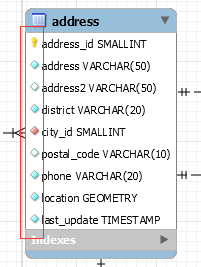
实验一报告

# 回答问题

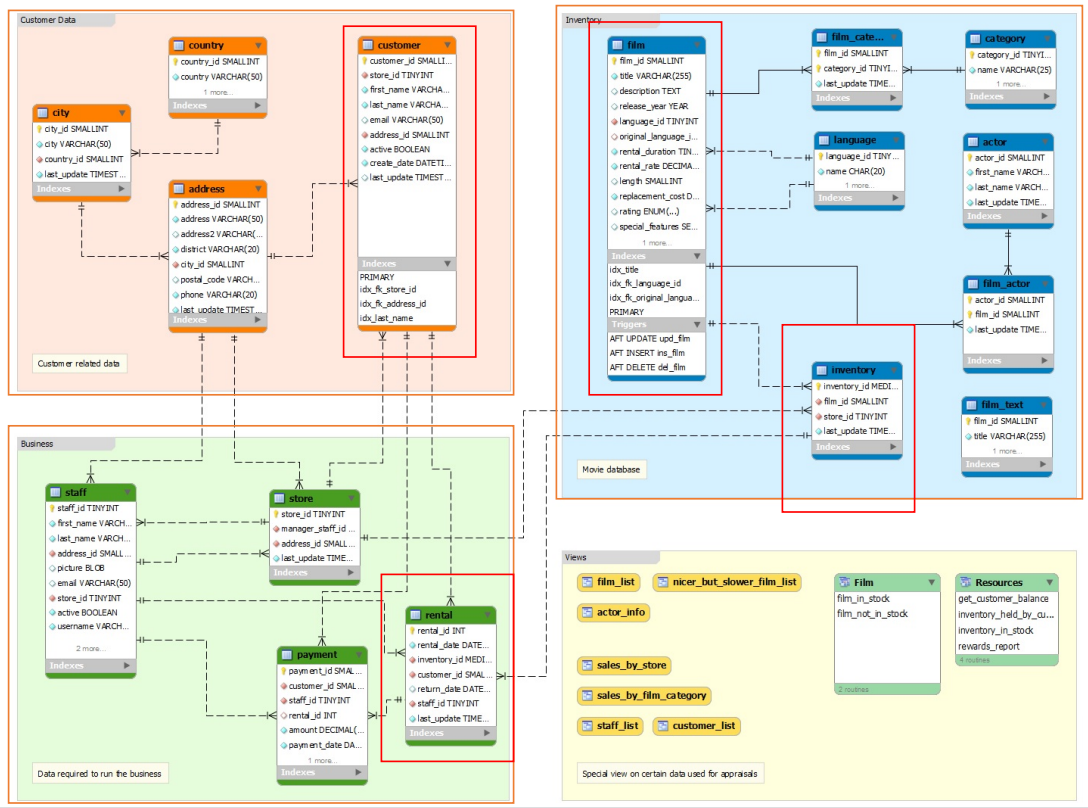
请一边熟悉sakila数据库，一边回答以下问题：

1. sakila.mwb模型中，表结构里每个字段前面的小标记分别表示什么意思？ （观察字段的属性）



|  |  |
| --- | --- |
| 标记 | 意义 |
|  | 主码（Primary Key）/主键，非空，其值能唯一标识一个元组，DBMS以主码为主要线索管理表中的各个元组。  该标记为为黄色时（如左框），表示它只是一个主键；为红色时（C:\Users\lenovo\Documents\WeChat Files\wxid_xgzr63lnxlp912\FileStorage\Temp\1669646677730.png），表示它既是一个主键，也是一个外键。 |
|  | 简单属性，非主码或外码，非空 |
|  | 简单属性，非主码或外码，可以为空 |
|  | 外码（Forei Key）/外键，表中的一个属性组，它不是该表的候选码，但是与另一张表的候选码相对应，非空。 |

