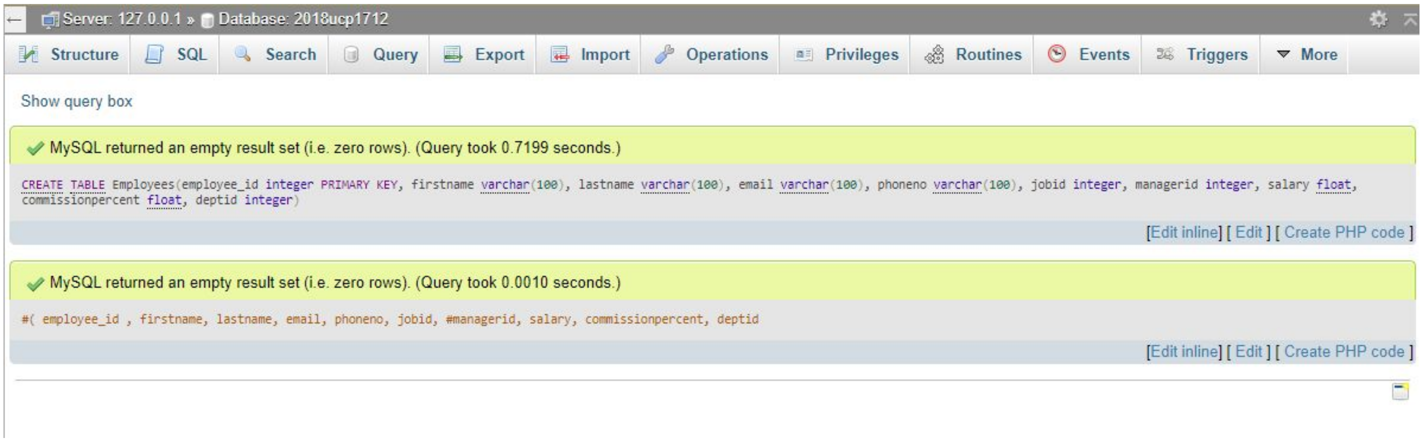
23/11/2020

DBMS Lab Assignment 3

By Abdullah Jamal 2018UCP1712

Schema (Q1 to Q5): Employees (employee_id , firstname, lastname, email, phoneno, jobid, managerid, salary, commissionpercent, deptid);



✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.7199 seconds.)

```
CREATE TABLE Employees(employee_id integer PRIMARY KEY, firstname varchar(100), lastname varchar(100), email varchar(100), phoneno varchar(100), jobid integer, managerid integer, salary float, commissionpercent float, deptid integer)
```

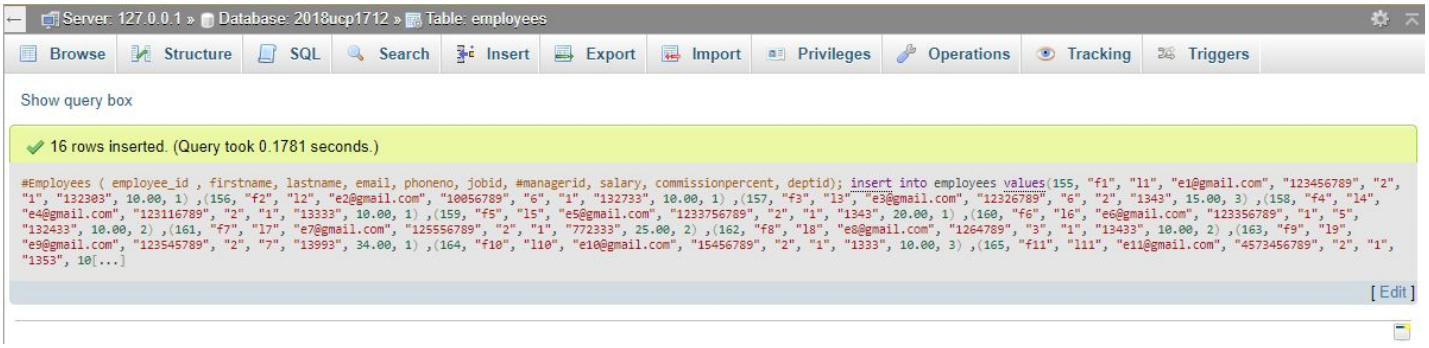
✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0010 seconds.)

```
#( employee_id , firstname, lastname, email, phoneno, jobid, #managerid, salary, commissionpercent, deptid
```

1. Insert data with employee id from 155 to 170.

```
insert into employees values(155, "f1", "l1", "e1@gmail.com", "123456789", "2", "161", "132303", 10.00, 1)
,(156, "f2", "l2", "e2@gmail.com", "10056789", "6", "1", "132733", 10.00, 1)
,(157, "f3", "l3", "e3@gmail.com", "12326789", "6", "2", "1343", 15.00, 3)
,(158, "f4", "l4", "e4@gmail.com", "123116789", "2", "162", "13333", 10.00, 1)
,(159, "f5", "l5", "e5@gmail.com", "1233756789", "2", "1", "1343", 20.00, 1)
,(160, "f6", "l6", "e6@gmail.com", "123356789", "1", "5", "132433", 10.00, 2)
,(161, "f7", "l7", "e7@gmail.com", "125556789", "2", "1", "772333", 25.00, 2)
,(162, "f8", "l8", "e8@gmail.com", "1264789", "3", "1", "13433", 10.00, 2)
,(163, "f9", "l9", "e9@gmail.com", "123545789", "2", "7", "13993", 34.00, 1)
,(164, "f10", "l10", "e10@gmail.com", "15456789", "2", "1", "1333", 10.00, 3)
,(165, "f11", "l11", "e11@gmail.com", "4573456789", "2", "1", "1353", 10.00, 1)
,(166, "f12", "l12", "e12@gmail.com", "12465589", "3", "161", "13243", 67.00, 1)
,(167, "f13", "l13", "e13@gmail.com", "123456756", "2", "1", "16333", 10.00, 2)
,(168, "f14", "l14", "e14@gmail.com", "1234523449", "2", "162", "133433", 10.00, 1)
```

,(169, "f15", "l15", "e15@gmail.com", "123456243", "2", "1", "13243", 10.00, 1)
,(170, "f16", "l16", "e16@gmail.com", "1216243", "4", "3", "13043", 16.00, 4);

