

LAB RECORD

NAME: SAQUIB NAUSHAD

USN: 1BM19CS144

DEPT: CSE

SECTION : C

COURSE NAME: DATABASE MANAGEMENT SYSTEMS

LAB_BATCH:C-3

LAP PROGRAM 1:-

PROGRAM 1

Consider the following schema:

Consider the Insurance database given below. The primary keys are underlined and the data types are specified.

PERSON (driver-id #: String, name: String, address: String)

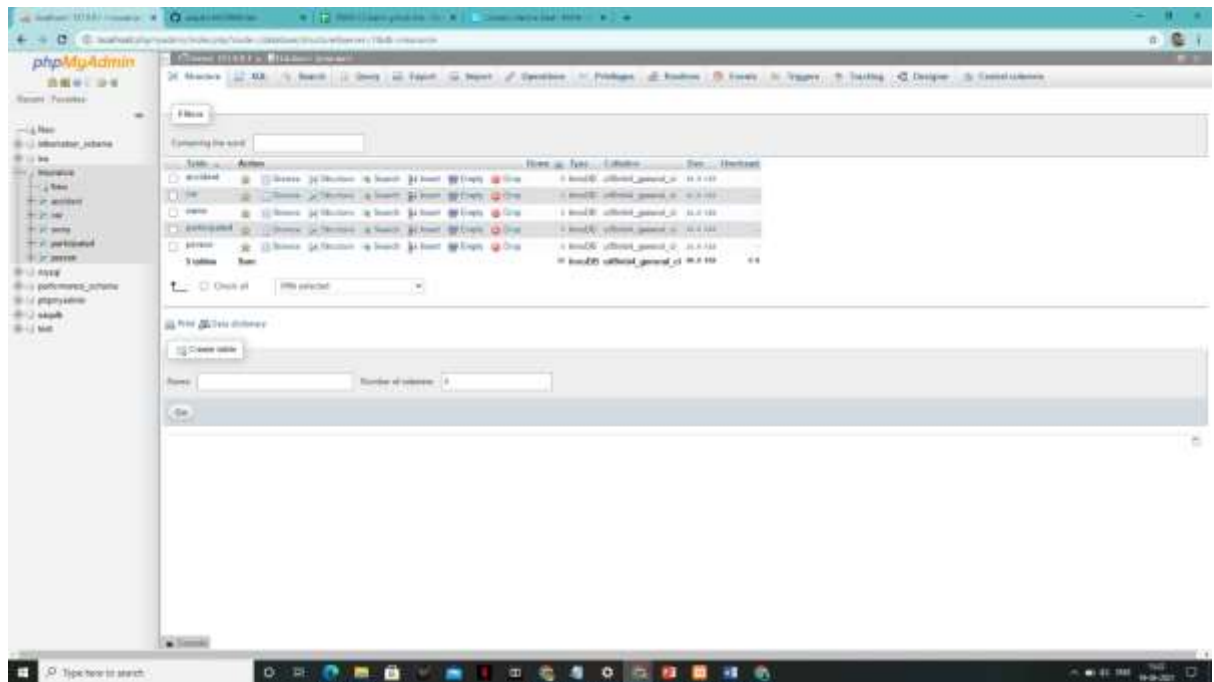
CAR (Regno: String, model: String, year: int)

ACCIDENT (report-number: int, date: date, location: String)

OWNS (driver-id #: String, Regno: String)

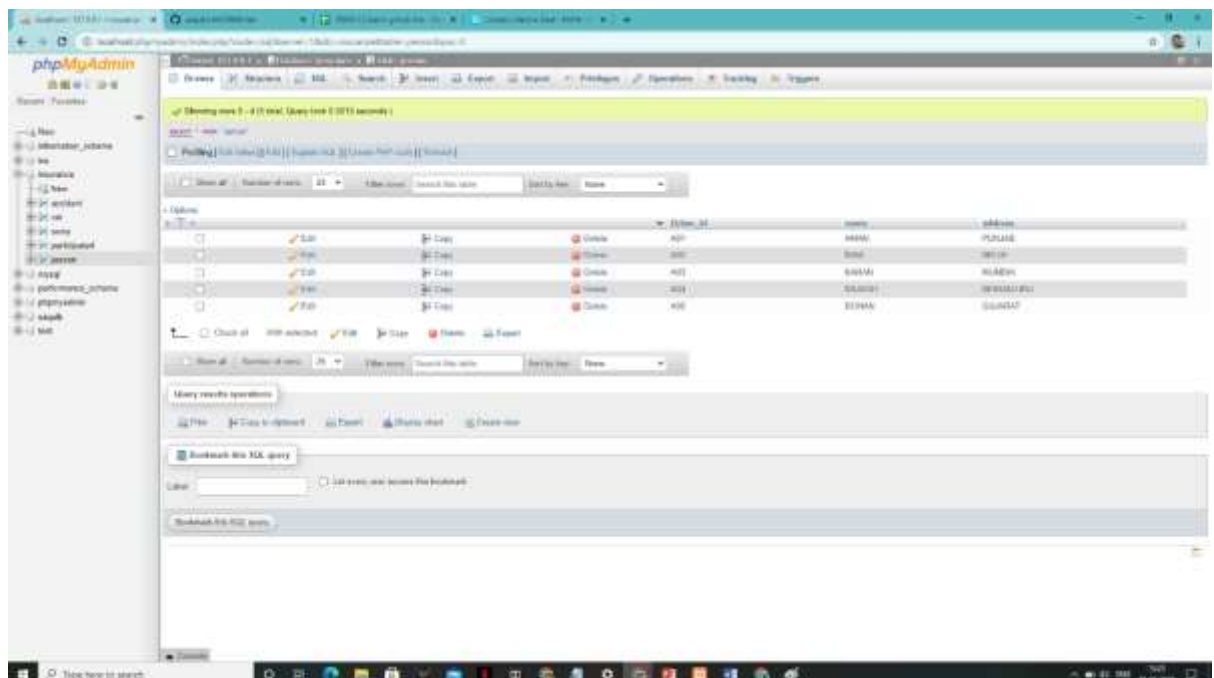
PARTICIPATED (driver-id: String, Regno: String, report-number: int, damage-amount: int)

- 1) Create the above tables y properly specifying the primary keys and the foreign keys.



2) Enter at least 5 tuples for each relation.

‘PERSON’ table:



‘CAR’ table:

The screenshot shows the phpMyAdmin interface with the 'PERSON' table selected. The table structure is displayed with the following columns:

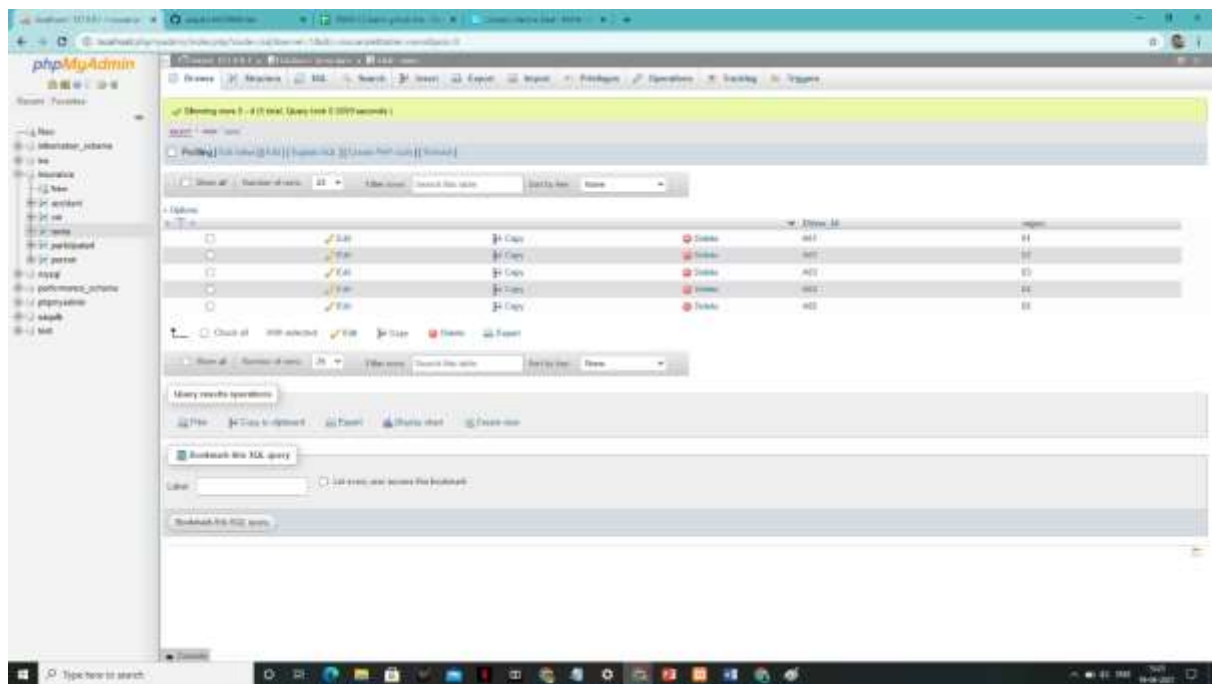
Column	Type	Nullable	Default	Extra
id	int(11)	NO		PRIMARY
name	varchar(255)	NO		
age	int(11)	NO		
sex	enum('M', 'F')	NO		
height	float(10,2)	NO		
weight	float(10,2)	NO		

‘ACCIDENT’ table:

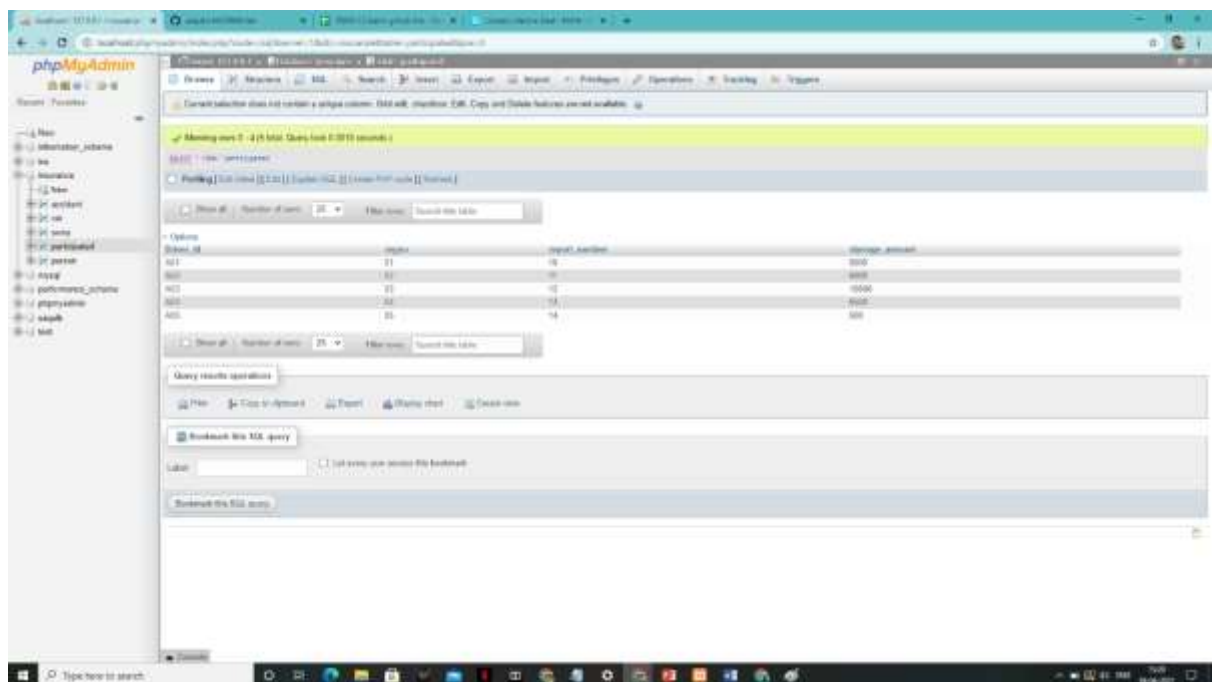
The screenshot shows the phpMyAdmin interface with the 'ACCIDENT' table selected. The table structure is displayed with the following columns:

Column	Type	Nullable	Default	Extra
id	int(11)	NO		PRIMARY
date	date	NO		
location	varchar(255)	NO		
number	int(11)	NO		
person	int(11)	NO		
height	float(10,2)	NO		
weight	float(10,2)	NO		

‘OWNS’ table:

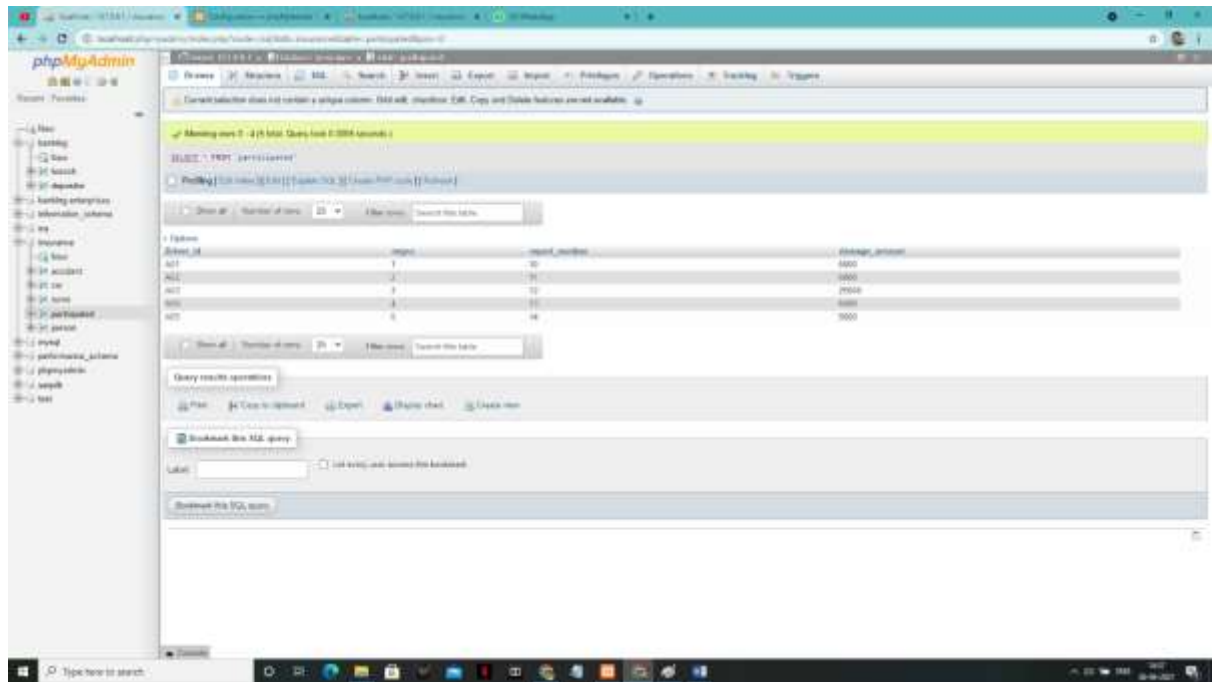


‘PARTICIPATED’ table:

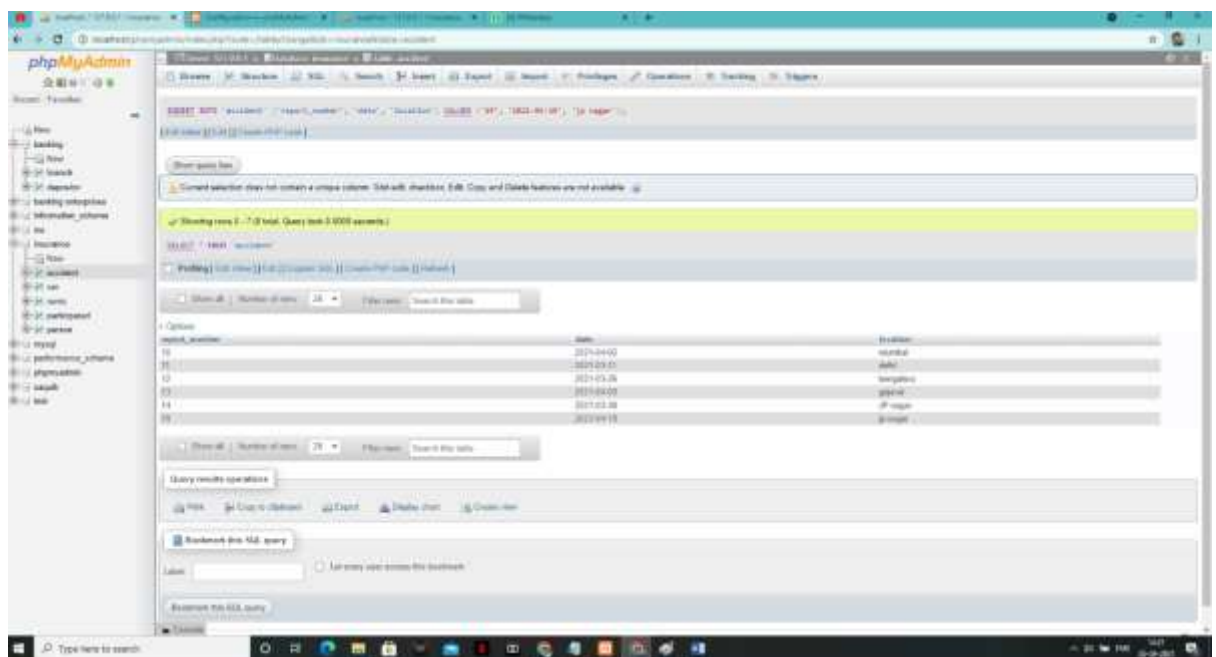
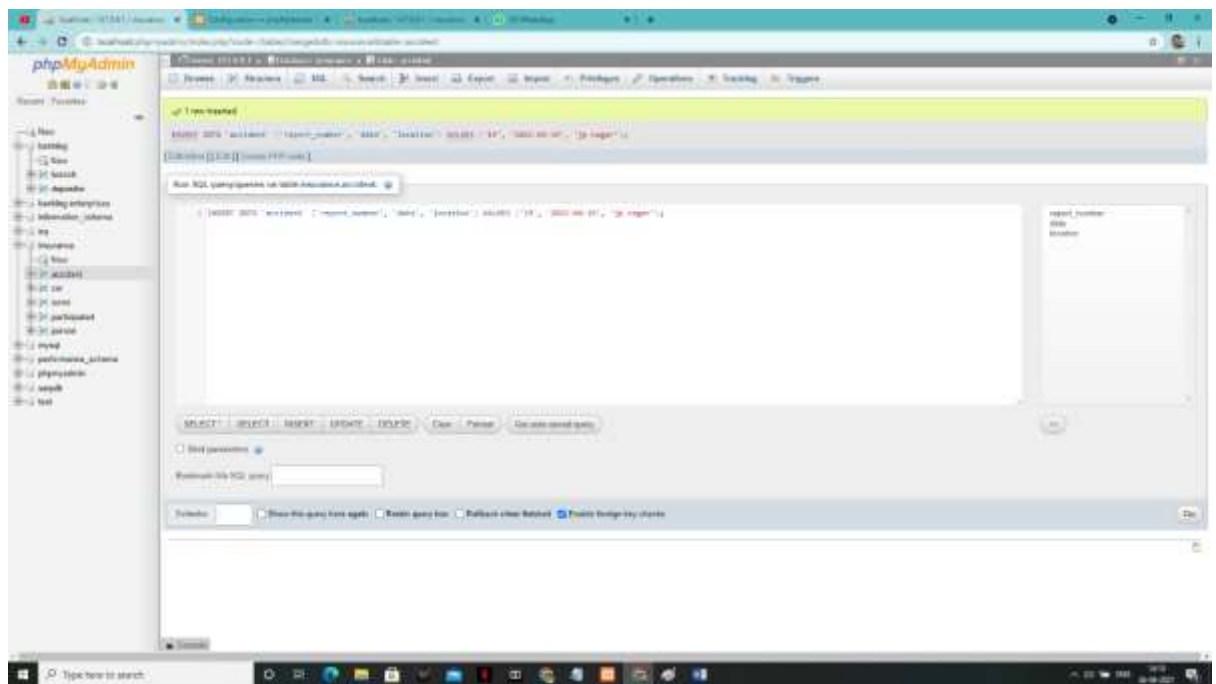


3) Demonstrate how you

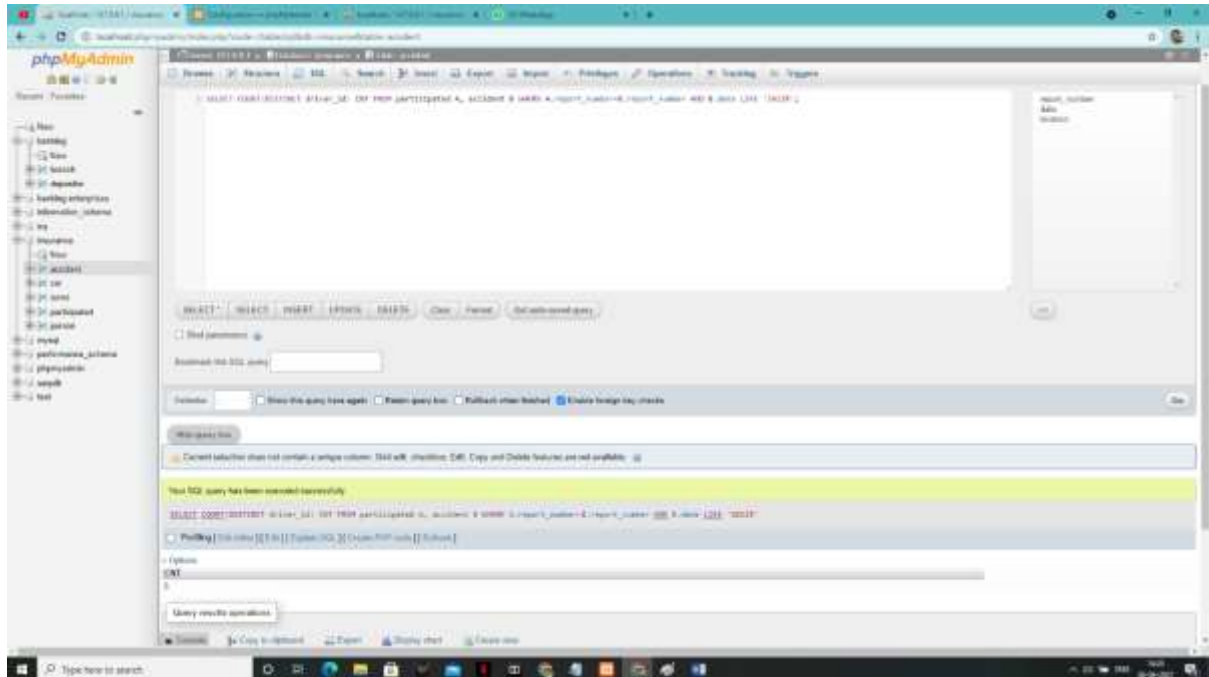
a) Update the damage amount for the car with a specific Regno in the accident with report number 12 to 25000.