2. 图中哪部分体现影片-演员关系？换句话说，如果要找出演某个影片的演员名字，访问哪几张表可以获得信息？



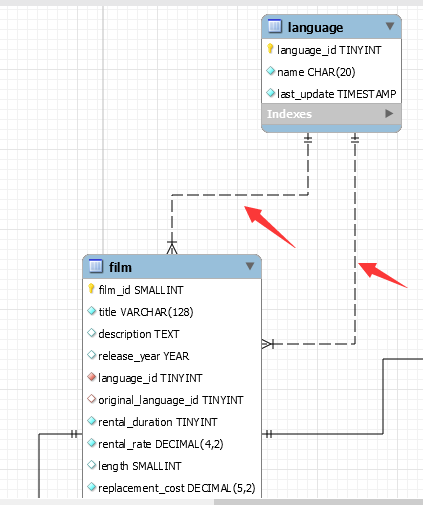
如上图，蓝色框Movie database可以体现影片-演员关系，具体来说film\_actor中记录了影片和演员的具体对应关系。若要找出演某个影片的演员名字，访问film、film\_actor和actor这三张表即可，具体来说，通过film表找到某个影片名字title对应的影片序号film\_id，然后通过film\_actor表找到出演该电影的演员序号，即film\_id对应的actor\_id，然后通过actor表找到该演员序号actor\_id对应的演员名字first\_name和last\_name。

3. 如果已知某个顾客姓名，要找到他租借的所有影片名，需要访问哪几张表？

需要访问customer、rental、inventory和film这四张表，具体来说，通过

customer表找到该顾客姓名first\_name last\_name对应的顾客序号customer\_id，然后通过rental表找到该customer\_id对应的所有inventory\_id，然后通过inventory表找到这些inventory\_id对应的电影序号film\_id，最后通过film表找到这些film\_id对应的影片名title。

4. film和language表间的2条虚线表示什么意思？



这两条虚线一条指film表中的外键language\_id对应到language表中的主键language\_id，可以唯一确定该电影的语言，而一个language\_id可以对应film表中的多个元组；另一条指film表中的外键original\_language\_id对应到language表中的主键language\_id，可以唯一确定该电影的原始语言，其中一个language\_id可以对应film表中的多个元组。

图中的虚线表示非决定关系（non-identifying relationship），指该表中的主键并不是这个外键的一部分，因此该外键在该表中无法对应到唯一的元组，如上例中的language\_id在film表中无法唯一的确定一部电影。若某表的主键是外键的一部分，那么该外键在该表中可以唯一的确定一个元组，此时则要用实线，表示决定关系（identifying relationship）。

# 实验截图

*（注意截图清晰，截图时需要体现SQL语句、执行结果、Output窗口）*

1. 请列出所有商店的详细地址，显示商店id，商店地址，所在区域，所在城市，所在国家；

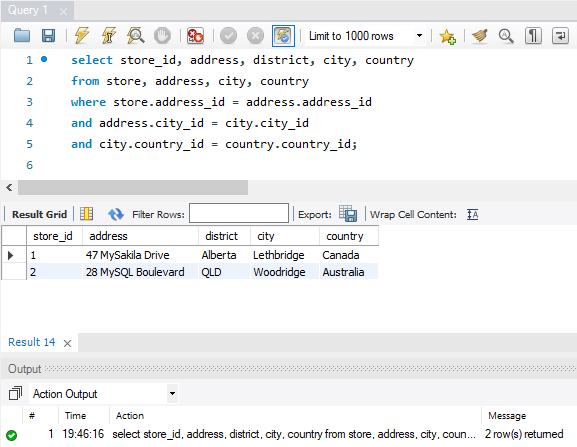
select store\_id, address, district, city, country

from store, address, city, country

where store.address\_id = address.address\_id

and address.city\_id = city.city\_id

and city.country\_id = country.country\_id;



1. 哪些演员出演过影片《ROCKY WAR》？请列出他的first\_name, last\_name；

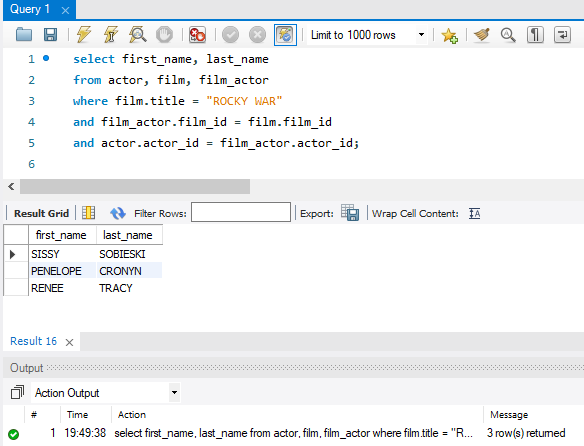
select first\_name, last\_name

from actor, film, film\_actor

where film.title = "ROCKY WAR"

and film\_actor.film\_id = film.film\_id

and actor.actor\_id = film\_actor.actor\_id;