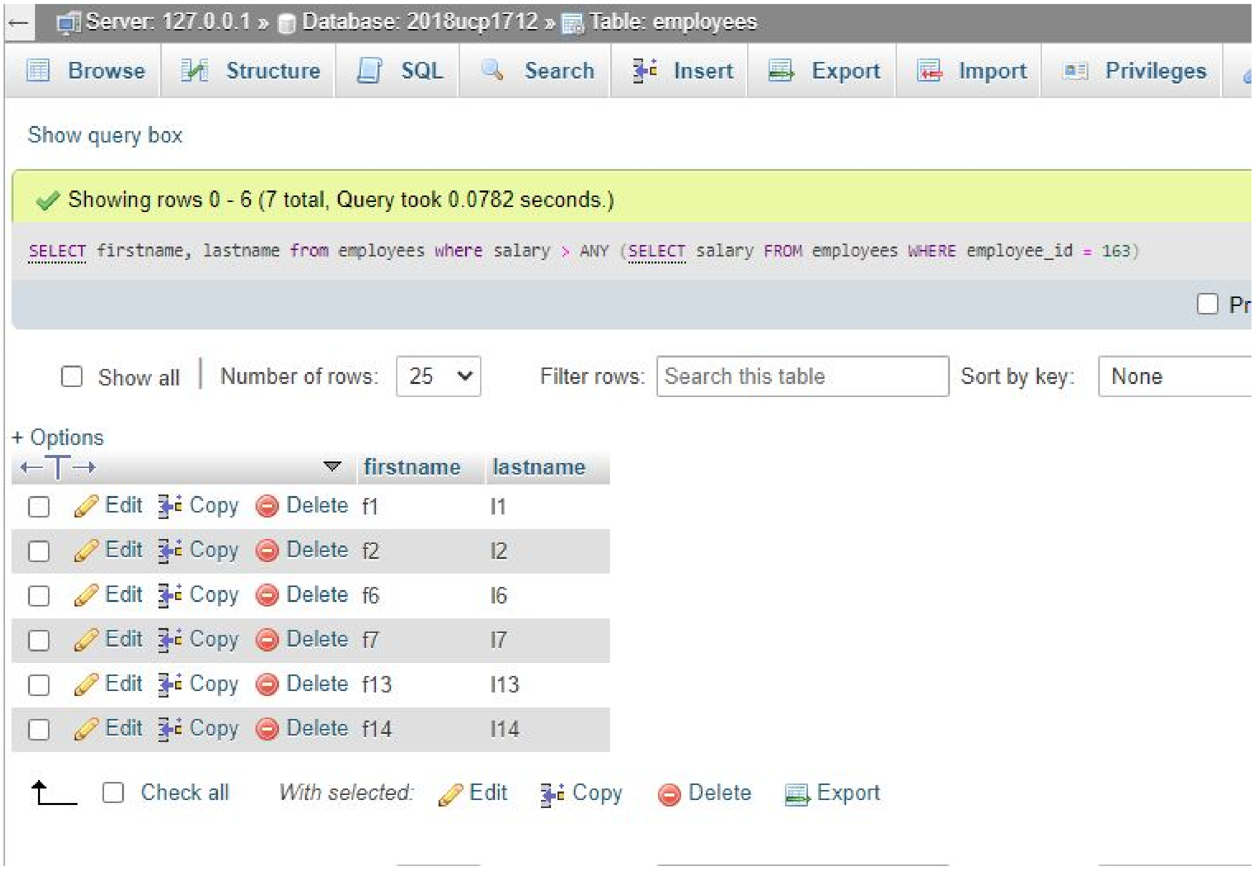


✓ 16 rows inserted. (Query took 0.1781 seconds.)

```
#Employees ( employee_id , firstname, lastname, email, phoneno, jobid, #managerid, salary, commissionpercent, deptid); insert into employees values(155, "f1", "l1", "e1@gmail.com", "123456789", "2",
"1", "132303", 10.00, 1) ,(156, "f2", "l2", "e2@gmail.com", "10056789", "6", "1", "132733", 10.00, 1) ,(157, "f3", "l3", "e3@gmail.com", "12326789", "6", "2", "1343", 15.00, 3) ,(158, "f4", "l4",
"e4@gmail.com", "123116789", "2", "1", "13333", 10.00, 1) ,(159, "f5", "l5", "e5@gmail.com", "1233756789", "2", "1", "1343", 20.00, 1) ,(160, "f6", "l6", "e6@gmail.com", "123356789", "1", "5",
"132433", 10.00, 2) ,(161, "f7", "l7", "e7@gmail.com", "125556789", "2", "1", "772333", 25.00, 2) ,(162, "f8", "l8", "e8@gmail.com", "1264789", "3", "1", "13433", 10.00, 2) ,(163, "f9", "l9",
"e9@gmail.com", "123545789", "2", "7", "13993", 34.00, 1) ,(164, "f10", "l10", "e10@gmail.com", "15456789", "2", "1", "1333", 10.00, 3) ,(165, "f11", "l11", "e11@gmail.com", "4573456789", "2", "1",
"1353", 10[...]
```

2. Write a query to display the name (first name and last name) for those employees who get more salary than the employee whose ID is 163.

SELECT firstname, lastname from employees where salary > ANY (SELECT salary FROM employees WHERE employee_id = 163);



✓ Showing rows 0 - 6 (7 total, Query took 0.0782 seconds.)

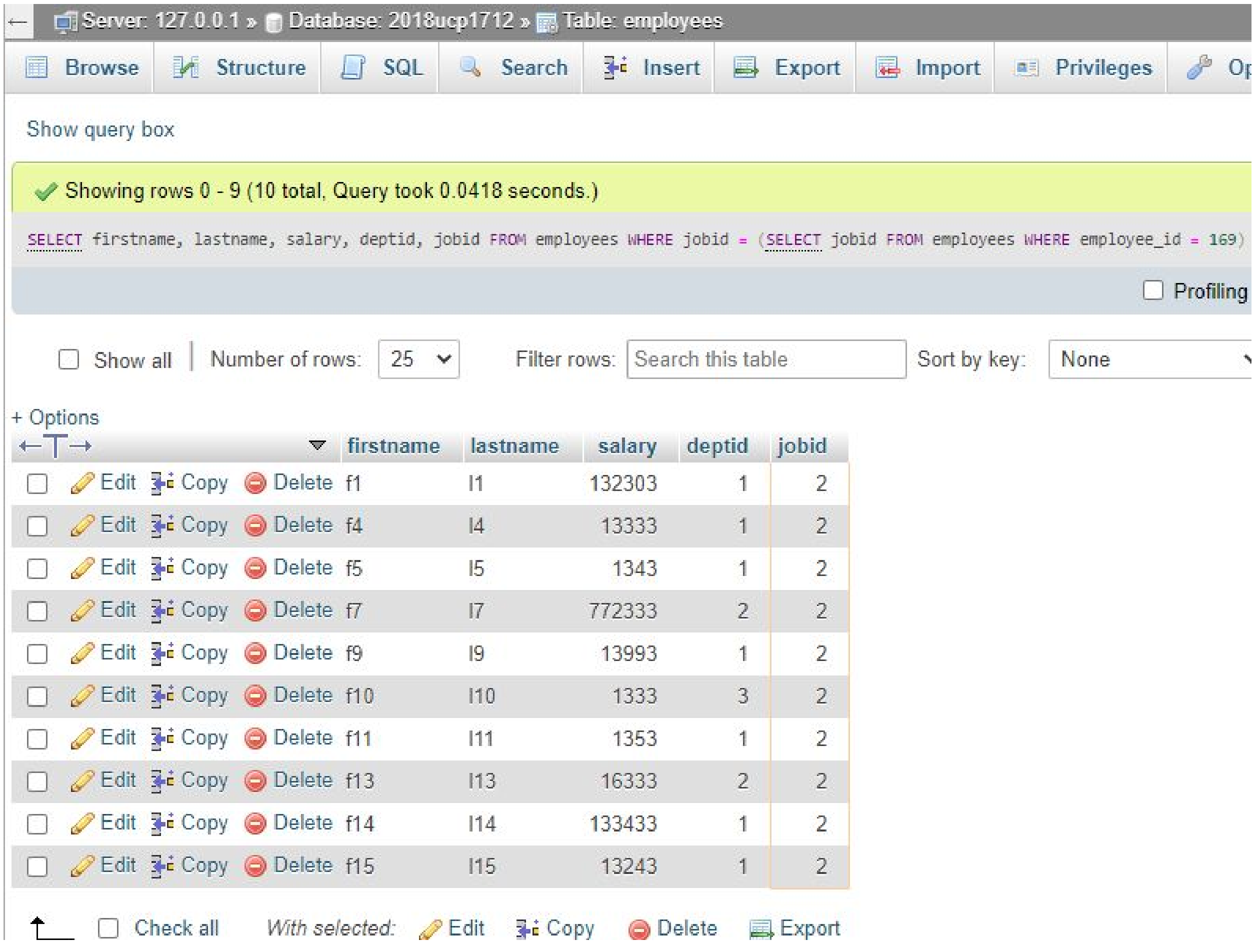
```
SELECT firstname, lastname from employees where salary > ANY (SELECT salary FROM employees WHERE employee_id = 163)
```

□ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	firstname	lastname
□ Edit Copy Delete	f1	l1
□ Edit Copy Delete	f2	l2
□ Edit Copy Delete	f6	l6
□ Edit Copy Delete	f7	l7
□ Edit Copy Delete	f13	l13
□ Edit Copy Delete	f14	l14

3. Write a query to display the name (first name and last name), salary, department id, job id for those employees who work in the same designation as the employee whose id is 169.

