LAB03: View (Solution)

Submission:

- Submit a lab file named "int205_lab03_xxxxxxxxxxx.docx/.pdf" into the LEB2 system. xxxxxxxxxxx = your student id

Due Date & Time:

- Lecturer will inform the LAB03 due date and time in lab class.

What is a View?

A view is a logical table based on a table or another view. A view contains no data of its own but is like a window through which data from tables can be viewed or changed. The tables on which a view is based are called ***base tables***. The view is stored as a SELECT statement in the data dictionary.

EMPLOYEES table

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALA
100	Steven	King	SKING	515.123.4567	17-JUN-87	AD_FRES	240
101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89	AD_VP	170
102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-93	AD_VP	170
103	Alexander	Hunold	AHUNO_D	590.423.4567	03-JAN-90	IT_PROG	90
104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-91	IT_PROG	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-99	IT_PROG	42
124	Kevin	Mourgos	KMOURGOS	650.123.5234	16-NOV-99	ST_MAN	58
141	Trenna	Rajs	TRAJS	650.121.8009	17-OCT-95	ST_CLERK	35
142	Curtis	Davies	CDAVIES	650.121.2994	29-JAN-97	ST_CLERK	31
143	Randall	Matos	RMATOS	650.121.2874	15-MAR-98	ST_CLERK	26
144	Peter	Vargas	PVARGAS	650.121.2004	09-JUL-98	ST_CLERK	25
149	Eleni	Zlotkey	EZLOTKEY	011.44.1344.429018	29-JAN-00	SA_MAN	105
174	Ellen	Abel	EABEL	011.44.1644.429267	11-MAY-96	SA_REP	110
176	Jonathon	Taylor	JTAYLOR	011.44.1644.429265	24-MAR-98	SA_REP	86
178	Kimberely	Grant	KGRANT	011.44.1644.429263	24-MAY-99	SA_REP	70
200	Jennifer	Whalen	JWHALEN	515.123.4444	17-SEP-87	AD_ASST	44
201	Michael	Hartstein	MHARTSTE	515.123.5555	17-FEB-96	MK_MAN	130
202	Pat	Fay	PFAY	603.123.6666	17-AUG-97	MK_REP	60
205	Shelley	Higgins	SHIGGINS	515.123.8080	07-JUN-94	AC_MGR	120
206	William	Gietz	WGIETZ	515.123.8181	07-JUN-94	AC_ACCOUNT	83

20 rows selected.

Syntax for creating a view

```
CREATE[OR REPLACE]
    VIEW view_name [(column_list)]
    AS select_statement ;
```

Updatable Views:

- A simple view is one that:
  - Derives data from only one table
  - Contains no functions or groups of data
  - Can perform DML operations through the view

Non-updatable Views:

- A complex view is one that:
  - Derives data from many tables
  - Contains functions or groups of data
  - Does not always allow DML operations through the view

The Syntax of CREATE VIEW statement:
Documentation: https://dev.mysql.com/doc/refman/8.0/en/create-view.html

Note: The MySQL error code 1064 is a syntax error. This means the reason there's a problem is because MySQL doesn't understand what you're asking it to do.

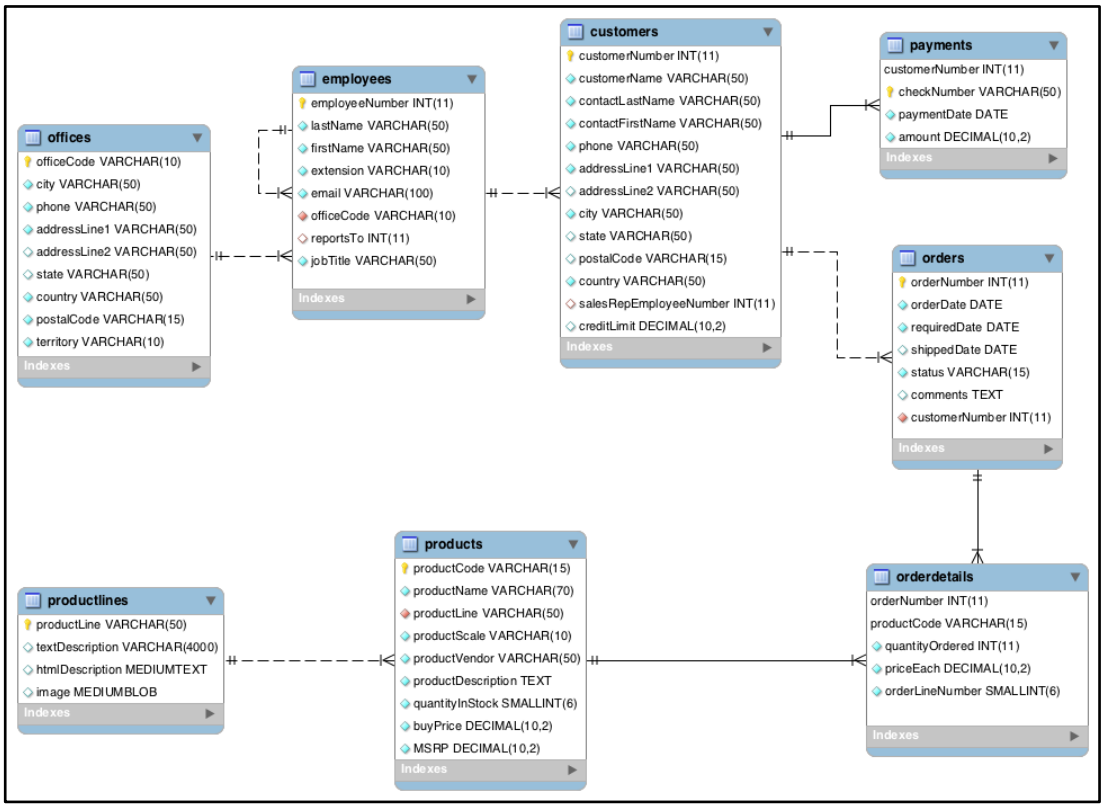
Switch to SQL Editor

- You should specify the classicmodels database before writing SQL statements using the following command:

  USE db_name;

The USE statement tells MySQL to use the named database as the default (current) database for subsequent statements. This statement requires some privilege for the database or some object within it.