1. 找出租DVD花费最高的前5名，请列出他们的first\_name, last\_name和每个人花费的金额；

select first\_name, last\_name, sum(amount)

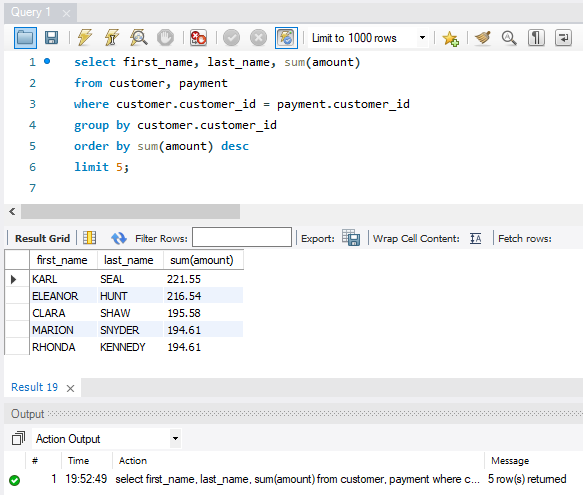
from customer, payment

where customer.customer\_id = payment.customer\_id

group by customer.customer\_id

order by sum(amount) desc

limit 5;



1. 哪个影片获得了总体最高的租金？请列出影片id、影片名、总租金；

select film.film\_id, title, sum(amount)

from film, payment, inventory, rental

where film.film\_id = inventory.film\_id

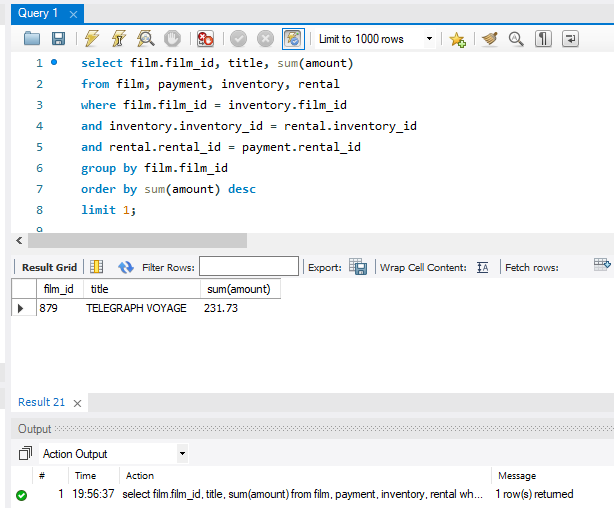
and inventory.inventory\_id = rental.inventory\_id

and rental.rental\_id = payment.rental\_id

group by film.film\_id

order by sum(amount) desc

limit 1;



但此种方法无法解决有两部电影同时获得总体最高租金的情况，下面给出一种略复杂但可以解决该问题的做法：

select film.film\_id, title, sum(amount)

from film, payment, inventory, rental

where film.film\_id = inventory.film\_id

and inventory.inventory\_id = rental.inventory\_id

and rental.rental\_id = payment.rental\_id

group by film.film\_id

having sum(amount) >= all (

select sum(amount)

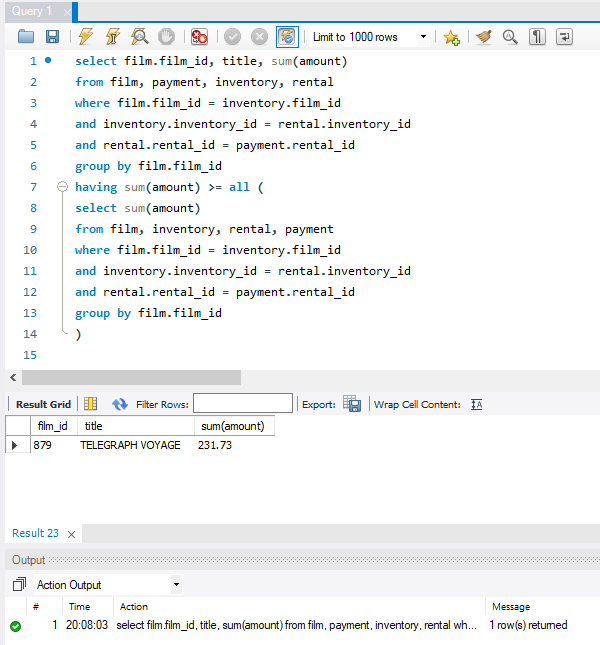
from film, inventory, rental, payment

where film.film\_id = inventory.film\_id

and inventory.inventory\_id = rental.inventory\_id

and rental.rental\_id = payment.rental\_id

group by film.film\_id)