SELECT firstname, lastname, salary, deptid, jobid FROM employees WHERE jobid = (SELECT jobid FROM employees WHERE employee_id = 169);



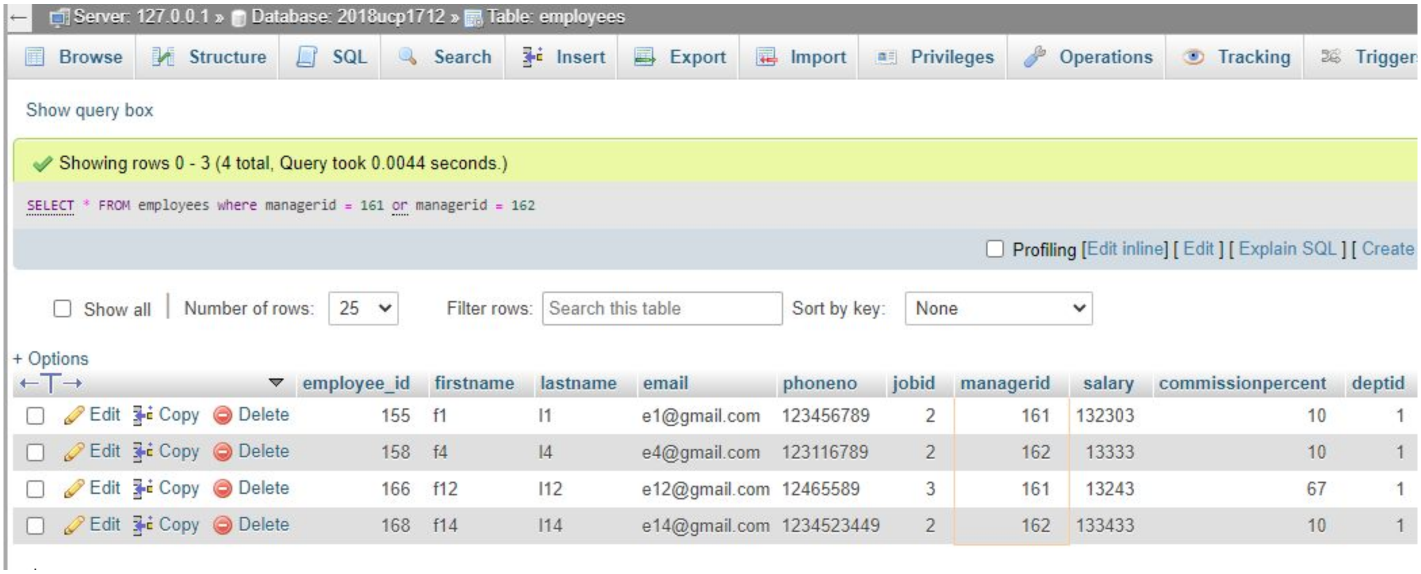
✓ Showing rows 0 - 9 (10 total, Query took 0.0418 seconds.)

```
SELECT firstname, lastname, salary, deptid, jobid FROM employees WHERE jobid = (SELECT jobid FROM employees WHERE employee_id = 169)
```

	firstname	lastname	salary	deptid	jobid
□ Edit Copy Delete	f1	l1	132303	1	2
□ Edit Copy Delete	f4	l4	13333	1	2
□ Edit Copy Delete	f5	l5	1343	1	2
□ Edit Copy Delete	f7	l7	772333	2	2
□ Edit Copy Delete	f9	l9	13993	1	2
□ Edit Copy Delete	f10	l10	1333	3	2
□ Edit Copy Delete	f11	l11	1353	1	2
□ Edit Copy Delete	f13	l13	16333	2	2
□ Edit Copy Delete	f14	l14	133433	1	2
□ Edit Copy Delete	f15	l15	13243	1	2

4. Write a query to display all the information of an employee whose reporting person id is 161 and 162 respectively.

SELECT * FROM employees where managerid = 161 or managerid = 162;



✓ Showing rows 0 - 3 (4 total, Query took 0.0044 seconds.)

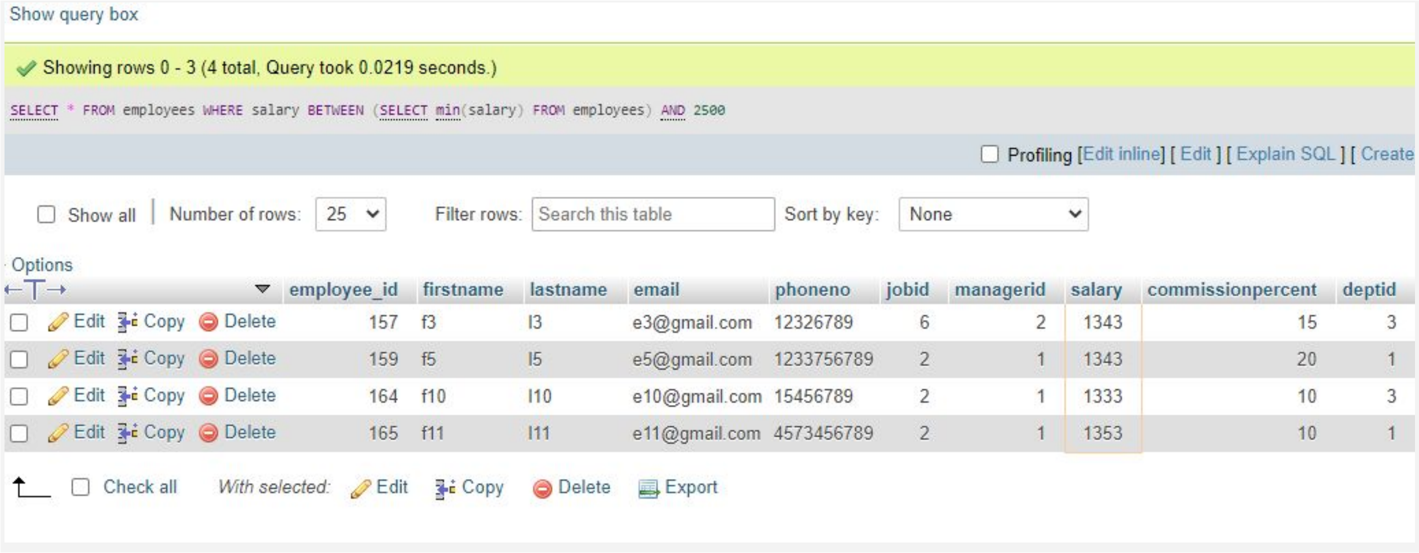
```
SELECT * FROM employees where managerid = 161 or managerid = 162
```

□ Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

	employee_id	firstname	lastname	email	phoneno	jobid	managerid	salary	commissionpercent	deptid
□ Edit Copy Delete	155	f1	l1	e1@gmail.com	123456789	2	161	132303	10	1
□ Edit Copy Delete	158	f4	l4	e4@gmail.com	123116789	2	162	13333	10	1
□ Edit Copy Delete	166	f12	l12	e12@gmail.com	12465589	3	161	13243	67	1
□ Edit Copy Delete	168	f14	l14	e14@gmail.com	1234523449	2	162	133433	10	1

5. Write a query to display all the information of the employees whose salary is within the range of smallest salary and 2500.