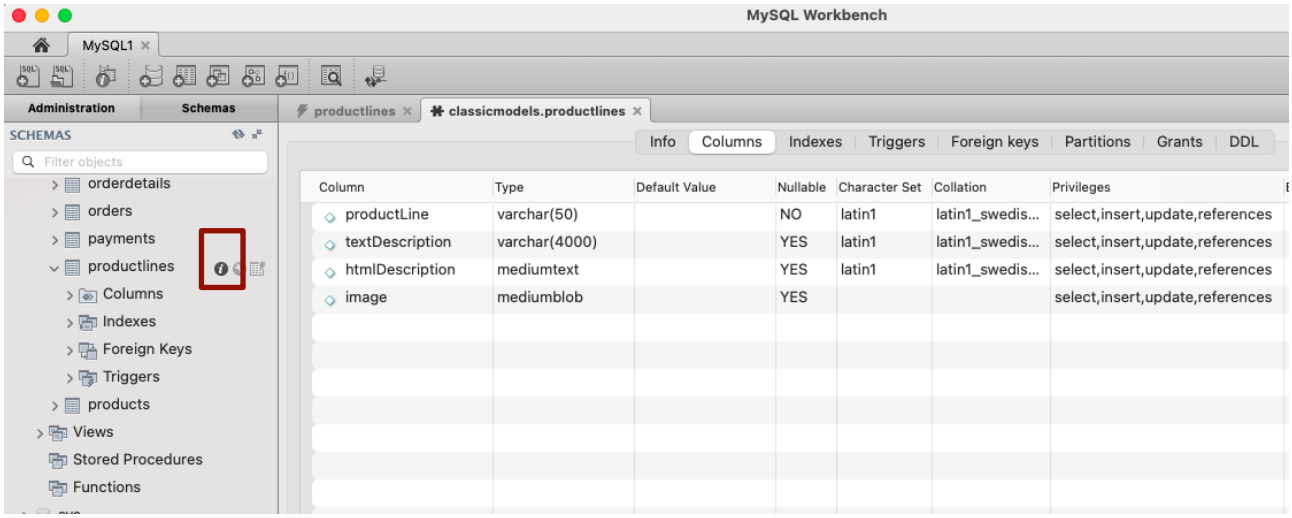
The ER diagram for the classicmodels.



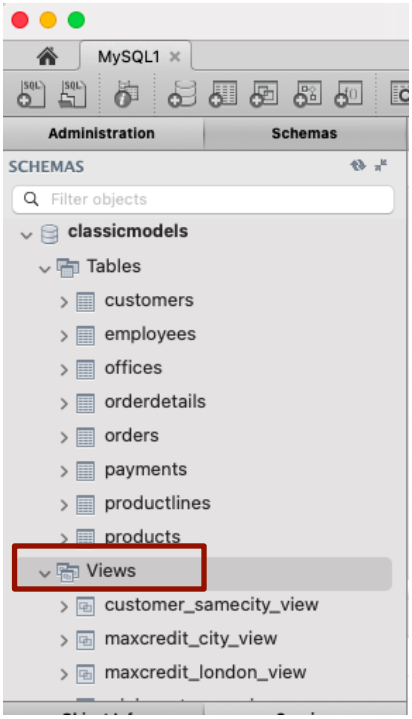
Note: The MSRP is "Manufacturer's suggested retail price" (ราคาขายปลีกแนะนำของผู้ผลิต).

MySQL Workbench:

- You can see details of a table by clicking i button below:



- You can see the existing view by clicking "Views" menu below:

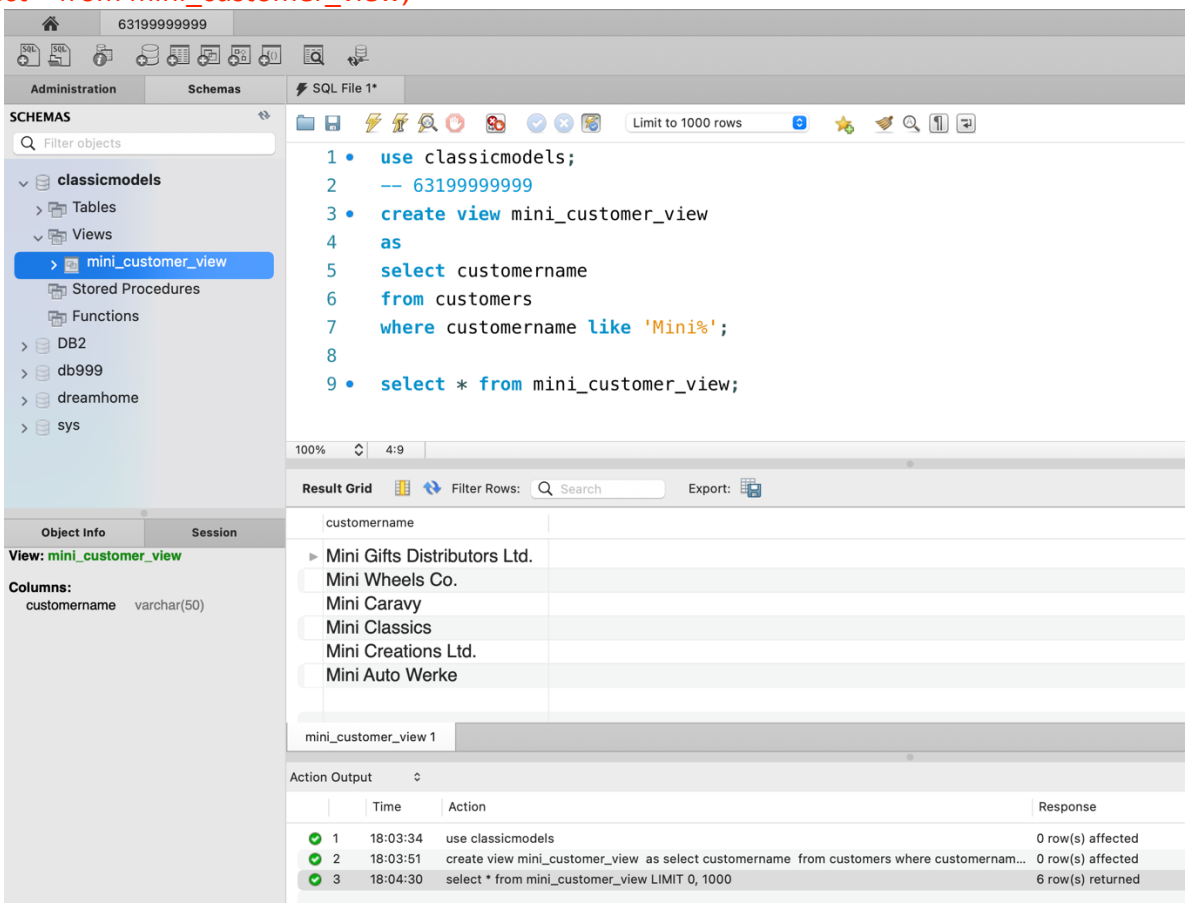


Task 1: Using the "classicmodels" database and write SQL statements to answer the following questions.

use classicmodels;

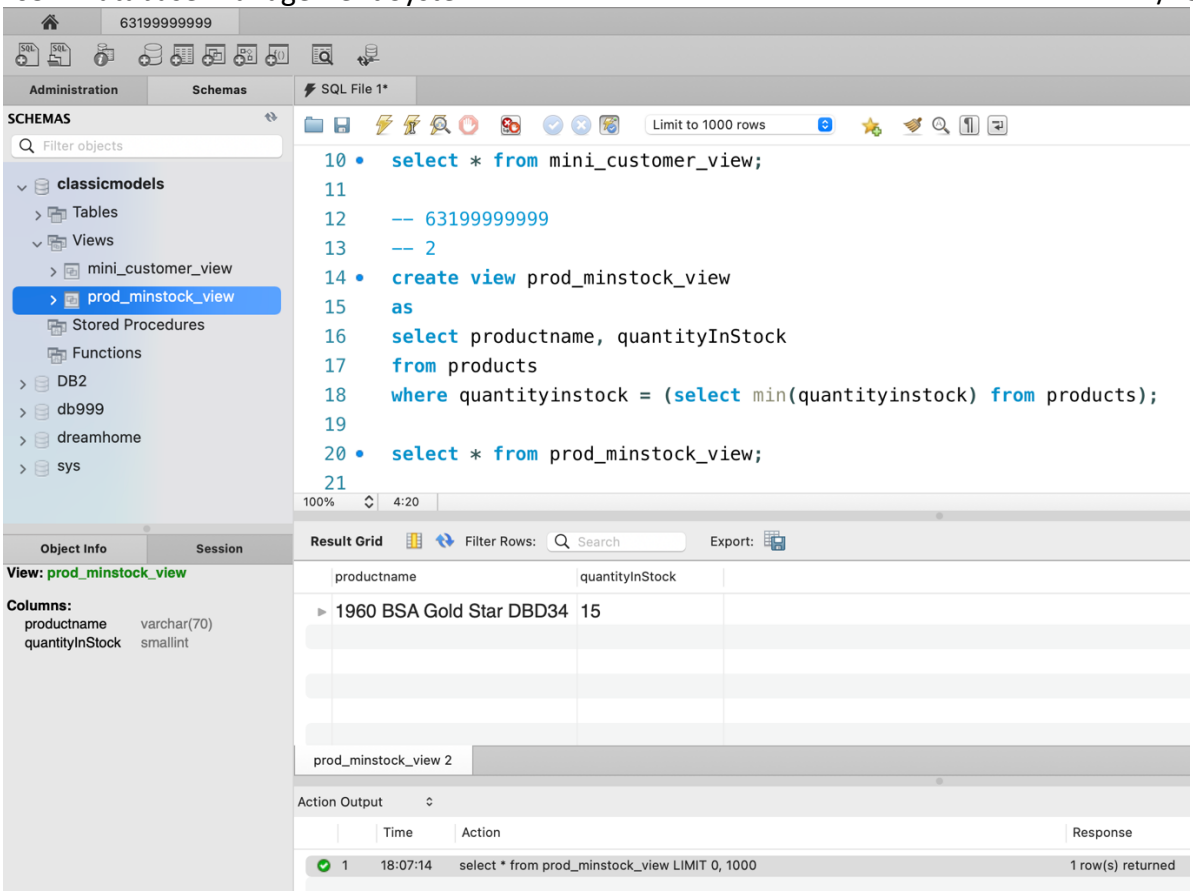
1. Create a view named "mini_customer_view" to display the customer name of all customers whose names start with the word "Mini". Please verify by querying data from this view.

```
-- Write a statement here, the screen of querying data from this view is optional
create view mini_customer_view
as
select customername
from customers
where customername like 'Mini%';

select * from mini_customer_view;
```