1. 哪个演员出演的电影超过35部？ 请列出演员id、演员名、出演的电影数；

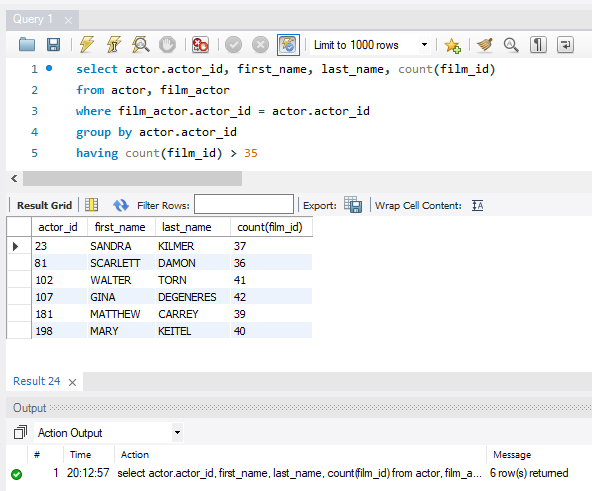
select actor.actor\_id, first\_name, last\_name, count(film\_id)

from actor, film\_actor

where film\_actor.actor\_id = actor.actor\_id

group by actor.actor\_id

having count(film\_id) > 35



1. 请找出没有租借过电影《TELEGRAPH VOYAGE》的顾客姓名；

select first\_name, last\_name

from customer

where not exists

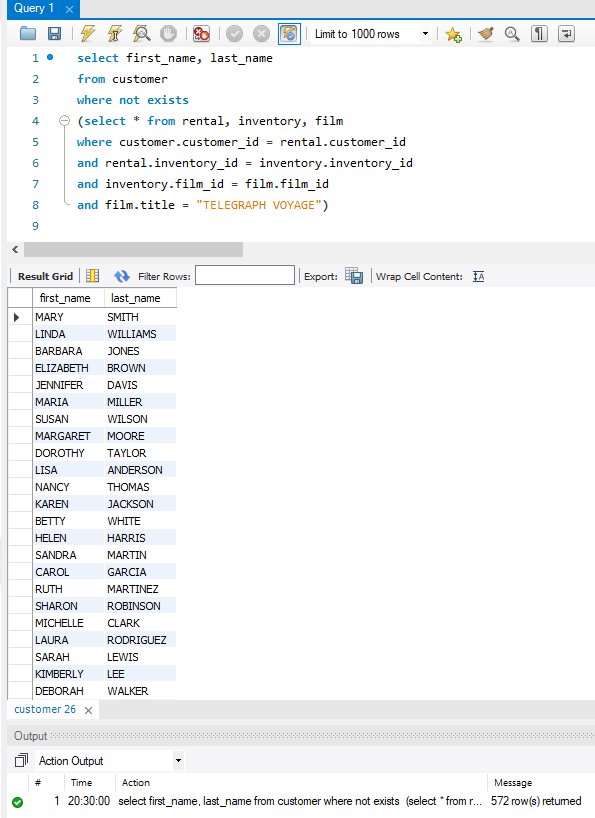
(select \* from rental, inventory, film

where customer.customer\_id = rental.customer\_id

and rental.inventory\_id = inventory.inventory\_id

and inventory.film\_id = film.film\_id

and film.title = "TELEGRAPH VOYAGE")



1. 查询演过《ELEPHANT TROJAN》和《SPLASH GUMP》这两部电影的演员，列出其姓名；

select distinct first\_name, last\_name

from actor, film\_actor as fa1, film\_actor as fa2, film as film1, film as film2

where actor.actor\_id = fa1.actor\_id

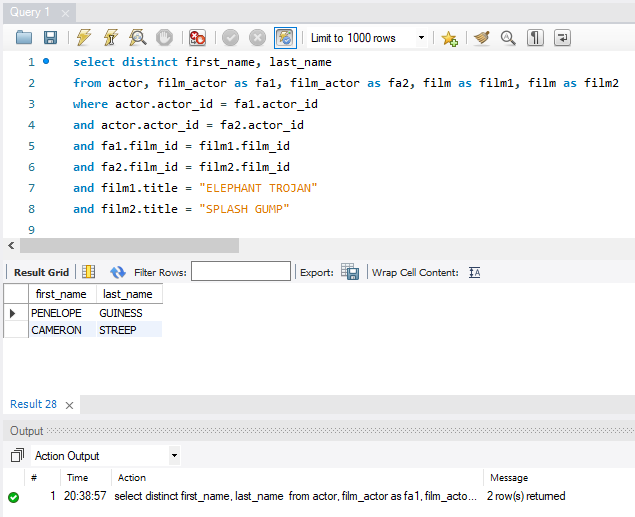
and actor.actor\_id = fa2.actor\_id

and fa1.film\_id = film1.film\_id

and fa2.film\_id = film2.film\_id

and film1.title = "ELEPHANT TROJAN"

and film2.title = "SPLASH GUMP"