SELECT * FROM employees WHERE salary BETWEEN (SELECT min(salary) FROM employees) AND 2500;



✓ Showing rows 0 - 3 (4 total, Query took 0.0219 seconds.)

```
SELECT * FROM employees WHERE salary BETWEEN (SELECT min(salary) FROM employees) AND 2500
```

□ Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

	employee_id	firstname	lastname	email	phoneno	jobid	managerid	salary	commissionpercent	deptid
□ Edit Copy Delete	157	f3	l3	e3@gmail.com	12326789	6	2	1343	15	3
□ Edit Copy Delete	159	f5	l5	e5@gmail.com	1233756789	2	1	1343	20	1
□ Edit Copy Delete	164	f10	l10	e10@gmail.com	15456789	2	1	1333	10	3
□ Edit Copy Delete	165	f11	l11	e11@gmail.com	4573456789	2	1	1353	10	1

Schema (Q6 to Q10):

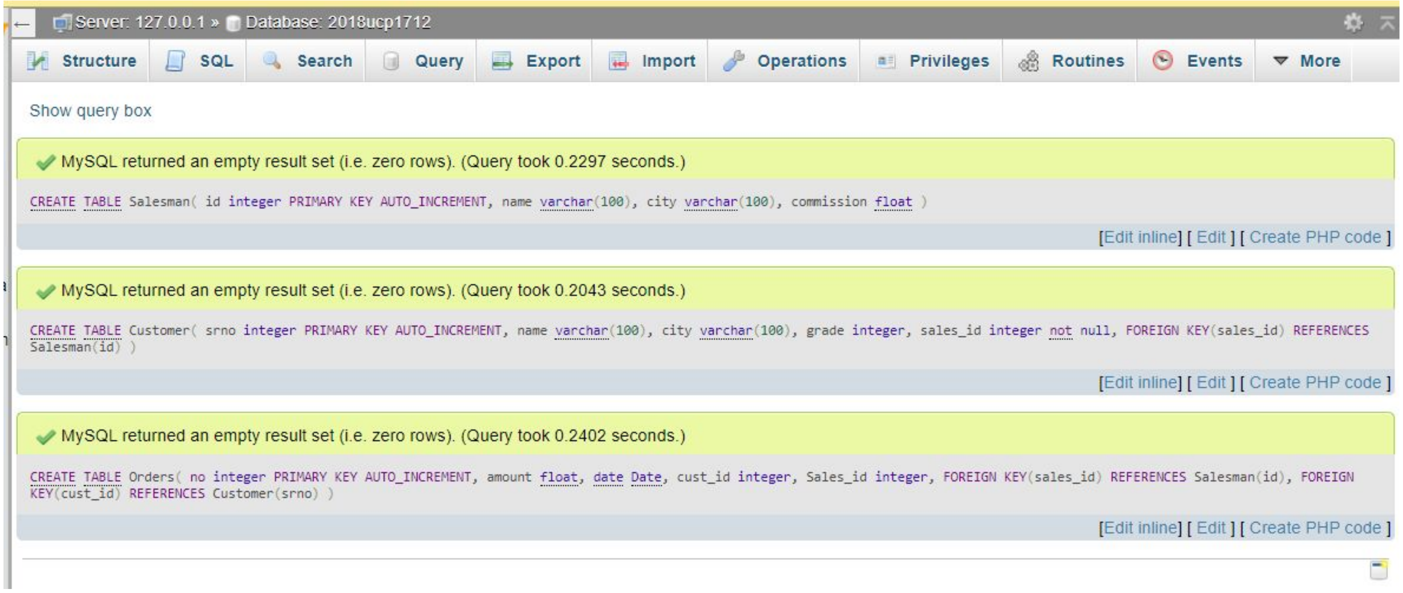
Salesman (id,name,city,commission);
Customer(srno,name,city,grade,sales_id);

Orders(no,amount,date,cust_id,sales_id);

```
CREATE TABLE Salesman( id integer PRIMARY KEY
AUTO_INCREMENT,
                     name varchar(100),
                     city varchar(100),
                     commission float
                     );

CREATE TABLE Customer( srno integer PRIMARY KEY
AUTO_INCREMENT,
                     name varchar(100),
                     city varchar(100),
                     grade integer,
                     sales_id integer not null,
                     FOREIGN KEY(sales_id) REFERENCES Salesman(id)
                    );

CREATE TABLE Orders( no integer PRIMARY KEY AUTO_INCREMENT,
                   amount float,
                   date Date,
                   cust_id integer,
                   Sales_id integer,
                   FOREIGN KEY(sales_id) REFERENCES Salesman(id),
                   FOREIGN KEY(cust_id) REFERENCES Customer(srno)
                   );
```



MySQL returned an empty result set (i.e. zero rows). (Query took 0.2297 seconds.)

```
CREATE TABLE Salesman( id integer PRIMARY KEY AUTO_INCREMENT, name varchar(100), city varchar(100), commission float )
```

MySQL returned an empty result set (i.e. zero rows). (Query took 0.2043 seconds.)

```
CREATE TABLE Customer( srno integer PRIMARY KEY AUTO_INCREMENT, name varchar(100), city varchar(100), grade integer, sales_id integer not null, FOREIGN KEY(sales_id) REFERENCES
Salesman(id) )
```

MySQL returned an empty result set (i.e. zero rows). (Query took 0.2402 seconds.)