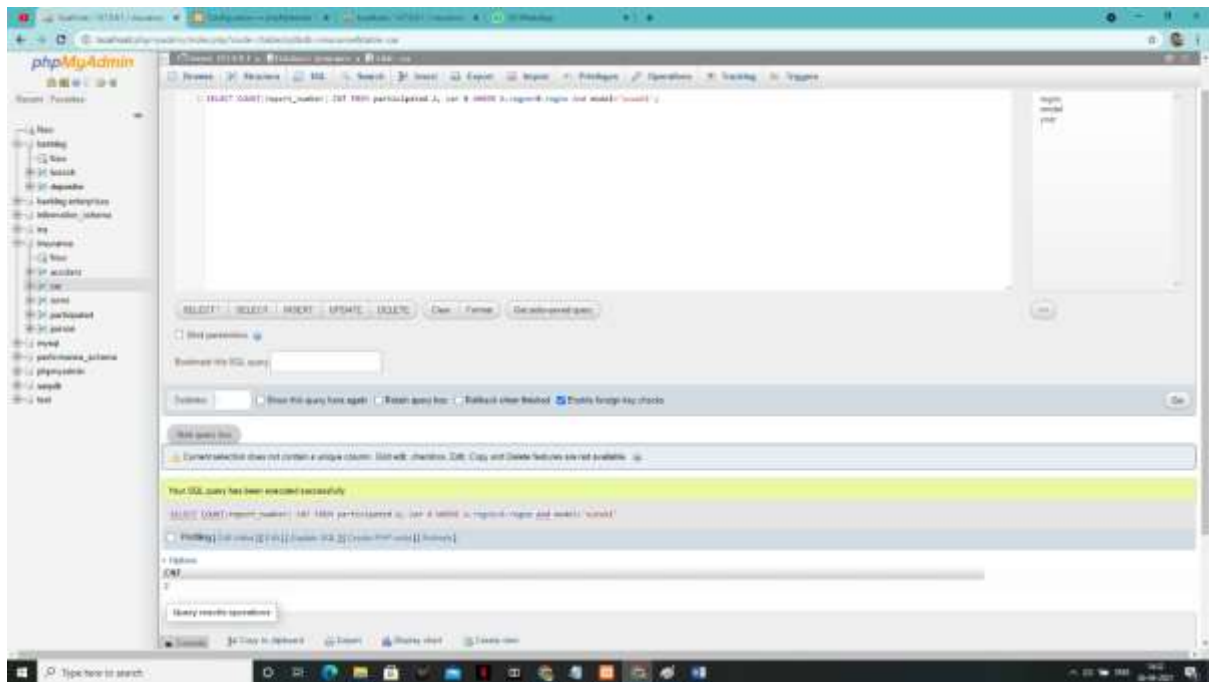
b) Add a new accident to the database.



4) Find the total number of people who owned cars that were involved in accidents in 2021.



5) Find the number of accidents in which cars belonging to a specific model (say 'Suzuki') were involved.



LAB PROGRAM 2:-

PROGRAM 2

Consider the following database for a banking enterprise.

BRANCH (branch-name: String, branch-city: String, assets: real)

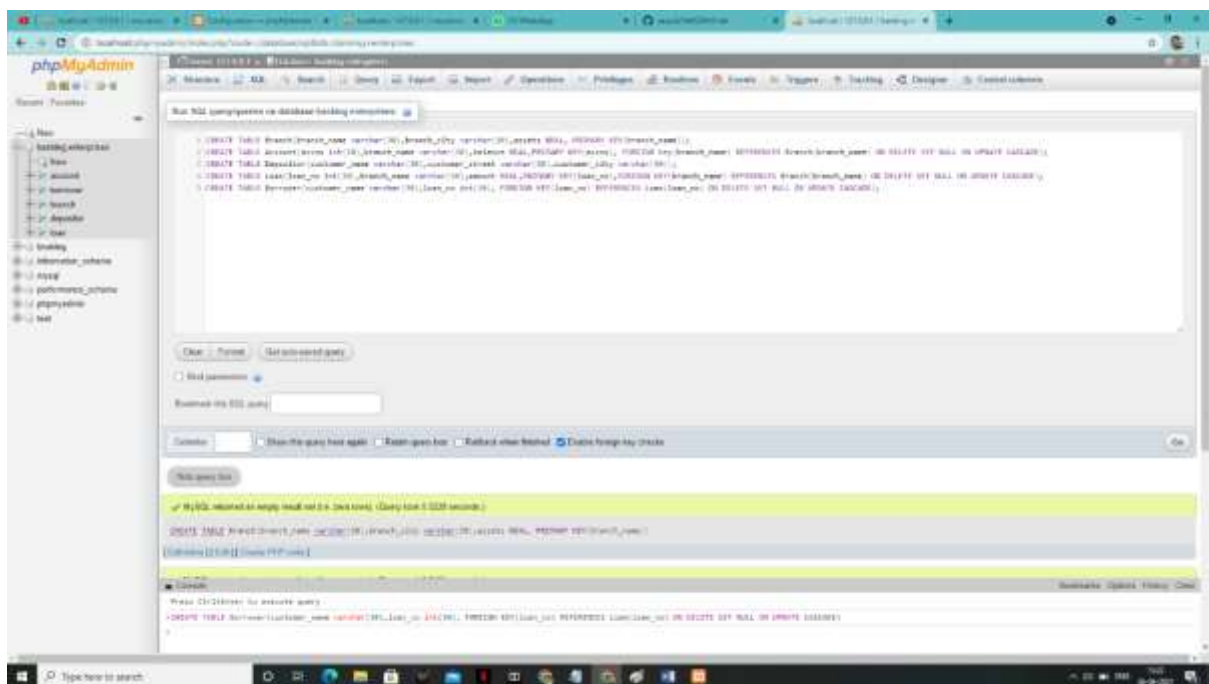
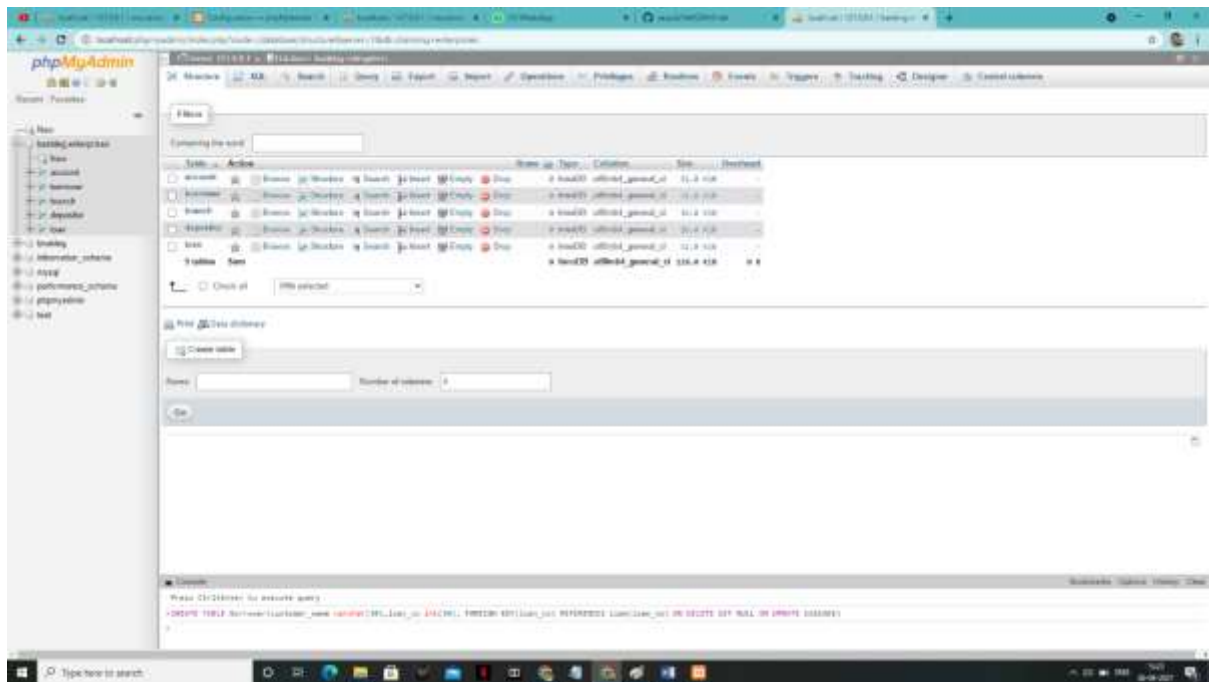
ACCOUNTS (accno: int, branch-name: String, balance: real)

DEPOSITOR (customer-name: String, customer-street: String, customer-city: String)

LOAN (loan-number: int, branch-name: String, amount: real)

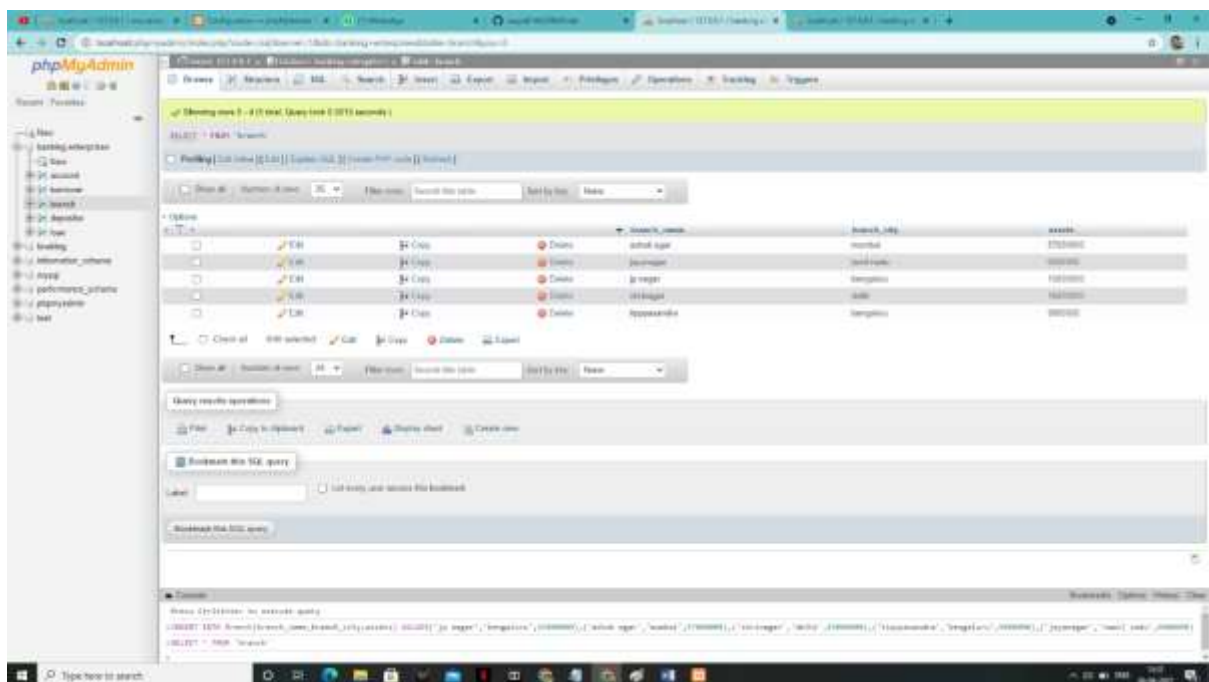
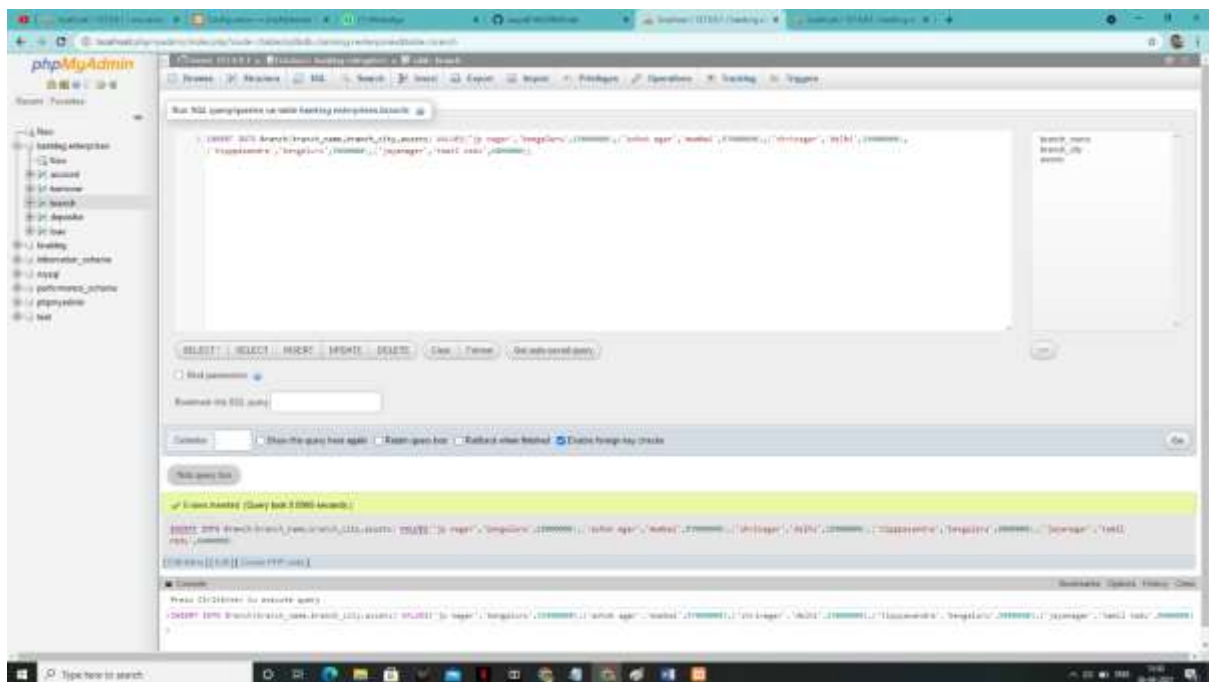
BORROWER (customer-name: String, loan-number: int)

- 1) Create the above tables by properly specifying the primary keys and the foreign keys.

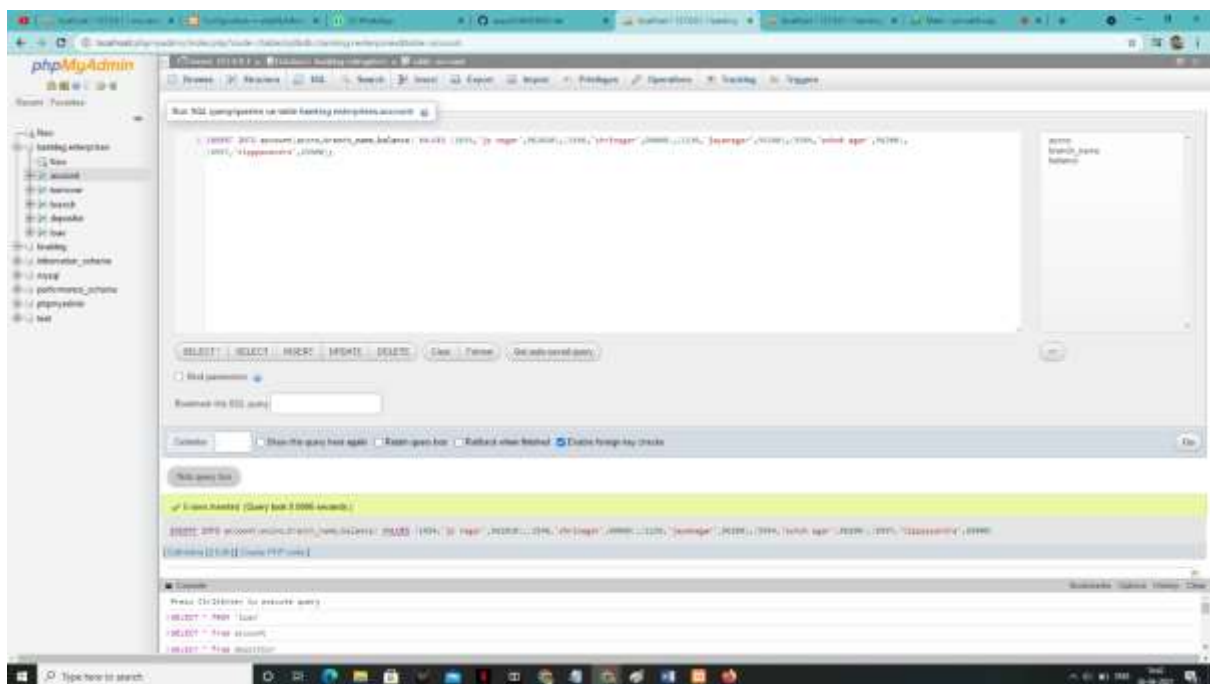
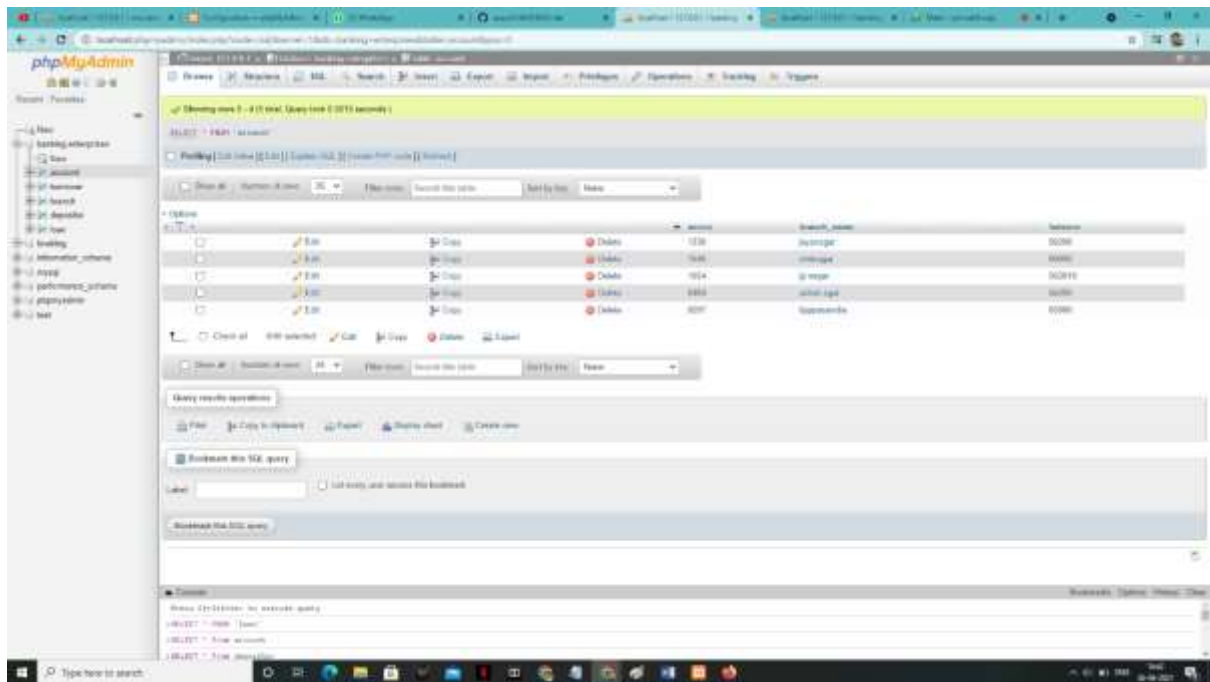


2) Enter at least 5 tuples for each relation.

