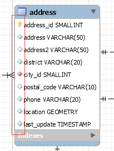
**实验一报告**

1. **回答问题**

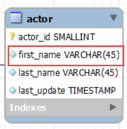
请一边熟悉sakila数据库，一边回答以下问题：

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 |
|  | 用来唯一标识表中每一行的键。 |
|  | 表的属性，其值必须存在 |
|  | 表的可选属性，即在某些情况下其值为空 |
|  | 该属性为外键，受外键约束且必须不为空 |

2. char和varchar类型的区别是什么？



CHAR类型的字段长度是固定的，长度为n。即使存储的字符串长度小于 n，数据库也会用空格（spaces）来填充

而VARCHAR长度可变，节省存储空间来的

3. 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？

film，actor，以及film\_actor做到了二者的联结

4. 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？

customer表，rental表，inventory表以及film表

1. **实验截图**

（请注意粘贴文本格式的*SQL*语句，截图执行结果和*Output*窗口）

1. 请列出所有country是“France”的客户的信息，显示customer\_id、客户姓名、地址、所在区域，所在城市（注意：客户姓名请以first\_name+空格+last\_name的格式，例如：SISSY SOBIESKI）；

SELECT customer.customer\_id,

CONCAT(customer.first\_name, ' ', customer.last\_name) AS customer\_name,

CONCAT\_WS(' ', address.address, address.address2) AS customer\_address,

address.district,

city.city

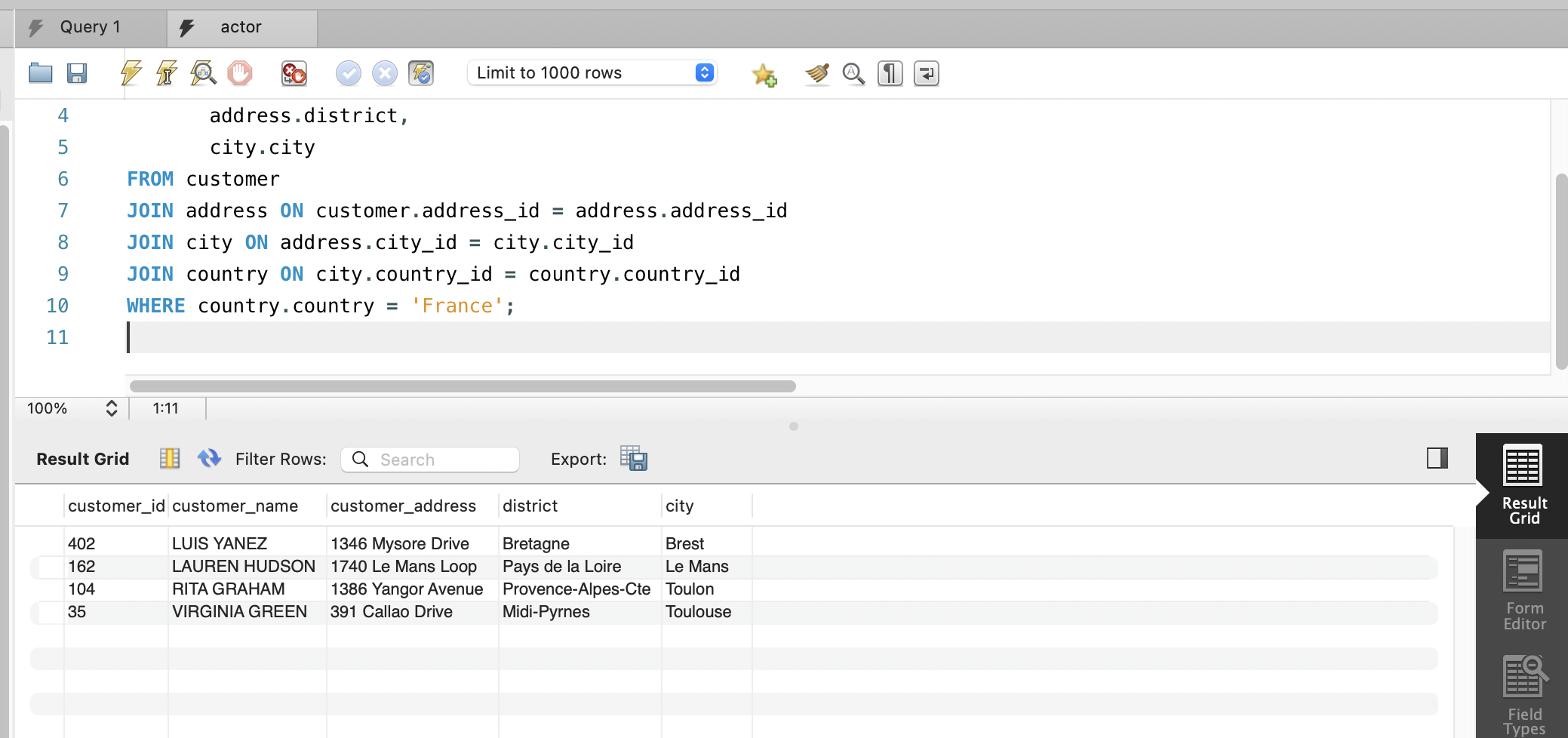
FROM customer

JOIN address ON customer.address\_id = address.address\_id

JOIN city ON address.city\_id = city.city\_id

JOIN country ON city.country\_id = country.country\_id

WHERE country.country = 'France';