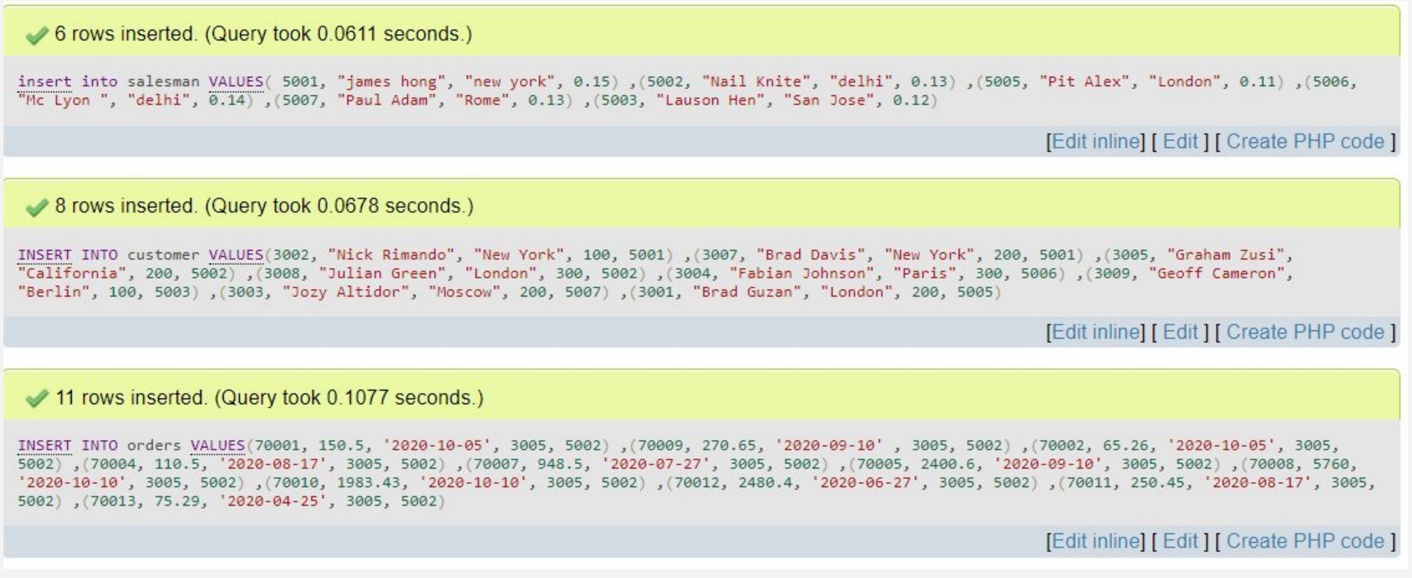
```
CREATE TABLE Orders( no integer PRIMARY KEY AUTO_INCREMENT, amount float, date Date, cust_id integer, Sales_id integer, FOREIGN KEY(sales_id) REFERENCES Salesman(id), FOREIGN
KEY(cust_id) REFERENCES Customer(srno) )
```

```
insert into salesman VALUES( 5001, "james hong", "new york", 0.15)
                                            ,(5002, "Nail Knite", "delhi", 0.13)
                   ,(5005, "Pit Alex", "London", 0.11)
                                            ,(5006, "Mc Lyon ", "delhi", 0.14)
                   ,(5007, "Paul Adam", "Rome", 0.13)
                   ,(5003, "Lauson Hen", "San Jose", 0.12);


INSERT INTO customer VALUES(3002, "Nick Rimando", "New York", 100,
5001)
                                            ,(3007, "Brad Davis", "New York",
200, 5001)
                   ,(3005, "Graham Zusi", "California", 200, 5002)
                   ,(3008, "Julian Green", "London", 300, 5002)
                   ,(3004, "Fabian Johnson", "Paris", 300, 5006)
                   ,(3009, "Geoff Cameron", "Berlin", 100, 5003)
                   ,(3003, "Jozy Altidor", "Moscow", 200, 5007)
                   ,(3001, "Brad Guzan", "London", 200, 5005);

INSERT INTO orders VALUES(70001, 150.5, '2020-10-05', 3005, 5001)
                                      ,(70009, 270.65, '2020-10-31' , 3001,
5002)
```

```
,(70002, 65.26, '2020-10-05', 3003, 5006)
,(70004, 110.5, '2020-08-17', 3005, 5006)
,(70007, 948.5, '2020-07-27', 3003, 5002)
,(70005, 2400.6, '2020-09-10', 3003, 5007)
,(70008, 5760, '2020-10-08', 3005, 5003)
,(70010, 1983.43, '2020-10-10', 3004, 5003)
,(70012, 2480.4, '2020-06-30', 3008, 5003)
,(70011, 250.45, '2020-08-17', 3008, 5002)
,(70013, 75.29, '2020-04-25', 3008, 5007);
```



✓ 6 rows inserted. (Query took 0.0611 seconds.)

```
insert into salesman VALUES( 5001, "james hong", "new york", 0.15) ,(5002, "Nail Knite", "delhi", 0.13) ,(5005, "Pit Alex", "London", 0.11) ,(5006,
"Mc Lyon ", "delhi", 0.14) ,(5007, "Paul Adam", "Rome", 0.13) ,(5003, "Lauson Hen", "San Jose", 0.12)
```

✓ 8 rows inserted. (Query took 0.0678 seconds.)

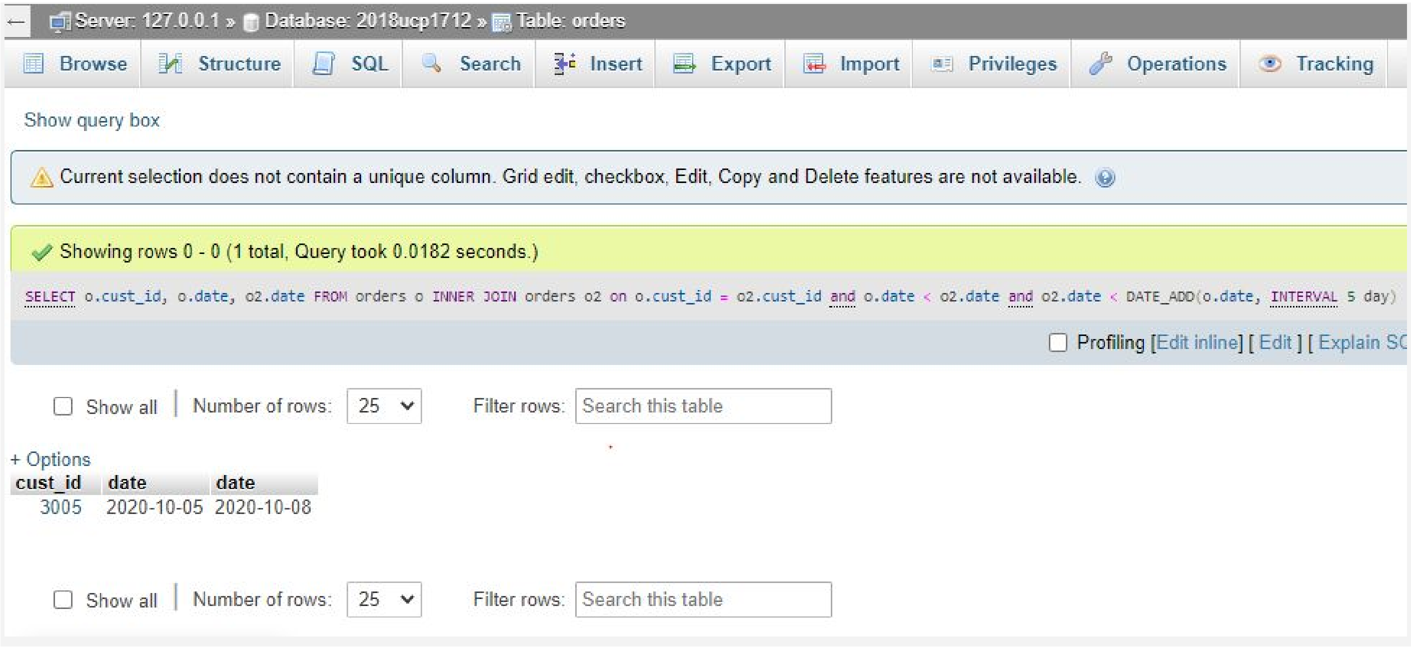
```
INSERT INTO customer VALUES(3002, "Nick Rimando", "New York", 100, 5001) ,(3007, "Brad Davis", "New York", 200, 5001) ,(3005, "Graham Zusi",
"California", 200, 5002) ,(3008, "Julian Green", "London", 300, 5002) ,(3004, "Fabian Johnson", "Paris", 300, 5006) ,(3009, "Geoff Cameron",
"Berlin", 100, 5003) ,(3003, "Jozy Altidor", "Moscow", 200, 5007) ,(3001, "Brad Guzan", "London", 200, 5005)
```

✓ 11 rows inserted. (Query took 0.1077 seconds.)

```
INSERT INTO orders VALUES(70001, 150.5, '2020-10-05', 3005, 5002) ,(70009, 270.65, '2020-09-10' , 3005, 5002) ,(70002, 65.26, '2020-10-05', 3005,
5002) ,(70004, 110.5, '2020-08-17', 3005, 5002) ,(70007, 948.5, '2020-07-27', 3005, 5002) ,(70005, 2400.6, '2020-09-10', 3005, 5002) ,(70008, 5760,
'2020-10-10', 3005, 5002) ,(70010, 1983.43, '2020-10-10', 3005, 5002) ,(70012, 2480.4, '2020-06-27', 3005, 5002) ,(70011, 250.45, '2020-08-17', 3005,
5002) ,(70013, 75.29, '2020-04-25', 3005, 5002)
```