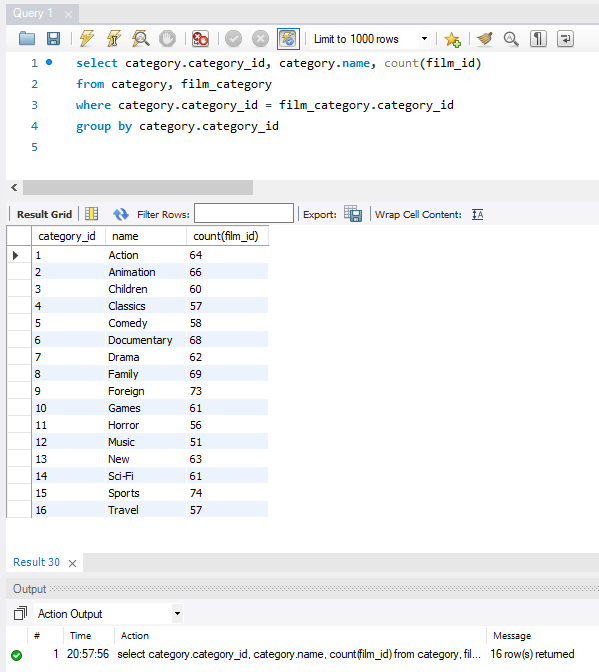
1. 统计每种类型的影片数，显示类型编号、类型名称、该类型影片数；

select category.category\_id, category.name, count(film\_id)

