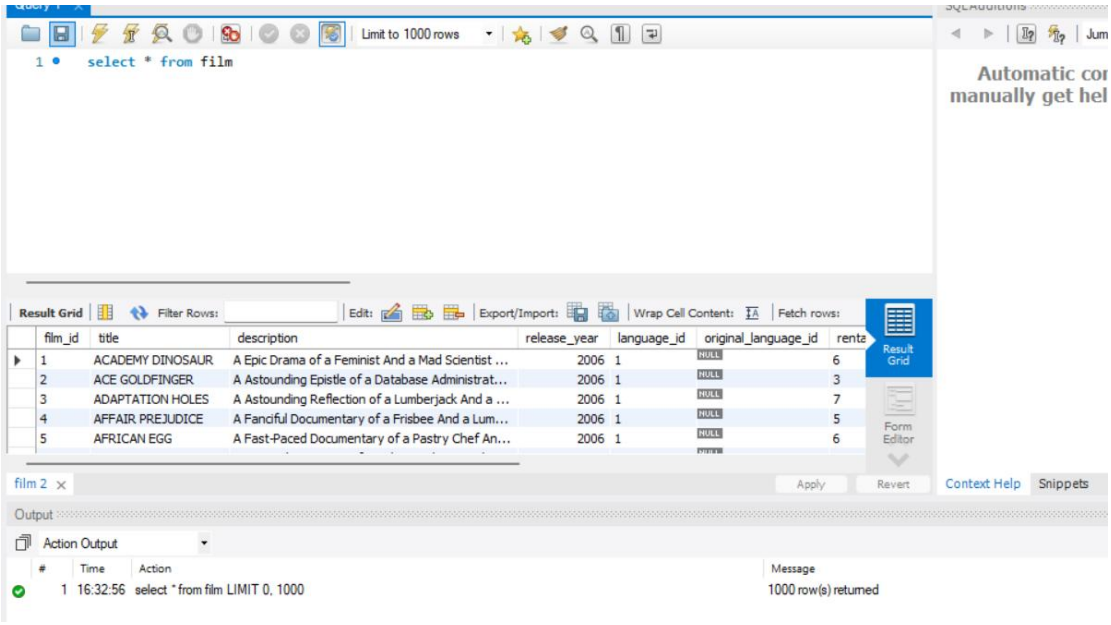


（3） 执行命令把 sakila 数据库的访问权限赋予 sakila_user 用户；

```
grant all privileges on sakila.* to 'sakila_user'@'localhost';
```



（4） 切换到 sakila_user 用户，执行 select * from film 操作。




```
mysql> select current_user();
+----------------------+
| current_user()       |
+----------------------+
| sakila_user@localhost |
+----------------------+
1 row in set (0.00 sec)

mysql> select count(*) from film;
+----------+
| count(*) |
+----------+
|     1000 |
+----------+
1 row in set (0.00 sec)
```

三、 设计并实现

根据应用场景，为 Sakila 数据库合理地设计并实现：

（截图语句和执行结果）

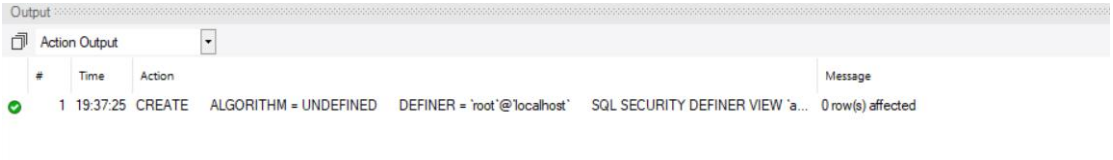
1. 设计 1 个视图，至少关联 2 个表；

（1） 执行新建视图的语句，并截图 SQL 和执行结果：

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `address_list` AS
    SELECT
        `c`.`country_id` AS `country_id`,
        `c`.`country` AS `country`,
```

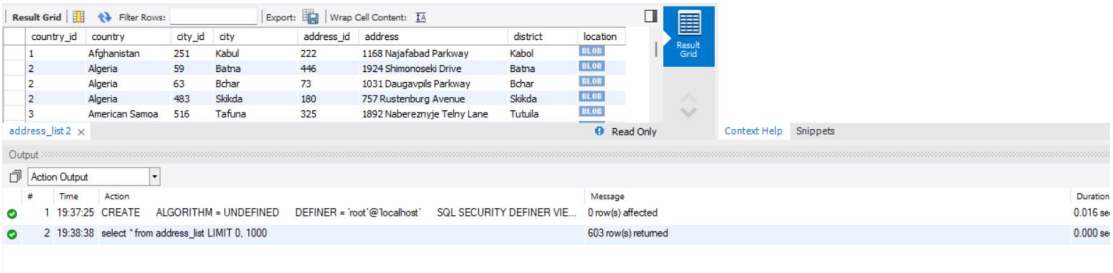
```
`ci`.`city_id` AS `city_id`,
`ci`.`city` AS `city`,
`a`.`address_id` AS `address_id`,
`a`.`address` AS `address`,
`a`.`district` AS `district`,
`a`.`location` AS `location`