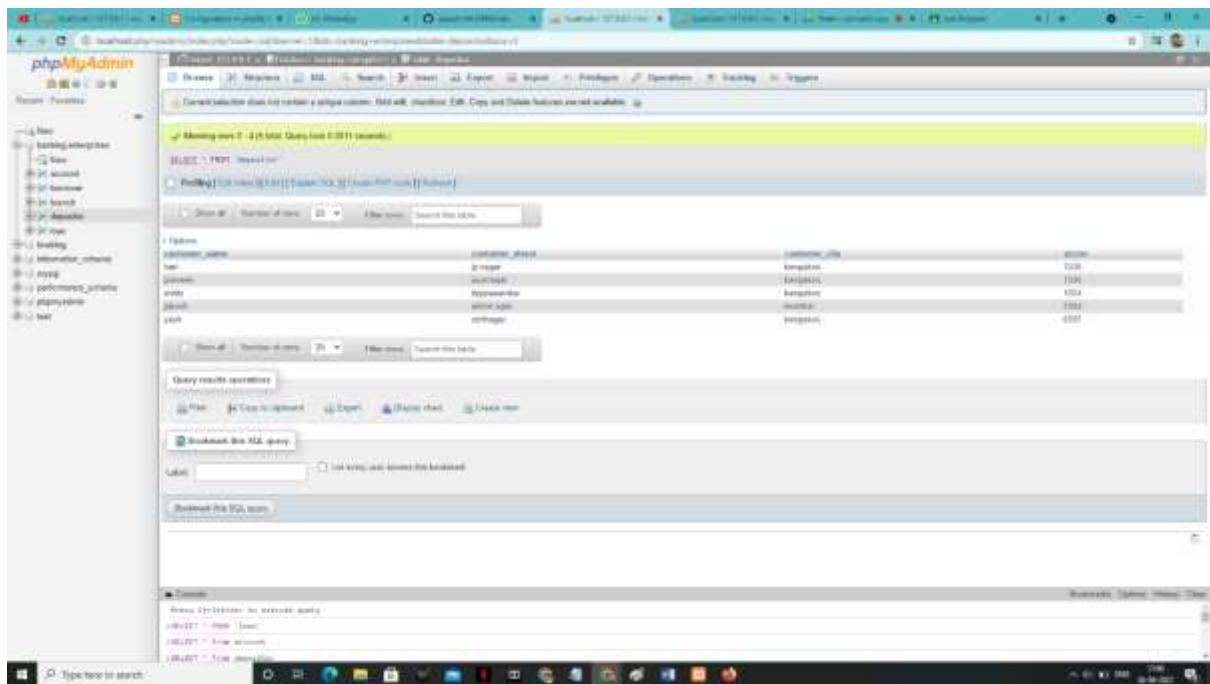
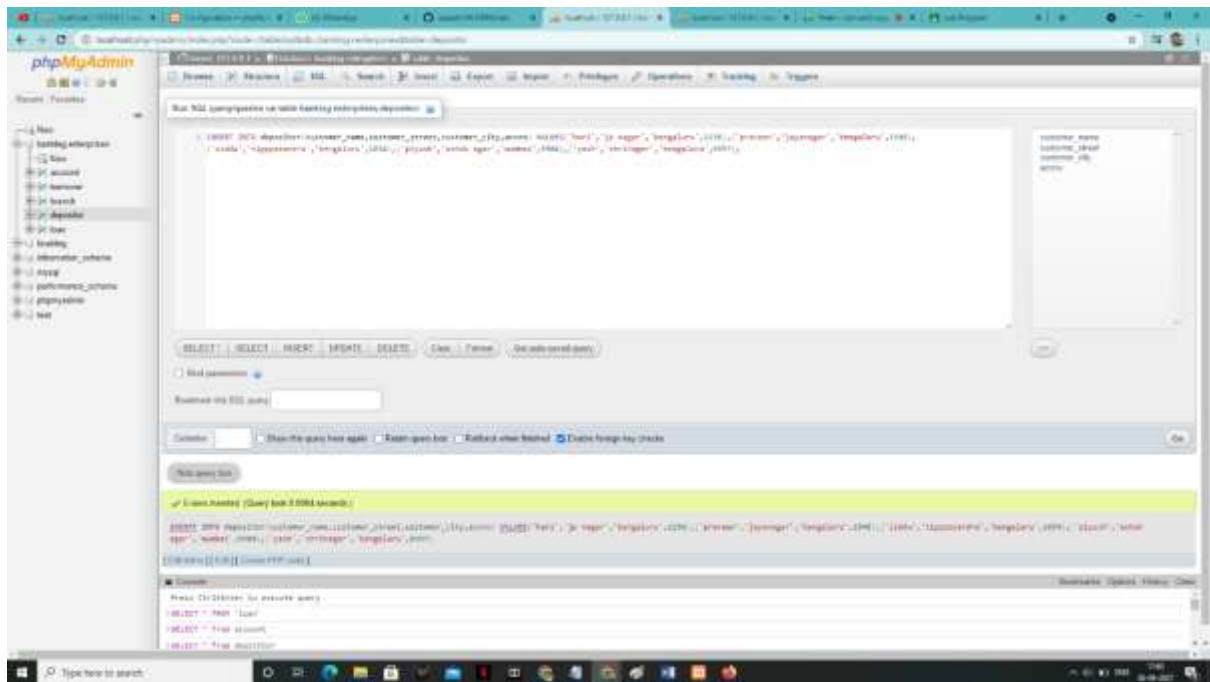
‘BRANCH’ table:



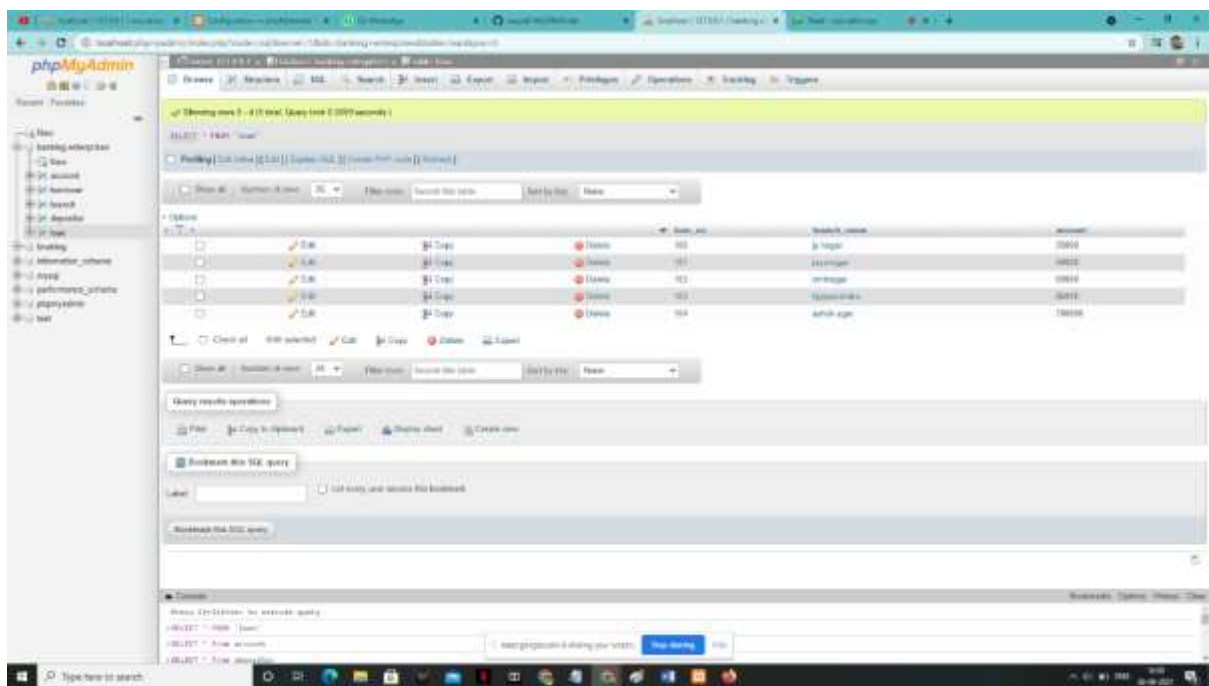
‘ACCOUNTS’ table:

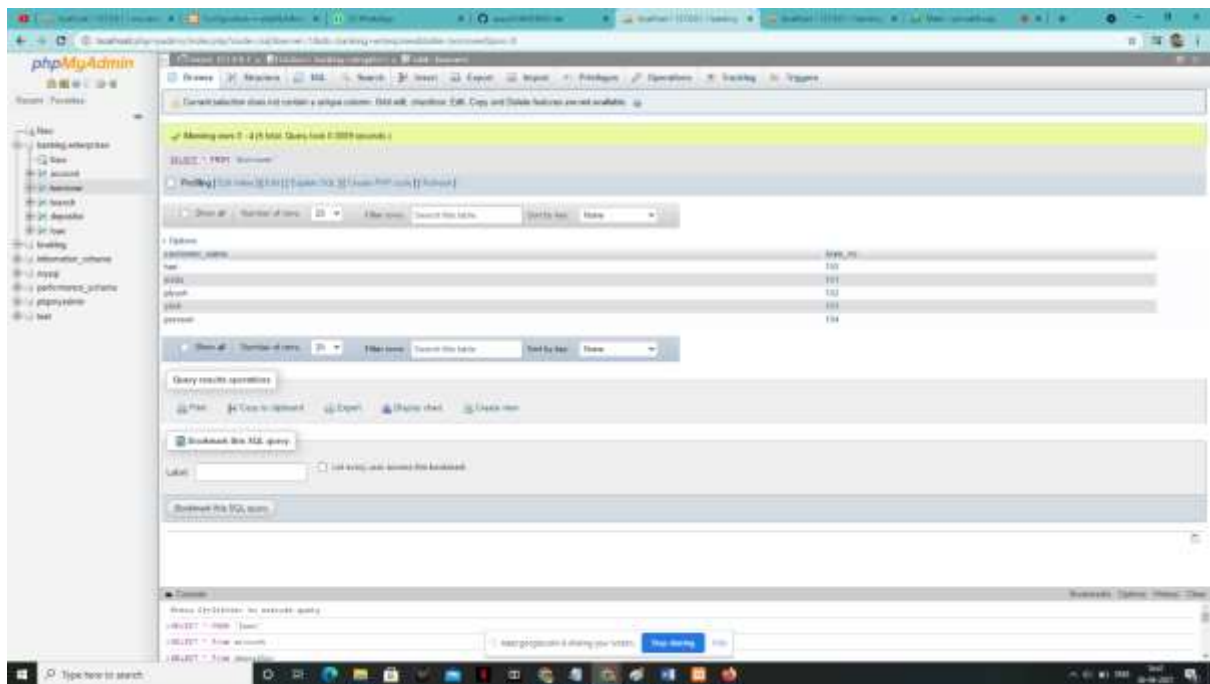


'DEPOSITOR' table:

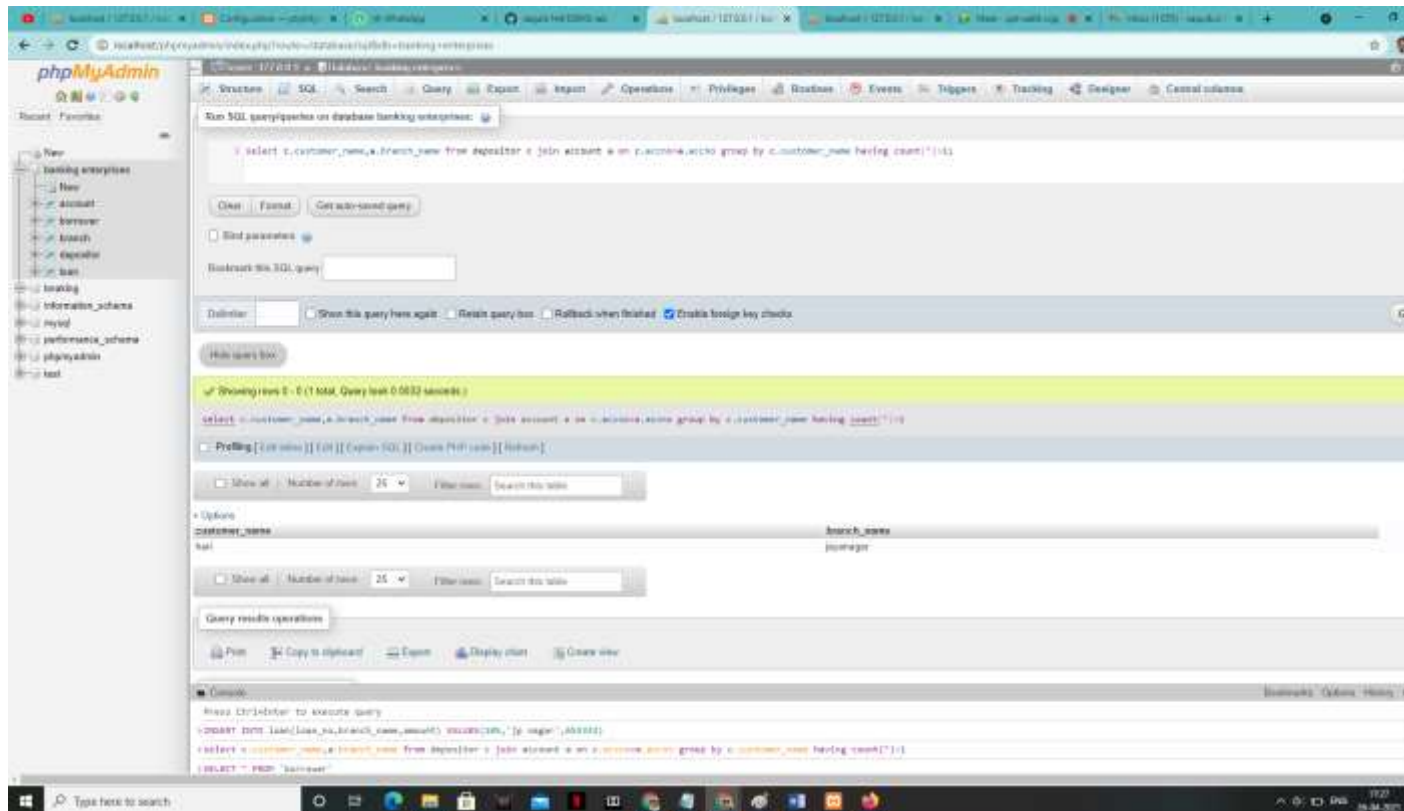


'LOAN' table:

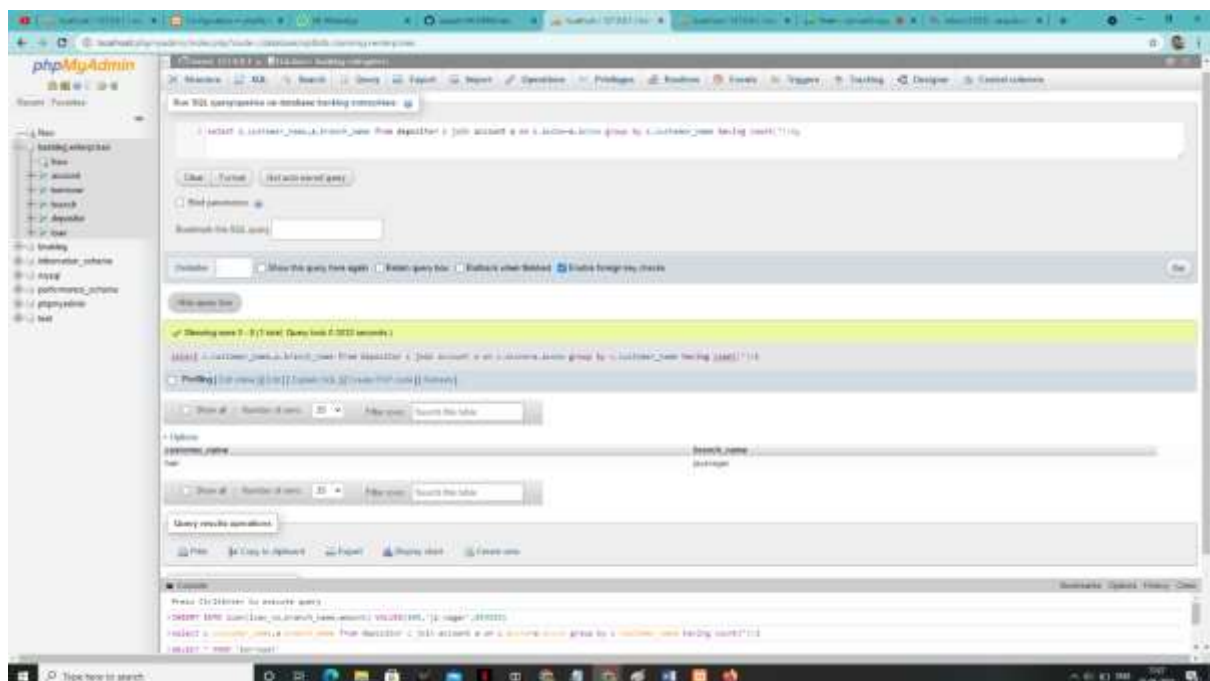




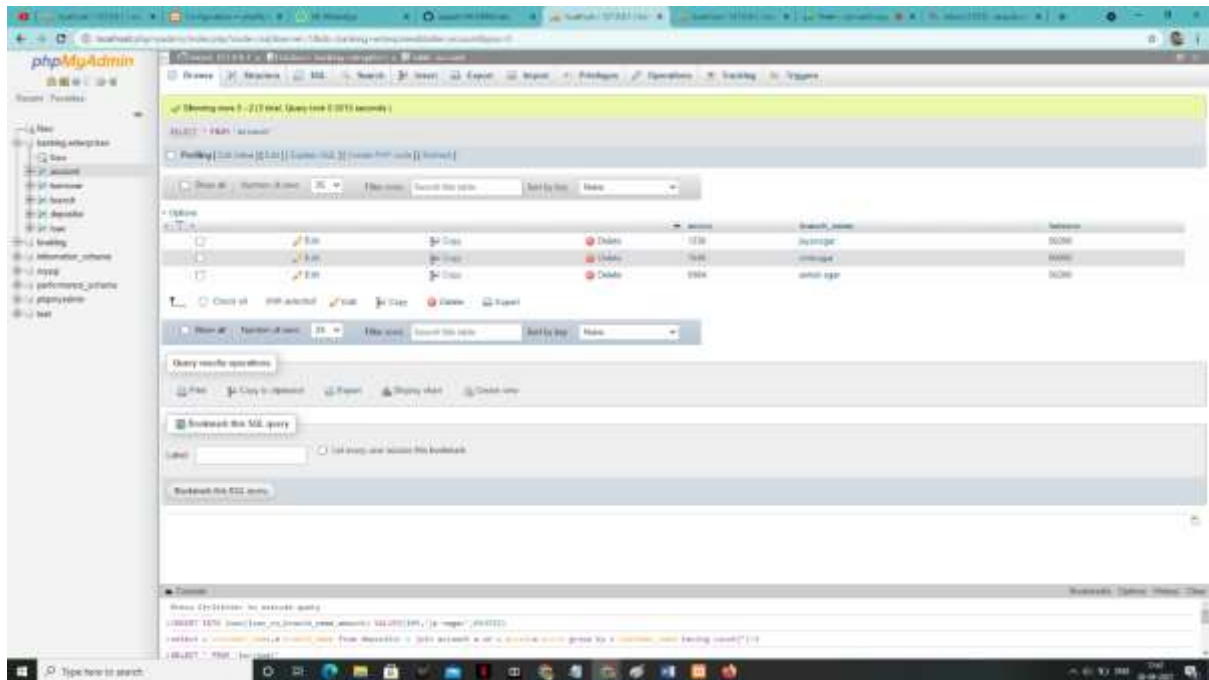
iii. Find all the customers who have at least two accounts at the Main branch.



iv. Find all the customers who have an account at all the branches located in a specific city.



v. Demonstrate how you delete all account tuples at every branch located in a specific city.



LAB PROGRAM 3:-

PROGRAM 3

Consider the following schema:

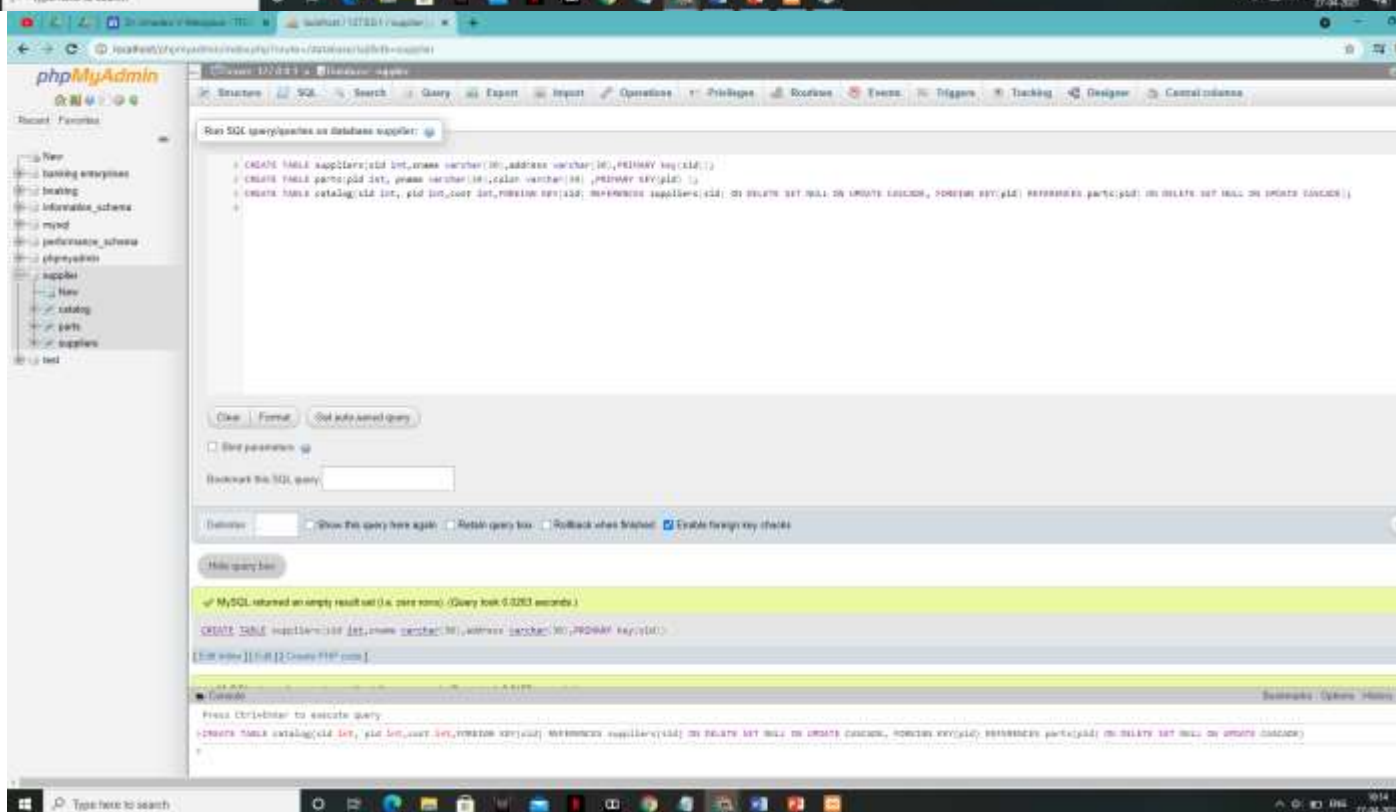
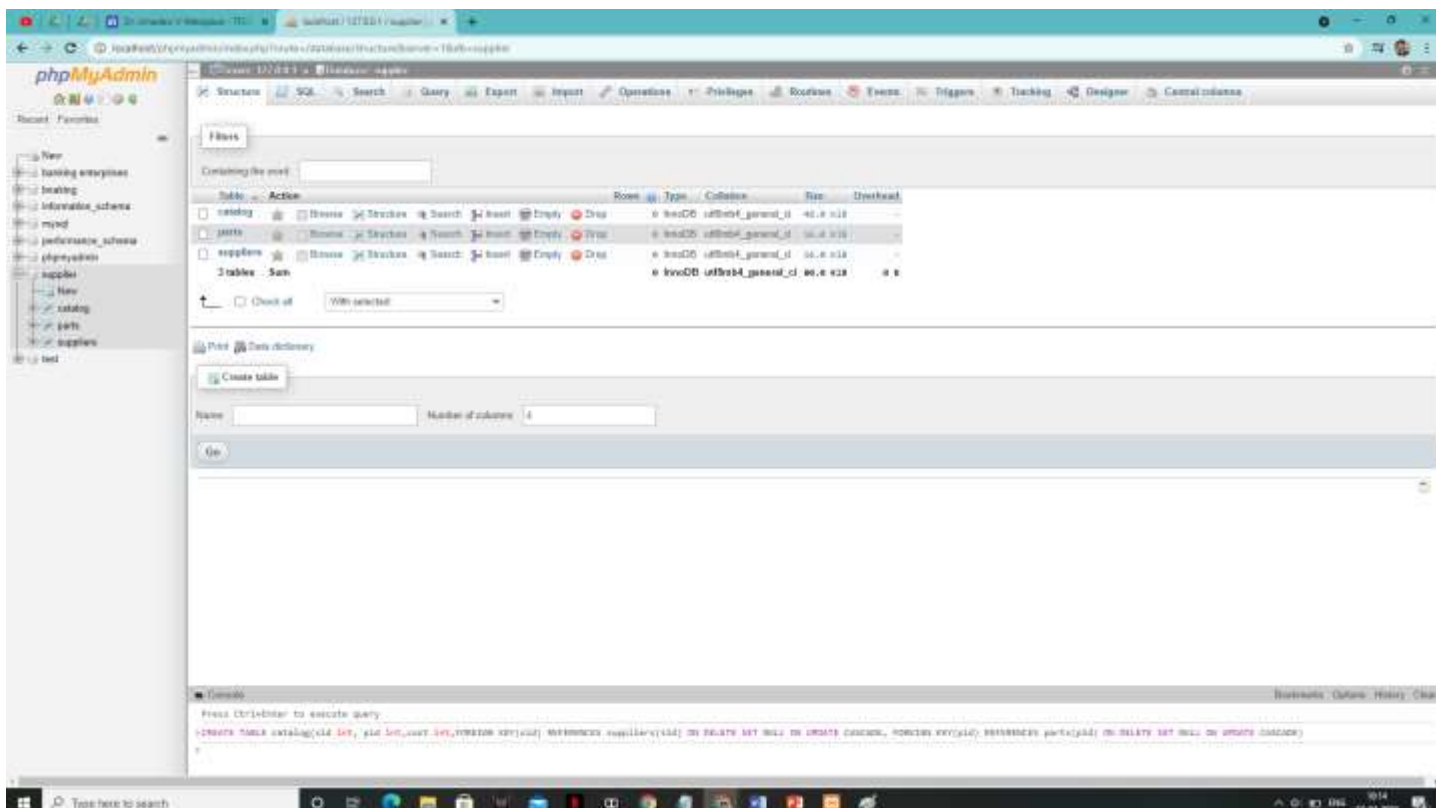
SUPPLIERS (sid: integer, sname: string, address: string)

PARTS (pid: integer, pname: string, color: string)

CATALOG (sid: integer, pid: integer, cost: real)

The Catalog relation lists the prices charged for parts by Suppliers.
Write the following queries in SQL:

2) Create the above tables by properly specifying the primary keys and the foreign keys.