6. List the customers with multiple orders in a 5 day period.

```
SELECT o.cust_id, o.date, o2.date
FROM orders o
INNER JOIN orders o2
on o.cust_id = o2.cust_id
and o.date < o2.date
and o2.date < DATE_ADD(o.date, INTERVAL 5 day);
```


Server: 127.0.0.1 » Database: 2018ucp1712 » Table: orders
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Tracking
Show query box

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 0 (1 total, Query took 0.0182 seconds.)

```
SELECT o.cust_id, o.date, o2.date FROM orders o INNER JOIN orders o2 on o.cust_id = o2.cust_id and o.date < o2.date and o2.date < DATE_ADD(o.date, INTERVAL 5 day)
```

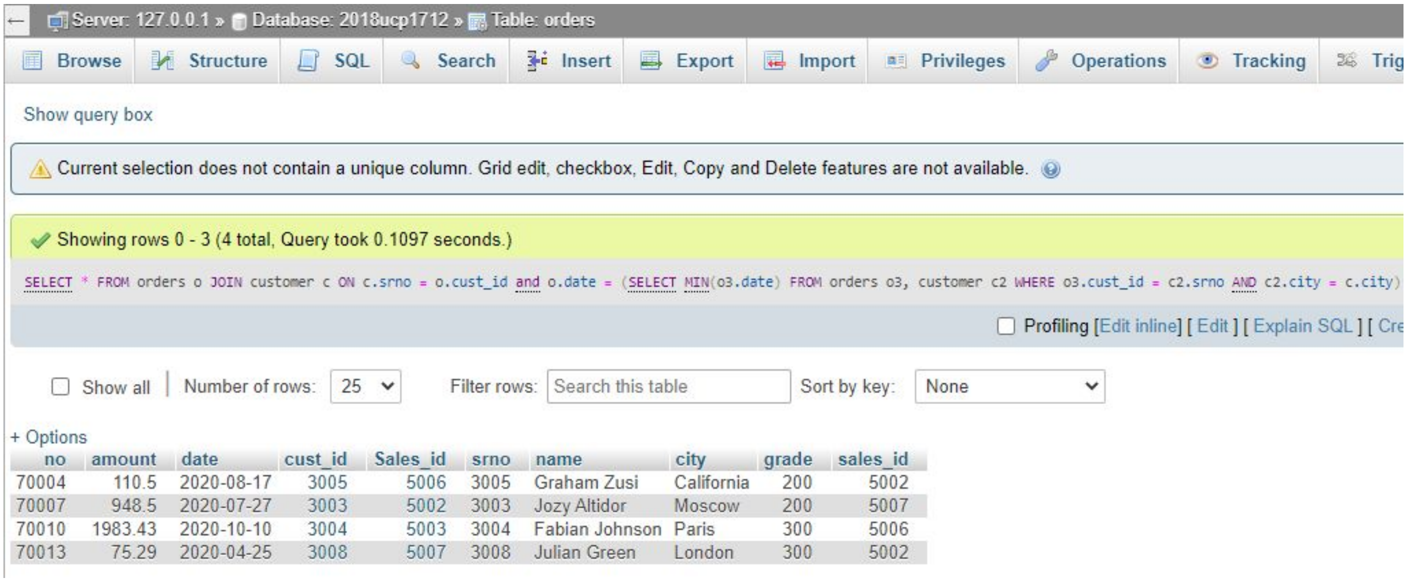
□ Profiling [Edit inline] [Edit] [Explain SC
□ Show all | Number of rows: 25 Filter rows: Search this table
+ Options

cust_id	date	date
3005	2020-10-05	2020-10-08

□ Show all | Number of rows: 25 Filter rows: Search this table

7. List the first order from each city.

```
SELECT * FROM orders o
JOIN customer c
ON c.srno = o.cust_id
and o.date = (SELECT MIN(o3.date) FROM orders o3, customer c2
WHERE o3.cust_id = c2.srno AND c2.city = c.city);
```


Server: 127.0.0.1 » Database: 2018ucp1712 » Table: orders
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Tracking
Trig
Show query box

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 3 (4 total, Query took 0.1097 seconds.)

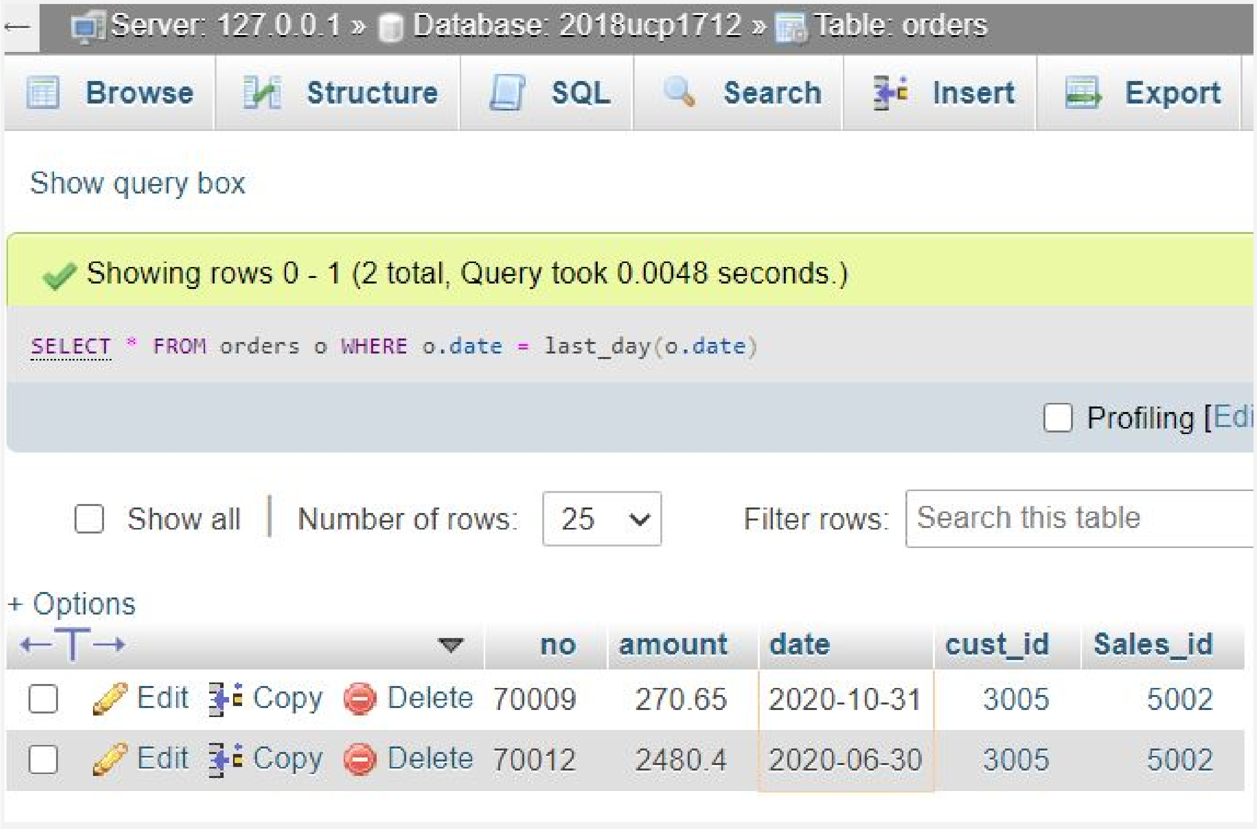
```
SELECT * FROM orders o JOIN customer c ON c.srno = o.cust_id and o.date = (SELECT MIN(o3.date) FROM orders o3, customer c2 WHERE o3.cust_id = c2.srno AND c2.city = c.city)
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Cre
□ Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
+ Options

no	amount	date	cust_id	Sales_id	srno	name	city	grade	sales_id
70004	110.5	2020-08-17	3005	5006	3005	Graham Zusi	California	200	5002
70007	948.5	2020-07-27	3003	5002	3003	Jozy Altidor	Moscow	200	5007
70010	1983.43	2020-10-10	3004	5003	3004	Fabian Johnson	Paris	300	5006
70013	75.29	2020-04-25	3008	5007	3008	Julian Green	London	300	5002

8. Show all orders made on the last day of the month.

