

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)

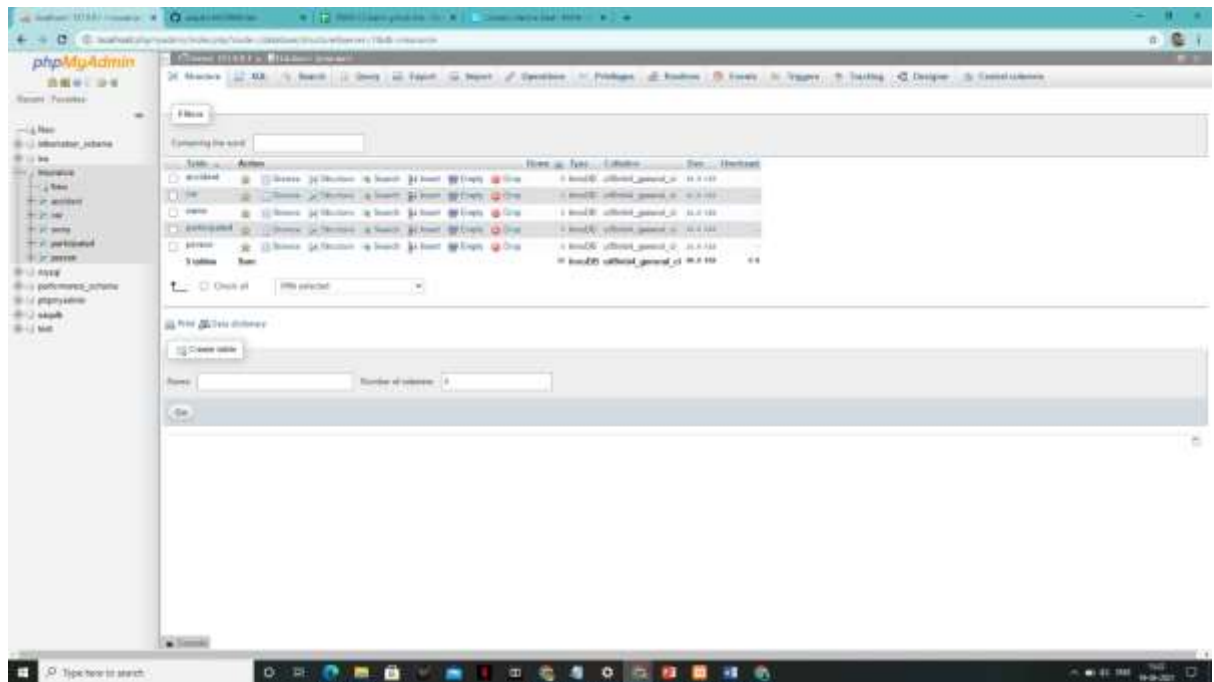
CAR (Regno: String, model: String, year: int)

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

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

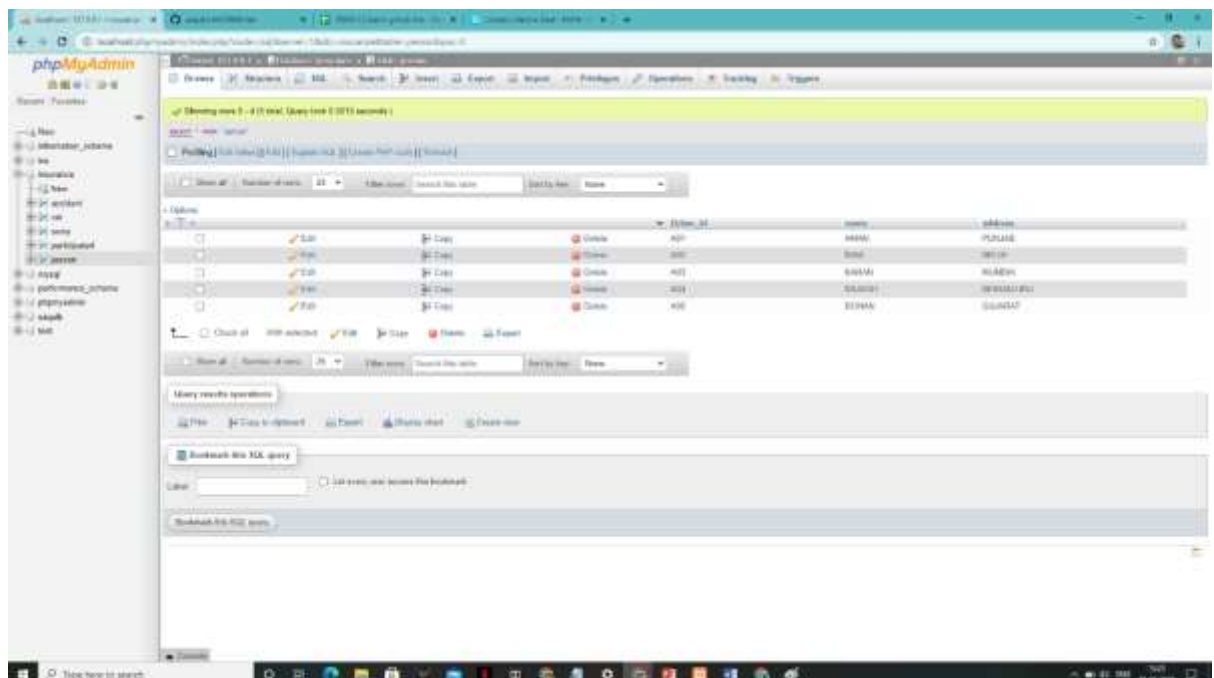
PARTICIPATED (driver-id: String, Regno: String, report-number: int, damage-amount: int)

- 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:

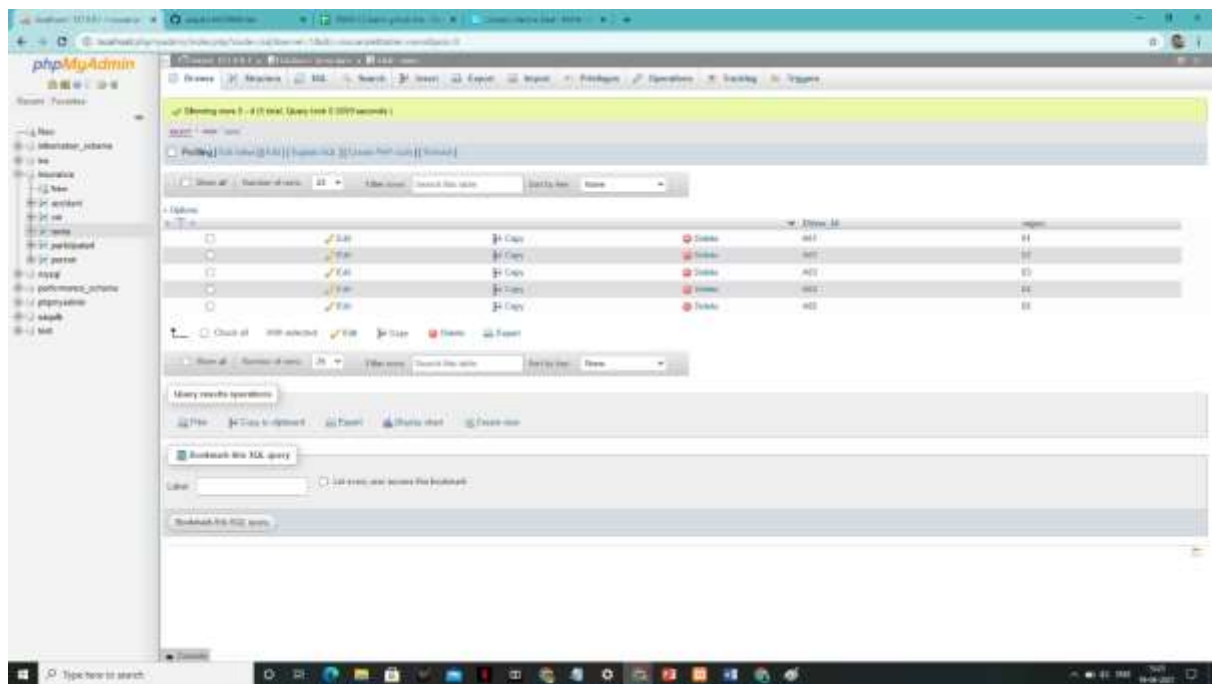
Column	Type	Nullable	Default	Extra
id	int(11)	NO		PRIMARY
name	varchar(255)	NO		
age	int(11)	NO		
sex	enum('M', 'F')	NO		
height	float(10,2)	NO		
weight	float(10,2)	NO		
eye_color	enum('blue', 'green', 'brown', 'grey', 'hazel', 'red')	NO		
hair_color	enum('black', 'brown', 'blond', 'red', 'grey', 'white')	NO		
skin_color	enum('fair', 'tan', 'dark')	NO		

‘ACCIDENT’ table:

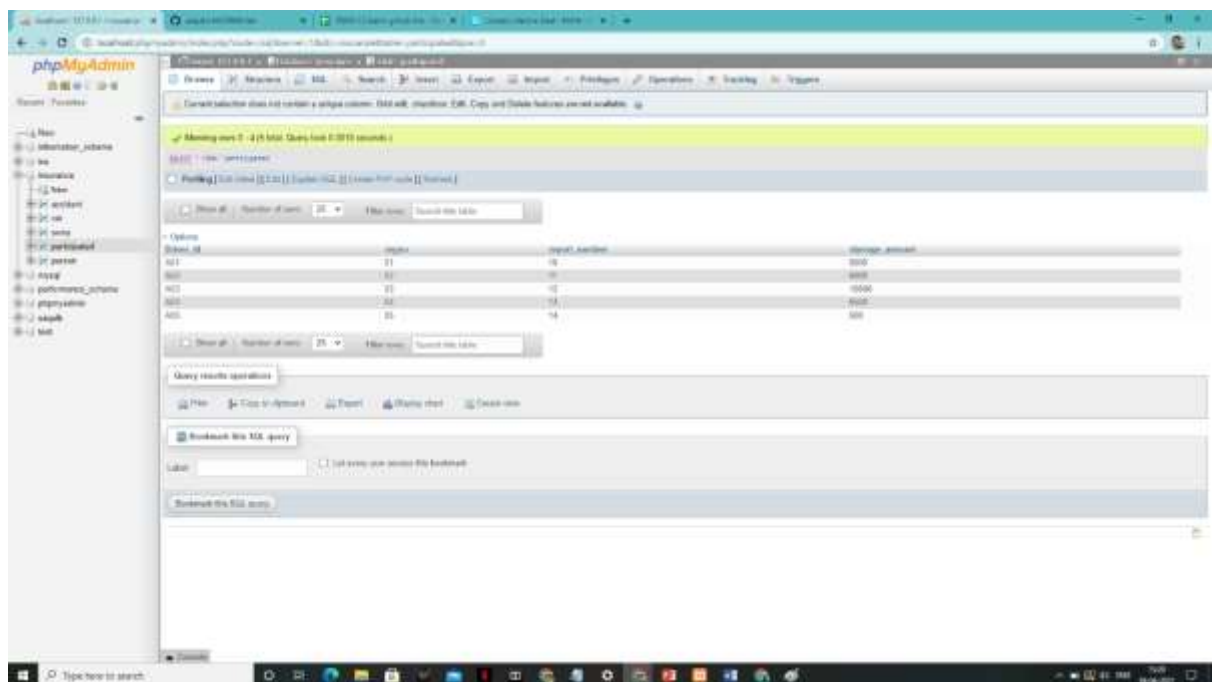
The screenshot shows the phpMyAdmin interface with the 'ACCIDENT' table selected. The table structure is displayed with the following columns:

Column	Type	Nullable	Default	Extra
id	int(11)	NO		PRIMARY
date	date	NO		
location	varchar(255)	NO		
number	int(11)	NO		
driver	enum('M', 'F')	NO		
passenger	enum('M', 'F')	NO		
object	enum('car', 'motorcycle', 'truck', 'bus', 'van', 'other')	NO		
type	enum('collision', 'pedestrian', 'fire', 'other')	NO		

‘OWNS’ table:

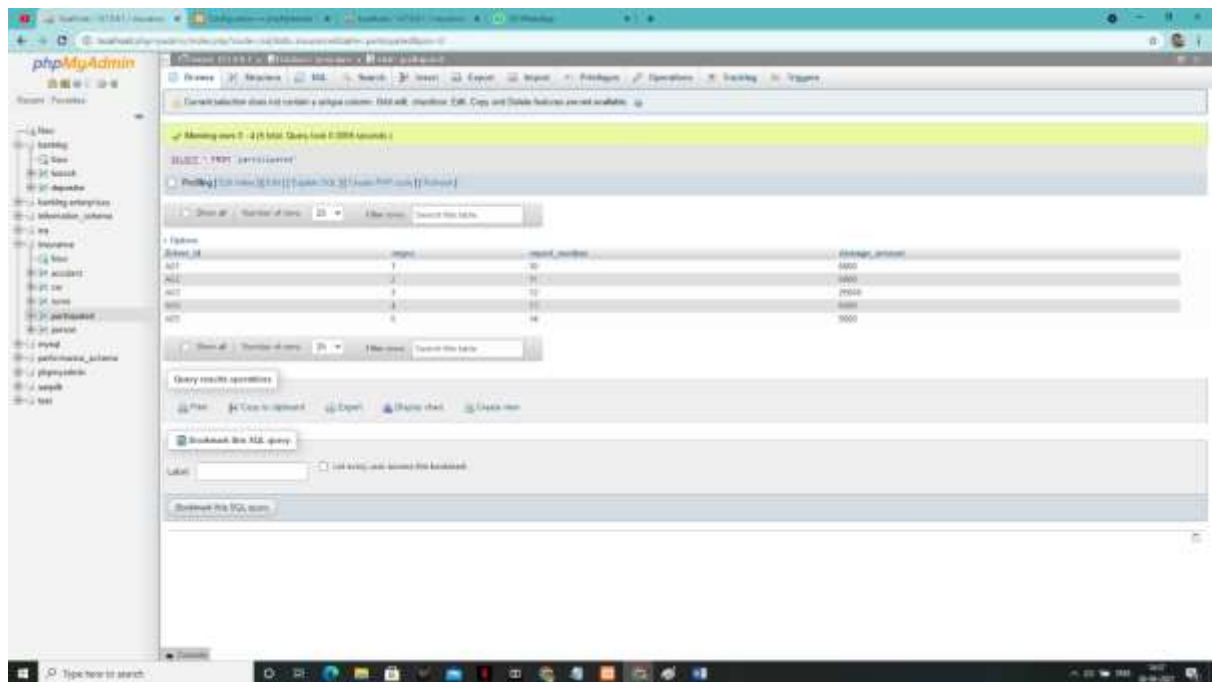


‘PARTICIPATED’ table:

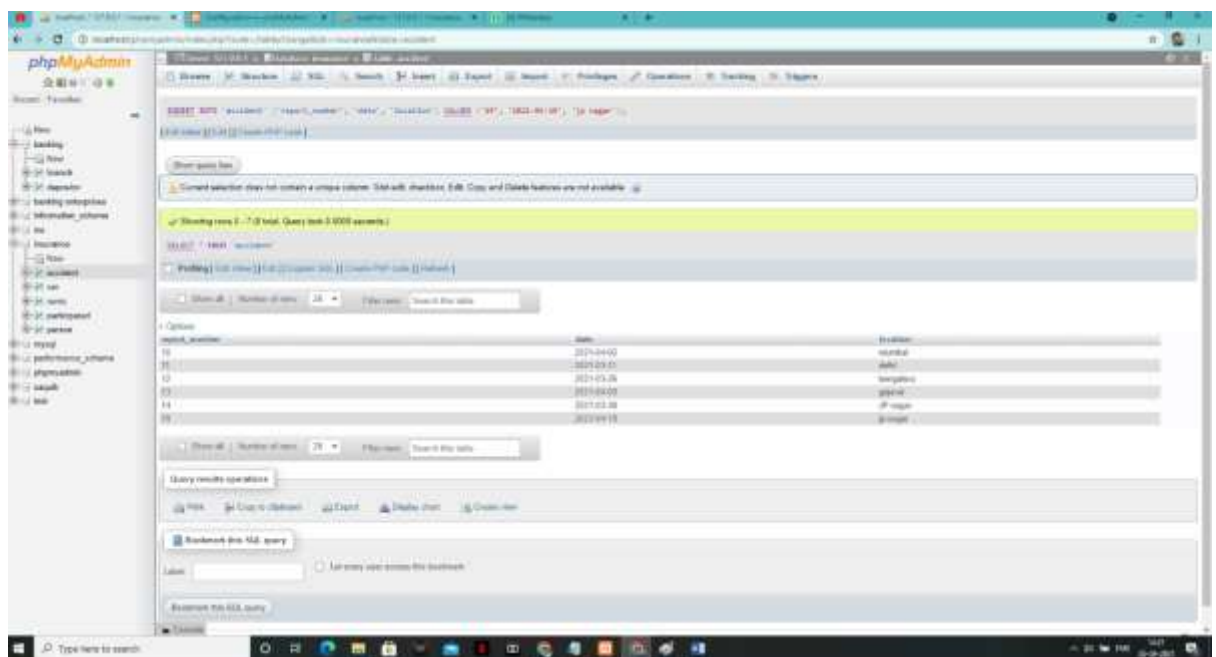
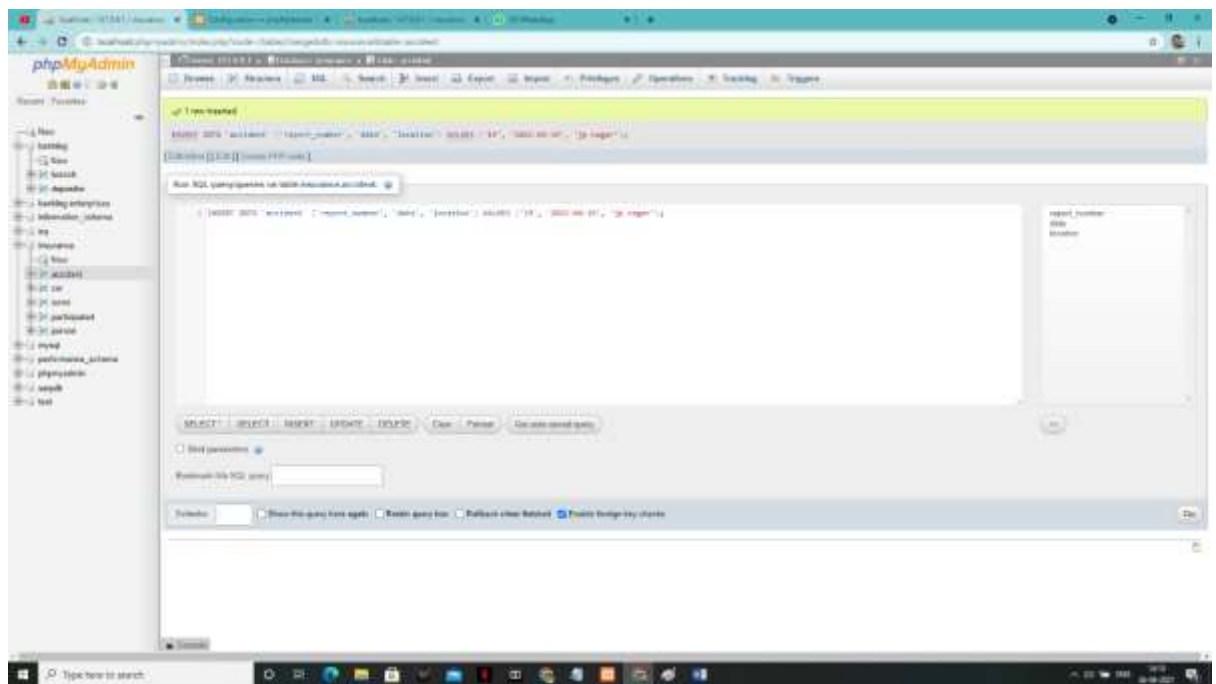