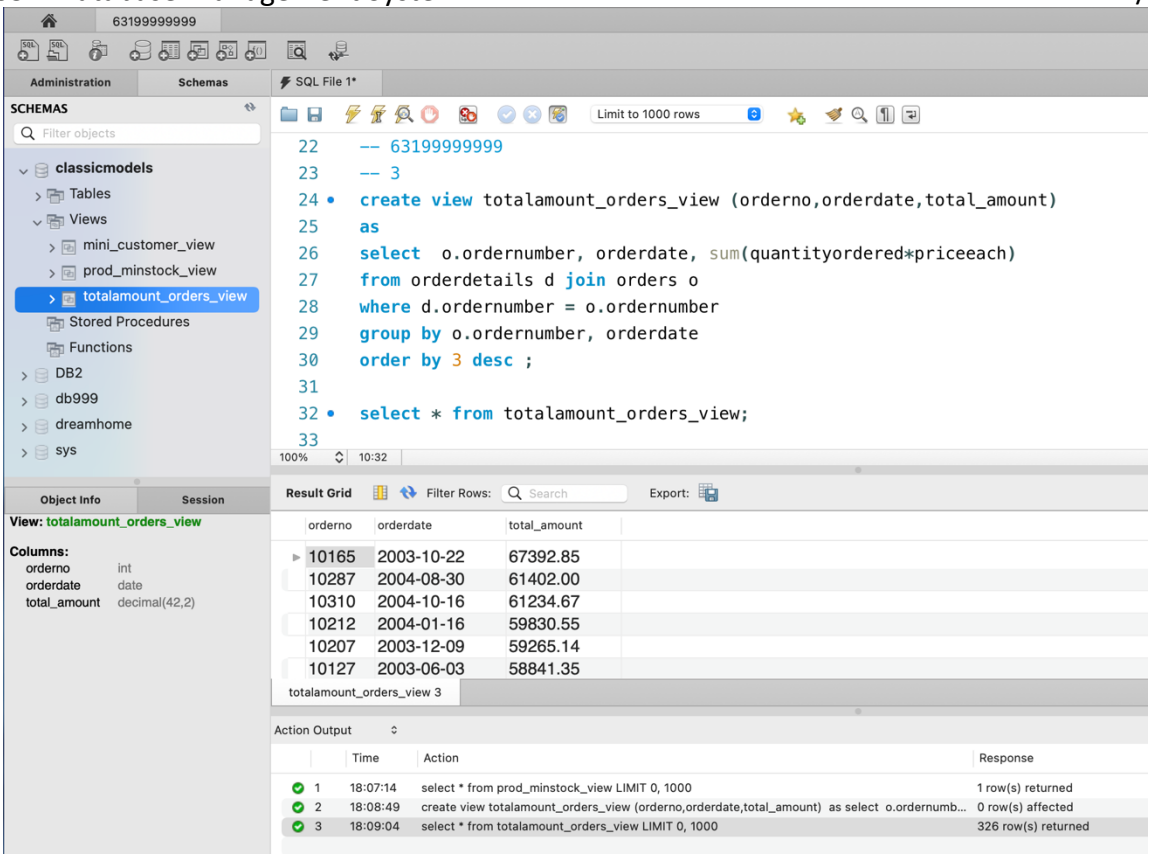
2. Create a view named "prod_minstock_view " to display the product name and quantity in stock of the product that has the minimum quantities in stock. Please verify by querying data from this view.

```
create view prod_minstock_view
as
select productname, quantityInStock
from products
where quantityinstock = (select min(quantityinstock) from products);

select * from prod_minstock_view;
```