```
SELECT * FROM orders o WHERE o.date = last_day(o.date);
```


←
Server: 127.0.0.1 » Database: 2018ucp1712 » Table: orders
Browse
Structure
SQL
Search
Insert
Export
Show query box

✓ Showing rows 0 - 1 (2 total, Query took 0.0048 seconds.)

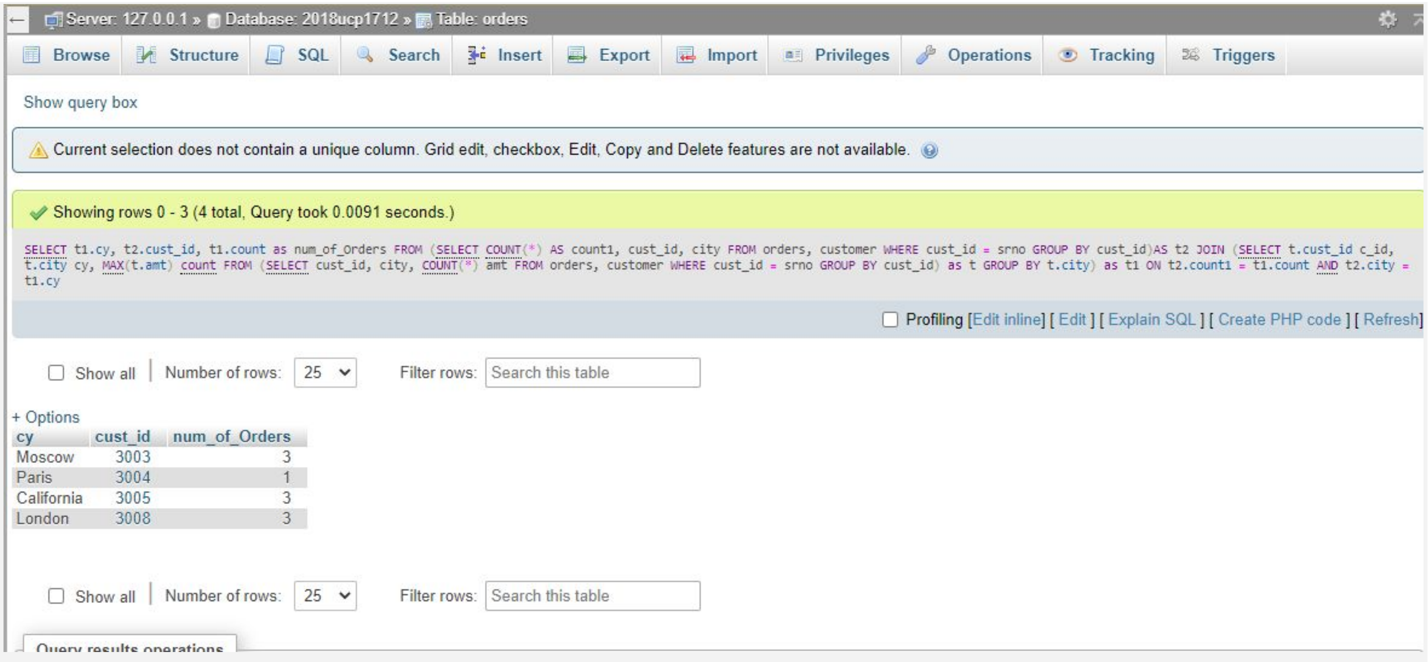
```
SELECT * FROM orders o WHERE o.date = last_day(o.date)
```

□ Profiling [Edi
□ Show all | Number of rows: 25
Filter rows: Search this table
+ Options
←T→

		no	amount	date	cust_id	Sales_id
□	Edit Copy Delete	70009	270.65	2020-10-31	3005	5002
□	Edit Copy Delete	70012	2480.4	2020-06-30	3005	5002

9. Find the customer with the highest number of orders in each city.

```
SELECT t1.cy, t2.cust_id, t1.count as num_of_Orders
FROM
(SELECT COUNT(*) AS count1, cust_id, city
 FROM orders, customer WHERE cust_id = srno GROUP BY cust_id)AS t2
 JOIN
 (SELECT t.cust_id c_id, t.city cy, MAX(t.amt) count
  FROM
  (SELECT cust_id, city, COUNT(*) amt FROM orders, customer WHERE
cust_id = srno GROUP BY
   cust_id) as t
  GROUP BY t.city) as t1
  ON t2.count1 = t1.count
  AND t2.city = t1.cy;
```



Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 3 (4 total, Query took 0.0091 seconds.)