FROM ((`country` `c` JOIN `city` `ci`) JOIN `address` `a`)
WHERE ((`c`.`country_id` = `ci`.`country_id`)
AND (`ci`.`city_id` = `a`.`city_id`))
```



创建的截图

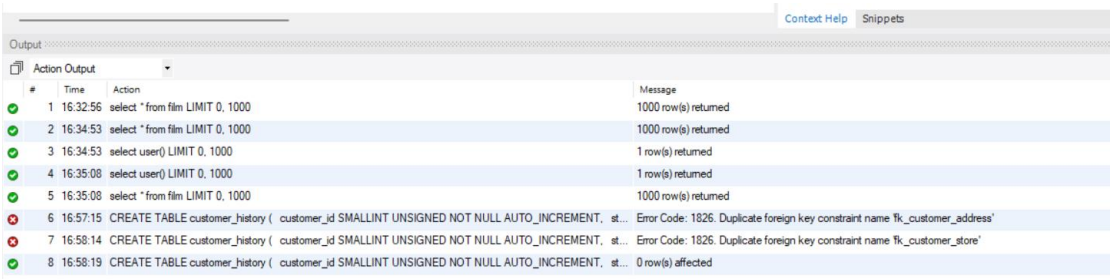
（2） 执行 select * from [视图名]，截图执行结果：



2. 设计 1 个触发器，需要体现触发器生效。