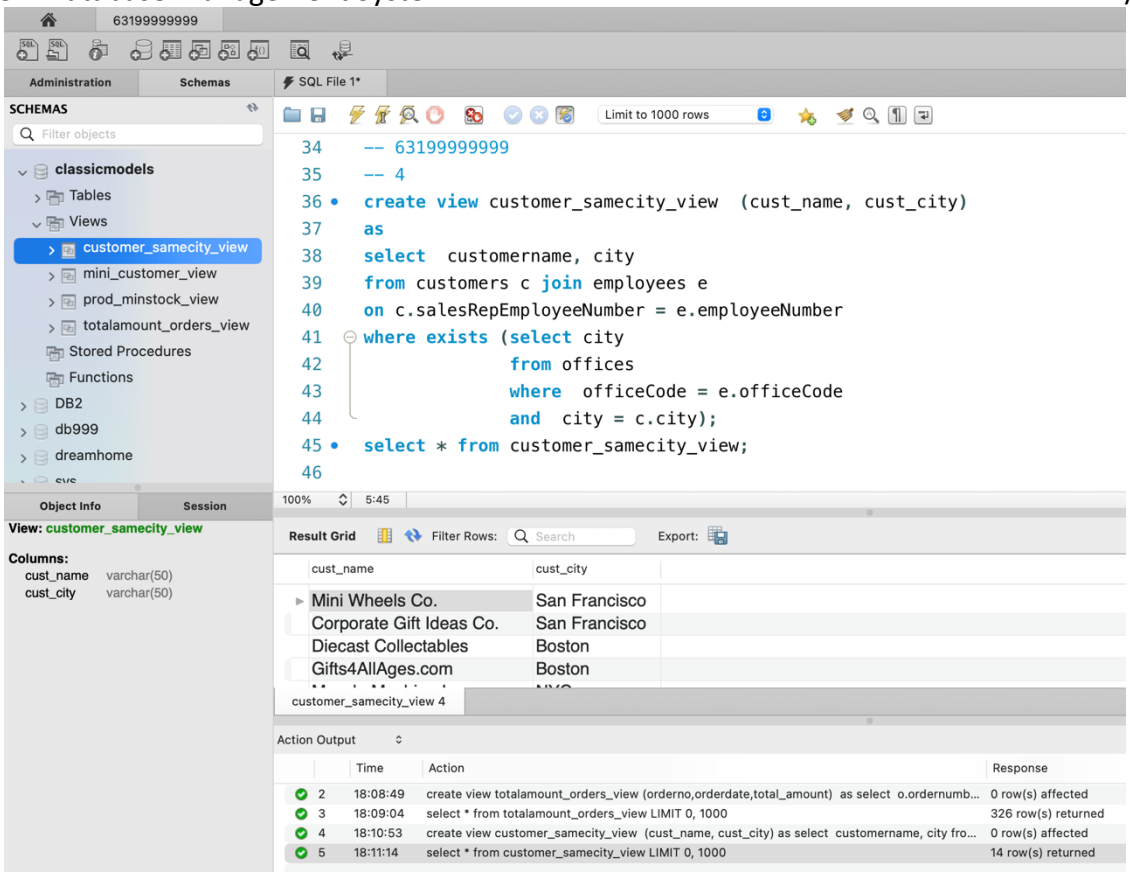
3. Create a view named "totalamount_orders_view" to display the order number, order date and the total amount of sales of all orders and sort the results in descending order by the total amount of sales. Name three columns of the view to orderno, orderdate and total_amount, respectively. Please verify by querying data from this view.

```
-- A1
create view totalamount_orders_view (orderno,orderdate,total_amount)
as
select  o.ordernumber, orderdate, sum(quantityordered*priceeach)
from orderdetails d join orders o
where d.ordernumber = o.ordernumber
group by o.ordernumber, orderdate
order by 3 desc ;
-- A2
create or replace view totalamount_orders_view
as
select  o.ordernumber as orderno, orderdate, sum(quantityordered*priceeach) as total_amount
from orderdetails d join orders o
where d.ordernumber = o.ordernumber
group by o.ordernumber, orderdate
order by total_amount desc ;

select * from totalamount_orders_view;
```



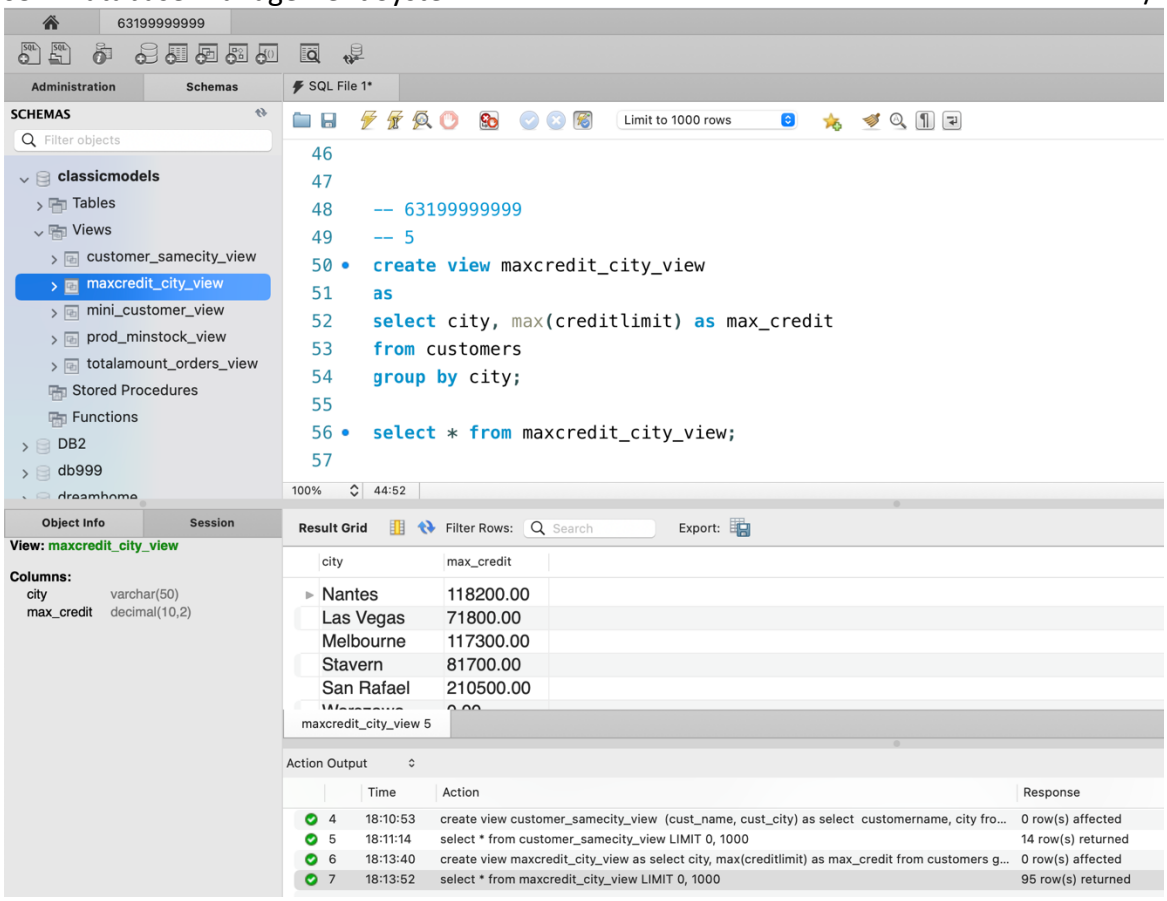
4. Create a view named "customer_samecity_view" to display the customer name and city of all customers who live in the same city of their sales rep employee's office city. Name two view columns to cust_name and cust_city, respectively. Please verify by querying data from this view.

```
-- A1
create view customer_samecity_view  (cust_name, cust_city)
as
select  customername, city
from customers c join employees e
on c.salesRepEmployeeNumber = e.employeeNumber
where exists (select city
                    from offices
                     where  officeCode = e.officeCode
                     and  city = c.city);
-- A2
create or replace view customer_samecity_view
as
select  customername as cust_name, c.city as cust_city
from customers c join employees e
on c.salesRepEmployeeNumber = e.employeeNumber
join offices o
on  o.officeCode = e.officeCode
where o.city = c.city;

select * from customer_samecity_view;
```