1. 列出属于“Children”类型并以“J”开头的电影名；

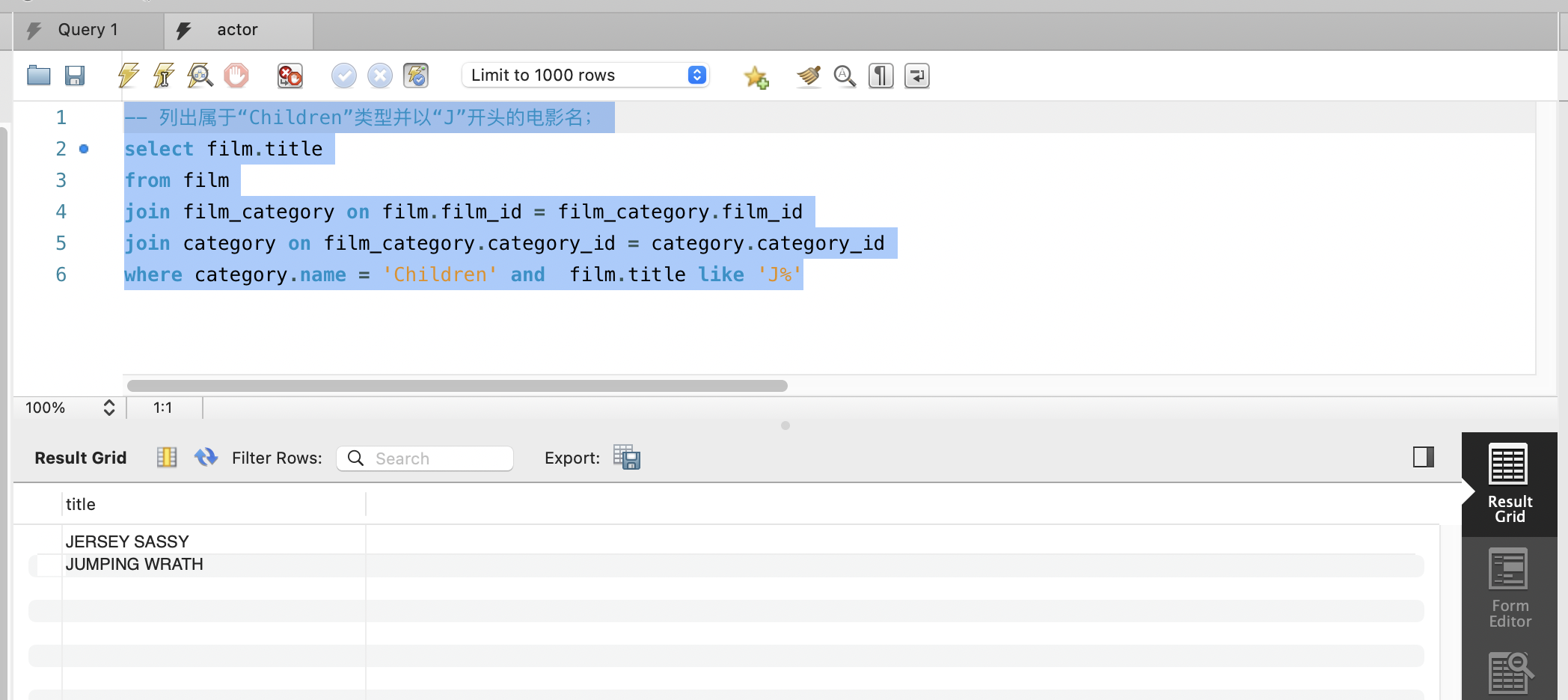
select film.title

from film

join film\_category on film.film\_id = film\_category.film\_id

join category on film\_category.category\_id = category.category\_id

where category.name = 'Children' and film.title like 'J%'



1. 找出费用在180至200之间的客户，列出他们的first\_name, last\_name和每个人花费的金额；

select customer.first\_name as first\_name,

customer.last\_name,

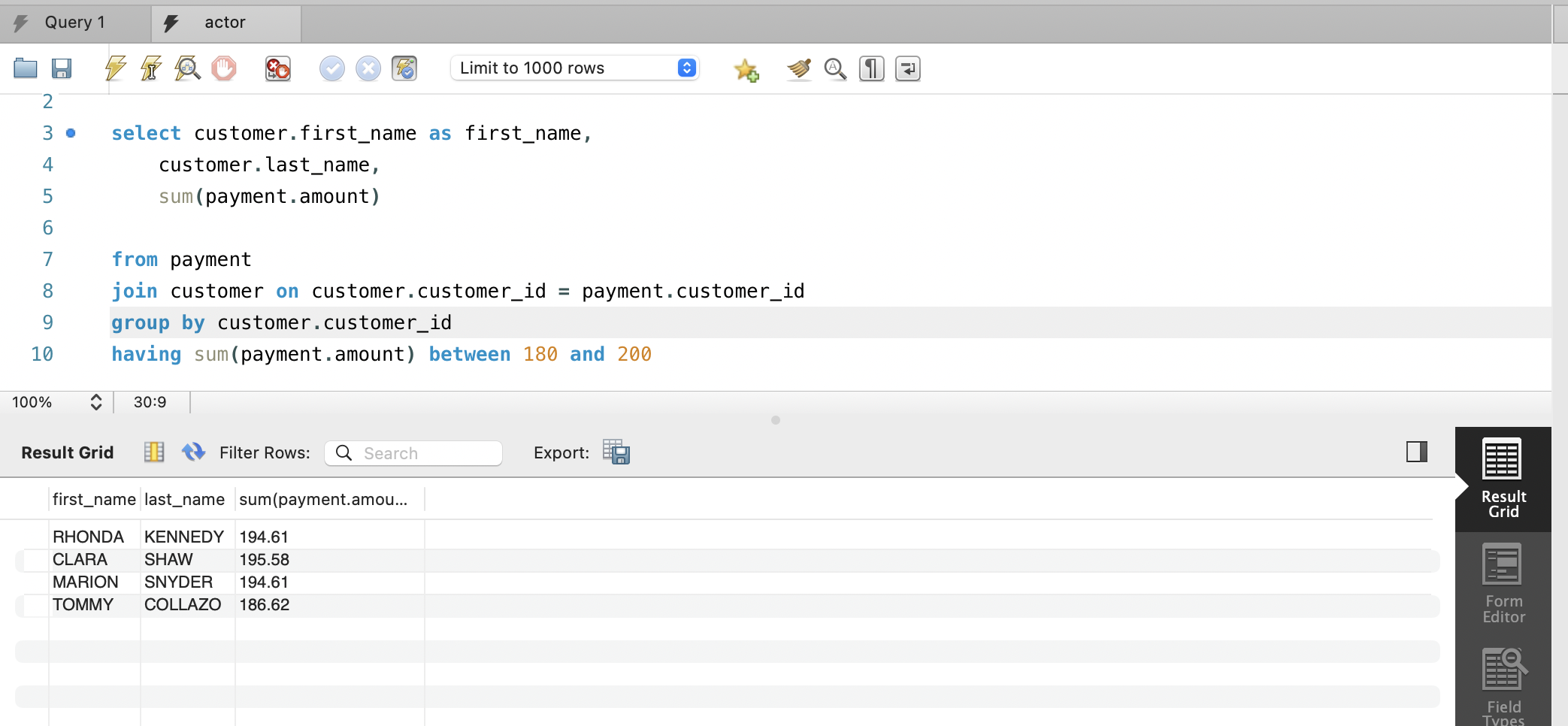
sum(payment.amount)

from payment

join customer on customer.customer\_id = payment.customer\_id

group by customer.customer\_id

having sum(payment.amount) between 180 and 200



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,

film.title,

sum(payment.amount) as total\_rental\_payment

from film

join inventory on inventory.film\_id = film.film\_id

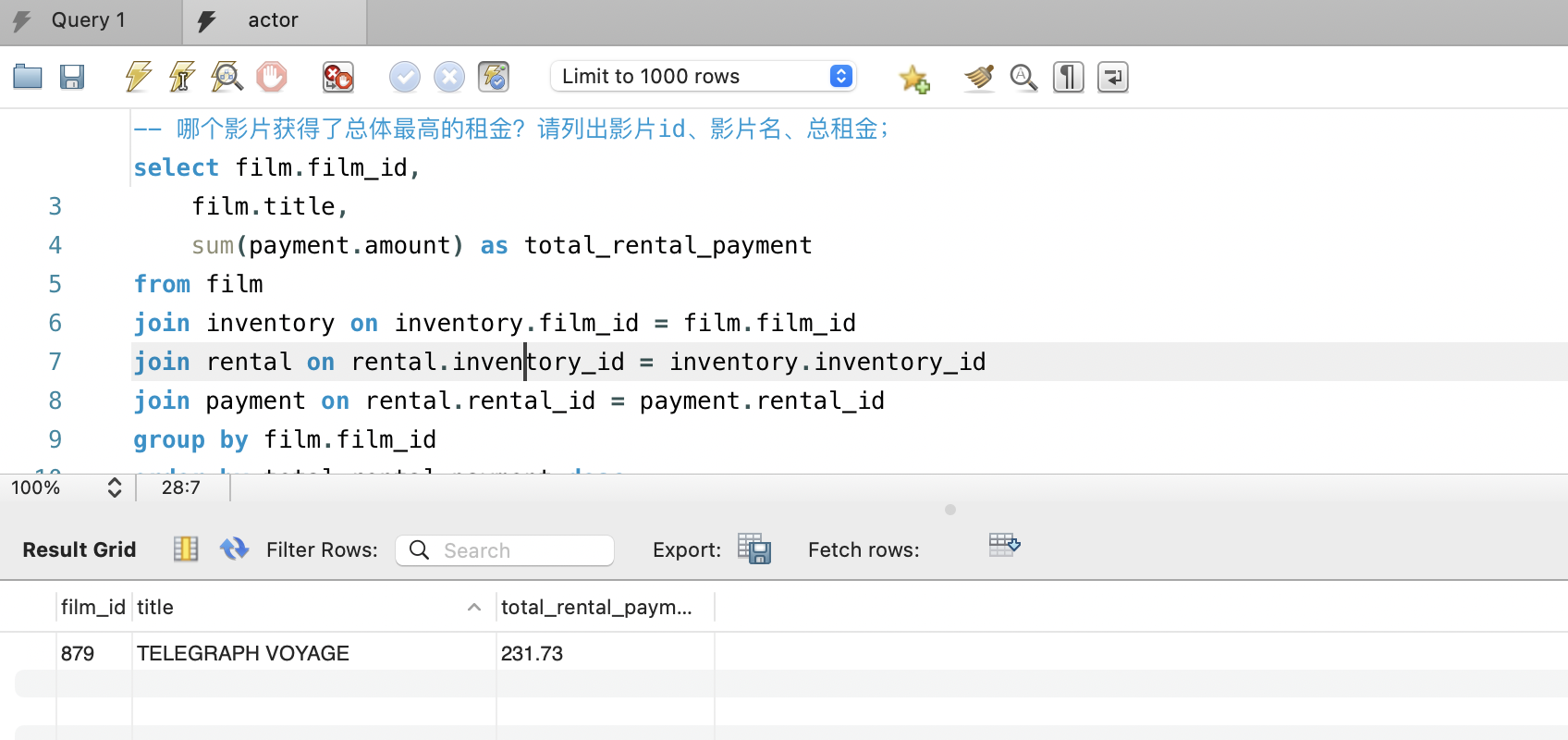
join rental on rental.inventory\_id = inventory.inventory\_id