3) Enter tuples for each relation.

'suppliers' table:

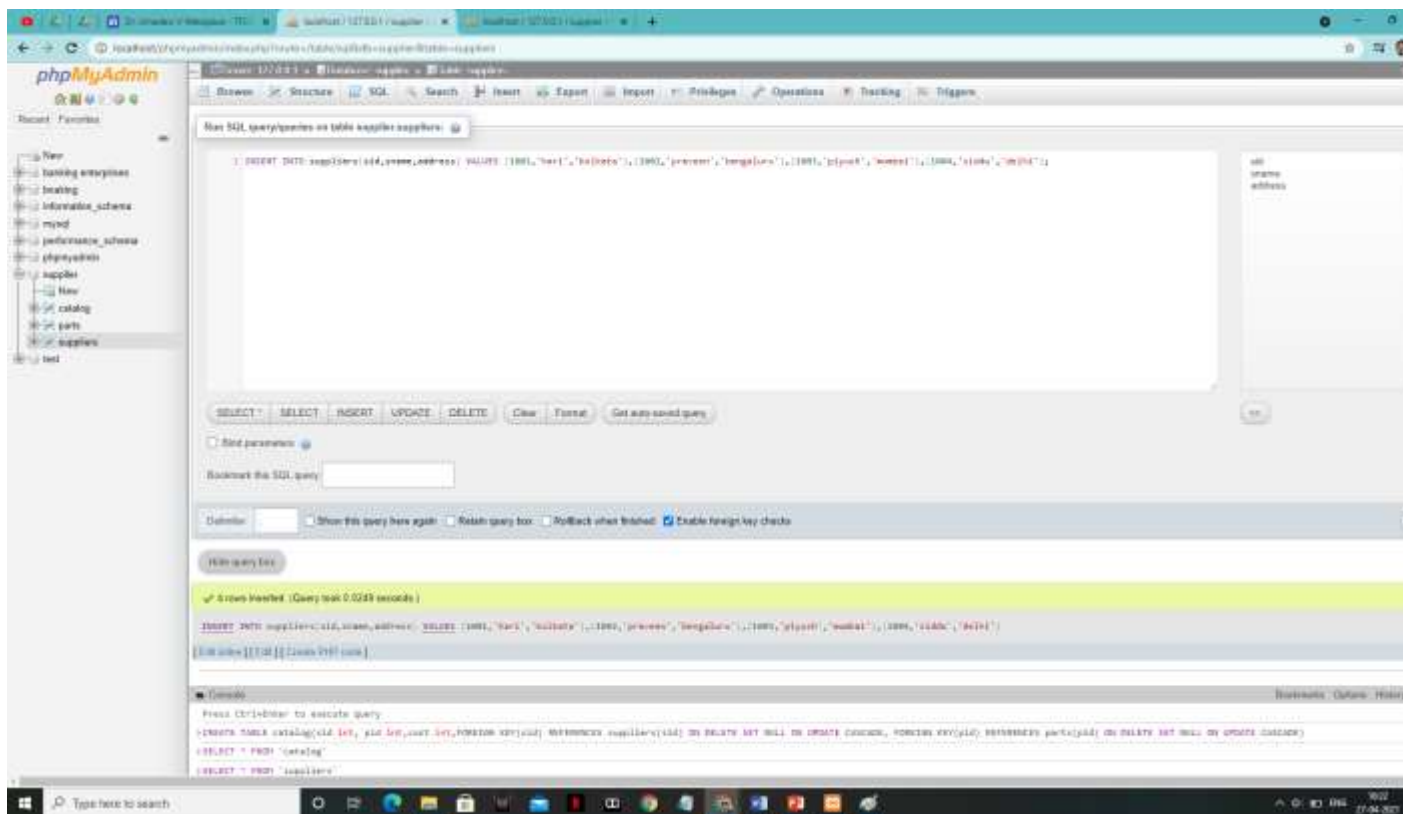
The screenshot shows the phpMyAdmin interface for a MySQL database. The 'suppliers' table is selected, and its structure and data are displayed. The table has columns: id, name, address, and address2. The data is as follows:

id	name	address	address2
1001	Sari	Indra	
1002	Arman	Indragiri	
1003	Yusuf	Indragiri	
1004	Yusuf	Indragiri	

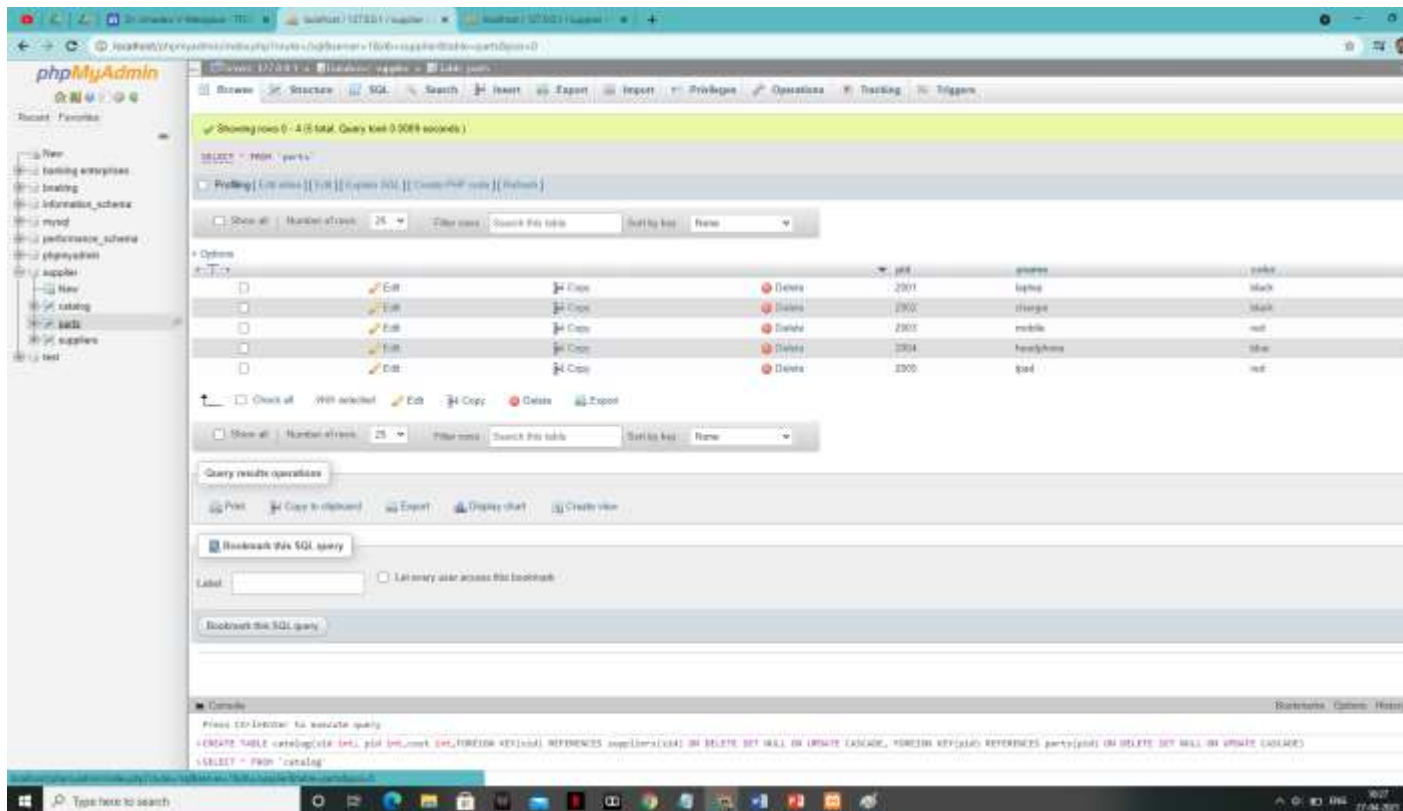
The console shows the following SQL queries:

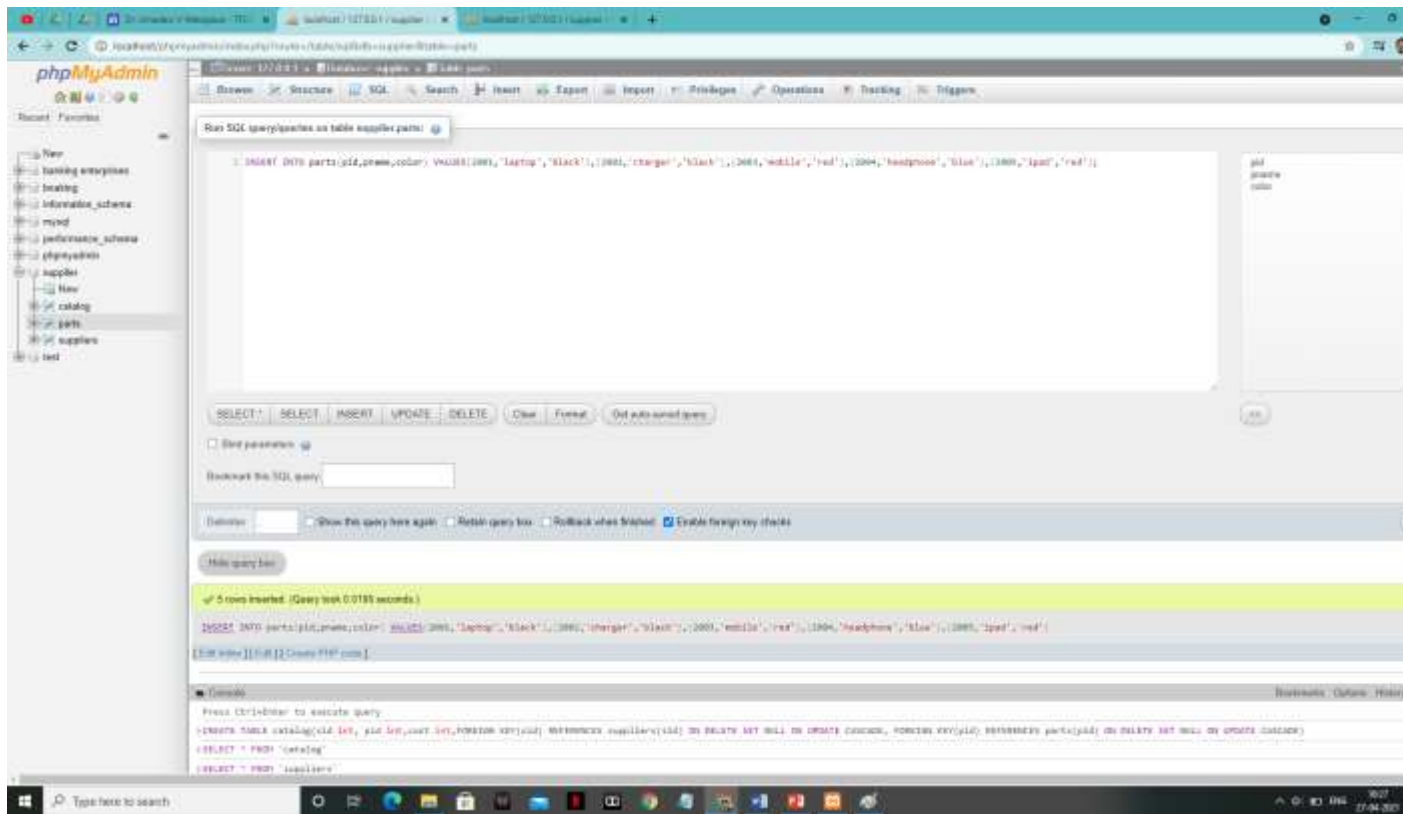
```
CREATE TABLE `suppliers` (
  `id` int(11) NOT NULL,
  `name` varchar(255) NOT NULL,
  `address` varchar(255) NOT NULL,
  `address2` varchar(255) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB;

INSERT INTO `suppliers` (`id`, `name`, `address`, `address2`) VALUES (1001, 'Sari', 'Indra', ''), (1002, 'Arman', 'Indragiri', ''), (1003, 'Yusuf', 'Indragiri', ''), (1004, 'Yusuf', 'Indragiri', '');
```



'PARTS' table:

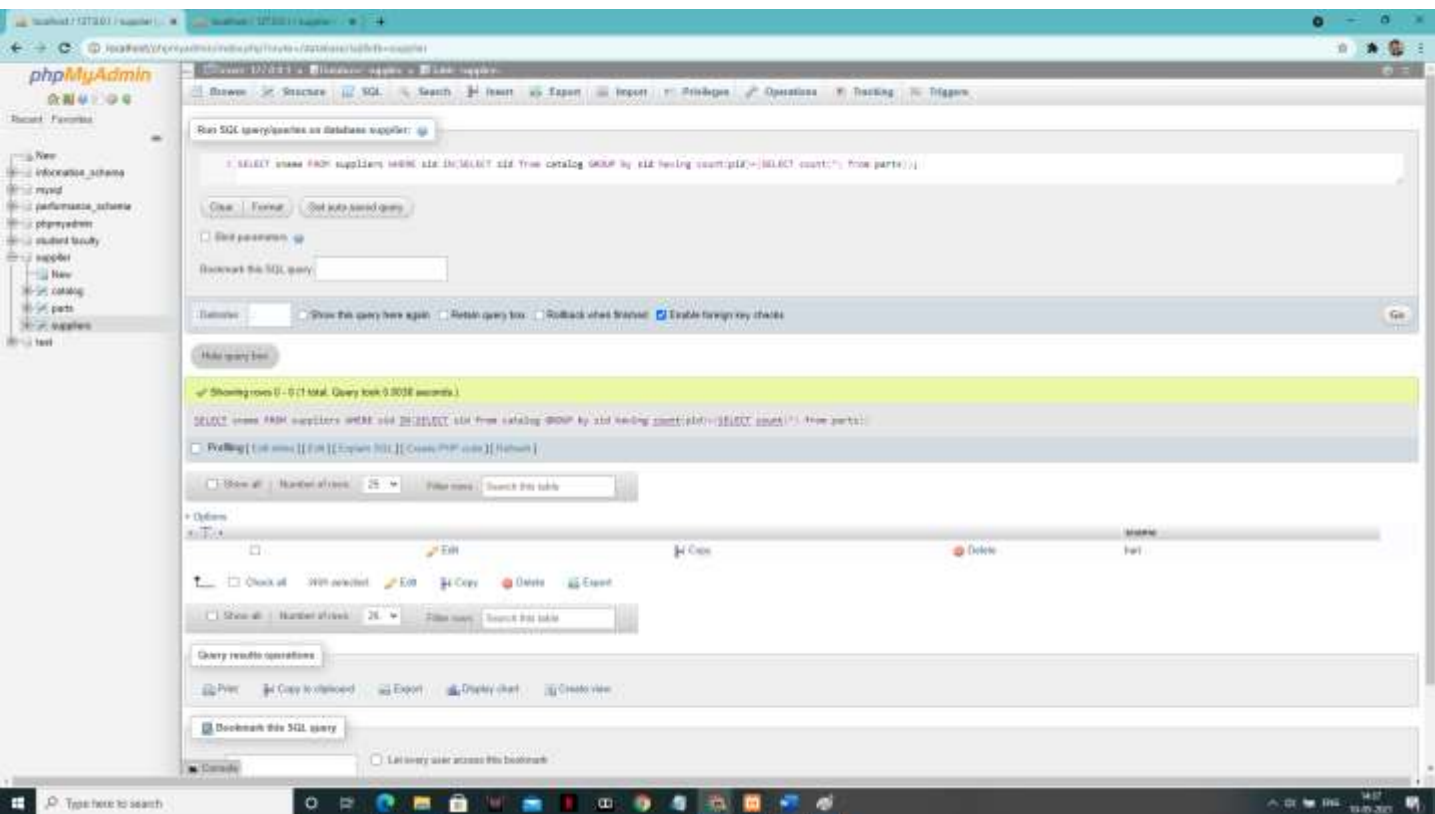




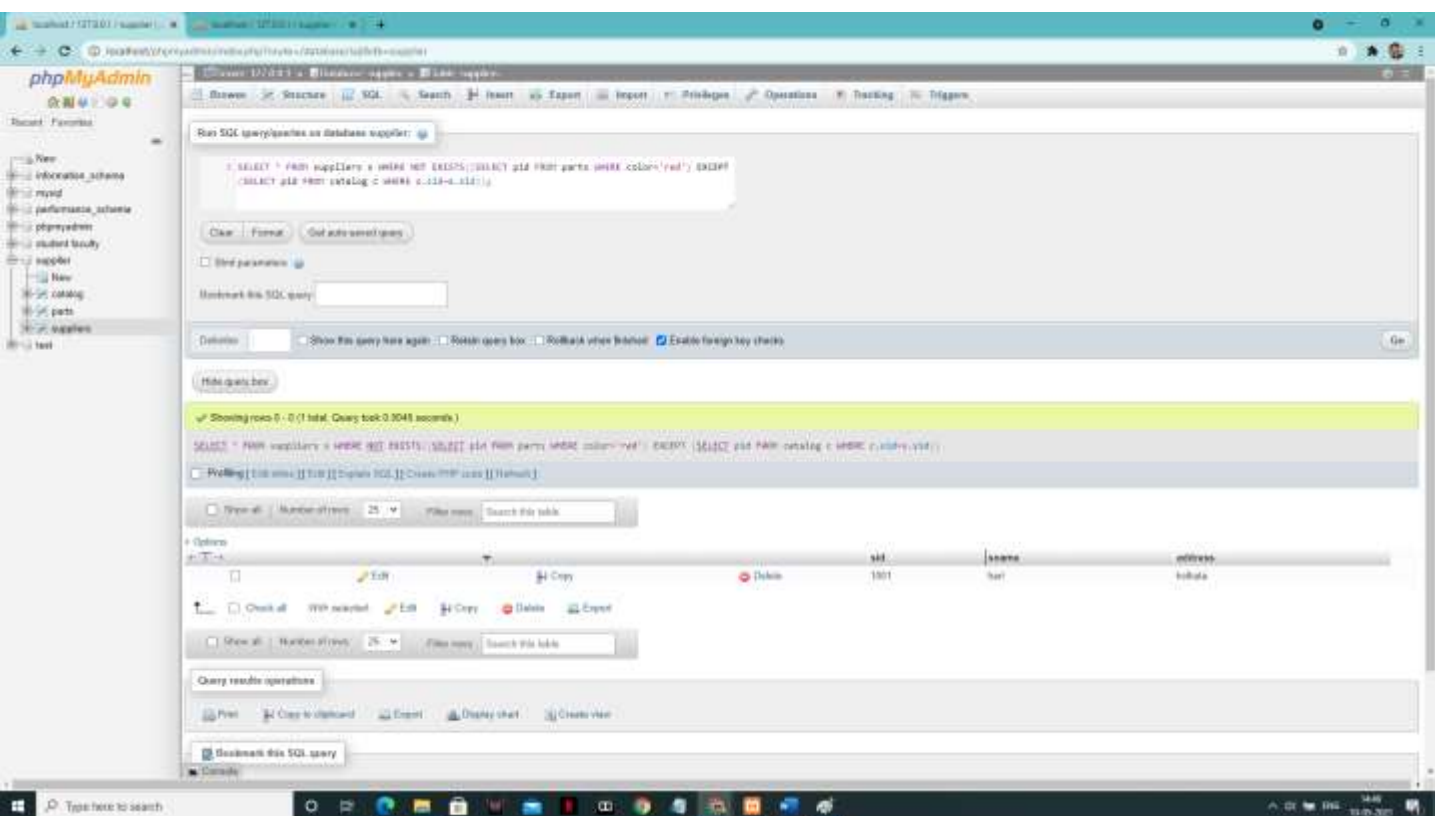
‘CATALOG’ table:



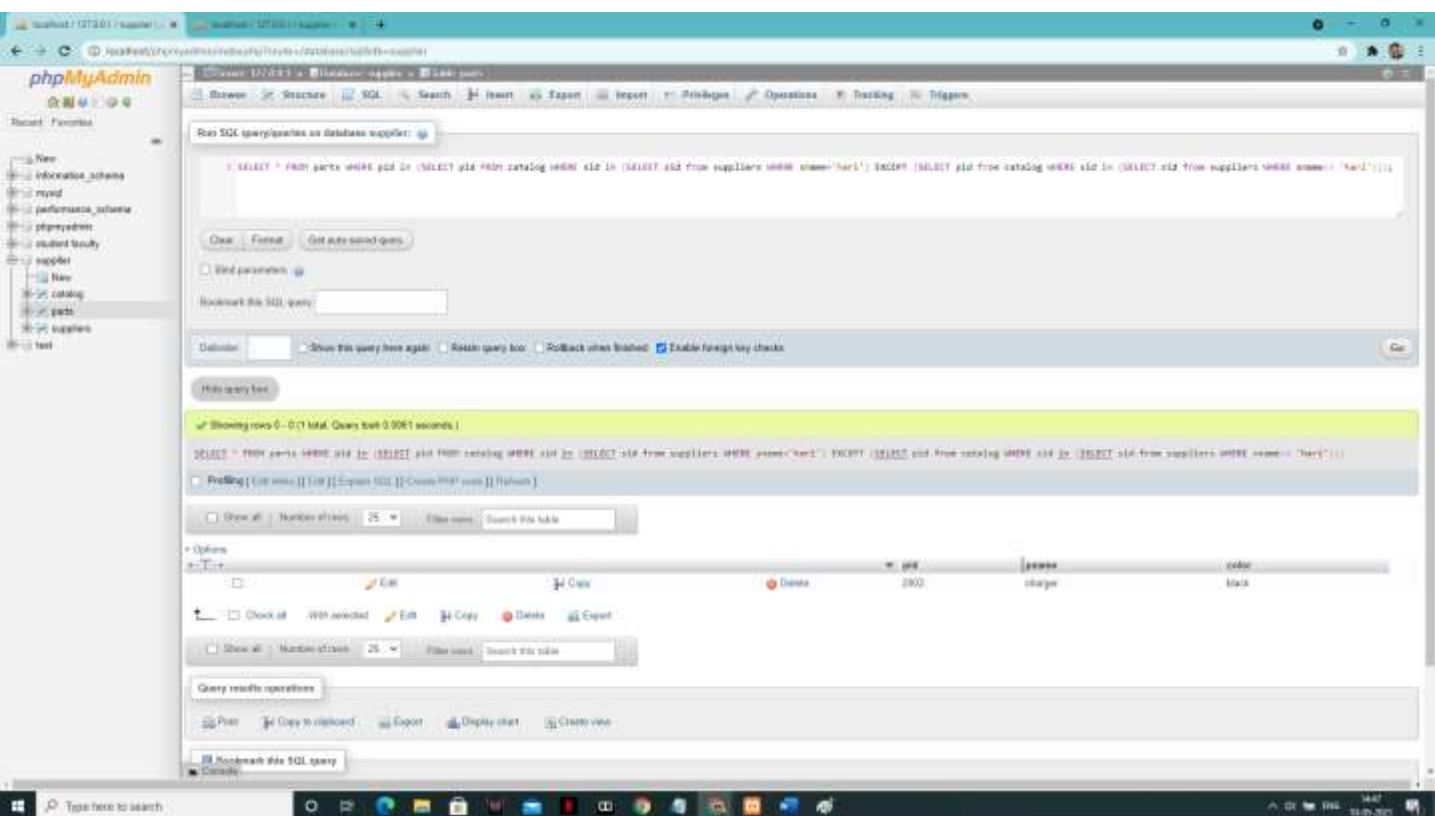
ii. Find the snames of suppliers who supply every part.



iii. Find the snames of suppliers who supply every red part.



iv. Find the pnames of parts supplied by hari Suppliers and by no one else.



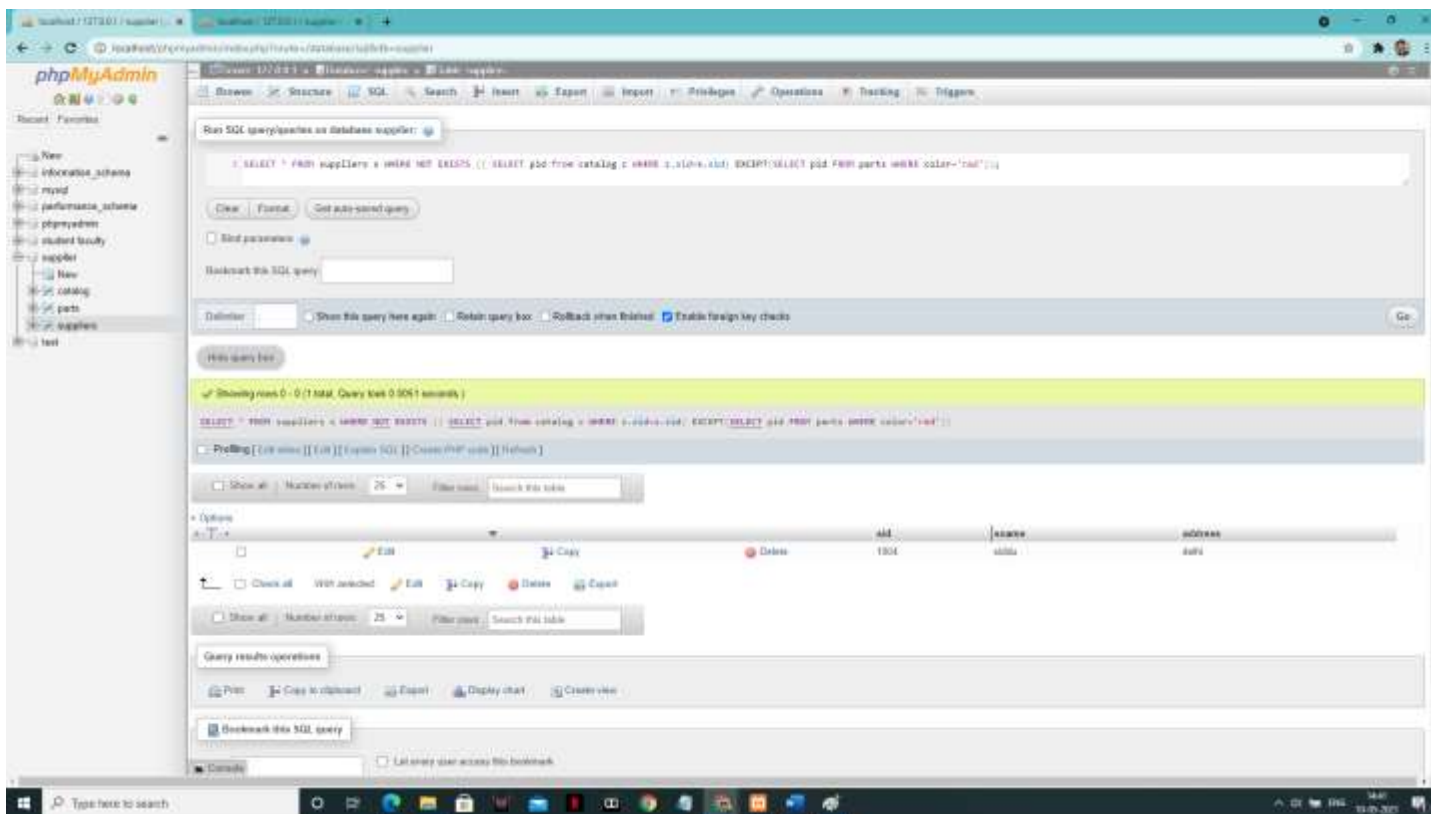
The screenshot shows the phpMyAdmin interface with a SQL query entered in the query editor. The query is:

```
SELECT * FROM suppliers s, parts p, catalog c WHERE s.sid=c.sid AND c.sid=p.sid AND cost < (SELECT MAX(cost) FROM catalog c1 WHERE c1.pid=p.pid)
```

The query has been executed, and the results are displayed in a table. The table has 9 columns: sid, sname, address, pid, pname, price, and two unnamed columns for cost. The results show 10 rows of data.

sid	sname	address	pid	pname	price	cost	cost	cost
1001	test	test	2001	chip	1001	2001	2001	10000
1001	test	test	2002	charger	1001	2002	2002	1000
1001	test	test	2003	module	1001	2003	2003	10000
1001	test	test	2004	handphone	1001	2004	2004	2000
1001	test	test	2005	card	1001	2005	2005	10000
1002	prawan	bangkulu	2001	chip	1002	2001	2001	10000
1002	prawan	bangkulu	2006	card	1002	2006	2006	10000
1002	prawan	marudi	2002	handphone	1002	2002	2002	2000

vii. Find the sids of suppliers who supply only red parts.



LAB PROGRAM 4:-

LAB-4 (program)

PROGRAM 4: STUDENT FACULTY DATABASE

Consider the following database for student enrollment for course :

STUDENT (snum: integer, sname: string, major: string, lvl: string, age: integer)

CLASS (cname: string, meets at: time, room: string, fid: integer)

ENROLLED (snum: integer, cname: string)

FACULTY (fid: integer, fname: string, deptid: integer)

Write the following queries in SQL. No duplicates should be printed in any of the answers.

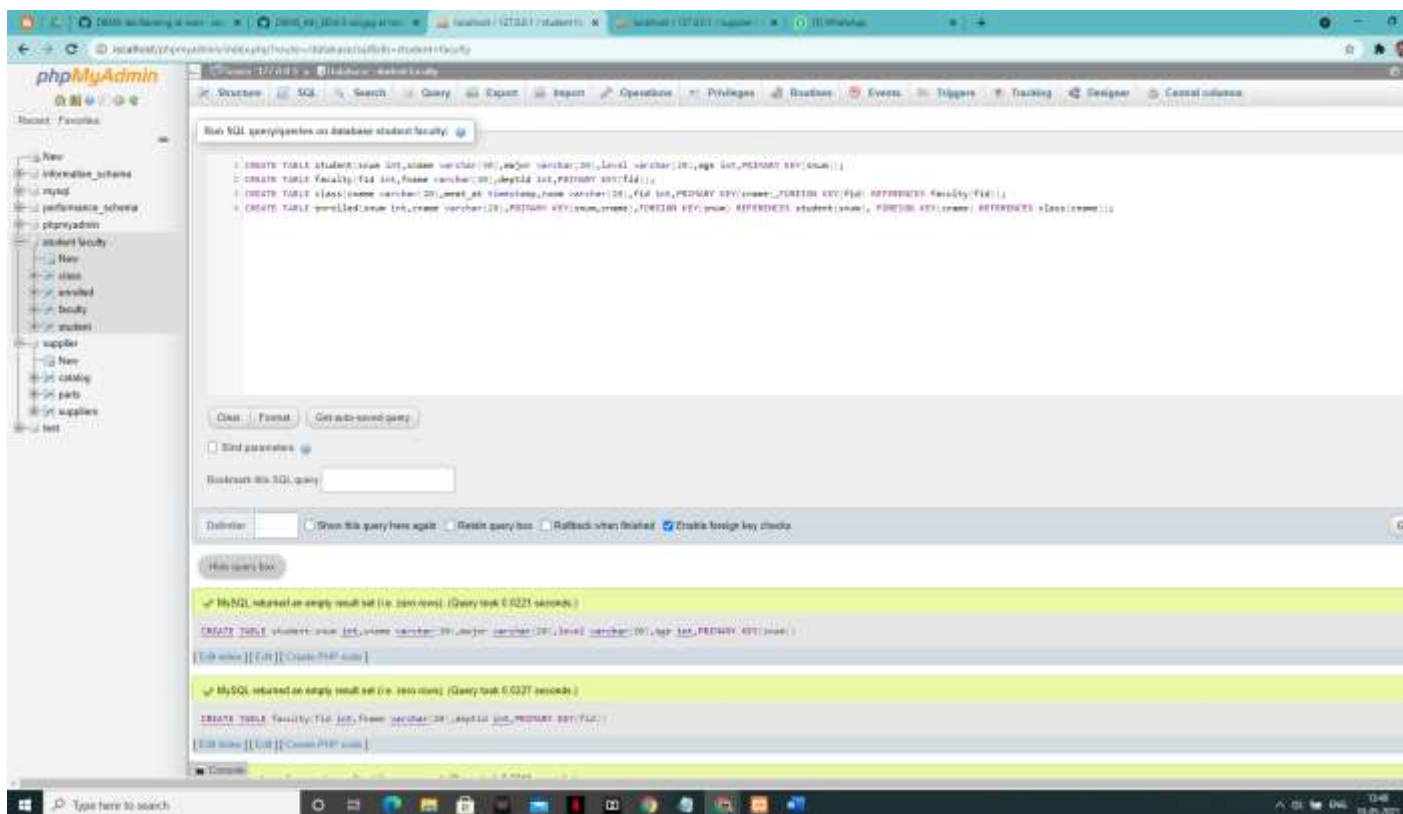
i. Create above mentioned tables

ii. insert records into each of the tables

- 3) Find the names of all Juniors (level = JR) who are enrolled in a class taught by
- 4) Find the names of all classes that either meet in room R128 or have five or more Students enrolled.
- 5) Find the names of all students who are enrolled in two classes that meet at the same time.
- 6) Find the names of faculty members who teach in every room in which some class is taught.
- 7) Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.
- 8) Find the names of students who are not enrolled in any class.
- 9) For each age value that appears in Students, find the level value that appears most often. For example, if there are more FR level students aged 18 than SR, JR, or SO students aged 18, you should print the pair (18, FR).

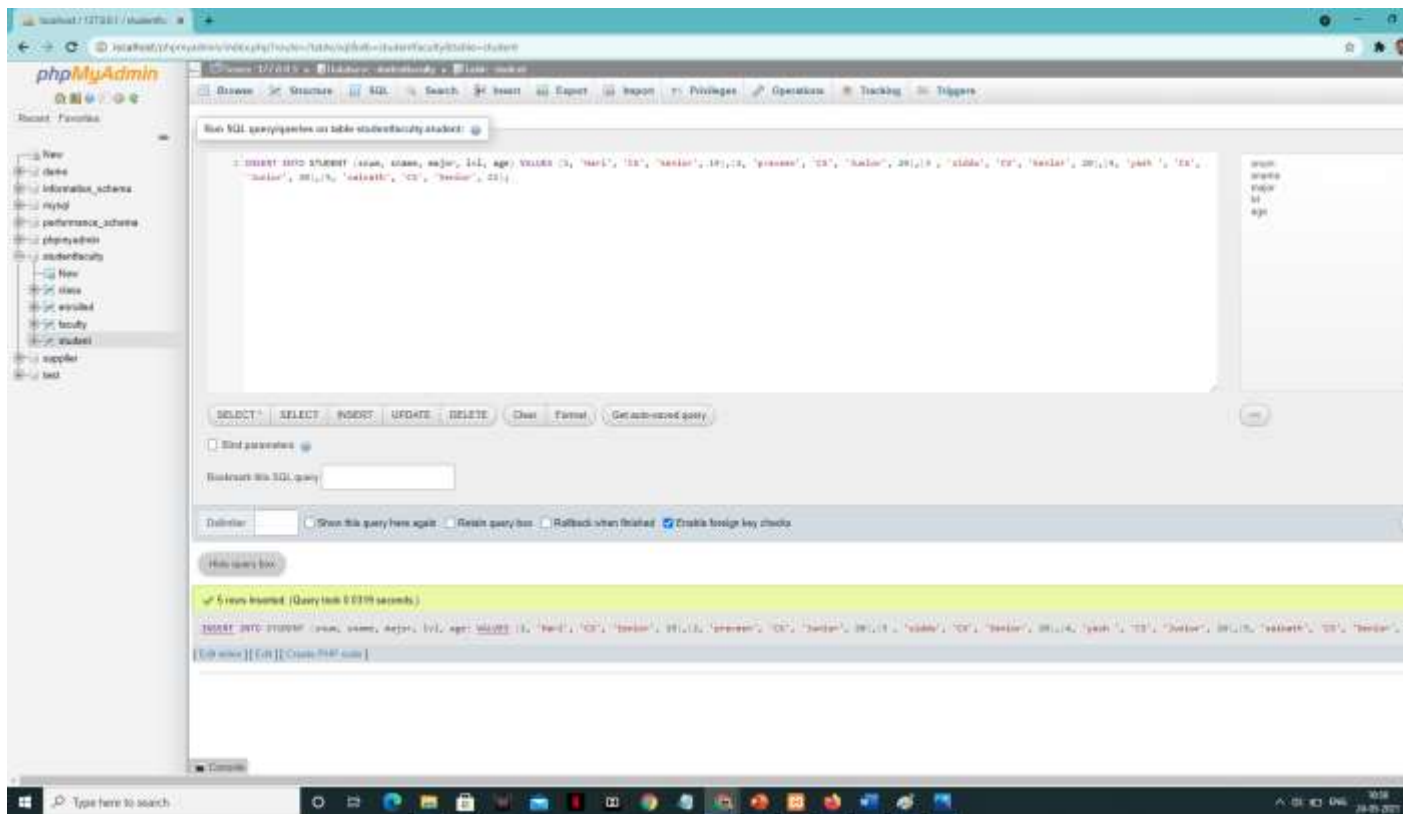
10) Create the above tables by properly specifying the primary keys and the foreign keys.

Create table:-

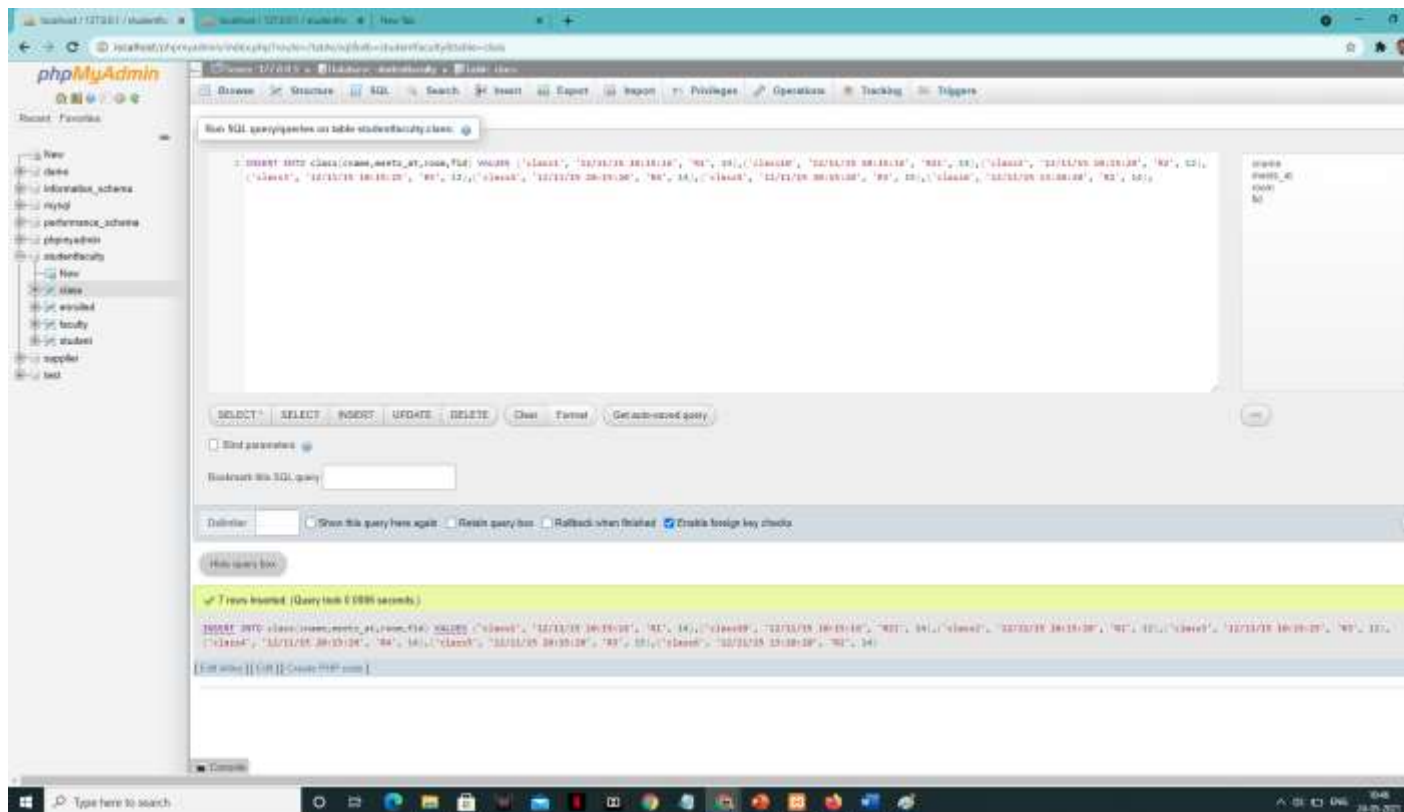


4) Enter tuples for each relation.