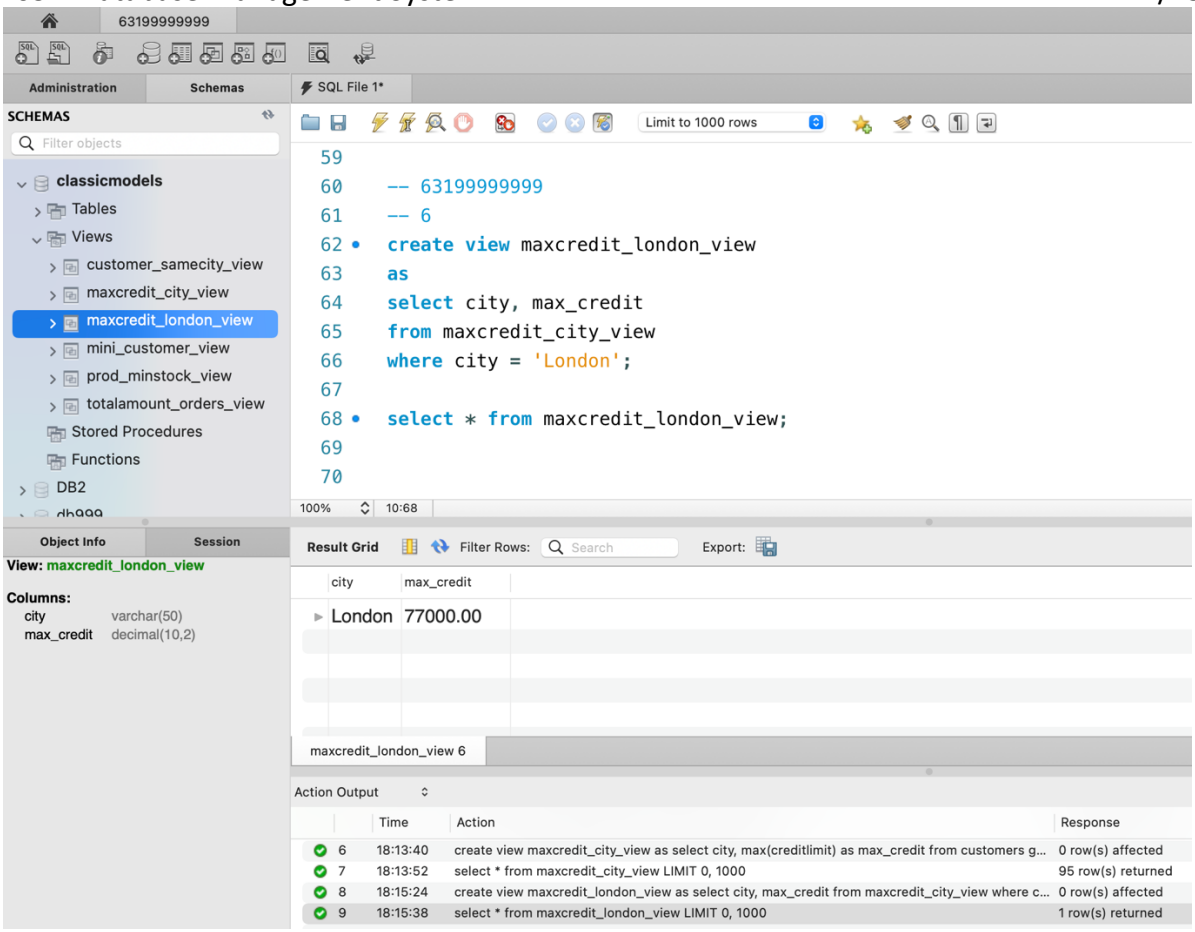
5. Create a view named "maxcredit_city_view" to display the city and the maximum credit limit of all customers in each city. Please verify by querying data from this view.

```
create view maxcredit_city_view
as
select city, max(creditlimit) as max_credit
from customers
group by city;

select * from maxcredit_city_view;
```