join payment on rental.rental\_id = payment.rental\_id

group by film.film\_id

order by total\_rental\_payment desc

limit 1



1. 哪些演员出演的电影超过38部？ 请列出演员名、出演的电影数；

select concat\_ws(" ", actor.first\_name, actor.last\_name),

count(film.film\_id) as total\_film\_starred

from actor

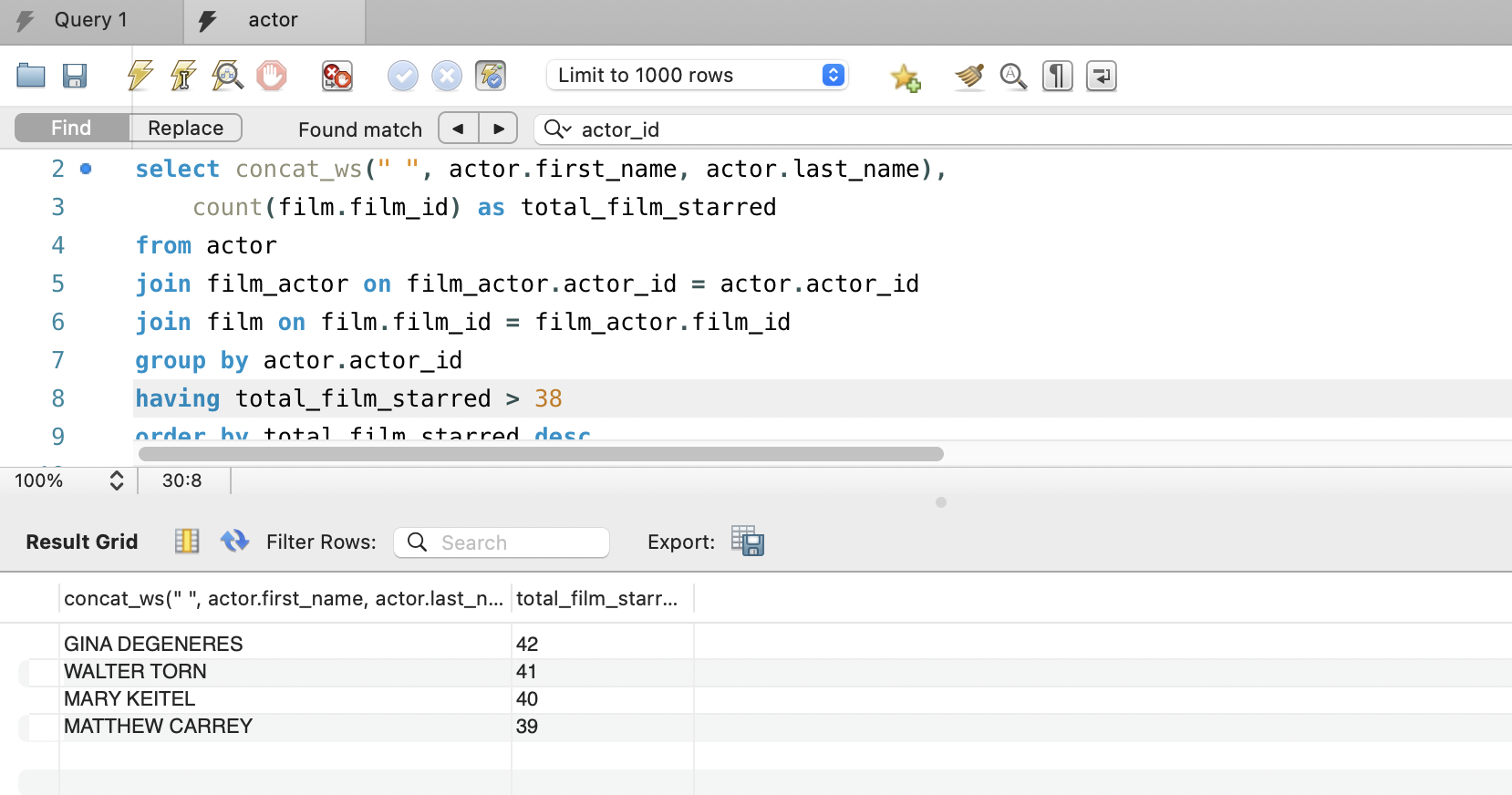
join film\_actor on film\_actor.actor\_id = actor.actor\_id

join film on film.film\_id = film\_actor.film\_id

group by actor.actor\_id

having total\_film\_starred > 28

order by total\_film\_starred desc



1. 请找出没有租借过电影《NATURAL STOCK》的顾客姓名；

select concat\_ws(' ', customer.first\_name, customer.last\_name) as customer\_name