```
CREATE TABLE customer_history (
  customer_id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
  store_id TINYINT UNSIGNED NOT NULL,
  first_name VARCHAR(45) NOT NULL,
  last_name VARCHAR(45) NOT NULL,
  email VARCHAR(50) DEFAULT NULL,
  address_id SMALLINT UNSIGNED NOT NULL,
  active BOOLEAN NOT NULL DEFAULT TRUE,
  create_date DATETIME NOT NULL,
  last_update TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY  (customer_id),
  KEY idx_fk_store_id (store_id),
  KEY idx_fk_address_id (address_id),
  KEY idx_last_name (last_name),
  CONSTRAINT fk_customer_address1 FOREIGN KEY (address_id) REFERENCES address (address_id) ON DELETE RESTRICT ON UPDATE CASCADE,
  CONSTRAINT fk_customer_store1 FOREIGN KEY (store_id) REFERENCES store (store_id) ON DELETE RESTRICT ON UPDATE CASCADE
```

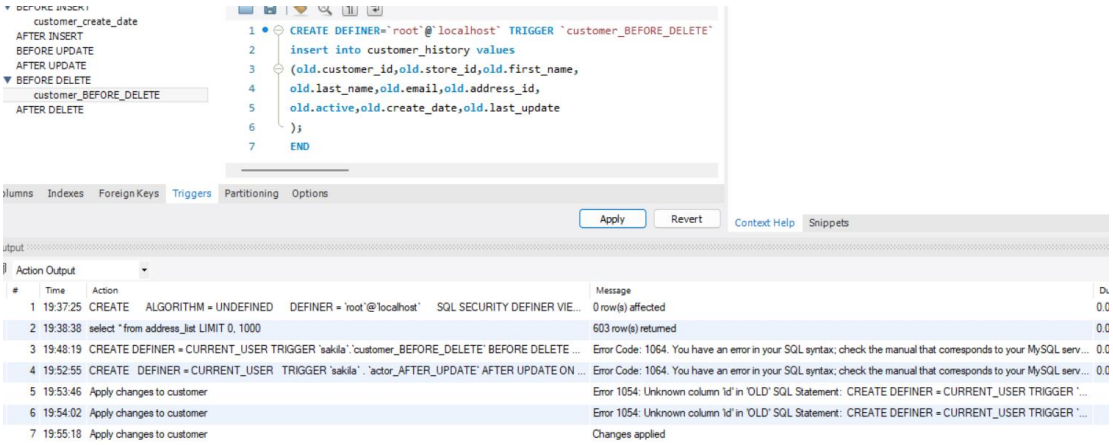
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;



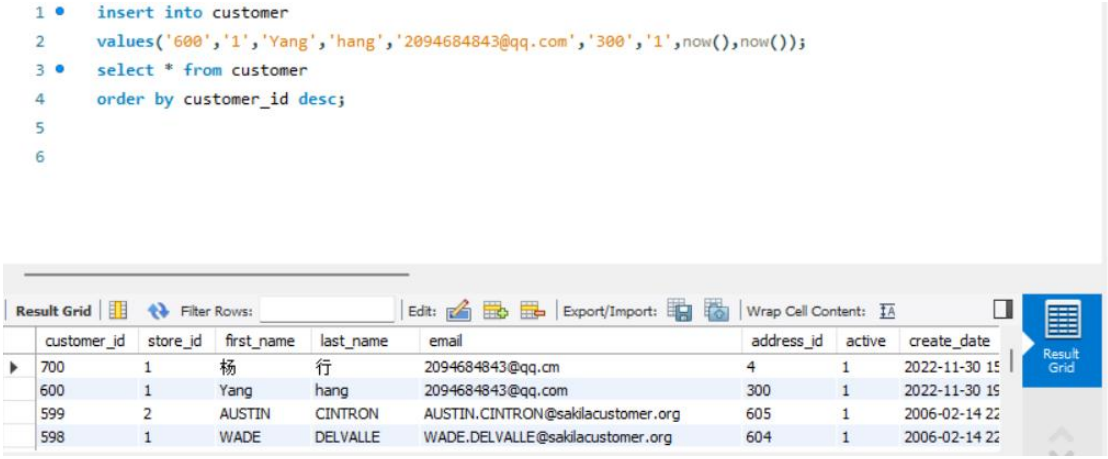
（1） 执行新建触发器的语句，并截图 SQL 和执行结果：

触发器语句：

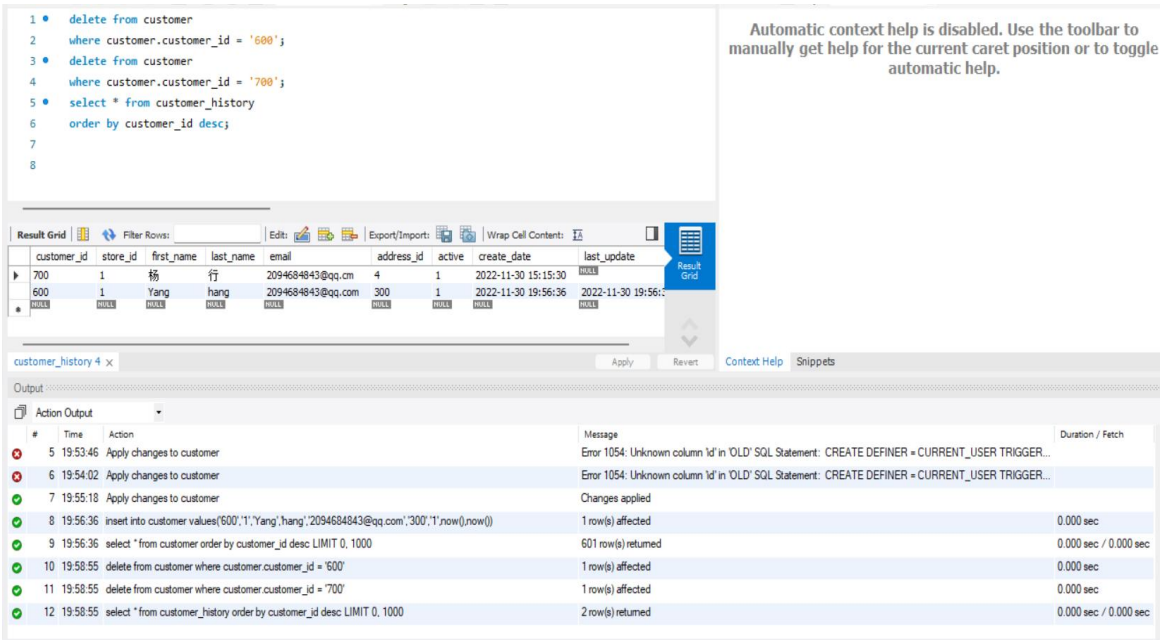
CREATE DEFINER = CURRENT_USER TRIGGER `sakila`.`customer_BEFORE_DELETE` BEFORE DELETE
ON `customer` FOR EACH ROW
BEGIN
insert into customer_history values
(old.id,old.store_id,old.first_name,
old.last_name,old.email,old.address_id,
old.active,old.create_date,old.last_update
);
END



（2） 验证触发器是否生效，截图验证过程：



插入两个顾客。



删除后打开 customer_history 有记录。

四、　　思考题

（这部分不是必做题，供有兴趣的同学思考）

在阿里开发规范里有一条“**【强制】不得使用外键与级联，一切外键概念必须在应用层解决。**”请分析一下原因。你认为外键是否没有存在的必要？

答：查阅相关资料得知 级联更新是强阻塞，存在数据库更新风暴的风险 ；且外键影响数据库的插入速度。外键应该还是有存在的必要，使用外键在大多数时候能够最大限度的保证数据的一致性和完整性