6. Create a view named "maxcredit_london_view" to display the city and the maximum credit limit of all customers who live in London city. You should create this view from the "maxcredit_city_view" view in Question 5. Please verify by querying data from this view.

```
-- A1
create view maxcredit_london_view
as
select city, max_credit
from maxcredit_city_view
where city = 'London';
-- A2
create or replace view maxcredit_london_view
as
select  *
from maxcredit_city_view
where city = 'London';

select * from maxcredit_london_view;
```

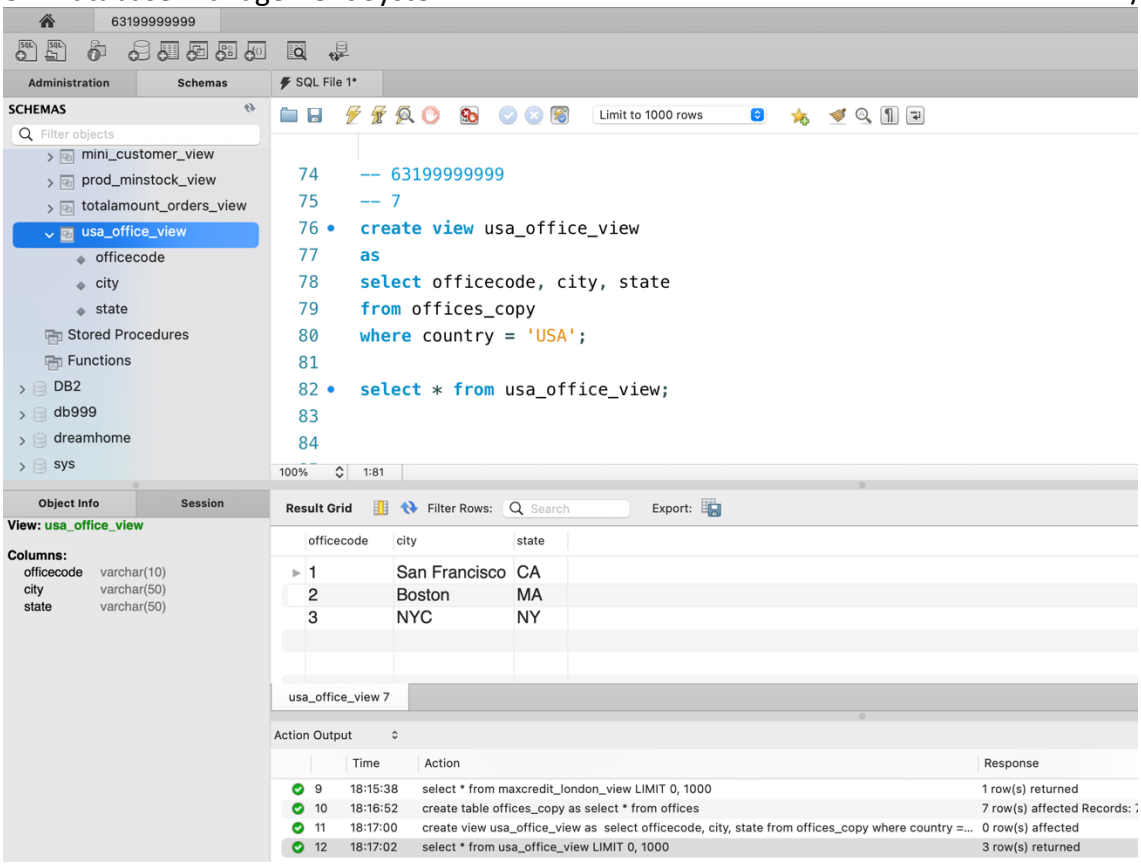


7. Create a table named "offices_copy" with copying the structure and data from the "offices" table using the following commands:

```
create table offices_copy
as select * from offices;
```

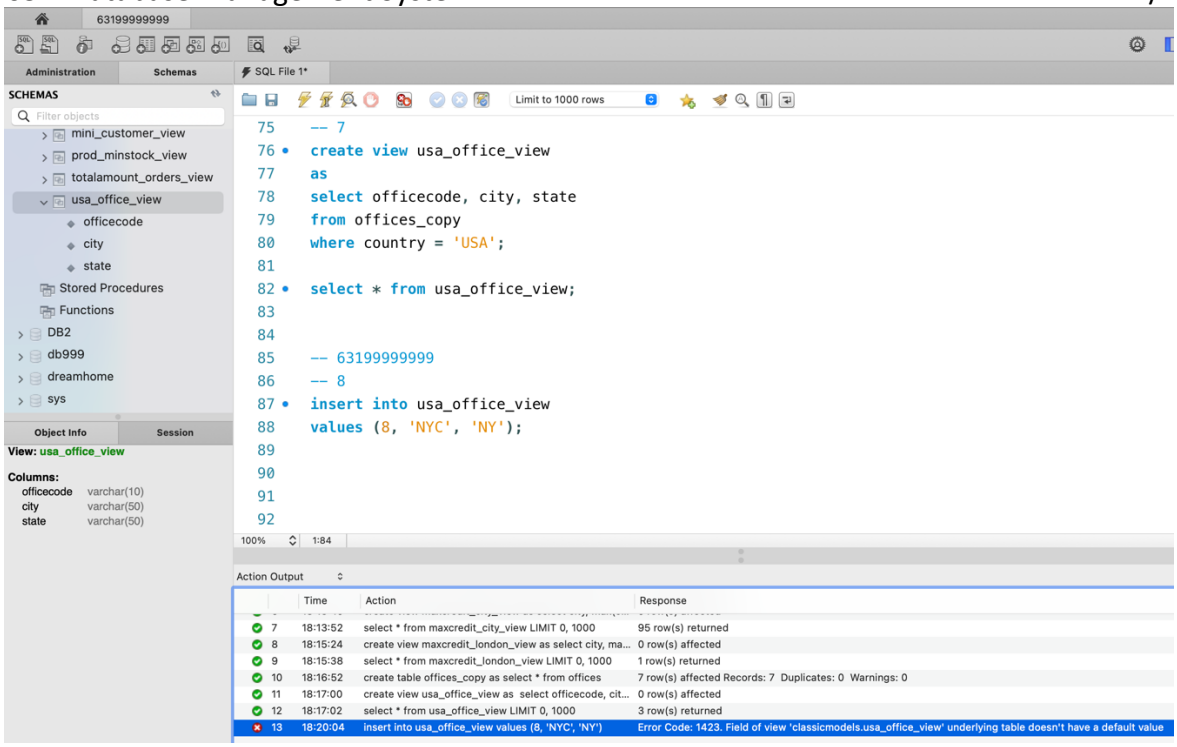
Create a view named "usa_office_view" to display office code, city and state of the country "USA" from the "offices_copy" table. Please verify by querying data from this view.

```
create table offices_copy
as select * from offices;

create view usa_office_view
as
select officecode, city, state
from offices_copy
where country = 'USA';

select * from usa_office_view;
```