from customer

where customer.customer\_id not in (

select customer.customer\_id

from inventory

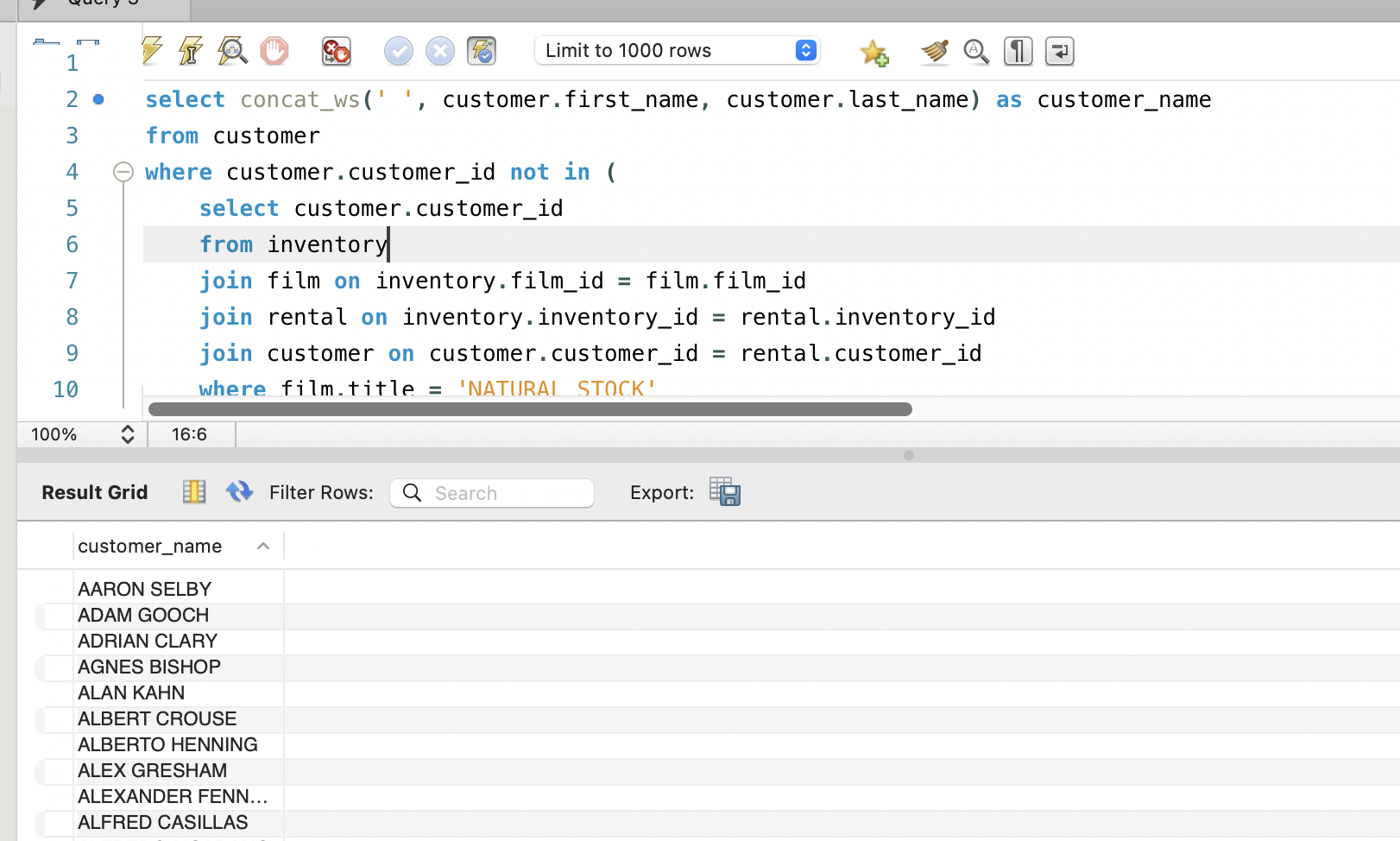
join film on inventory.film\_id = film.film\_id

join rental on inventory.inventory\_id = rental.inventory\_id

join customer on customer.customer\_id = rental.customer\_id

where film.title = 'NATURAL STOCK'

)



1. 查询既演过《ELEPHANT TROJAN》又演过《DOGMA FAMILY》的演员，列出其姓名；

select concat\_ws(' ', actor.first\_name, actor.last\_name) as actor\_name

from film

join film\_actor on film.film\_id = film\_actor.film\_id

join actor on actor.actor\_id = film\_actor.actor\_id

where film.title = 'ELEPHANT TROJAN'

and concat\_ws(' ', actor.first\_name, actor.last\_name) in

(

select concat\_ws(' ', actor.first\_name, actor.last\_name) as actor\_name

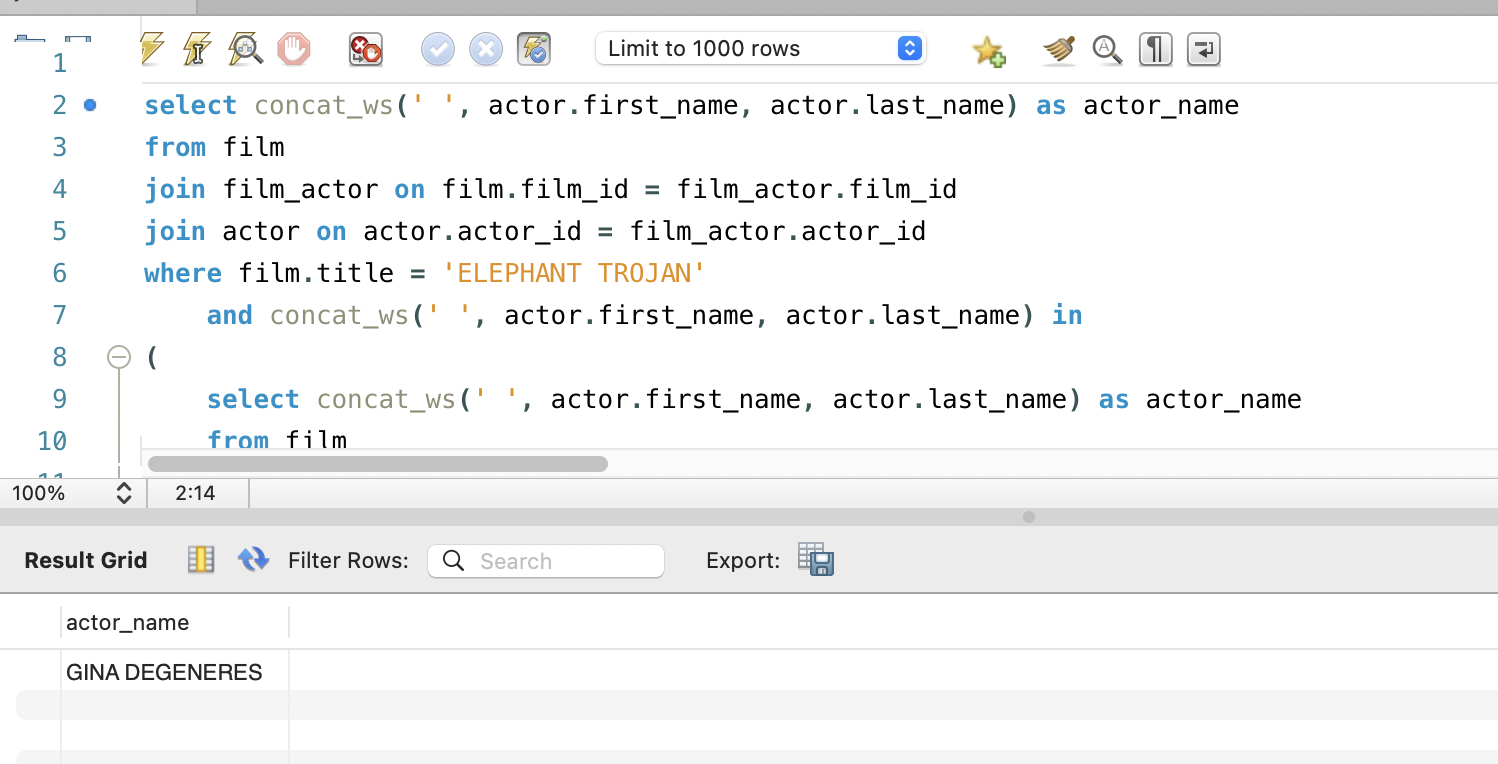
from film

join film\_actor on film.film\_id = film\_actor.film\_id

join actor on actor.actor\_id = film\_actor.actor\_id

where film.title = 'DOGMA FAMILY'

)