3) Demonstrate how you

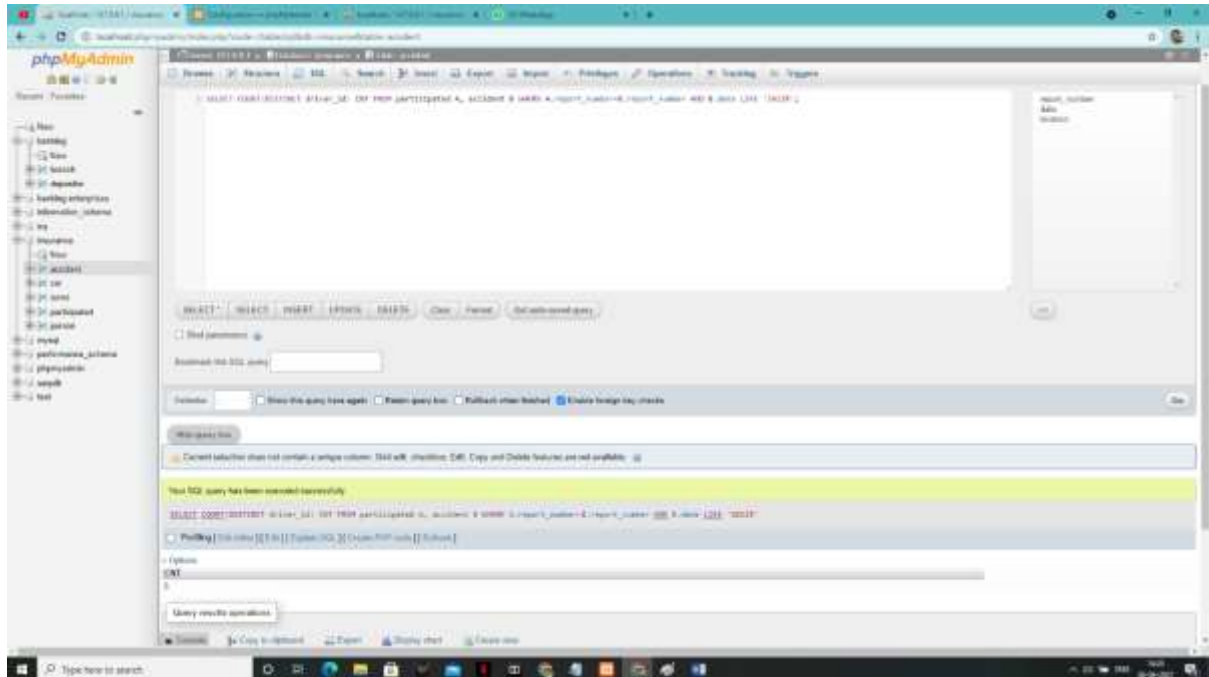
a) Update the damage amount for the car with a specific Regno in the accident with report number 12 to 25000.



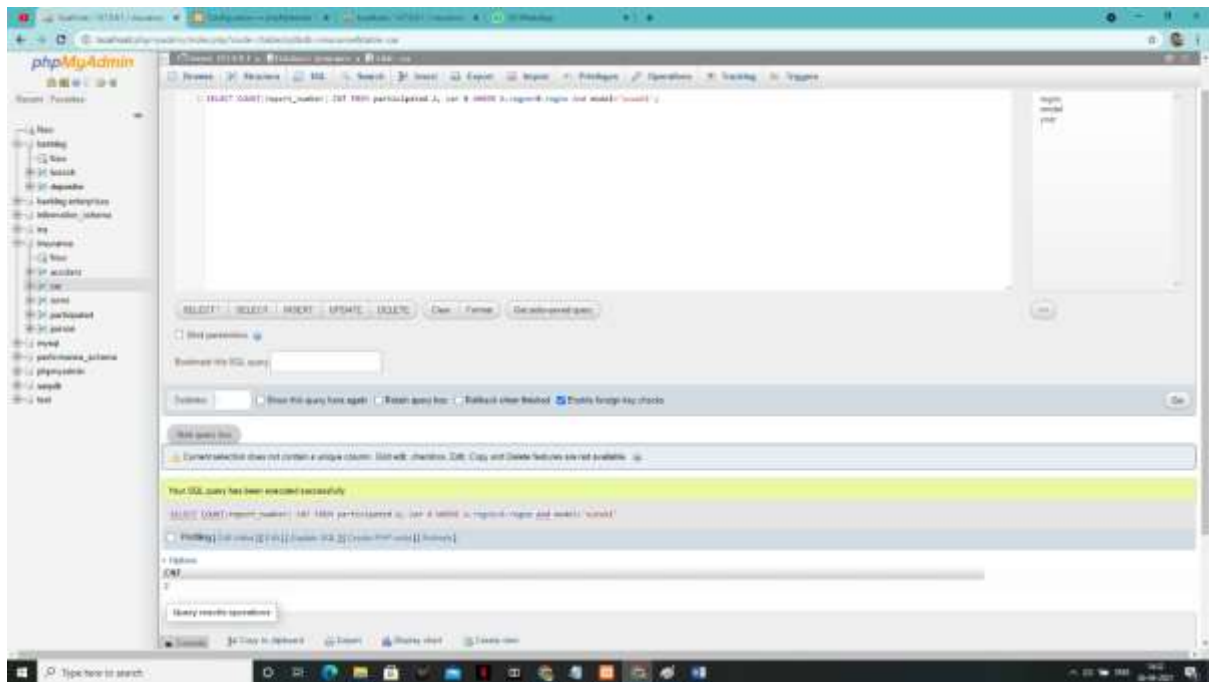
b) Add a new accident to the database.



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)