8. Try to insert a new row through the "usa_office_view" view created in Question 7. What happens about the data insertion into the "offices_copy" table? Please explain.
Hint: You can create data for a new row by yourself.

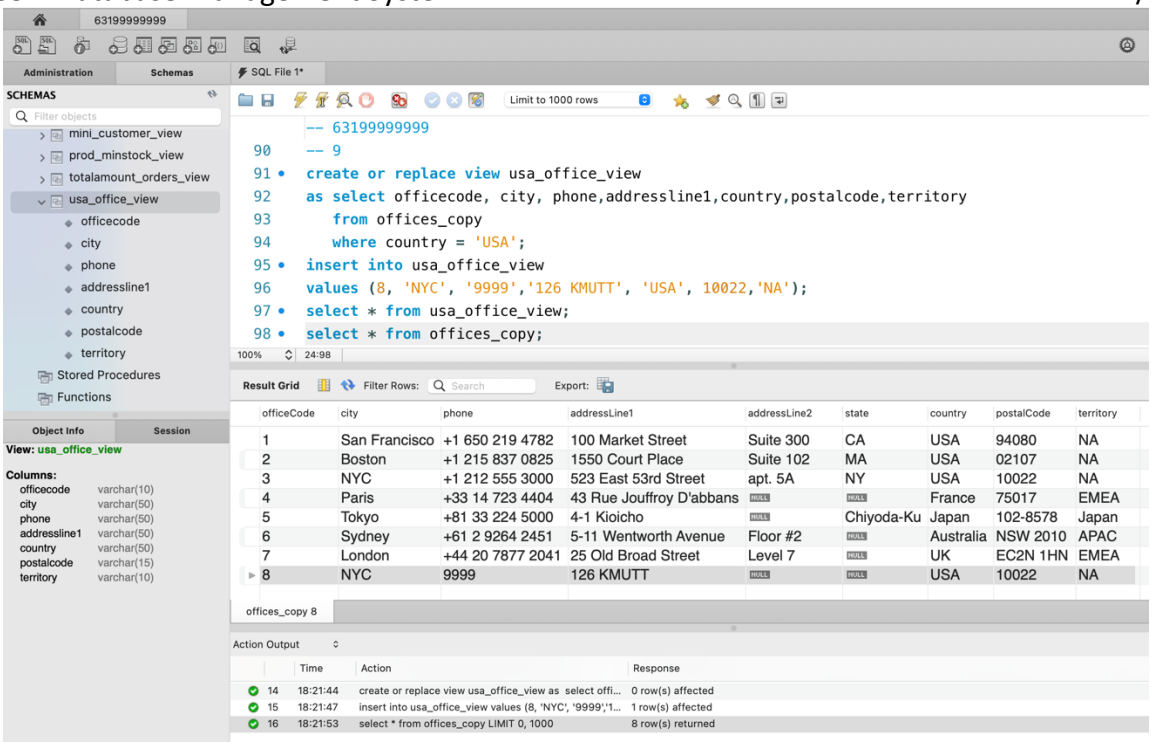
```
insert into usa_office_view
values (8, 'NYC', 'NY');

-- Error Code: 1423. Field of view 'classicmodels.usa_office_view' underlying table doesn't have a default value
-- The NOT NULL columns of the "offices_copy" table were not in the view definition. Therefore, you cannot insert new rows through this view.

select * from usa_office_view;
select * from offices_copy;
```



9. To resolve the problem found in Question 8, Please modify the "usa_office_view" view to ensure that you can insert a new row through this view (an updatable view). Please show the data insertion of the "offices_copy" table.

Hint: You can create data for a new row by yourself.

```
create or replace view usa_office_view
as
select officecode, city, phone,addressline1,country,postalcode,territory
from offices_copy
where country = 'USA';

insert into usa_office_view
values (8, 'NYC', '9999','126 KMUTT', 'USA', 10022,'NA');
commit;

select * from usa_office_view;
select * from offices_copy;
```



10 Please delete both the structure and data of the "offices_copy" table. What happens to an existing view that references the "offices_copy" table? Please explain.

drop table offices_copy;

select * from usa_office_view;

-- Error Code: 1356. View 'classicmodels.usa_office_view' references invalid table(s) or column(s) or function(s) or definer/invoker of view lack rights to use them
-- The “usa_office_view” still exists but it does not work because the base table does not exist.