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.film\_id) as film\_number

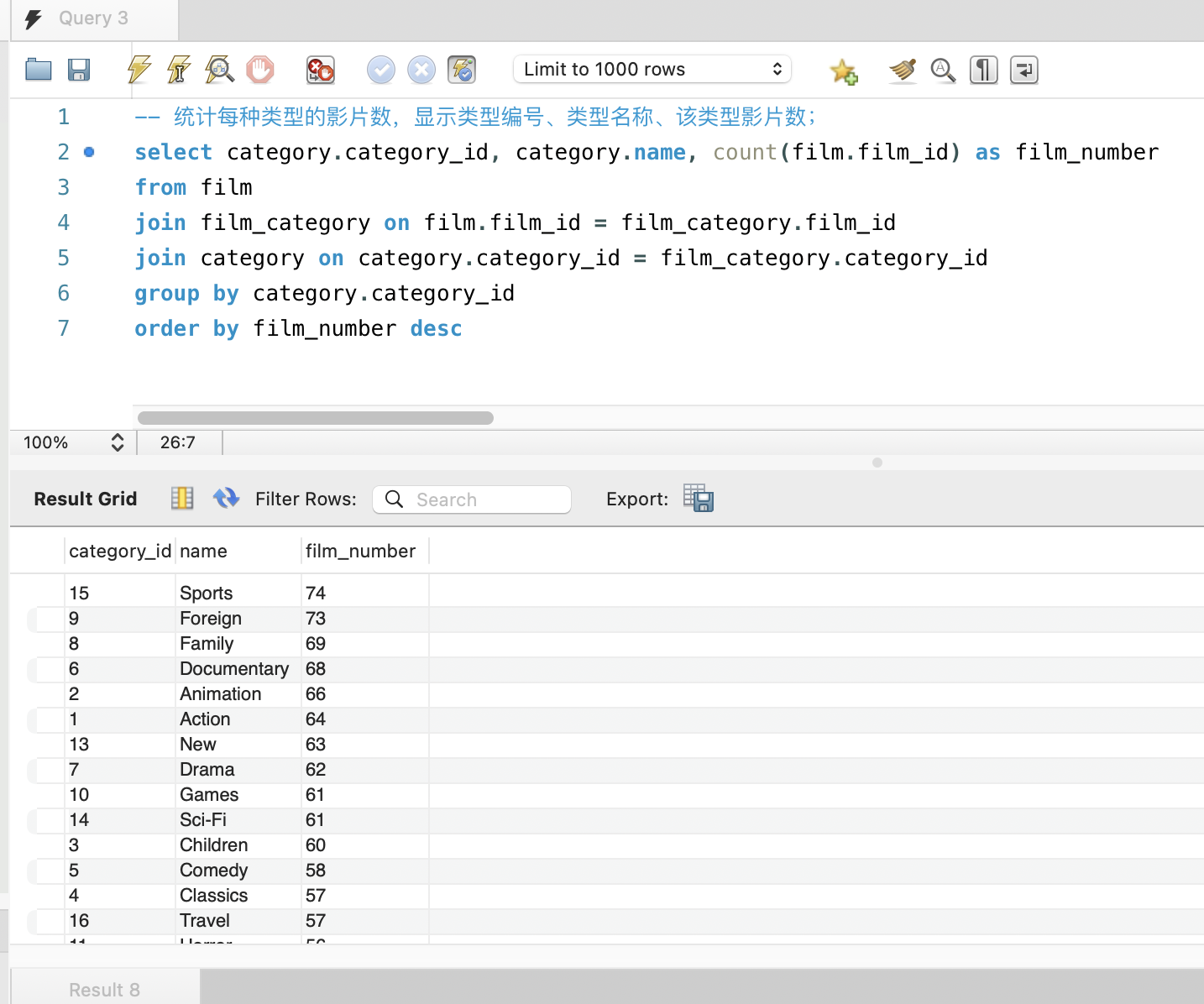
from film

join film\_category on film.film\_id = film\_category.film\_id

join category on category.category\_id = film\_category.category\_id

group by category.category\_id

order by film\_number desc



1. 找出最热门的（被最多不同人租借过）影片名，并显示租借人数；

select film.title, count(distinct customer.customer\_id) as lease\_number

from film

join inventory on inventory.film\_id = film.film\_id

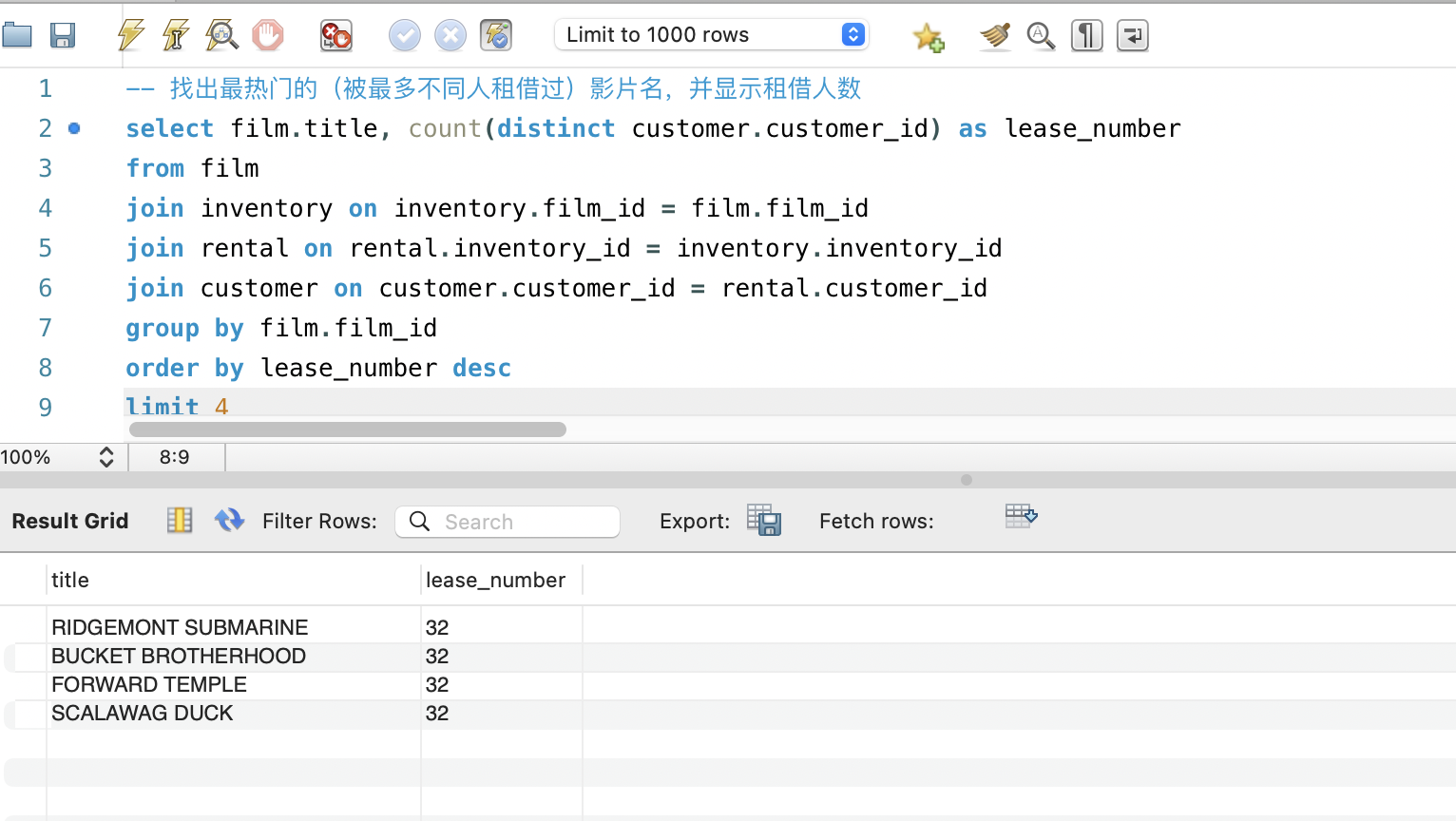
join rental on rental.inventory\_id = inventory.inventory\_id

join customer on customer.customer\_id = rental.customer\_id

group by film.film\_id

order by lease\_number desc

limit 4



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（单位秒）；

select

customer.first\_name,

customer.last\_name,

max(timestampdiff(second, rental.rental\_date, rental.return\_date))as lease\_period