from category, film\_category

where category.category\_id = film\_category.category\_id

group by category.category\_id



1. 有哪些影片是2个商店都有库存的？

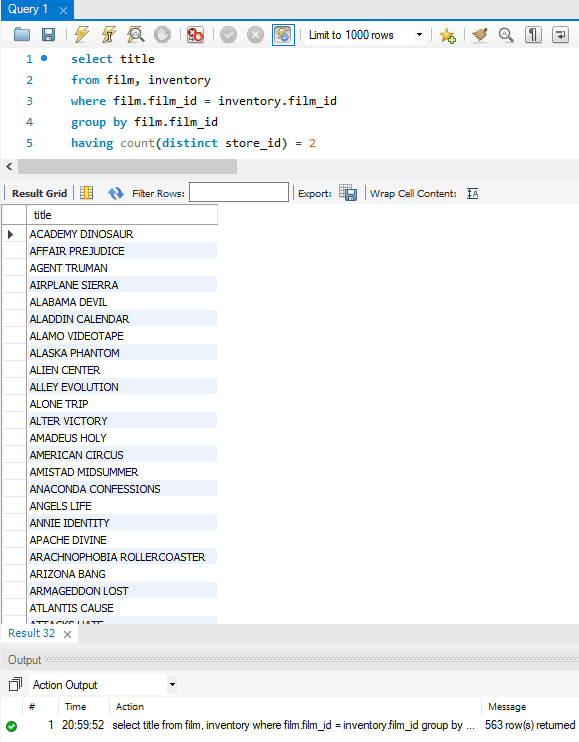
select title

from film, inventory

where film.film\_id = inventory.film\_id

group by film.film\_id

having count(distinct store\_id) = 2



使用连接也可以达到相同的效果：

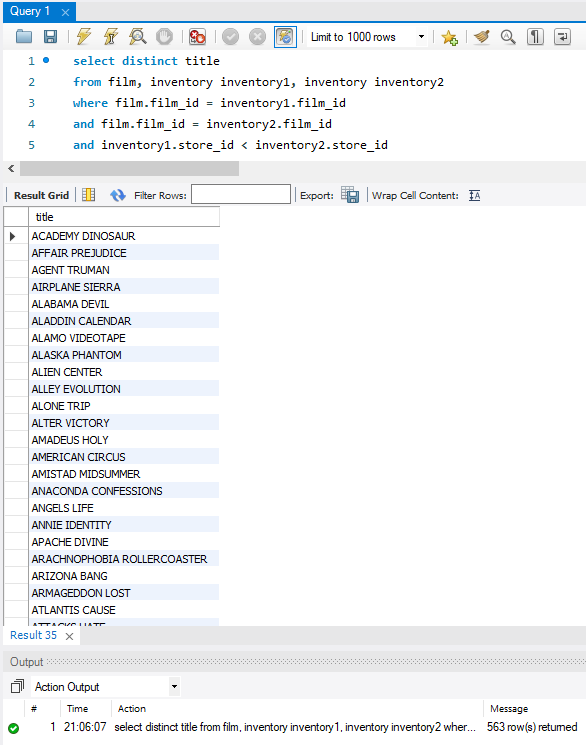
select distinct title

from film, inventory inventory1, inventory inventory2

where film.film\_id = inventory1.film\_id

and film.film\_id = inventory2.film\_id

and inventory1.store\_id < inventory2.store\_id



1. 查询单次租借影片时间最长的6位客户，列出其first\_name、last\_name和当次租借时长；

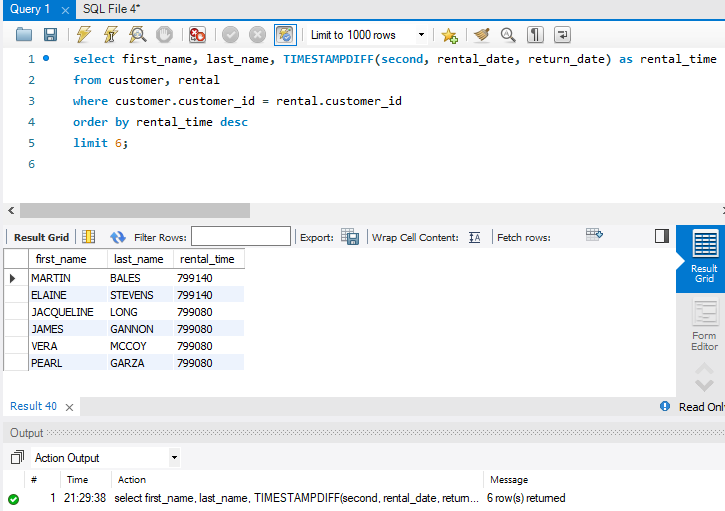
select first\_name, last\_name, TIMESTAMPDIFF(second, rental\_date, return\_date) as rental\_time

from customer, rental

where customer.customer\_id = rental.customer\_id

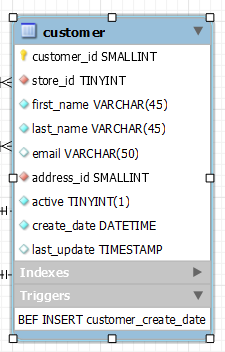
order by rental\_time desc

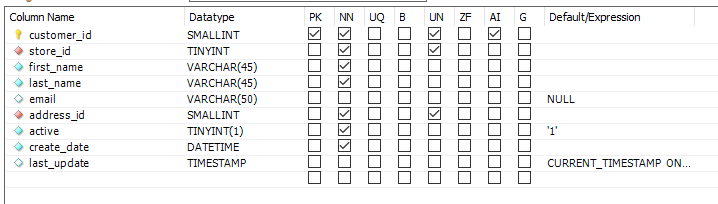
limit 6;



1. 在customer表中新增一条数据，注意customer表与其他表的关系；

观察如下sakila.mwb数据模型可知，在customer表中，属性store\_id和address\_id均为外键，因此在插入时只能选择其作为候选码的表中已有的值，如store\_id取值为1，address\_id取值为605。同时，create\_date属性存在trigger，会设置为insert该行时的时间；last\_update属性存在默认值，为last\_update该行的时间；active属性存在默认值为1。因此，在插入时，只需要设置customer\_id, store\_id, first\_name, last\_name, email, address\_id这几个属性值即可。