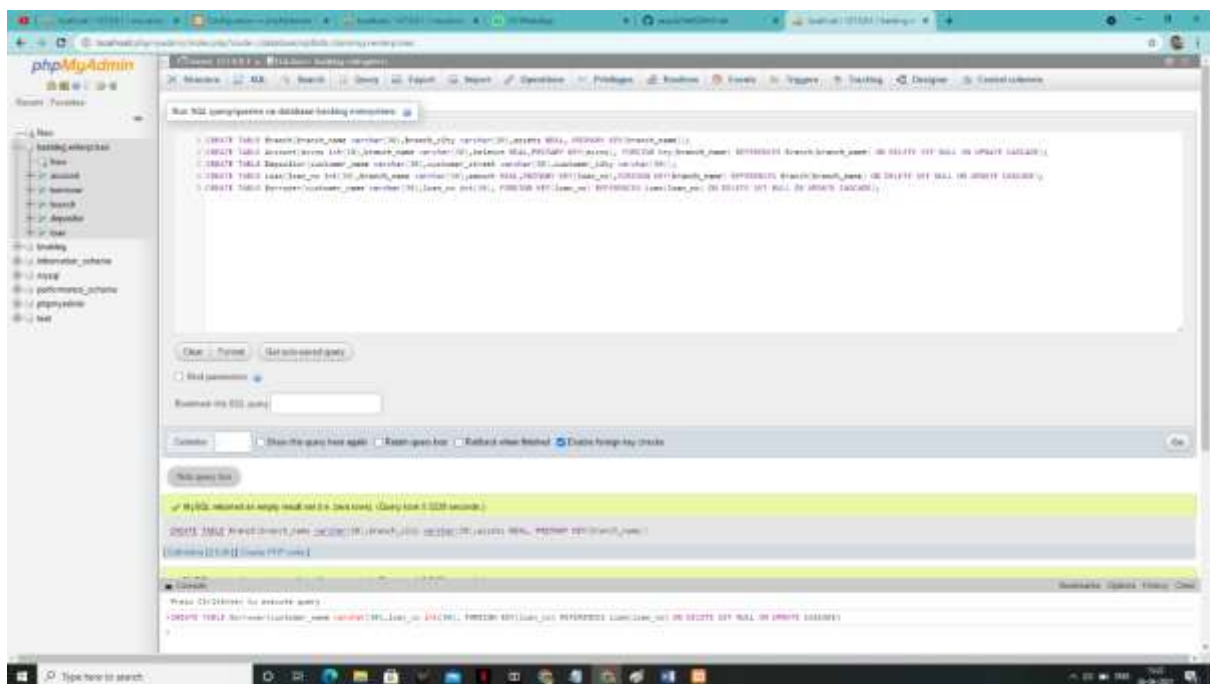
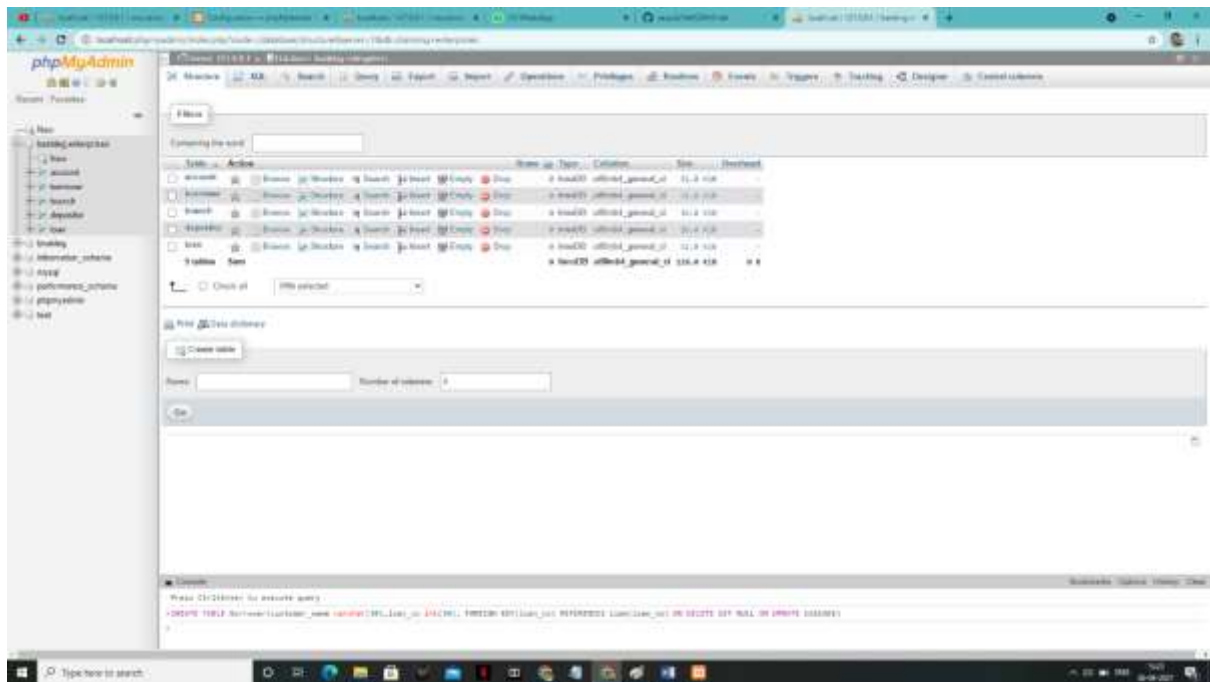
ACCOUNTS (accno: int, branch-name: String, balance: real)

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

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

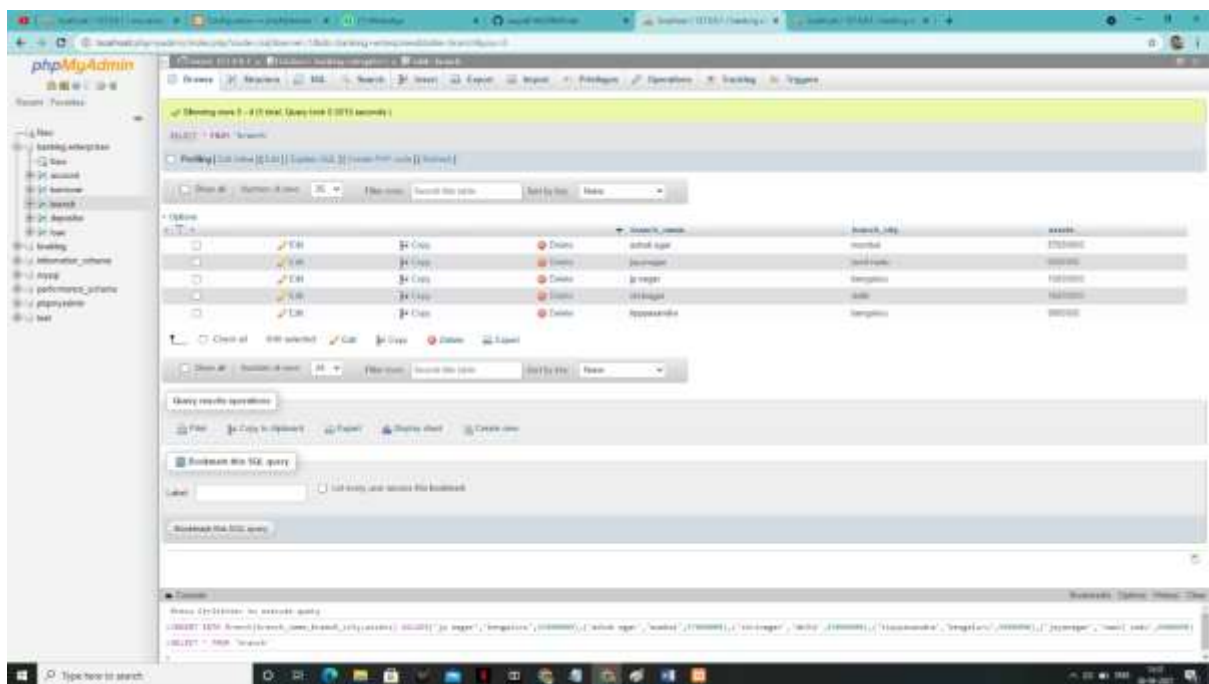
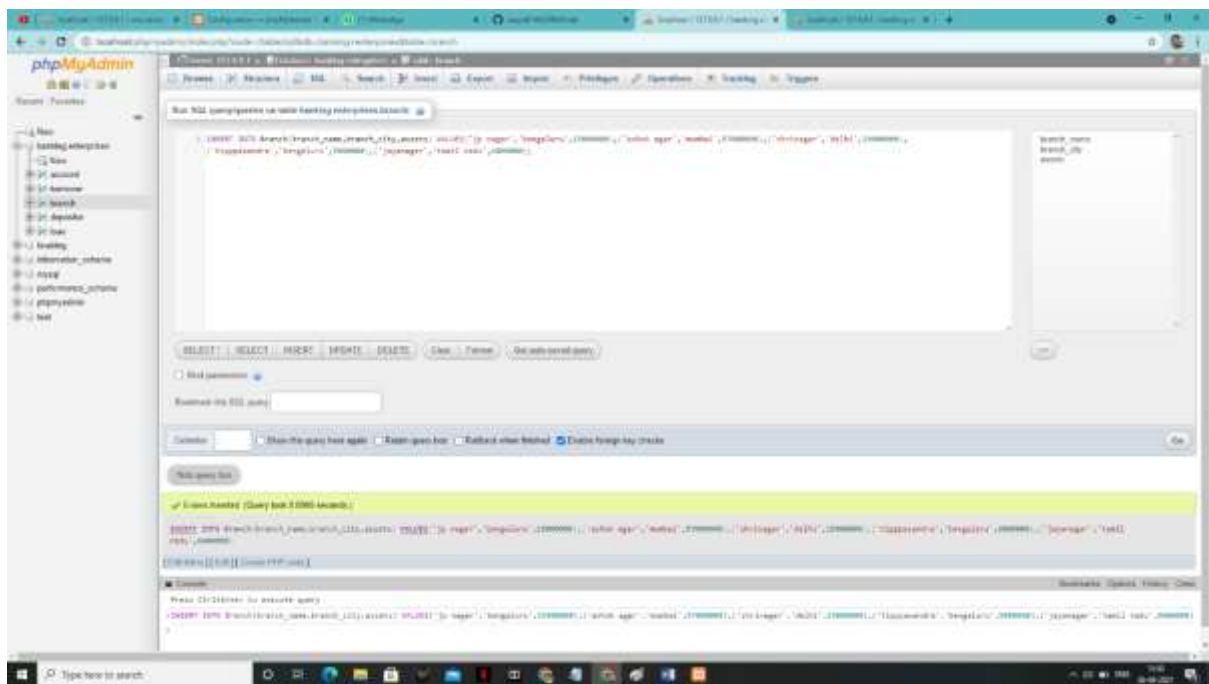
BORROWER (customer-name: String, loan-number: int)