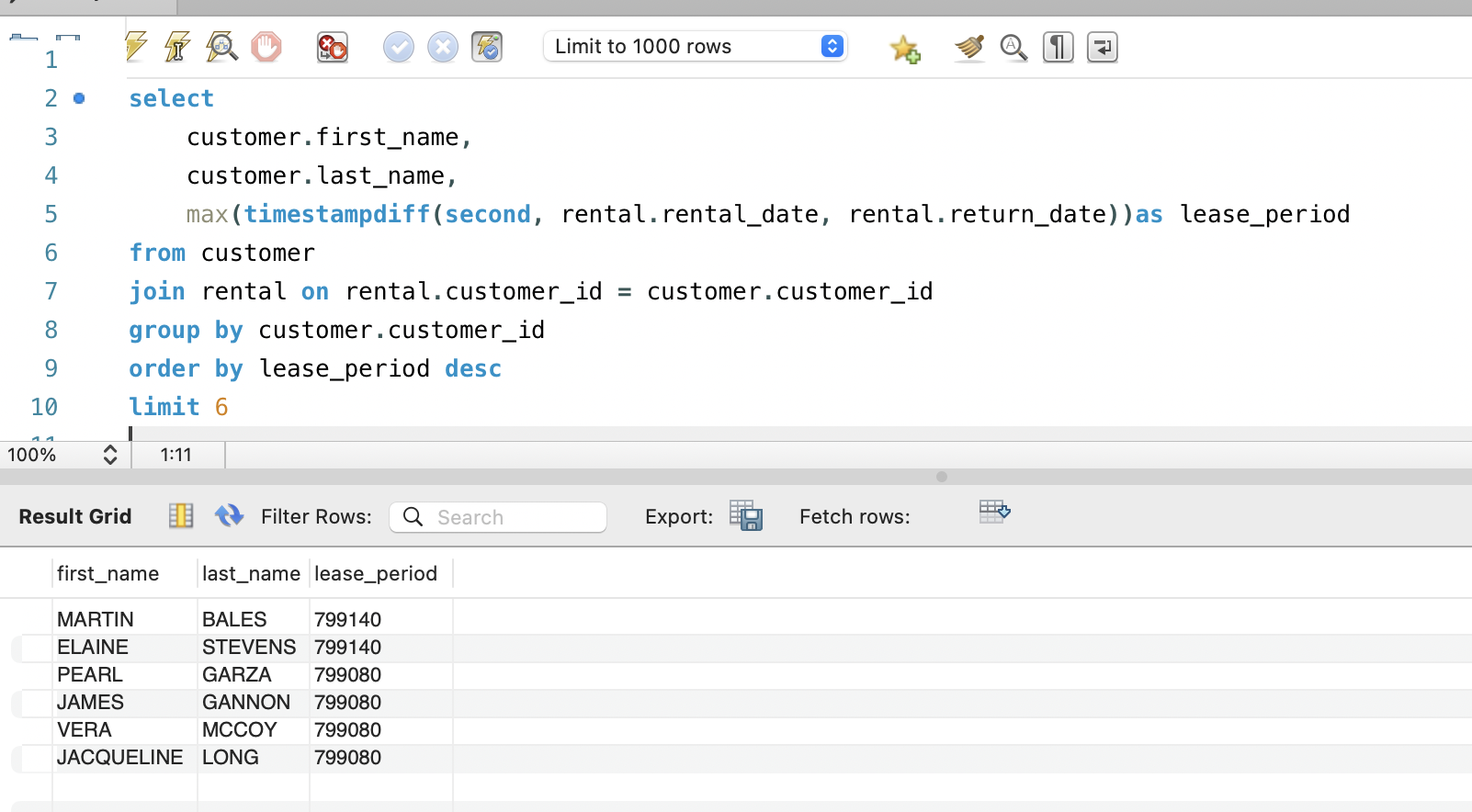
from customer

join rental on rental.customer\_id = customer.customer\_id

group by customer.customer\_id

order by lease\_period 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；

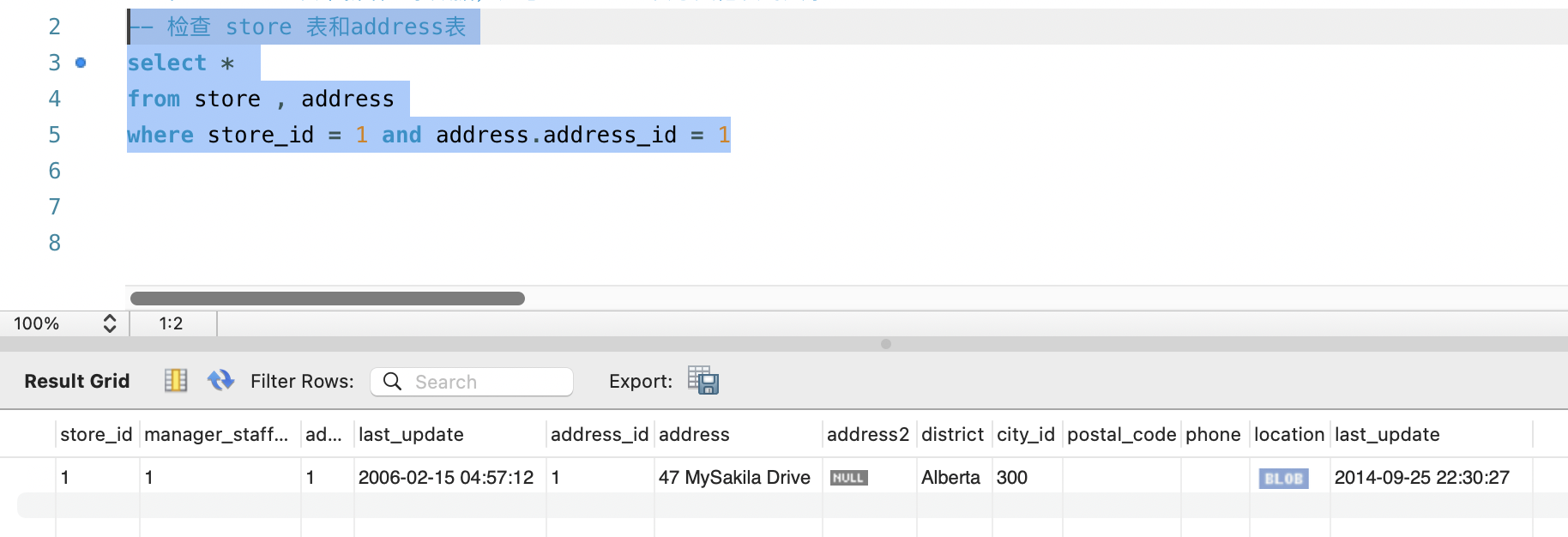
首先需要确保插入的customer的address\_id和store\_id存在：

-- 检查 store 表和address表

select \*

from store , address

where store\_id = 1 and address.address\_id = 1



如上图所示，确实存在，因此可以进行插入

通过查看customer表的属性，发现customer\_id是自增长的因此不需要手动设置customer\_id的值.