可在表中增加数据如下：

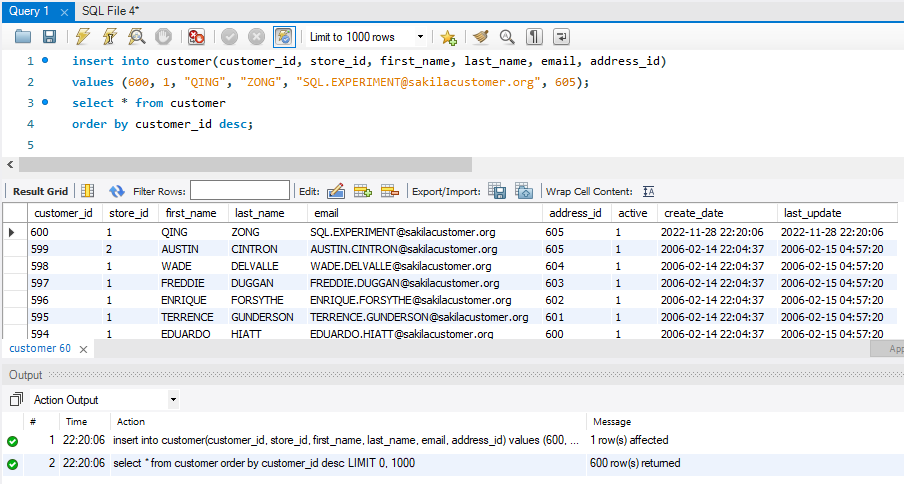
insert into customer(customer\_id, store\_id, first\_name, last\_name, email, address\_id)

values (600, 1, "QING", "ZONG", "SQL.EXPERIMENT@sakilacustomer.org", 605);

然后查看添加数据之后的表：

select \* from customer

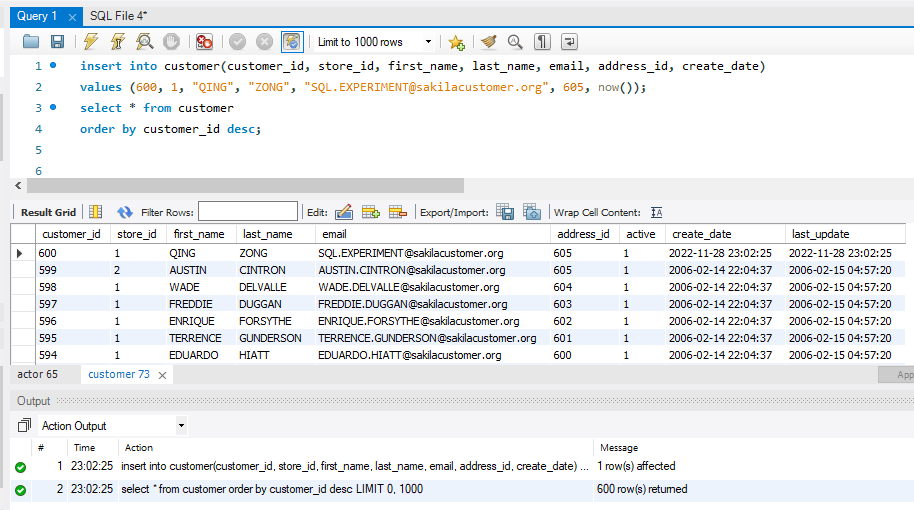
order by customer\_id desc;



当然，也可以用now()函数来设置create\_date属性：

insert into customer(customer\_id, store\_id, first\_name, last\_name, email, address\_id, create\_date)

values (600, 1, "QING", "ZONG", "SQL.EXPERIMENT@sakilacustomer.org", 605, now());



1. 修改刚才在customer表中新增的那条数据；

可在表中修改新增的数据如下：

update customer

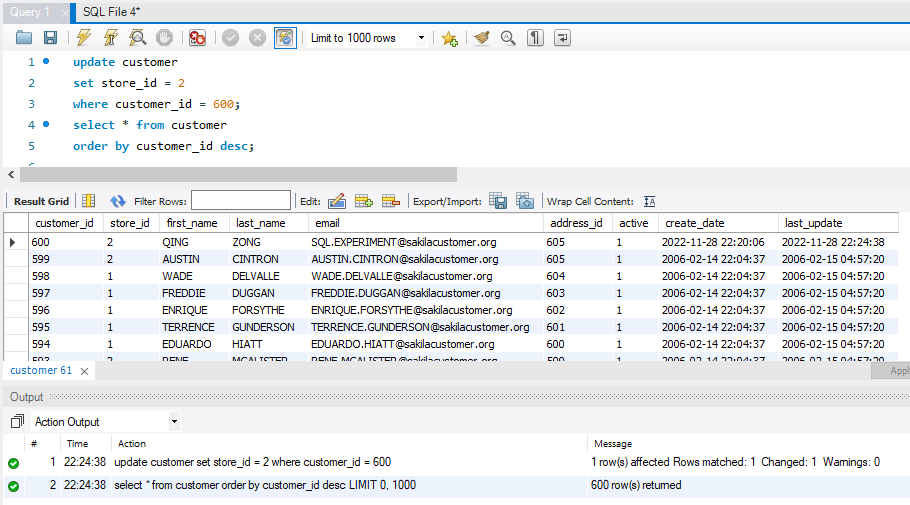
set store\_id = 2

where customer\_id = 600;