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

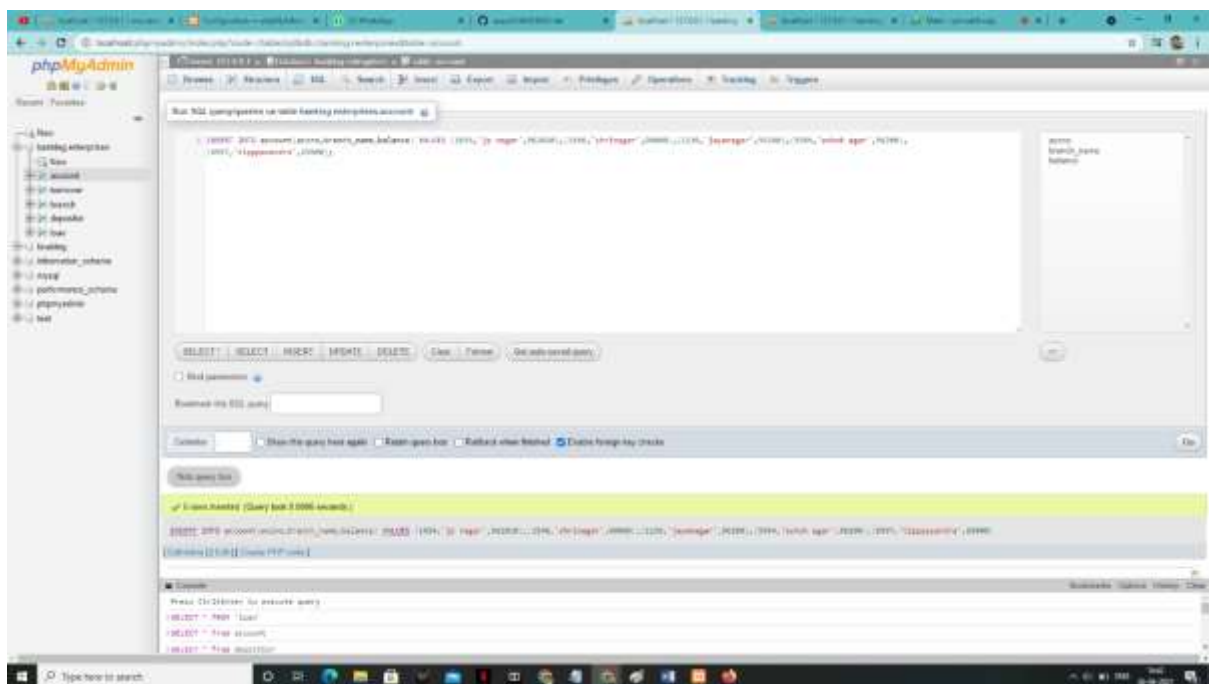
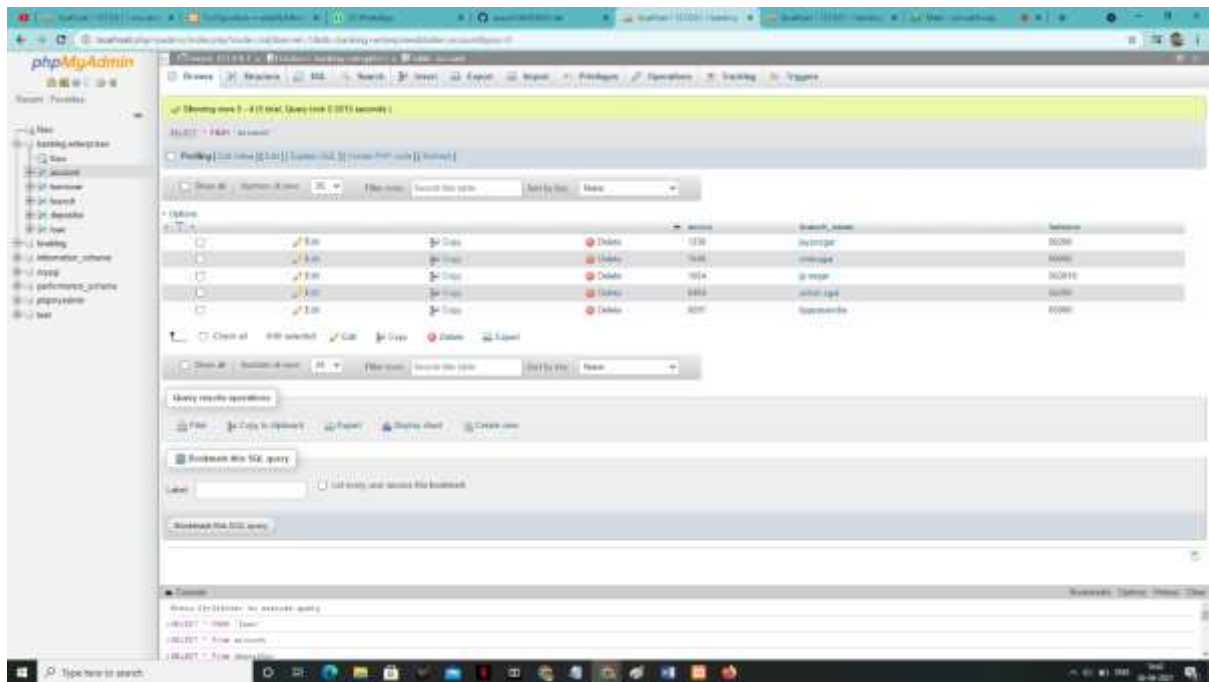


2) Enter at least 5 tuples for each relation.

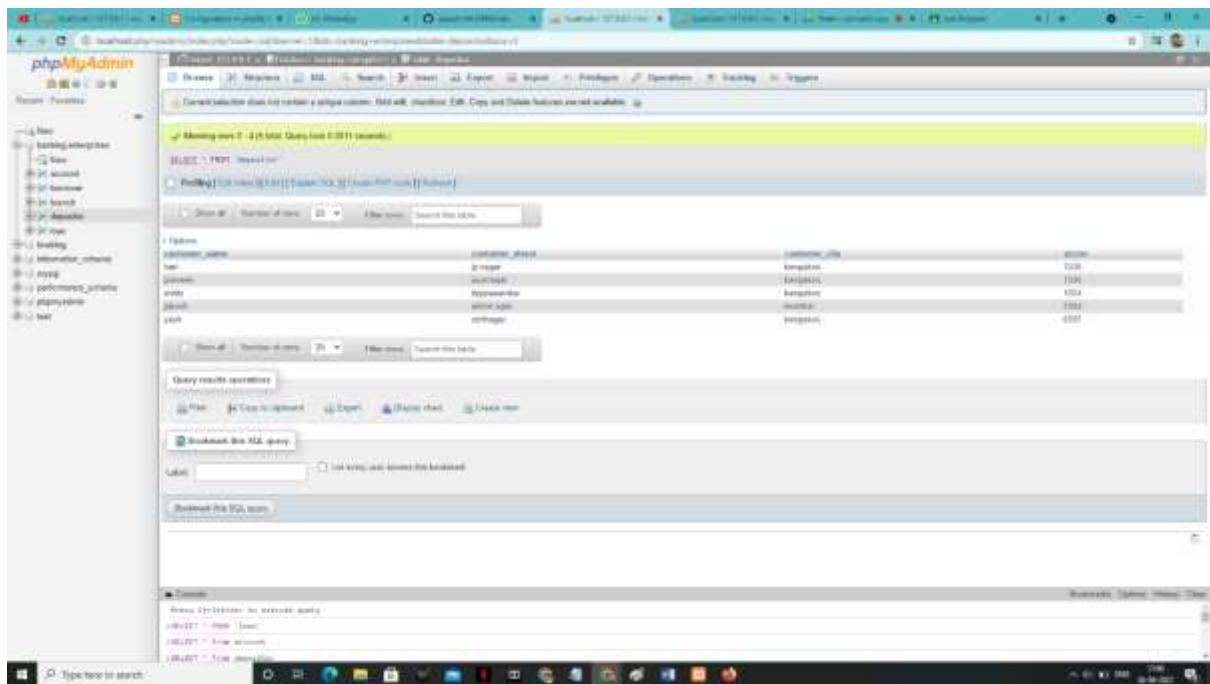
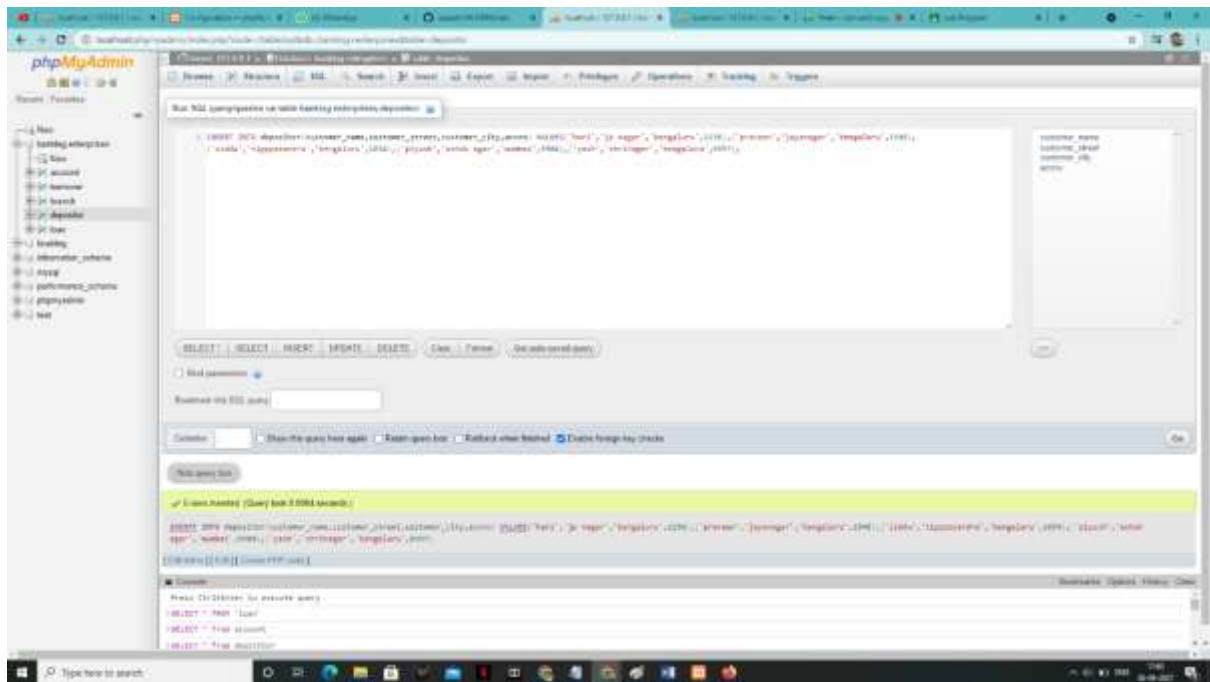
'BRANCH' table:



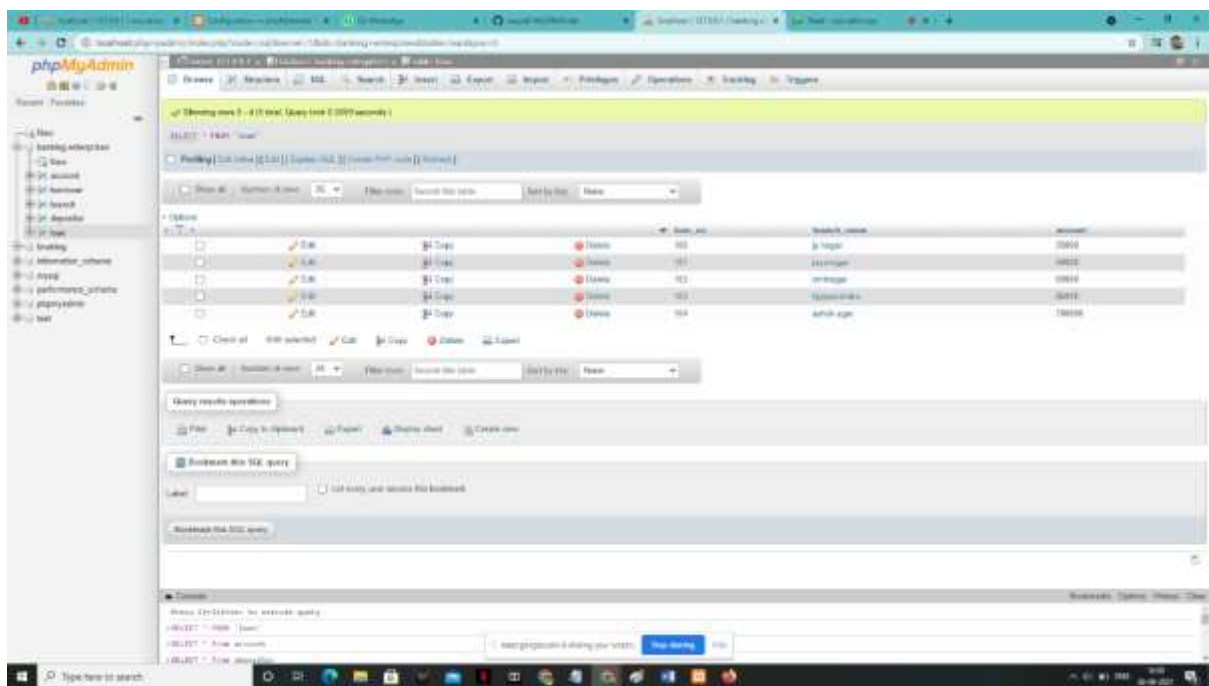
'ACCOUNTS' table:

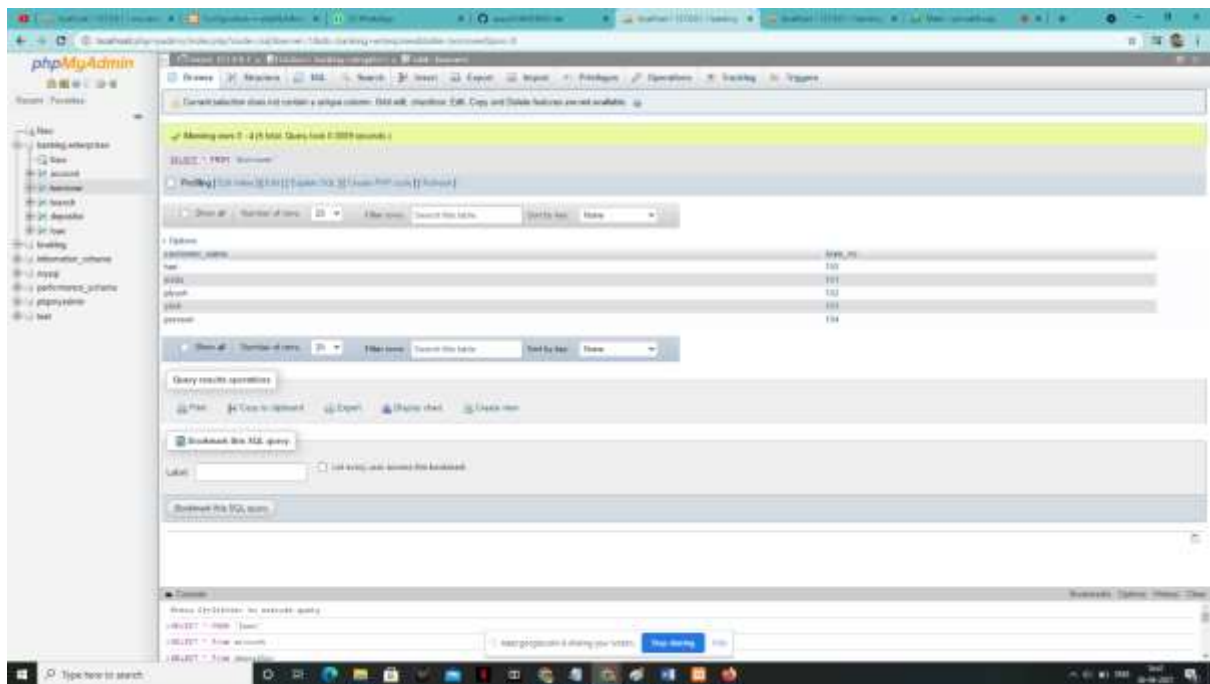


'DEPOSITOR' table:

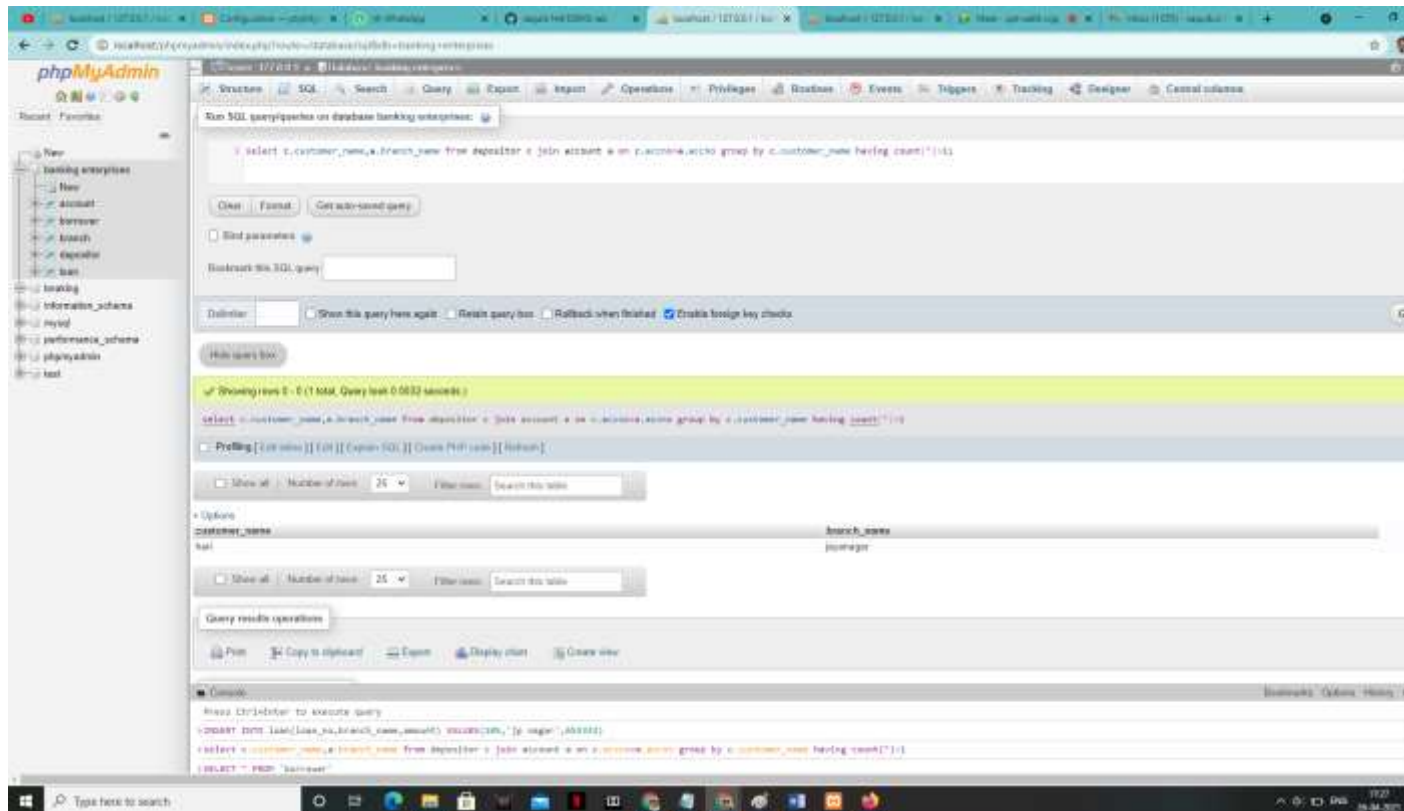


'LOAN' table:

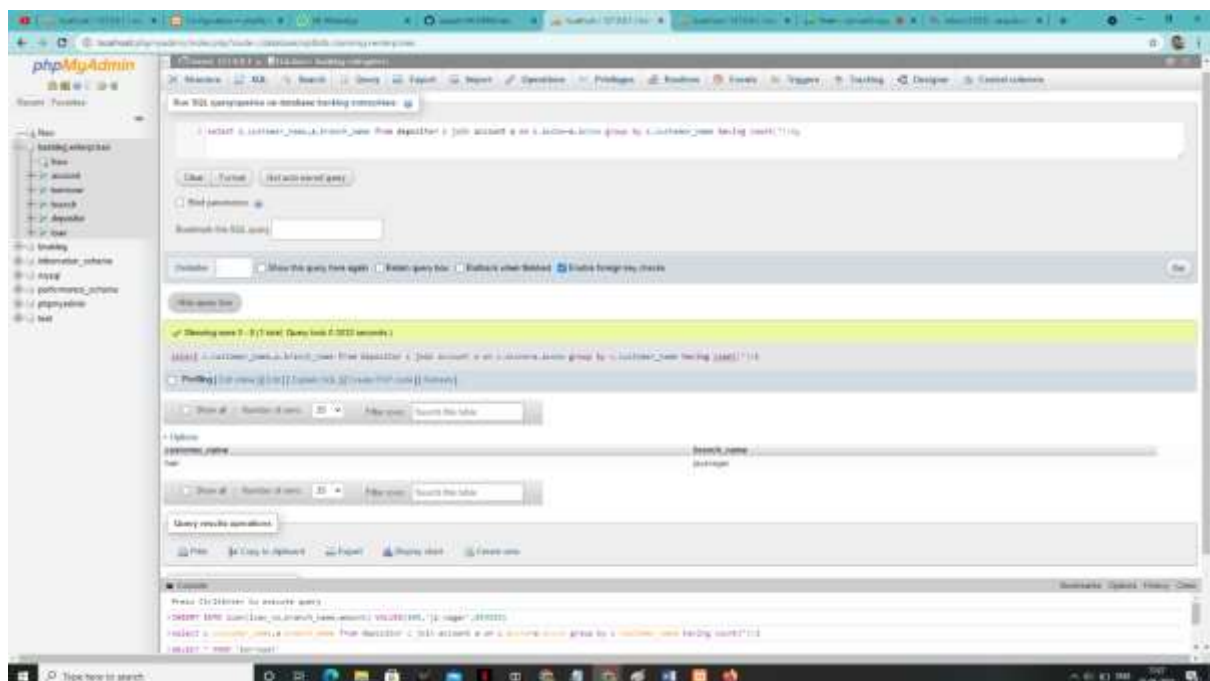




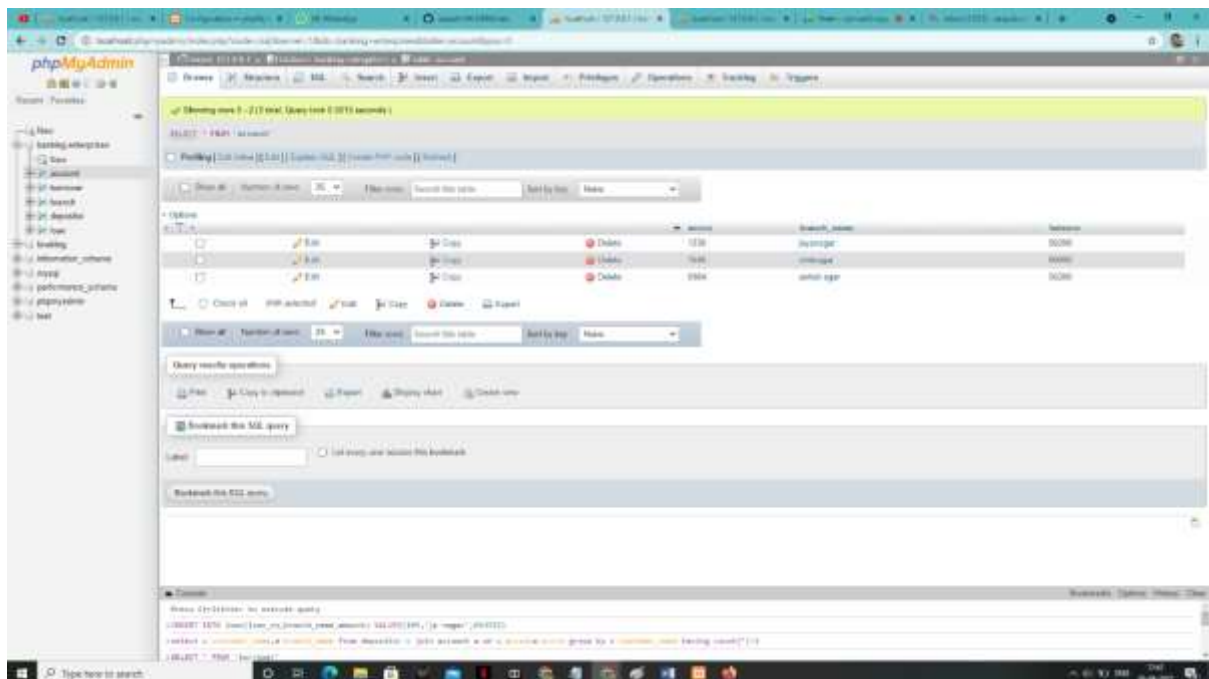
iii. Find all the customers who have at least two accounts at the Main branch.



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:

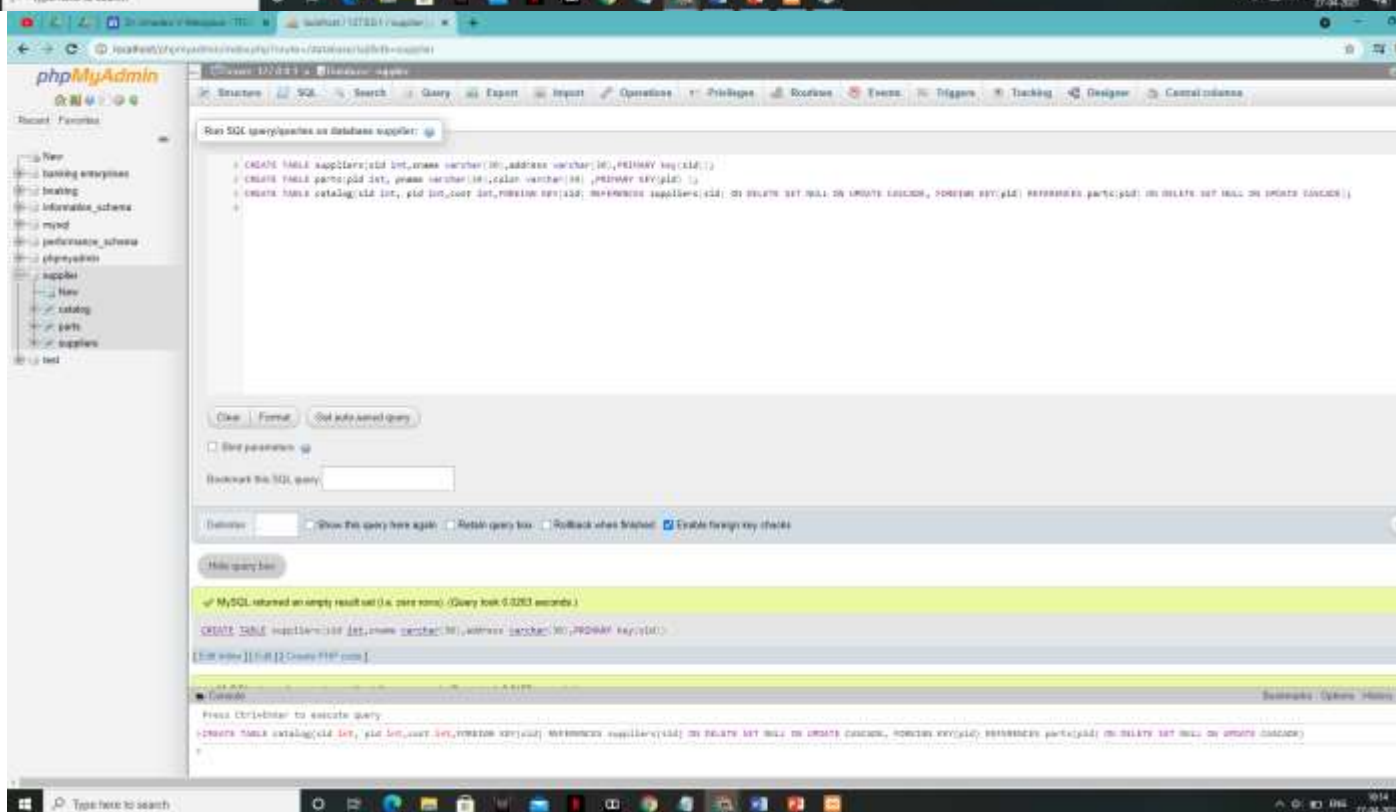
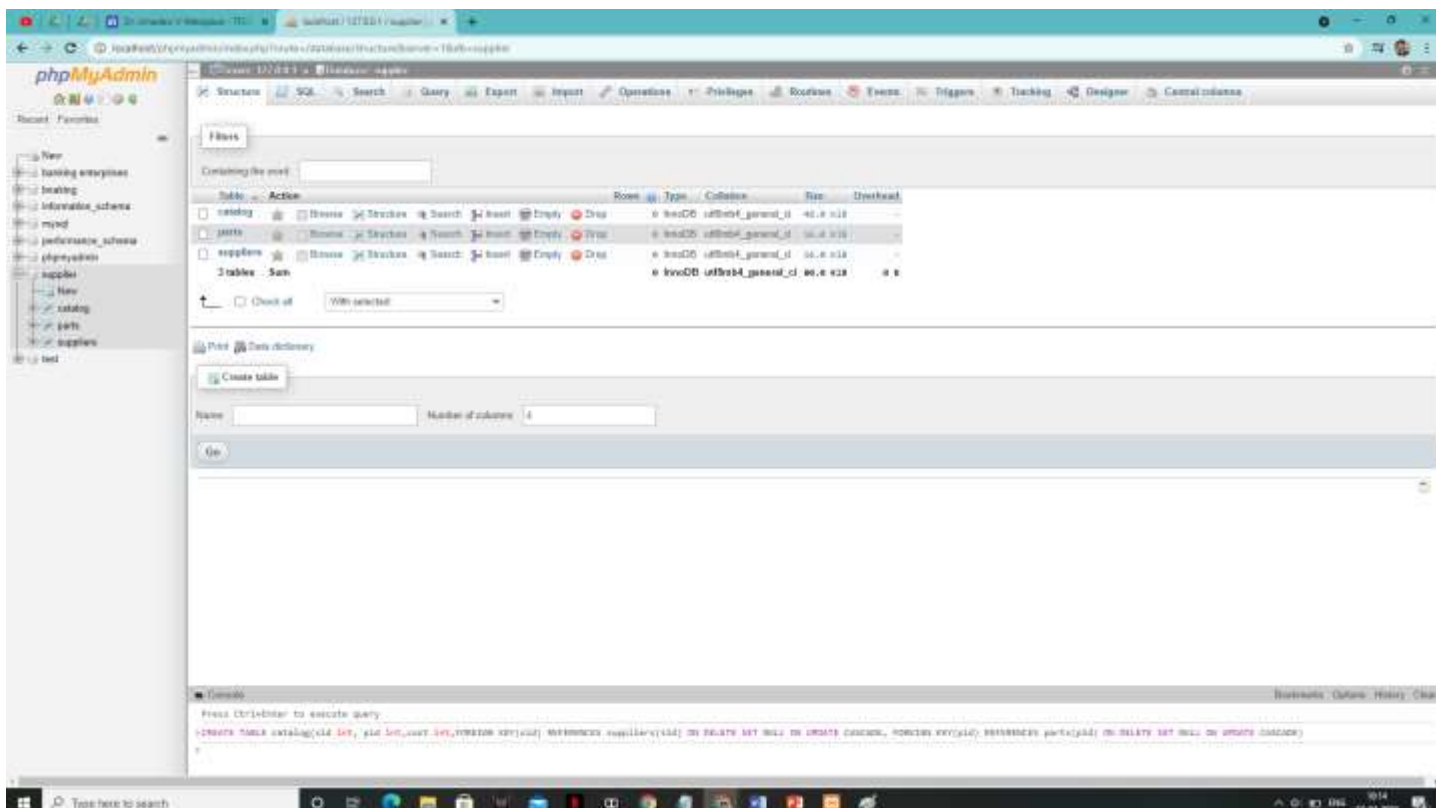
SUPPLIERS (sid: integer, sname: string, address: string)