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

values(

1,

'Junkun',

'Yang',

'email@email.com',

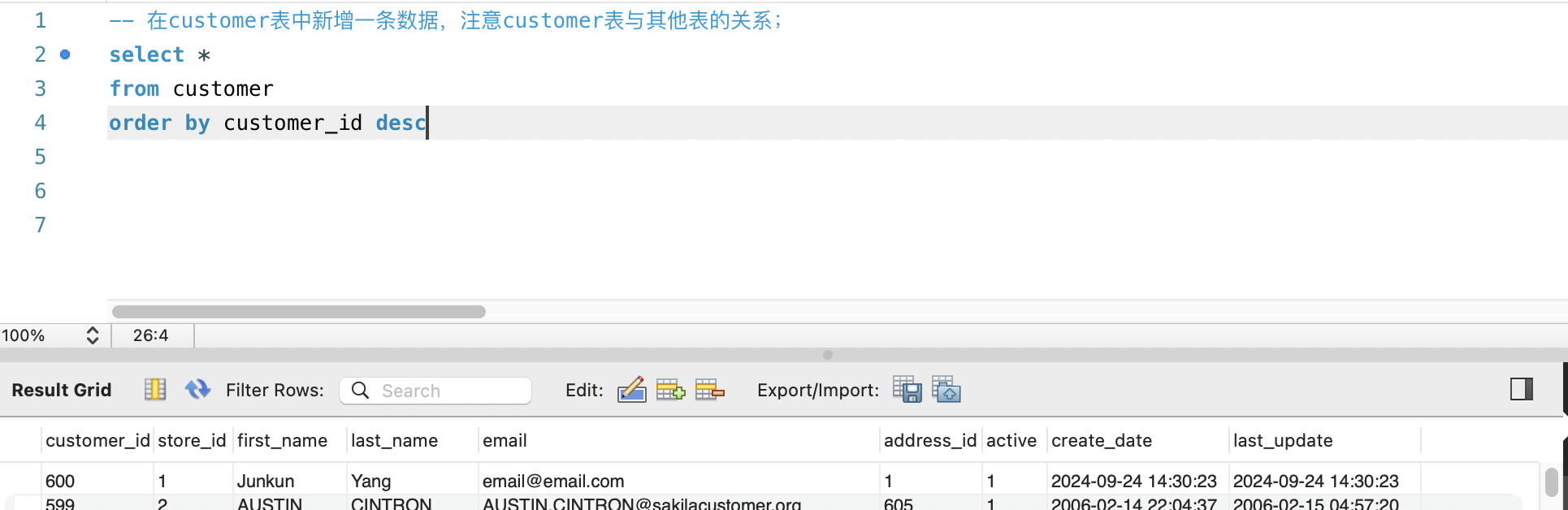
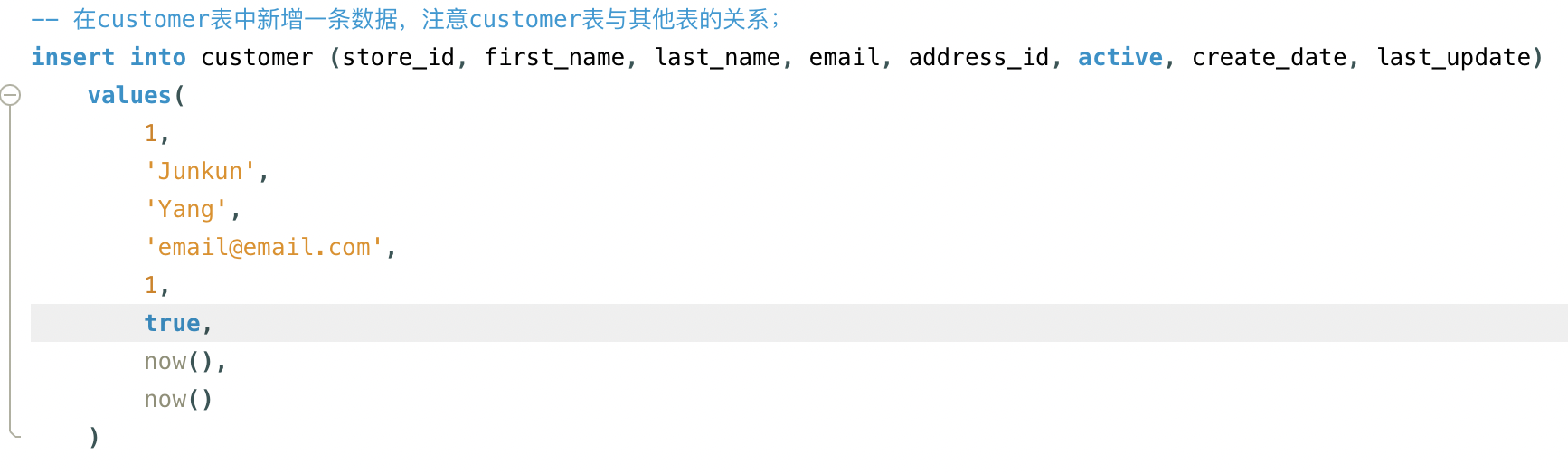
1,

true,

now(),

now()

)



如图查询可知插入成功

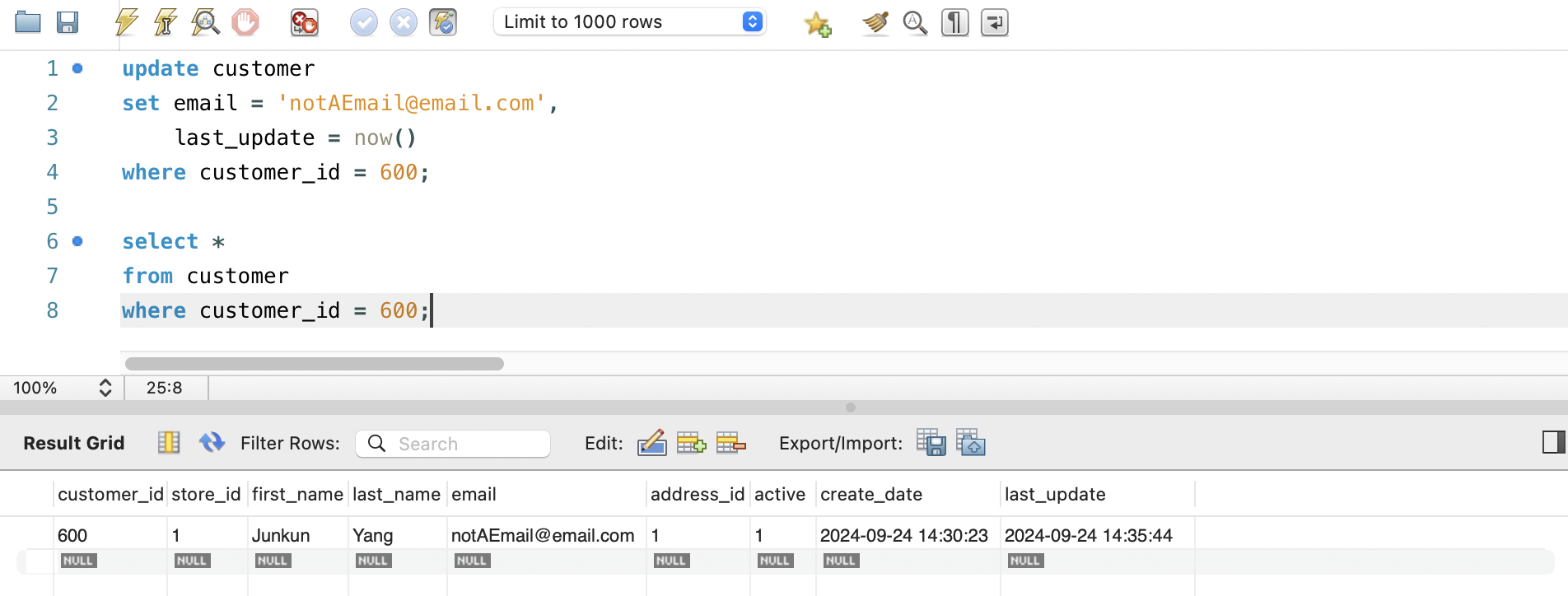
1. 修改刚才在customer表中新增的那条数据；

update customer

set email = 'notAEmail@email.com',

last\_update = now()

where customer\_id = 600;