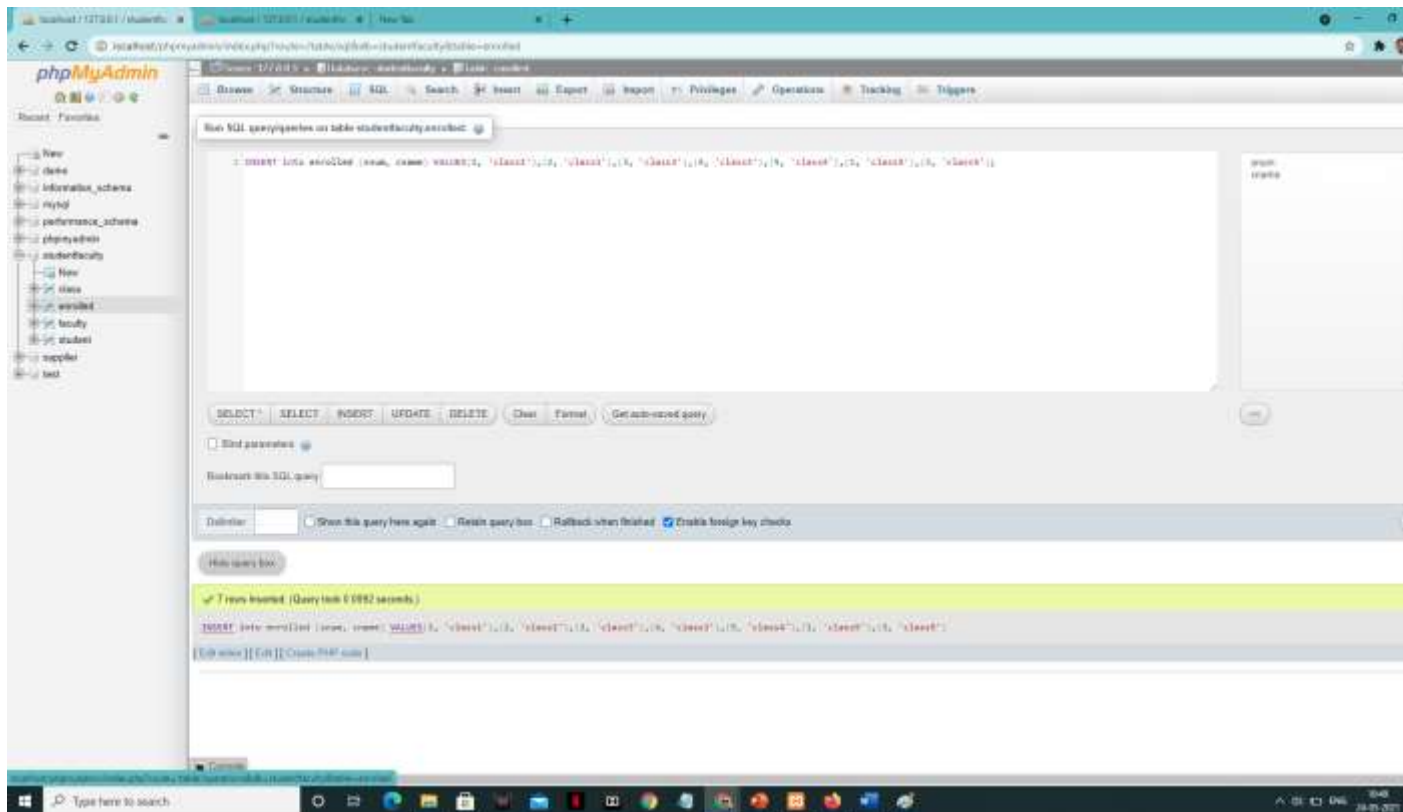
'student' table:



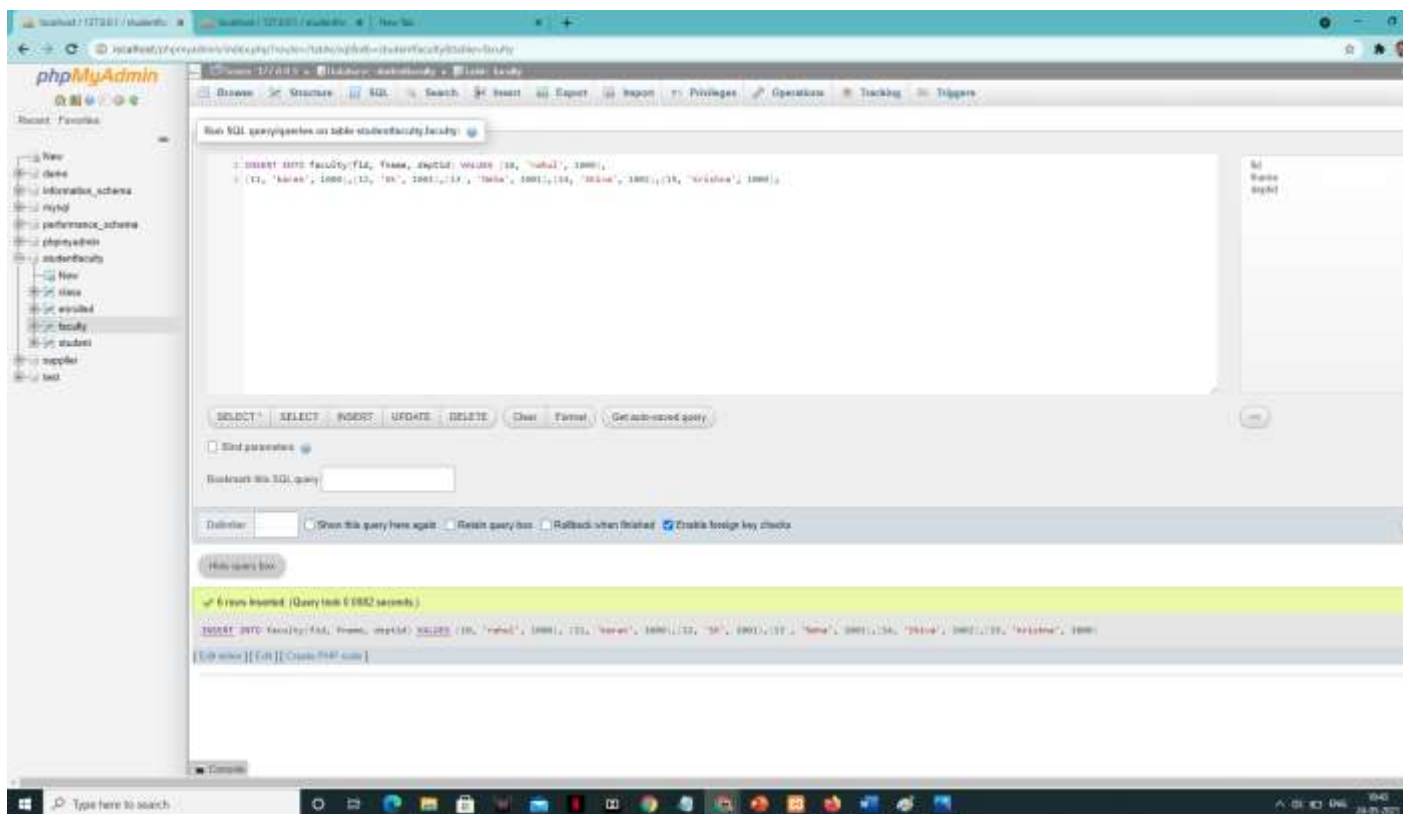
'class' table:



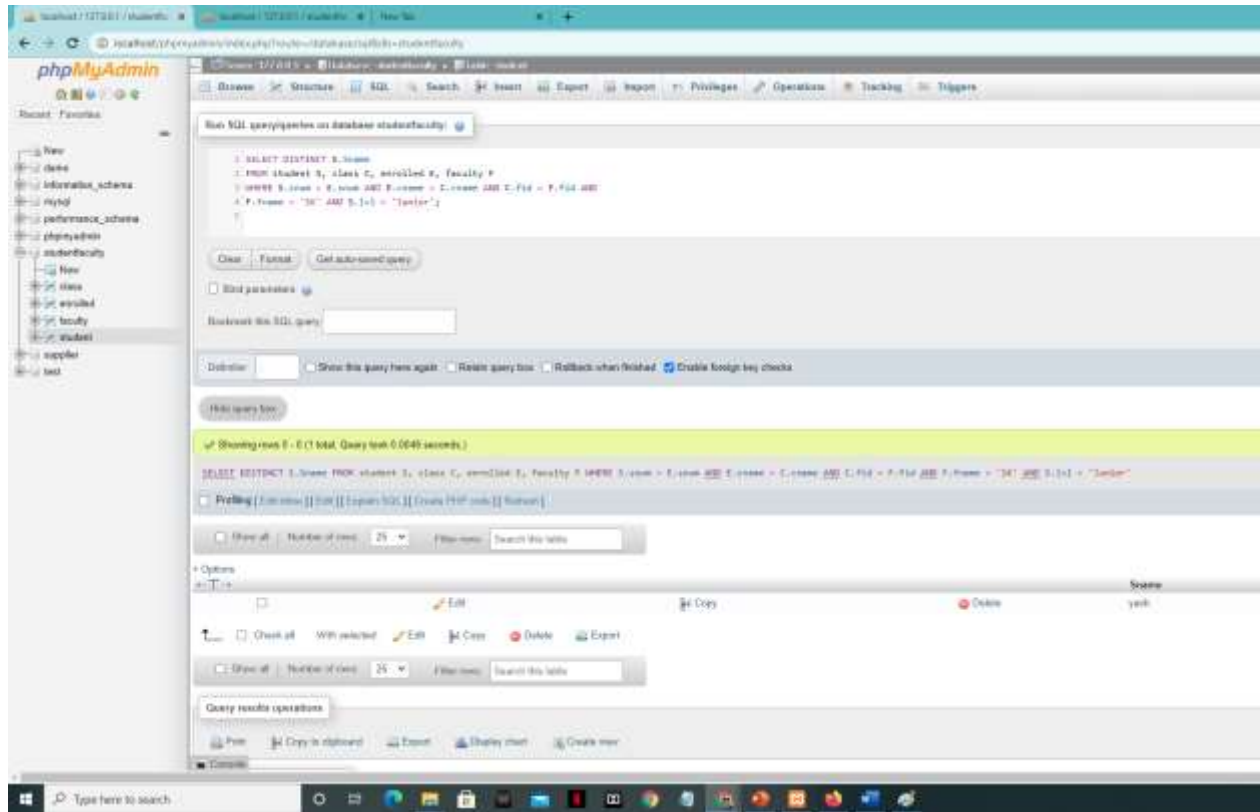
‘enrolled’ table:



Faculty value:-



1:-



2:-

Run SQL queries/queries on database studentfaculty

SQL Query: `SELECT * FROM class as c WHERE c.class="R01" or c.class="R" (SELECT * FROM enrolled GROUP BY class ORDER BY class)`

Showing rows 0 - 6 (1 total. Query took 0.0235 seconds.)

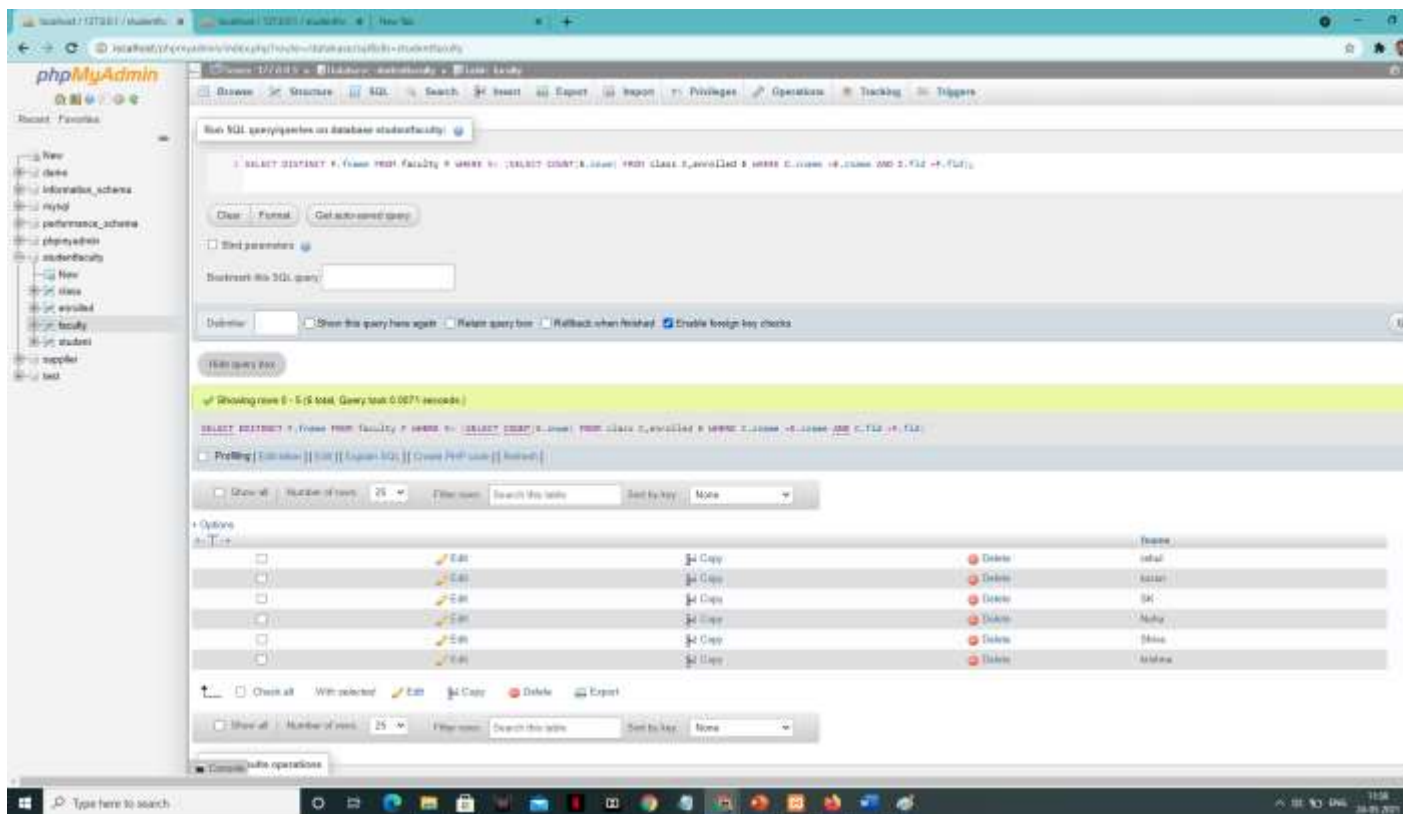
SELECT * FROM class as c WHERE c.class="R01" or c.class="R" (SELECT * FROM enrolled GROUP BY class ORDER BY class)

class	enrolled_at	room	id
class16	2012-11-16 10:10:16	R01	14

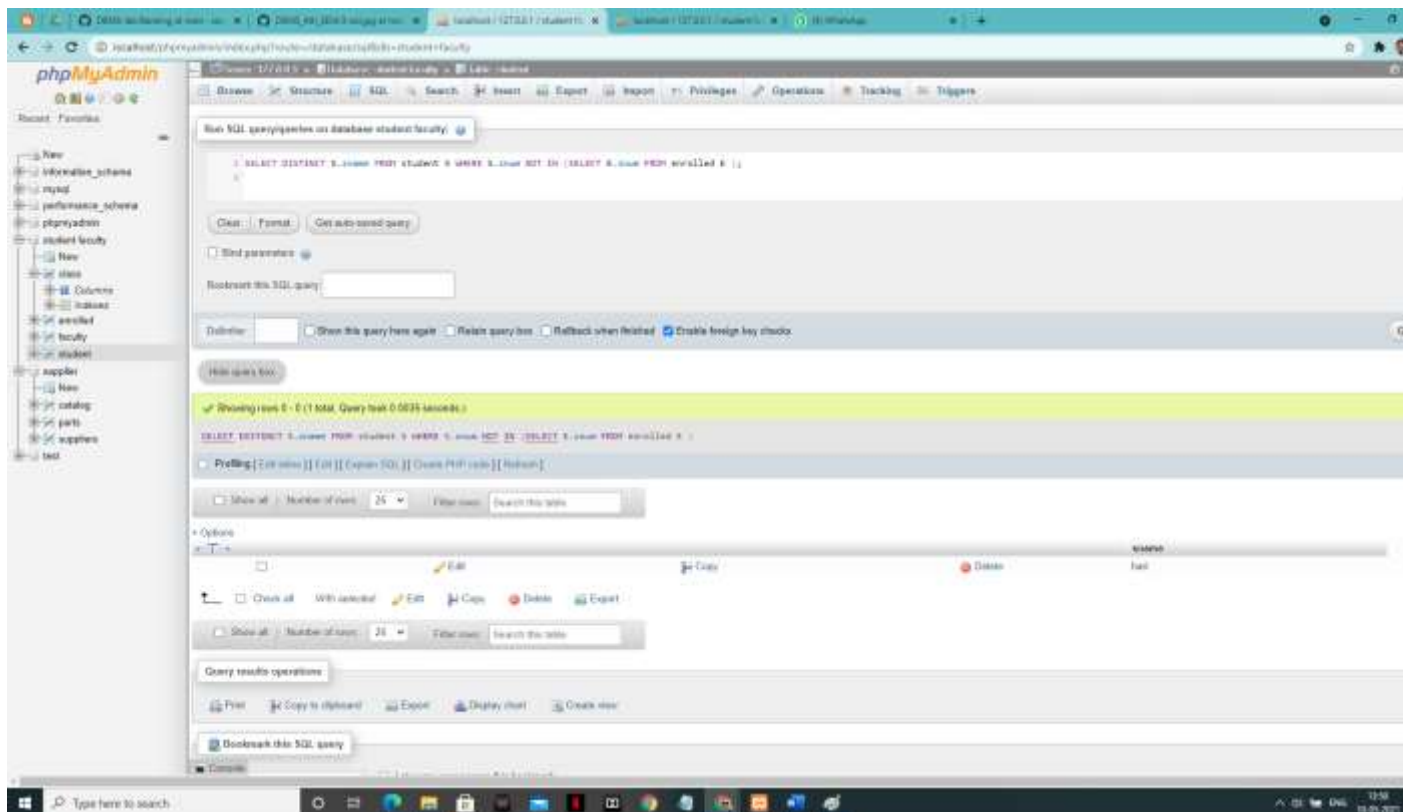
3:-

4:-

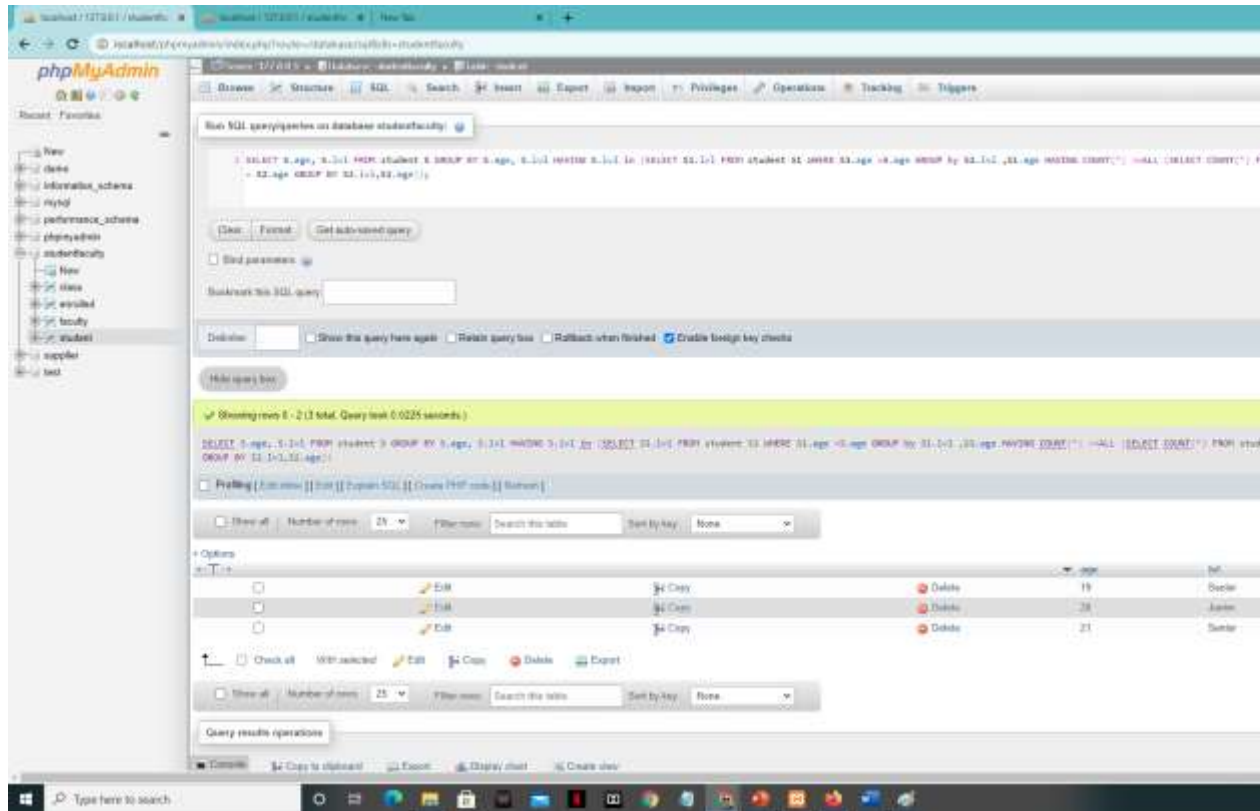
5:-



6:-



7:-



LAB PROGRAM 5:-

LAB-5 (program)

PROGRAM 5: AIRLINE FLIGHTS DATABASE

Consider the following database that keeps track of airline flight information:
FLIGHTS (flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)

AIRCRAFT (aid: integer, aname: string, cruisingrange: integer)

CERTIFIED (eid: integer, aid: integer)

EMPLOYEE (eid: integer, ename: string, salary: integer)

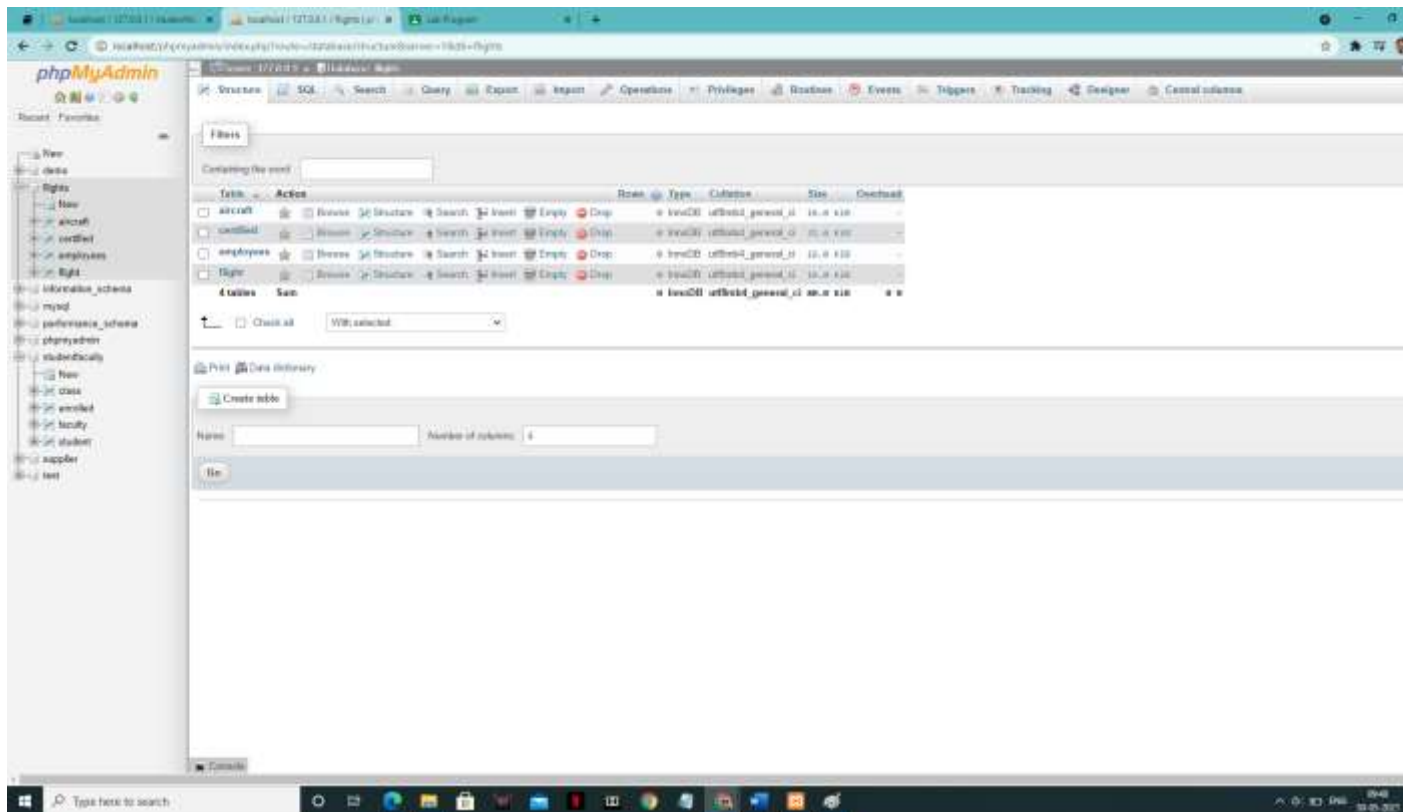
Note that the Employees relation describes pilots and other kinds of employees as well; Every pilot is certified

for some aircraft, and only pilots are certified to fly.