然后查看添加数据之后的表：

select \* from customer

order by customer\_id desc;



1. 删除第11步新增的那条数据。

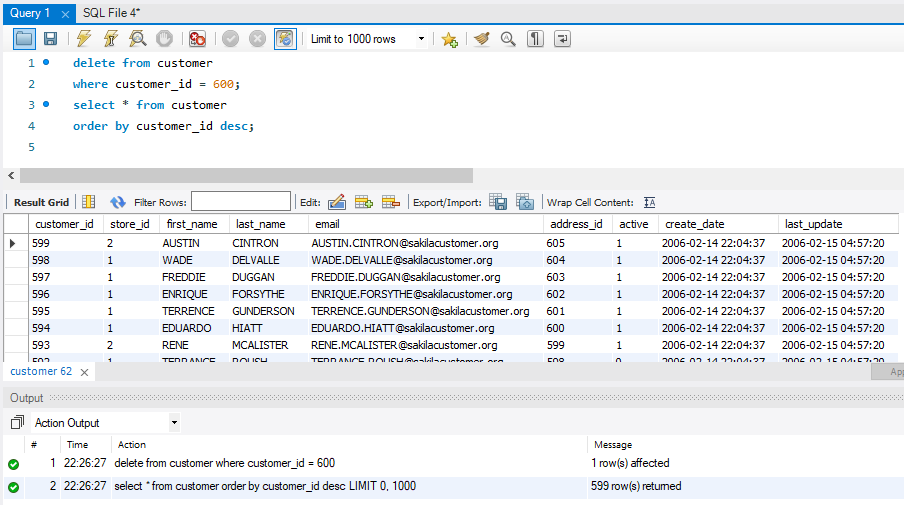
delete from customer

where customer\_id = 600;

然后查看添加数据之后的表：

select \* from customer

order by customer\_id desc;



# 思考题

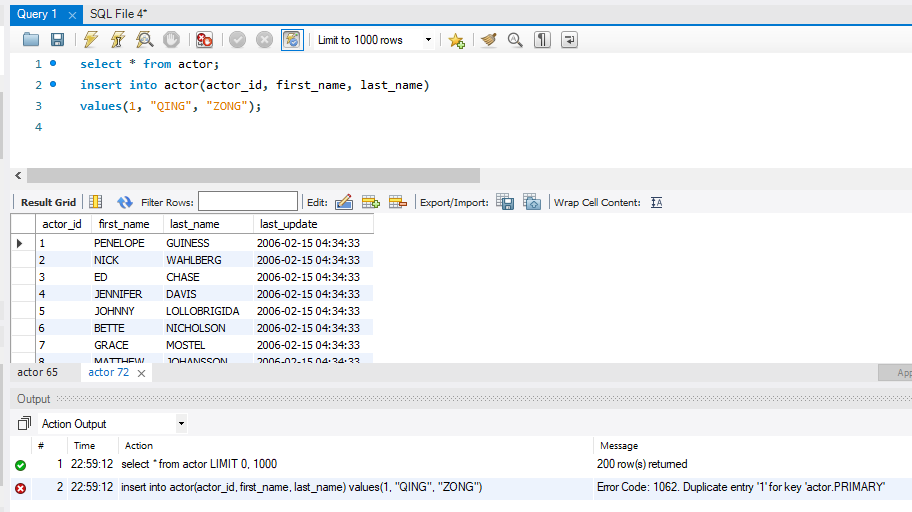
1. 如果insert一条数据到actor表，但actor\_id和已有数据重复，会发生什么？同学们请自己尝试一下，截图并分析原因。

插入一条actor\_id为1的数据：

insert into actor(actor\_id, first\_name, last\_name)

values(1, “QING”, “ZONGQ”);

会产生如下错误：



通过Message，我们发现错误原因是插入了相同主键的元组。因为主键需要满足唯一性，能唯一标识一个元组，因此只要插入了主键与现有元组中的主键相同的数据就会产生这样的错误。

1. insert语句还用了一个函数NOW()，是做什么的呢？

函数now()用于获取当前的日期和时间，以时间戳timestamp的格式显示：