1. 删除第11步新增的那条数据。

delete from customer

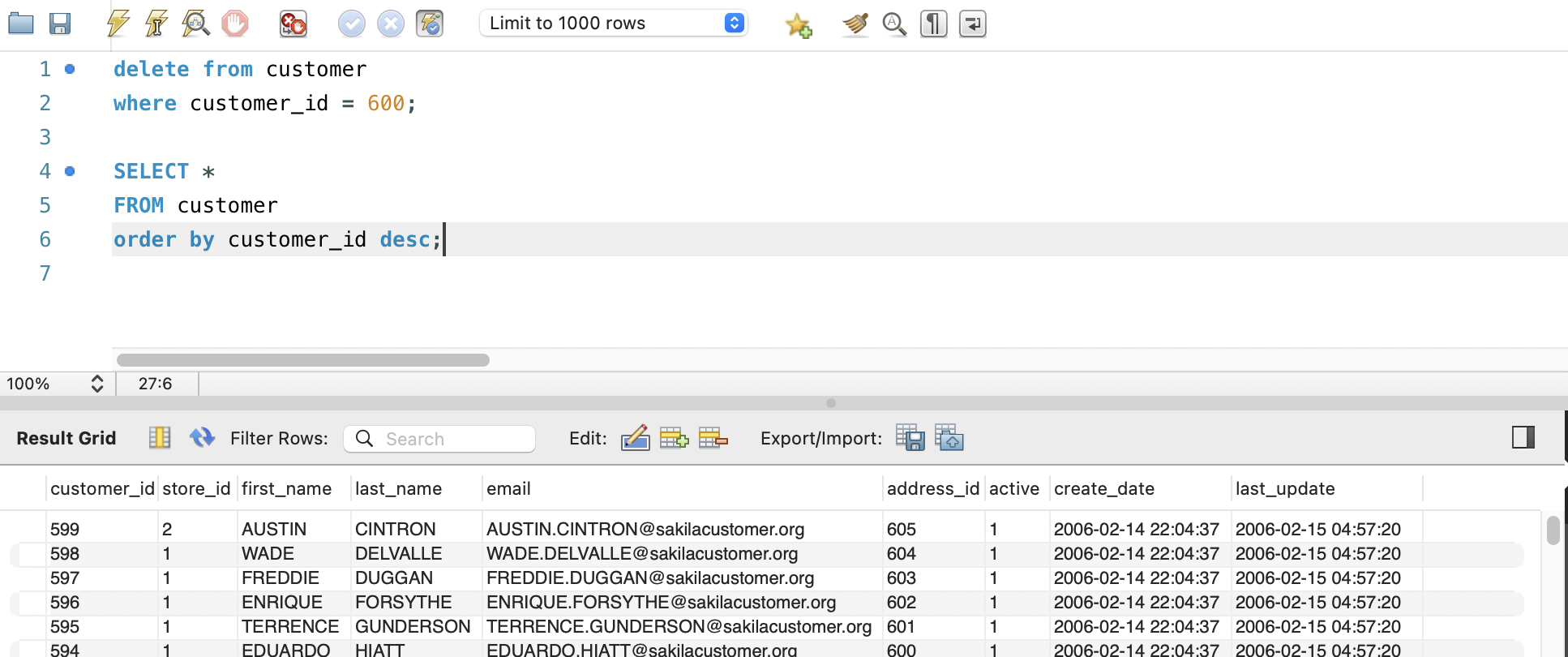
where customer\_id = 600;

SELECT \*

FROM customer

order by customer\_id desc;

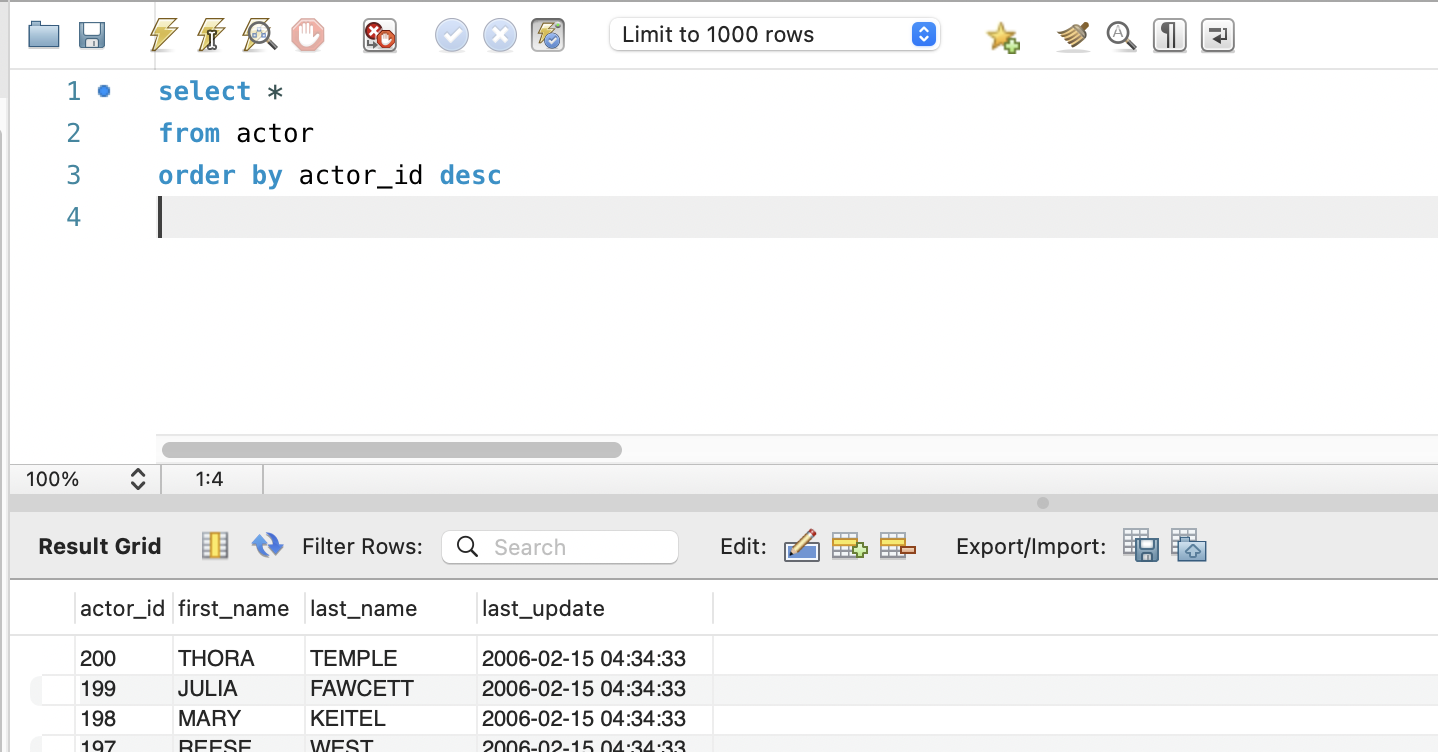
如图，customer\_id = 600的顾客已被删除

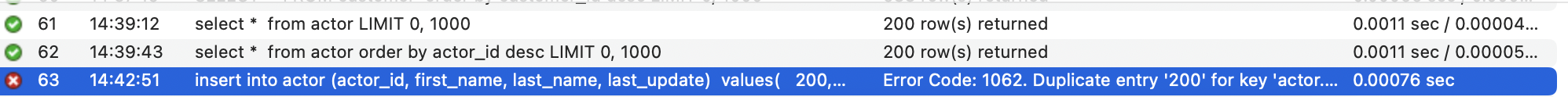
****

* **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。

首先查看到最多200名演员，然后插入actor\_id = 200的数据：





发现报错：Error Code: 1062. Duplicate entry '200' for key 'actor.PRIMARY'

这是因为actor\_id是主键，唯一标识符，不允许重复

1. insert语句还用了一个函数NOW()，是做什么的呢？

NOW() 返回数据库服务器的当前时间戳，通常包含 **日期** 和 **时间**，格式类似于 YYYY-MM-DD HH:MM:SS。

该函数通常用于插入或更新数据库中，需要记录当前时间的字段。