Write each of the following queries in SQL.

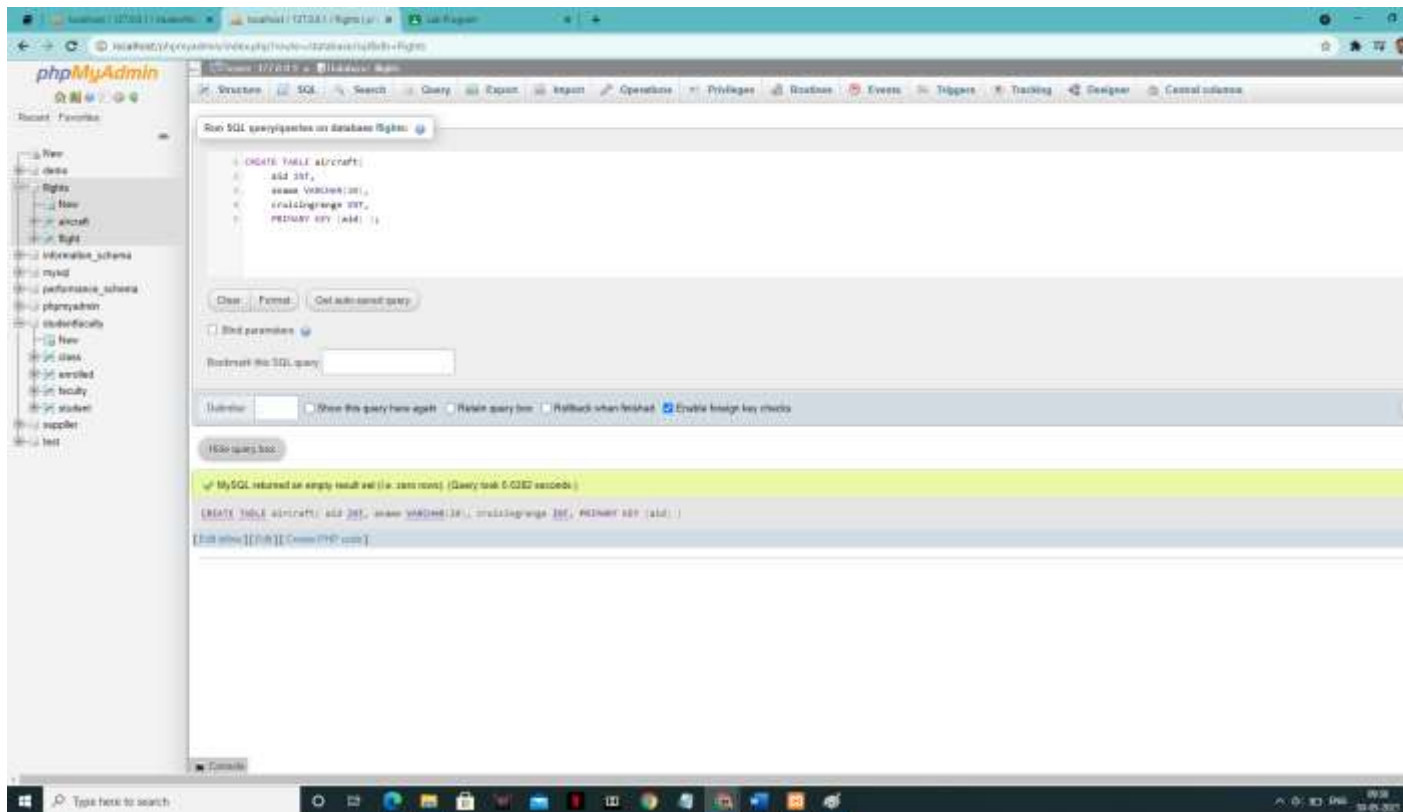
- i. Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80,000.**
- ii. For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruising range of the aircraft for which she or he is certified.**
- iii. Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.**
- iv. For all aircraft with cruising range over 1000 Kms, find the name of the aircraft and the average salary of all pilots certified for this aircraft.**
- v. Find the names of pilots certified for some Boeing aircraft.**
- vi. Find the aids of all aircraft that can be used on routes from Bengaluru to New Delhi.**
- vii. A customer wants to travel from Madison to New York with no more than two changes of flight. List the choice of departure times from Madison if the customer wants to arrive in New York by 6 p.m.**
- viii. Print the name and salary of every non-pilot whose salary is more than the average salary for pilots.**

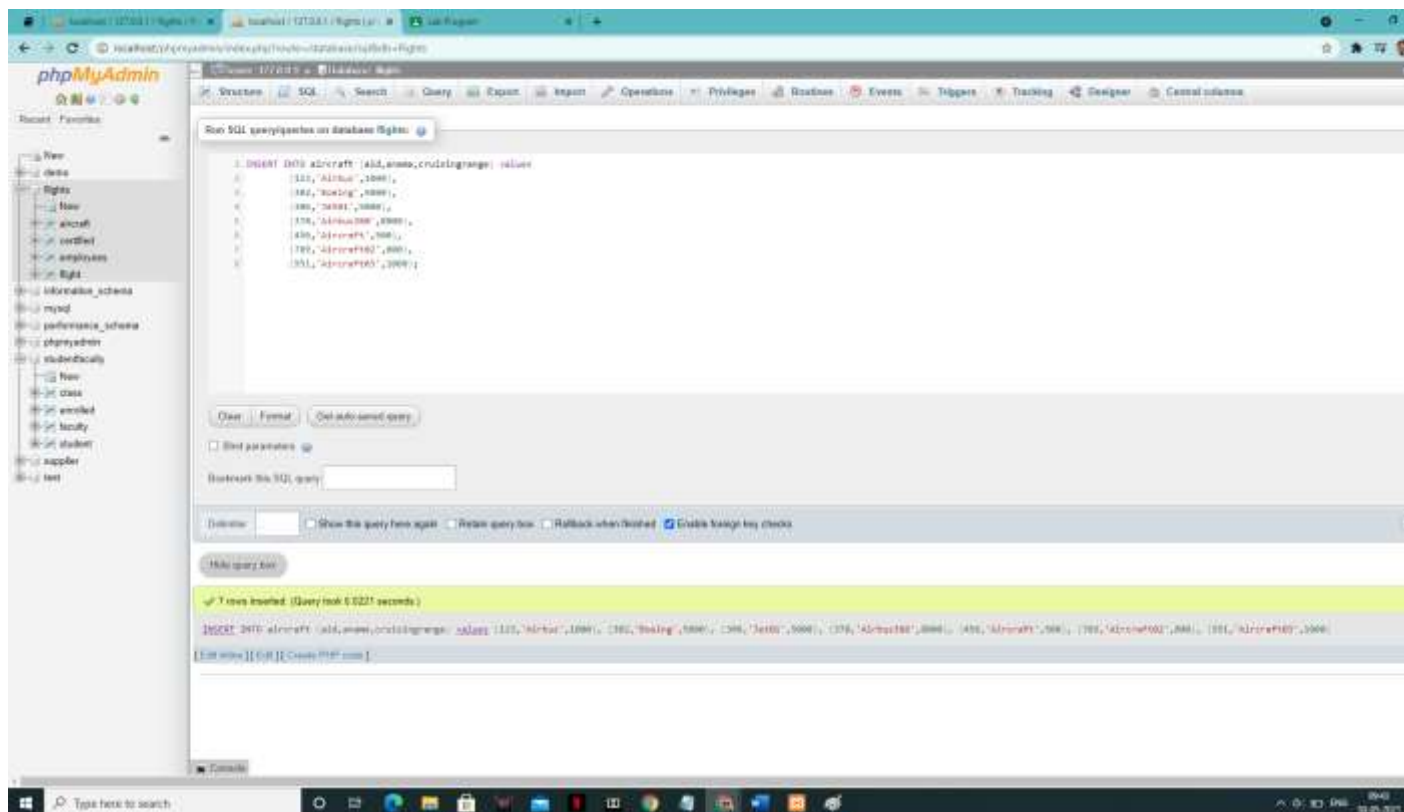
Create table:-



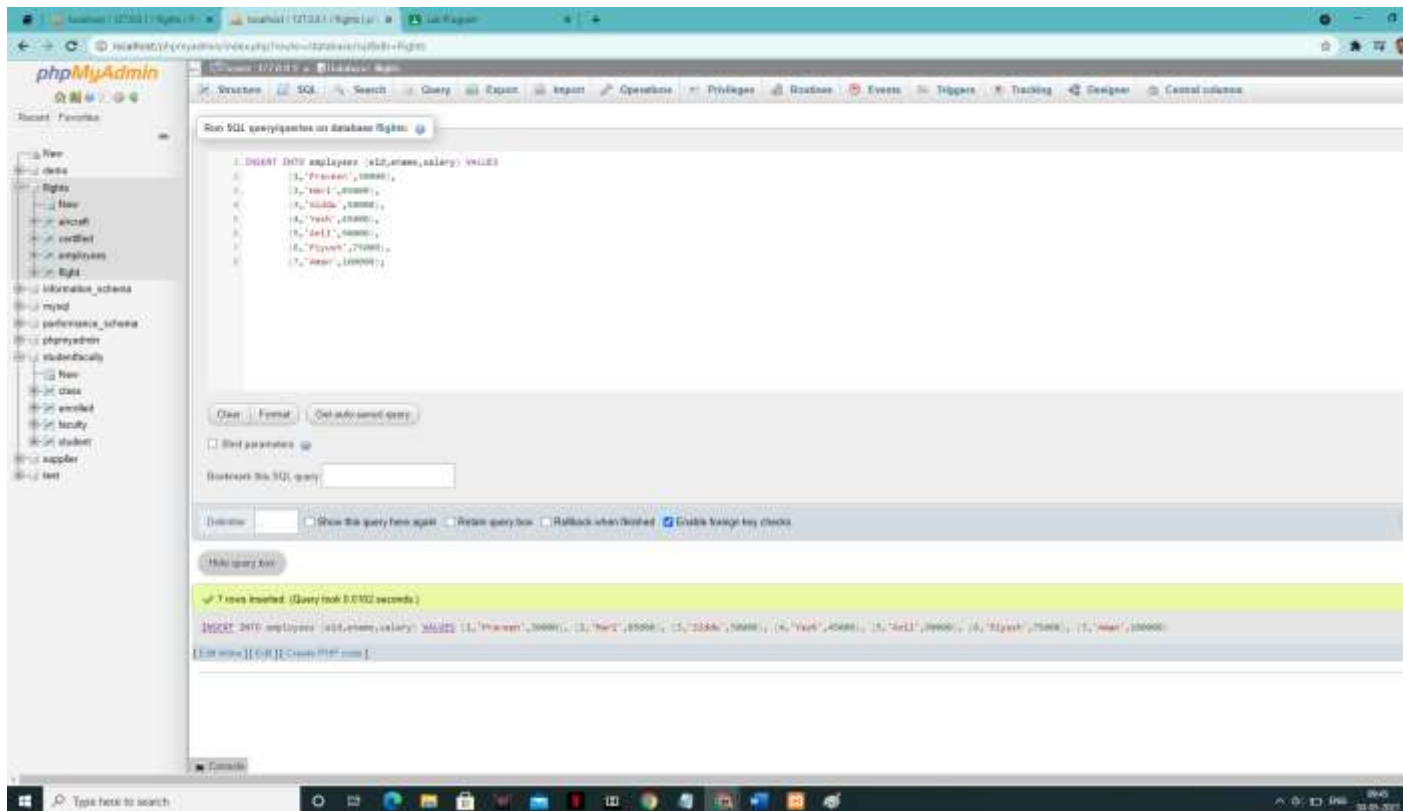
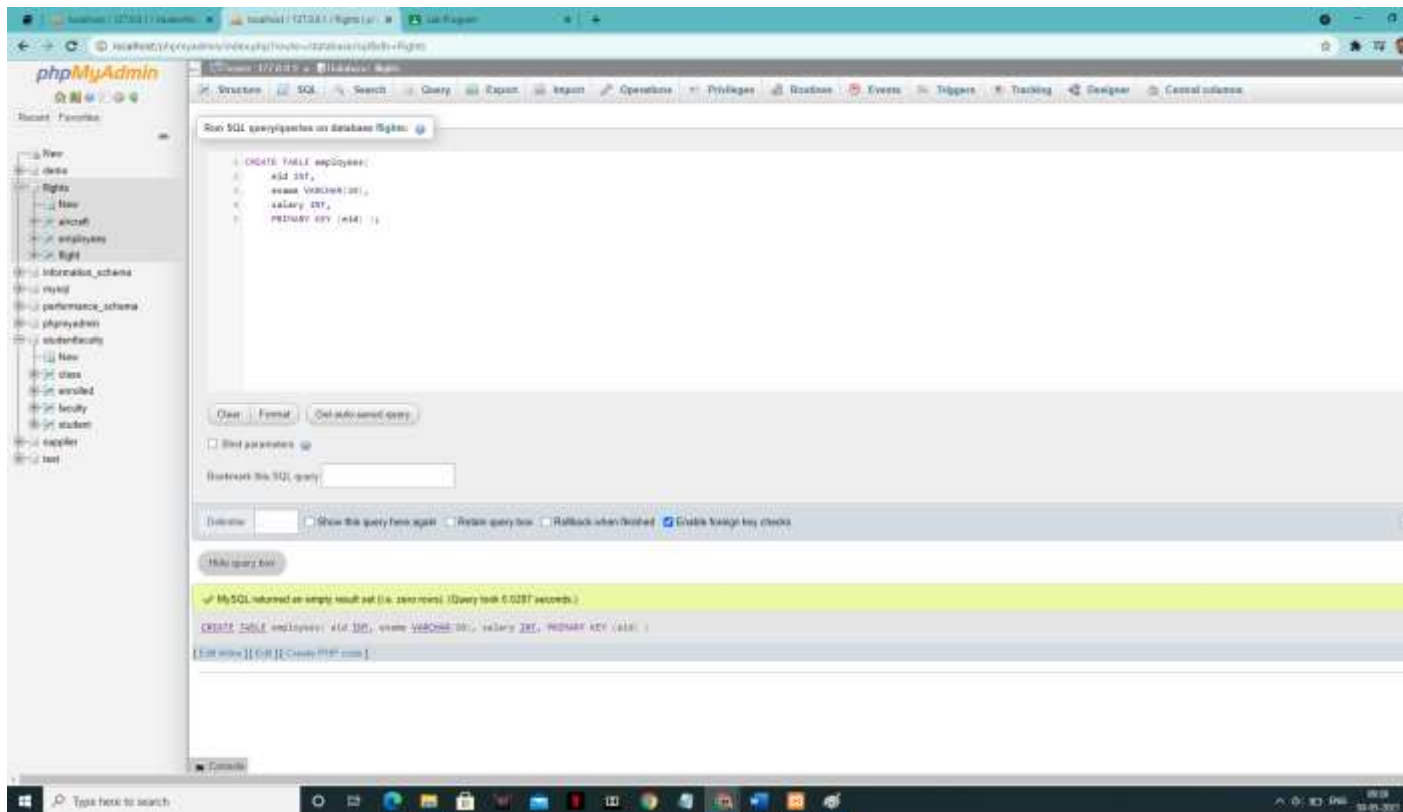
2) Enter tuples for each relation.

‘AIRCRAFT’ table:

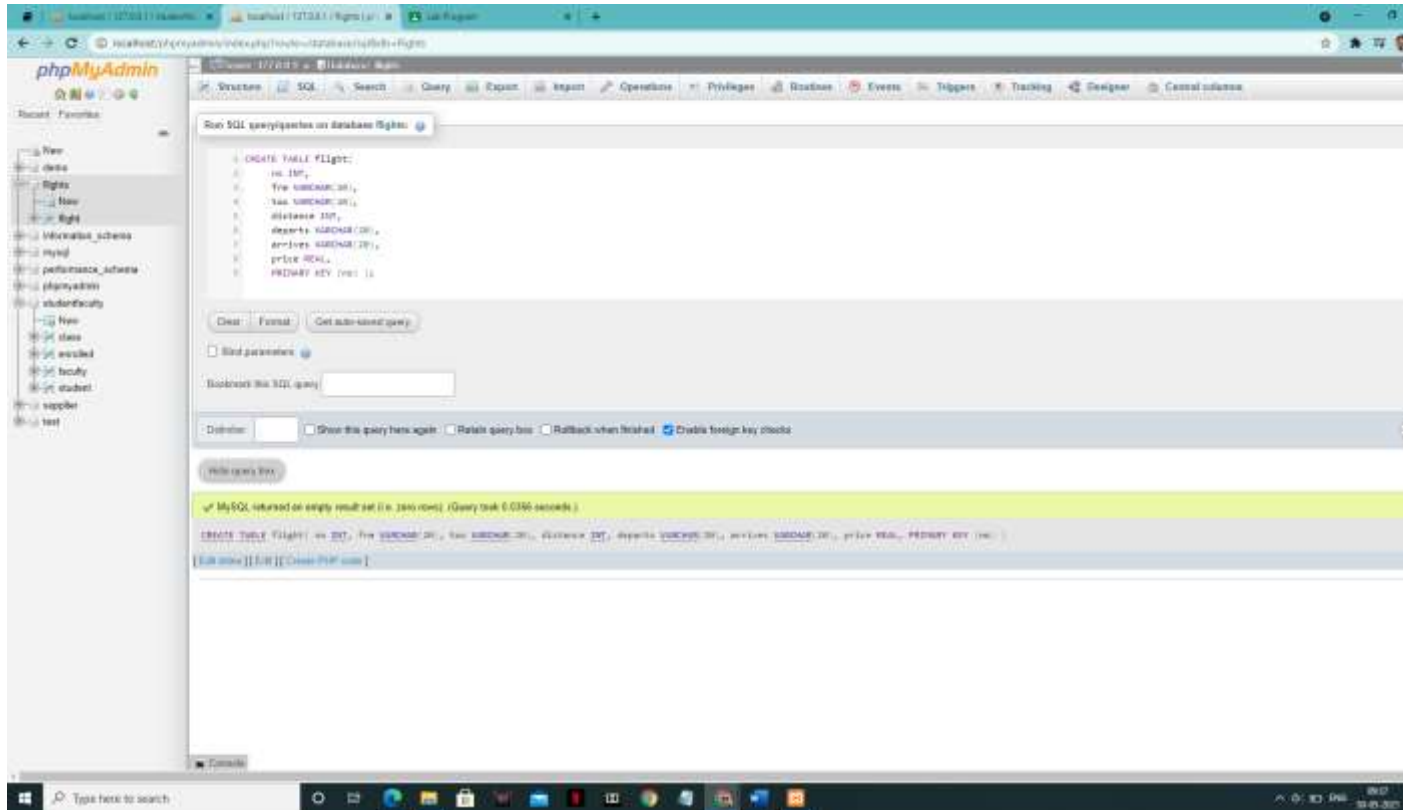


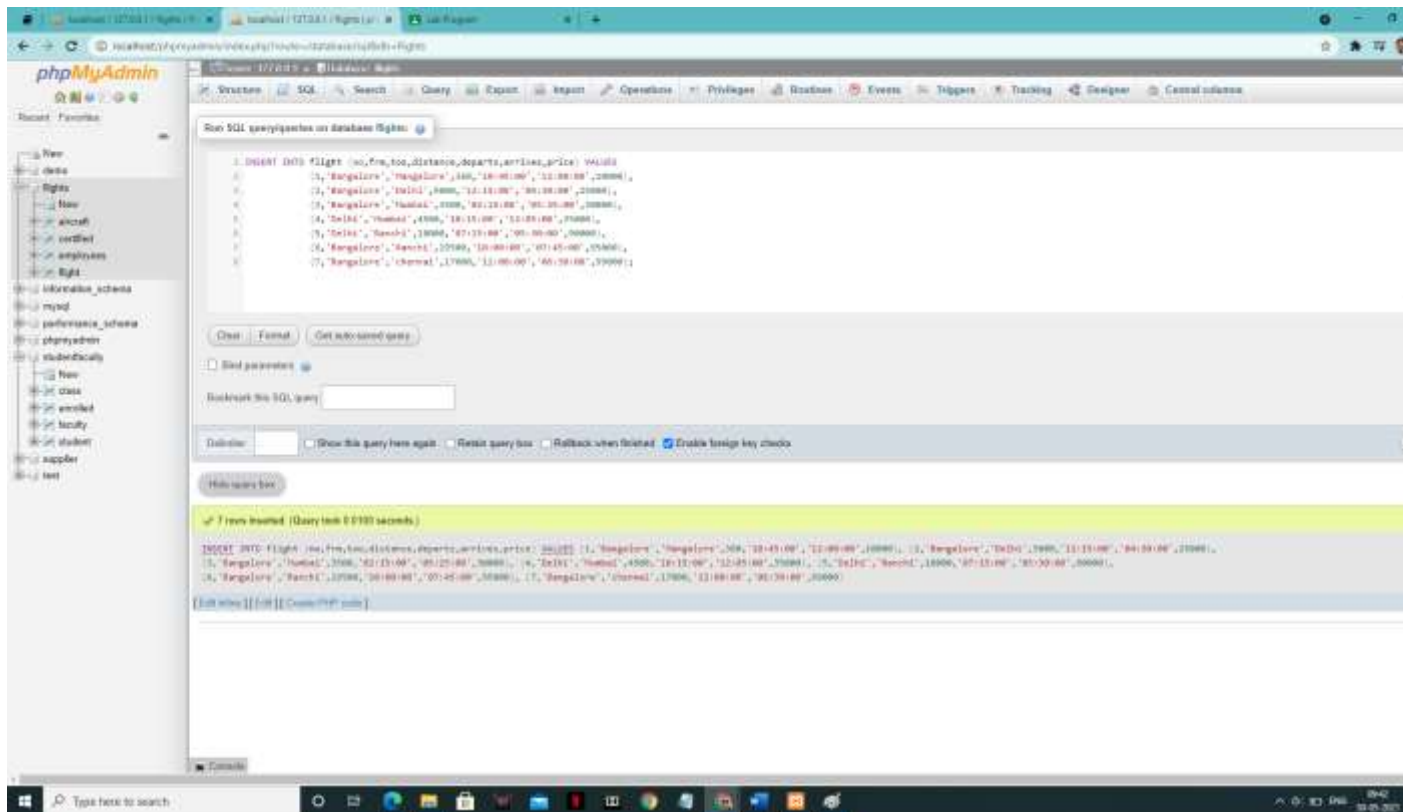


'EMPLOYEES' table:

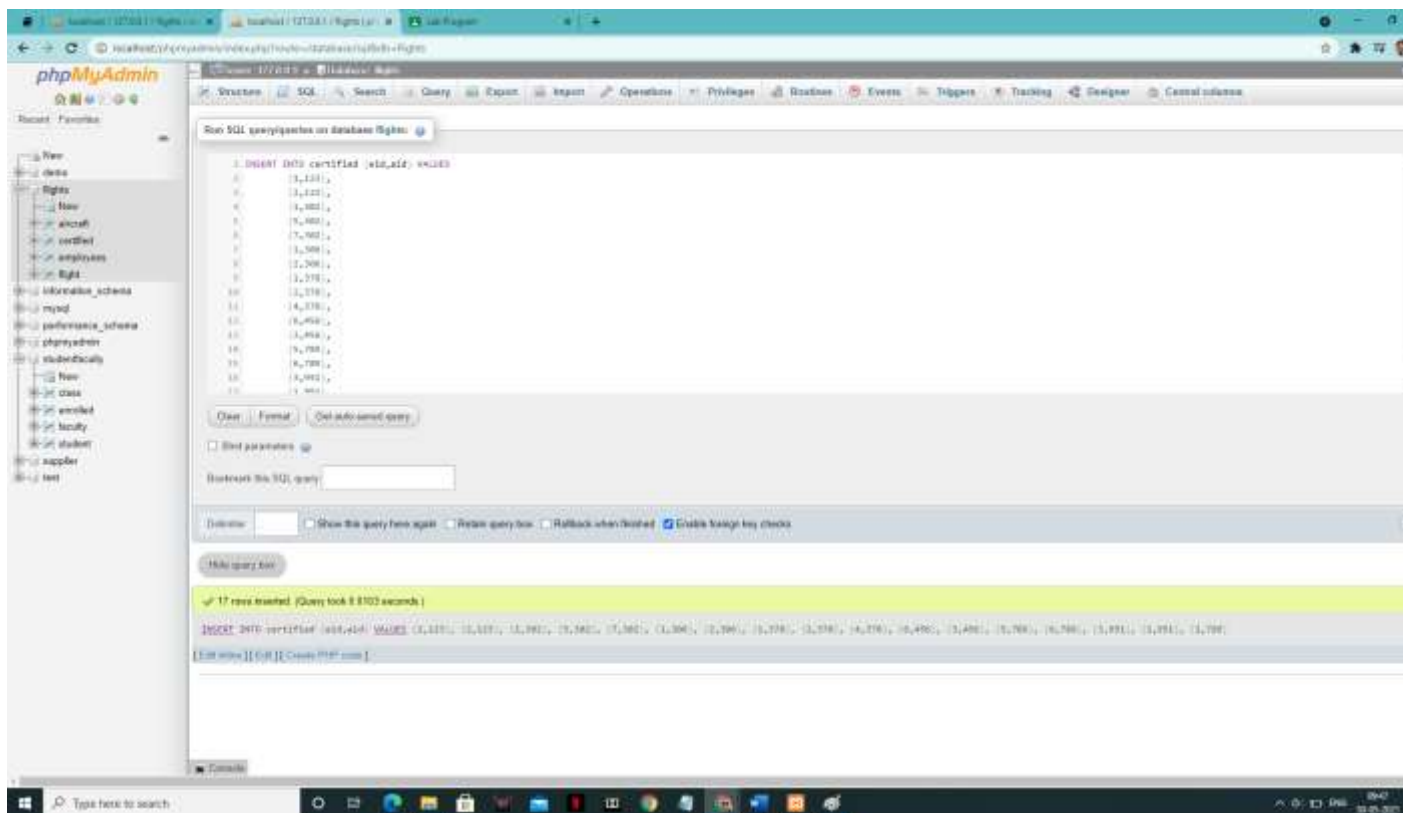


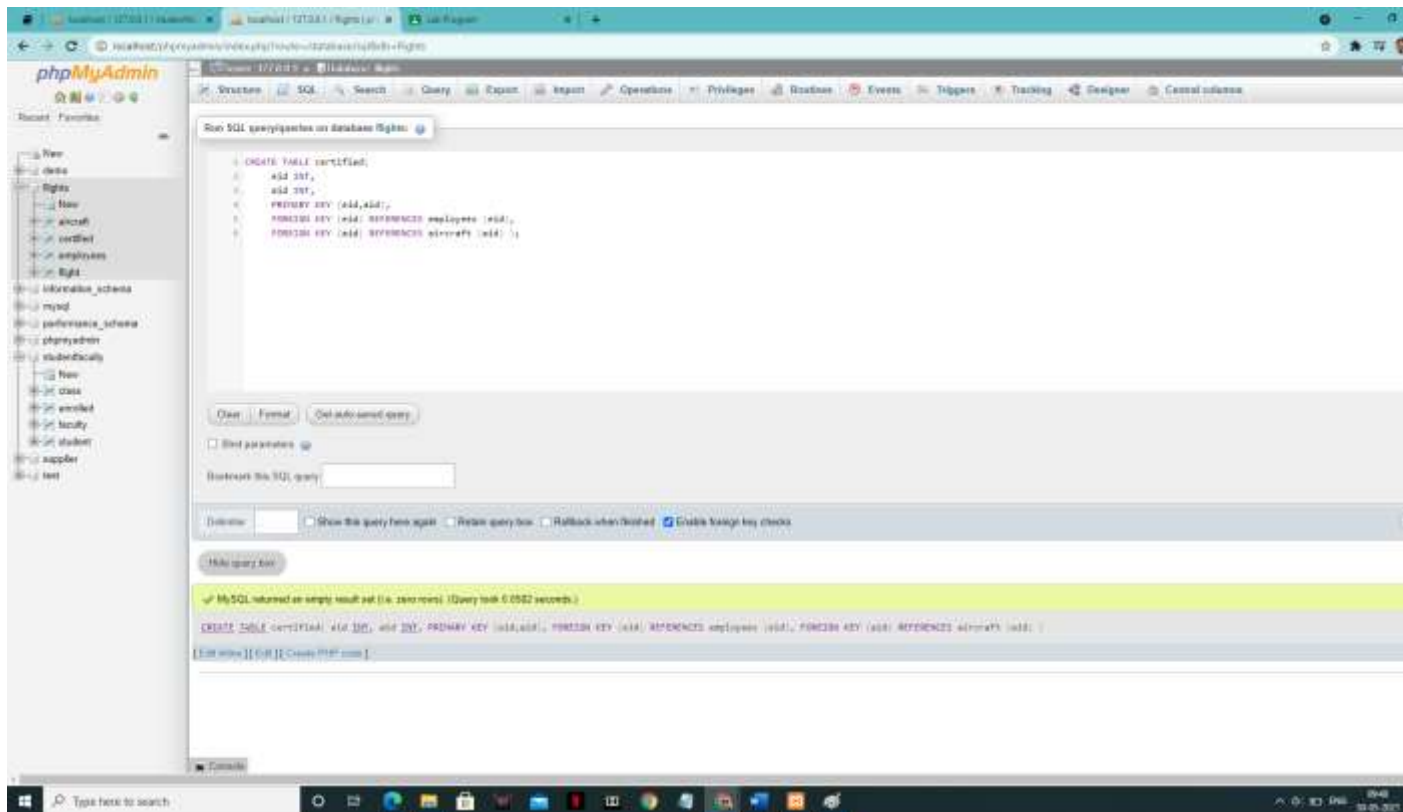
‘FLIGHT’ table:





CERTIFIED value: -





1:-

Run SQL query/queries on database flights:

```
1. SELECT DISTINCT a.name FROM aircraft a,certified c,employee e WHERE a.aid=c.aid AND c.aid=e.aid AND NOT EXISTS (SELECT * FROM employee e1 WHERE e1.aid=e.aid AND e1.salary=99999);
```

Clear Format Get auto-saved query

☐ Edit parameters

Breakdown this SQL query:

Delimiter: Show this query here again: ☐ Retain query log: ☐ Rollback when finished: ☐ Create foreign key checks: ☐

Hide query log

✓ Showing rows 1 - 4 (4 total. Query took 0.0116 seconds)

```
SELECT DISTINCT a.name FROM aircraft a,certified c,employee e WHERE a.aid=c.aid AND c.aid=e.aid AND NOT EXISTS (SELECT * FROM employee e1 WHERE e1.aid=e.aid AND e1.salary=99999);
```

☐ Preview [1 column] [Edit] [Expand SQL] [Create PHP code] [Refresh]

☐ Show all Number of rows: 25 Filter rows Search this table

Options

<input type="checkbox"/>	✎ Edit	📄 Copy	🗑 Delete		name
<input type="checkbox"/>	✎ Edit	📄 Copy	🗑 Delete		Album
<input type="checkbox"/>	✎ Edit	📄 Copy	🗑 Delete		Review
<input type="checkbox"/>	✎ Edit	📄 Copy	🗑 Delete		Artist
<input type="checkbox"/>	✎ Edit	📄 Copy	🗑 Delete		Artist333
<input type="checkbox"/>	✎ Edit	📄 Copy	🗑 Delete		Artist02

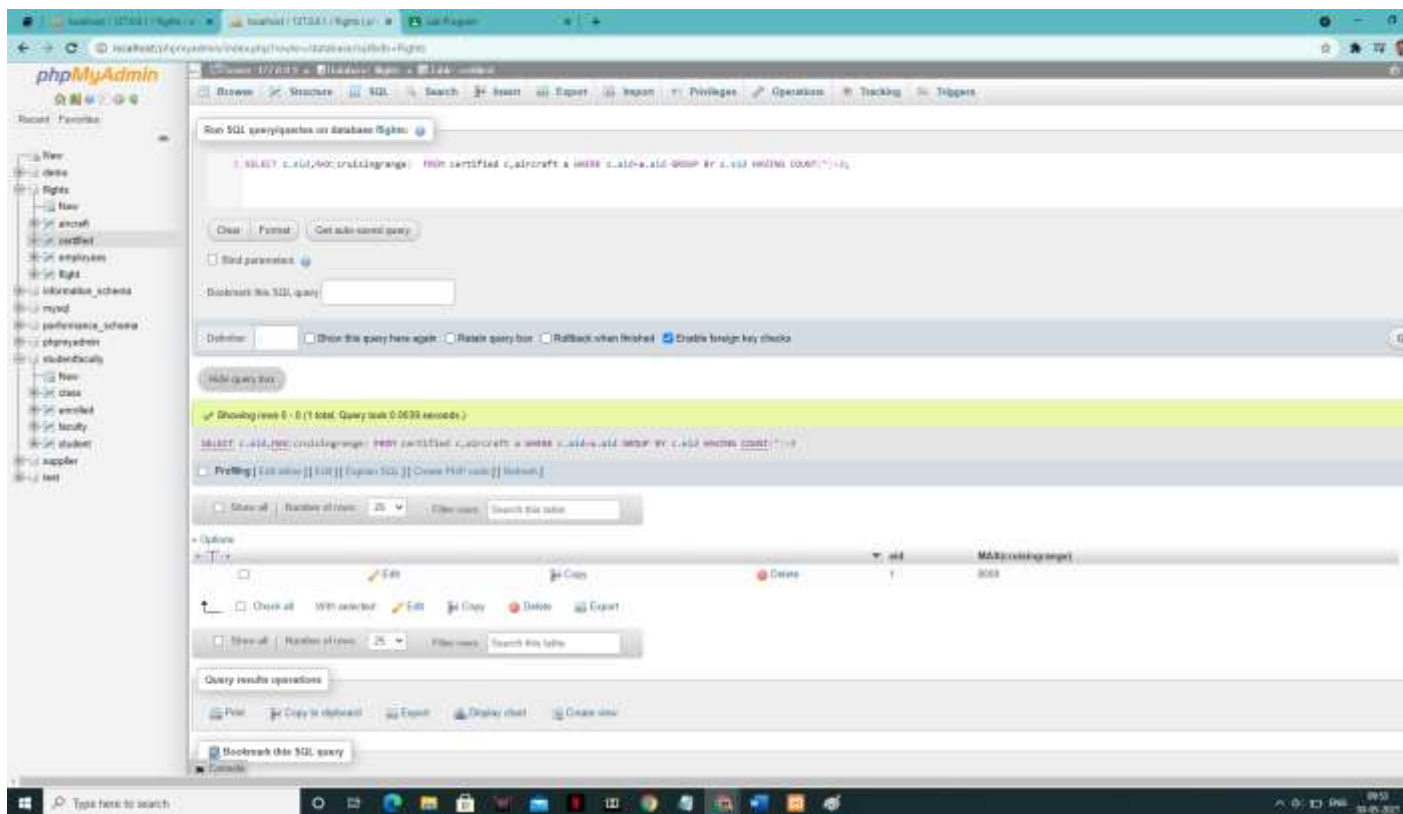
🔍 Check all With selected ✎ Edit 📄 Copy 🗑 Delete 📄 Export

☐ Show all Number of rows: 25 Filter rows Search this table

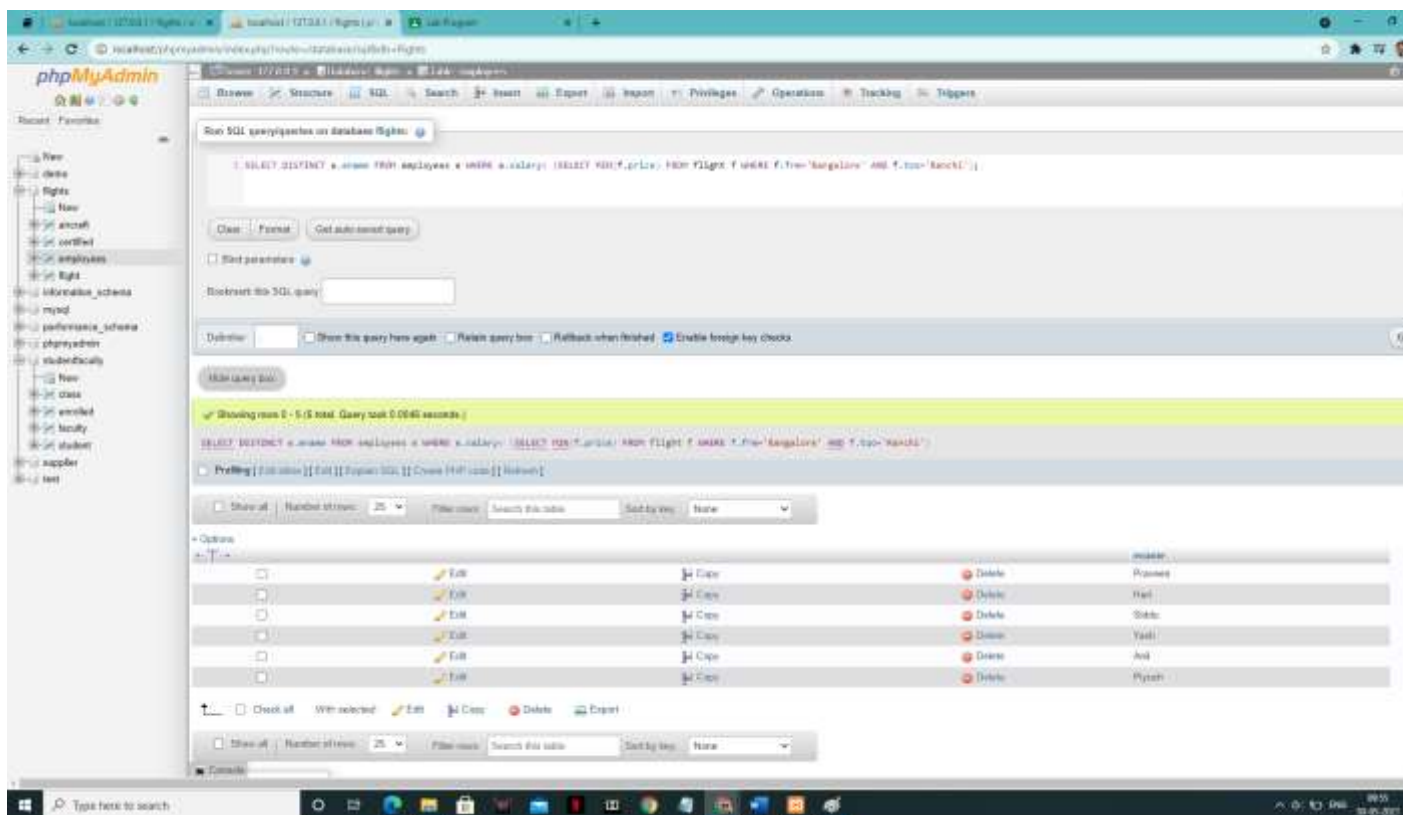
Query results operations

Console

2:-



3:-



4:-

The screenshot shows the phpMyAdmin interface with a SQL query executed. The query is:

```
SELECT a.airc,a.empno,a.salary FROM aircraft a,certified c,employees e WHERE a.airc=c.airc AND c.airc=a.airc AND a.cruisingrange>999 GROUP BY a.airc,a.empno
```

The result shows 3 rows:

airc	empno	salary
332	Swamp	73333 3333
333	ard1	37333 3333
334	Albus DB	33333 3333

5:-

The screenshot shows the phpMyAdmin interface with a SQL query executed. The query is:

```
SELECT distinct a.empno FROM employees a,aircraft c,certified e WHERE a.airc=c.airc AND c.airc=a.airc AND a.empno='Swamp'
```

The result shows 3 rows:

empno
Swamp
ard1
Albus

[illegible]

8:-

The screenshot shows the phpMyAdmin web interface. The left sidebar displays a database structure with various tables like 'airlines', 'aircraft', 'certified', 'employees', 'flight', 'information_schema', 'performance_schema', 'phpmyadmin', 'students', 'supplier', and 'test'. The main panel shows a SQL query editor with the following query:

```
SELECT NAME,SALARY FROM EMPLOYEES WHERE EID NOT IN(SELECT EID FROM CERTIFIED); AND SALARY<(SELECT MAX(SALARY) FROM EMPLOYEES WHERE EID IN (SELECT EID FROM CERTIFIED));
```

Below the query editor, there are buttons for 'SELECT*', 'SELECT', 'INSERT', 'UPDATE', 'DELETE', 'Clear', 'Format', and 'Get auto-generated query'. There is also a checkbox for 'Show parameters' and a text input for 'Reconnect this SQL query'. Below these are checkboxes for 'Distinct', 'Show this query here again', 'Refresh query box', 'Rollback when finished', and 'Create foreign key checks'. A 'Hide query box' button is also present.

The results section shows a green bar indicating 'Showing rows 0 - 0 (1 total, Query took 0.0043 seconds)'. Below this, the query is repeated. There is a 'Profiling' section with checkboxes for 'Edit view', 'SQL', 'Check HTTP code', and 'Refresh'. Below the profiling section are buttons for 'Show all', 'Number of rows', '25', 'Other rows', and 'Search this table'.

At the bottom, there is a table with columns 'NAME' and 'SALARY'. The table is currently empty. Below the table are buttons for 'Check all', 'With selected', 'Edit', 'Copy', 'Delete', and 'Export'. There is also a 'Show all' button, a 'Number of rows' dropdown set to '25', a 'Filter rows' input, and a 'Search this table' input.