```
SELECT t1.cy, t2.cust_id, t1.count as num_of_Orders FROM (SELECT COUNT(*) AS count1, cust_id, city FROM orders, customer WHERE cust_id = srno GROUP BY cust_id)AS t2 JOIN (SELECT t.cust_id c_id, t.city cy, MAX(t.amt) count FROM (SELECT cust_id, city, COUNT(*) amt FROM orders, customer WHERE cust_id = srno GROUP BY cust_id) as t GROUP BY t.city) as t1 ON t2.count1 = t1.count AND t2.city = t1.cy
```

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
☐ Show all | Number of rows: 25 Filter rows: Search this table
+ Options

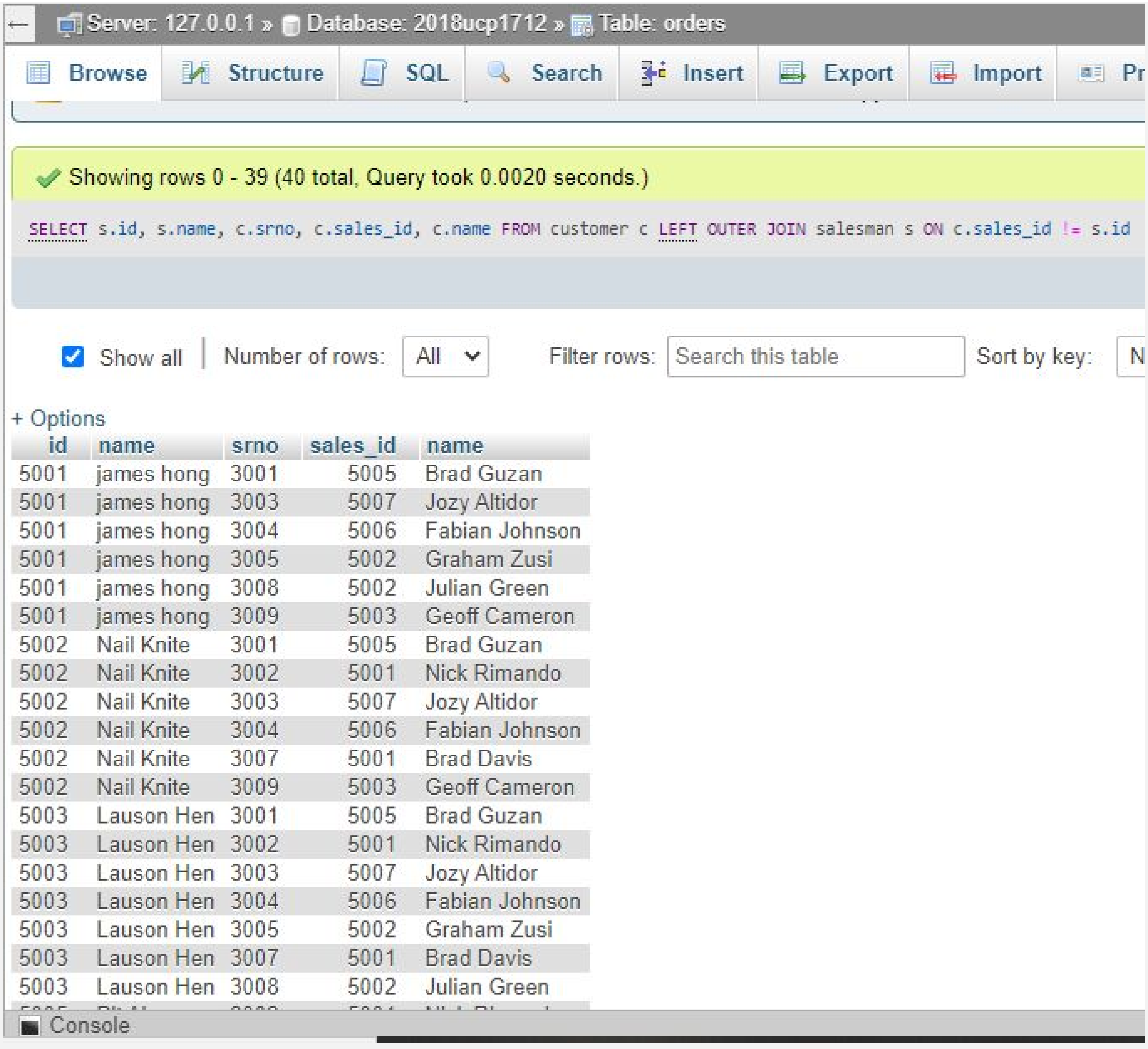
cy	cust_id	num_of_Orders
Moscow	3003	3
Paris	3004	1
California	3005	3
London	3008	3

☐ Show all | Number of rows: 25 Filter rows: Search this table
Query results operations

10. For each salesman, list the customers who have not ordered from him.

SELECT s.id, s.name, c.srno, c.sales_id, c.name FROM customer c LEFT OUTER JOIN salesman s ON c.sales_id != s.id;

ScreenShot



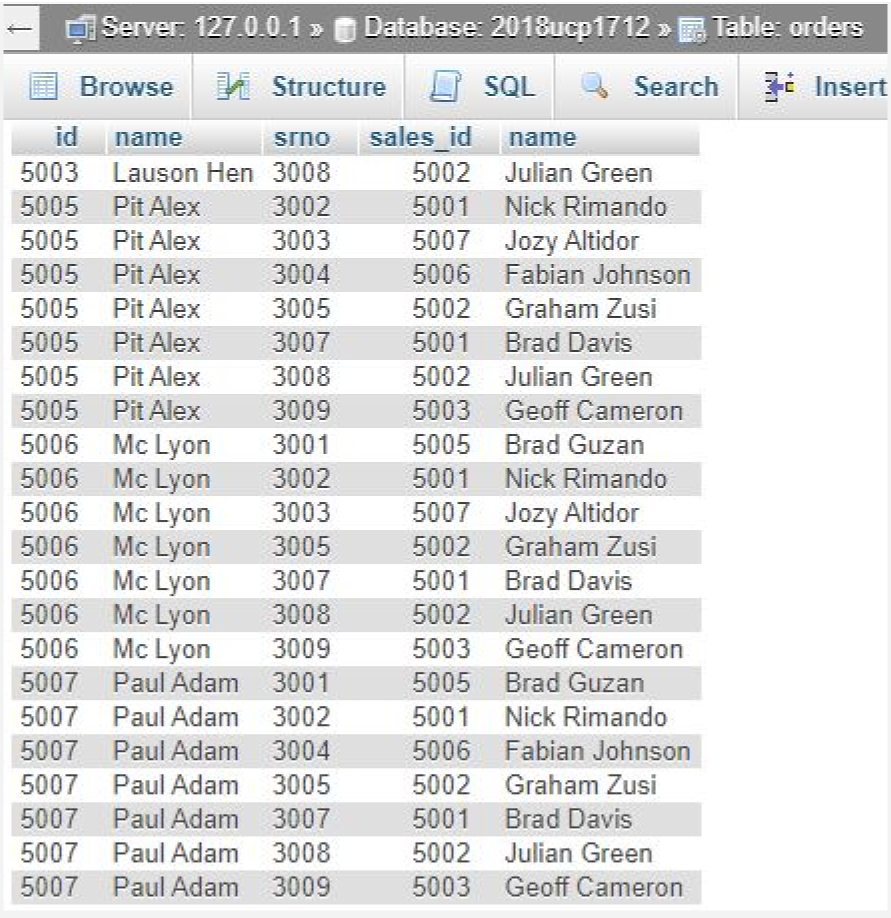
Showing rows 0 - 39 (40 total, Query took 0.0020 seconds.)

```
SELECT s.id, s.name, c.srno, c.sales_id, c.name FROM customer c LEFT OUTER JOIN salesman s ON c.sales_id != s.id
```

id	name	srno	sales_id	name
5001	james hong	3001	5005	Brad Guzan
5001	james hong	3003	5007	Jozy Altidor
5001	james hong	3004	5006	Fabian Johnson
5001	james hong	3005	5002	Graham Zusi
5001	james hong	3008	5002	Julian Green
5001	james hong	3009	5003	Geoff Cameron
5002	Nail Knite	3001	5005	Brad Guzan
5002	Nail Knite	3002	5001	Nick Rimando
5002	Nail Knite	3003	5007	Jozy Altidor
5002	Nail Knite	3004	5006	Fabian Johnson
5002	Nail Knite	3007	5001	Brad Davis
5002	Nail Knite	3009	5003	Geoff Cameron
5003	Lauson Hen	3001	5005	Brad Guzan
5003	Lauson Hen	3002	5001	Nick Rimando
5003	Lauson Hen	3003	5007	Jozy Altidor
5003	Lauson Hen	3004	5006	Fabian Johnson
5003	Lauson Hen	3005	5002	Graham Zusi
5003	Lauson Hen	3007	5001	Brad Davis
5003	Lauson Hen	3008	5002	Julian Green

continued..

--------->>>>



id	name	srno	sales_id	name
5003	Lauson Hen	3008	5002	Julian Green
5005	Pit Alex	3002	5001	Nick Rimando
5005	Pit Alex	3003	5007	Jozy Altidor
5005	Pit Alex	3004	5006	Fabian Johnson
5005	Pit Alex	3005	5002	Graham Zusi
5005	Pit Alex	3007	5001	Brad Davis
5005	Pit Alex	3008	5002	Julian Green
5005	Pit Alex	3009	5003	Geoff Cameron
5006	Mc Lyon	3001	5005	Brad Guzan
5006	Mc Lyon	3002	5001	Nick Rimando
5006	Mc Lyon	3003	5007	Jozy Altidor
5006	Mc Lyon	3005	5002	Graham Zusi
5006	Mc Lyon	3007	5001	Brad Davis
5006	Mc Lyon	3008	5002	Julian Green
5006	Mc Lyon	3009	5003	Geoff Cameron
5007	Paul Adam	3001	5005	Brad Guzan
5007	Paul Adam	3002	5001	Nick Rimando
5007	Paul Adam	3004	5006	Fabian Johnson
5007	Paul Adam	3005	5002	Graham Zusi
5007	Paul Adam	3007	5001	Brad Davis
5007	Paul Adam	3008	5002	Julian Green
5007	Paul Adam	3009	5003	Geoff Cameron

```
=================================================================
=================================================================
```