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 displays the phpMyAdmin interface for a MySQL database. The left sidebar shows the database structure, with 'suppliers' selected under the 'catalog' database. The main panel shows the 'suppliers' table structure and data.

Table Structure:

Field	Type	Length	Charset	Collate	Index
id	INT	11	utf8	utf8_general_ci	PRIMARY
name	VARCHAR	45	utf8	utf8_general_ci	
address	VARCHAR	128	utf8	utf8_general_ci	

Table Data:

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

Query Log:

```
CREATE TABLE catalog.suppliers (id INT(11) UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY, name VARCHAR(45) CHARACTER SET utf8 COLLATE utf8_general_ci, address VARCHAR(128) CHARACTER SET utf8 COLLATE utf8_general_ci);
```

Console:

```
SELECT * FROM 'suppliers';
```

The screenshot shows the phpMyAdmin interface with the 'suppliers' table selected. The SQL query entered is:

```
INSERT INTO suppliers(sid,sname,address) VALUES (1001,'bark','barkaba'),(1002,'pawent','bangalore'),(1003,'pipark','mumbai'),(1004,'stake','delhi');
```

The query was executed successfully, as indicated by the green message: "Query took 0.0289 seconds". The console output shows the command and the resulting table structure:

```
Press Ctrl+Enter to execute query
CREATE TABLE catalog(sid INT, sname VARCHAR(255), address VARCHAR(255)) REFERENCES suppliers(sid) ON DELETE SET NULL ON UPDATE CASCADE, FOREIGN KEY(sname) REFERENCES parts(sname) ON DELETE SET NULL ON UPDATE CASCADE)
SELECT * FROM 'catalog'
SELECT * FROM 'suppliers'
```

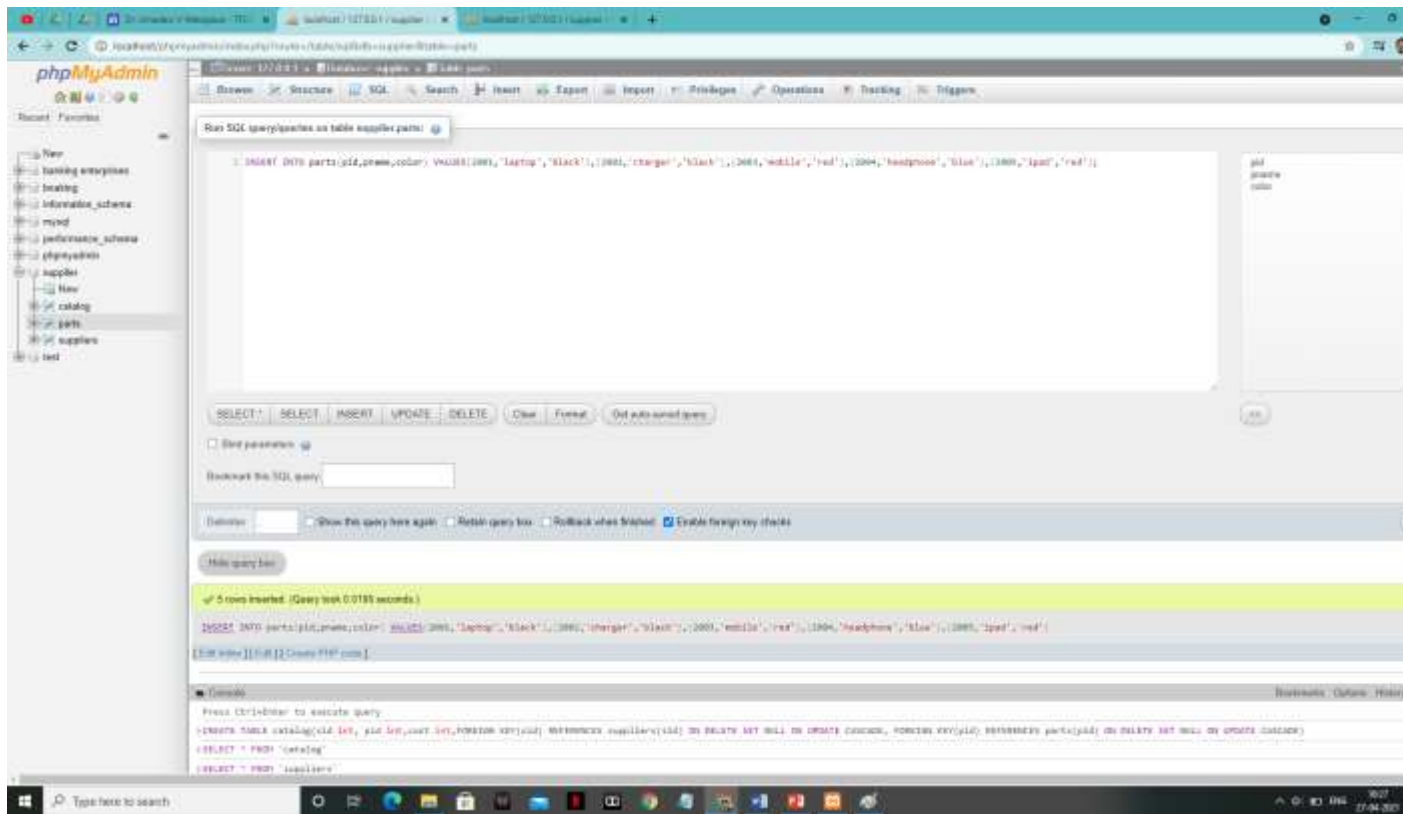
'PARTS' table:

The screenshot shows the phpMyAdmin interface with the 'PARTS' table selected. The table contains 4 rows of data:

sname	pid	quantity	price
bark	2001	1000	1000
pawent	2002	1000	1000
pipark	2003	1000	1000
stake	2004	1000	1000

The console output shows the command and the resulting table structure:

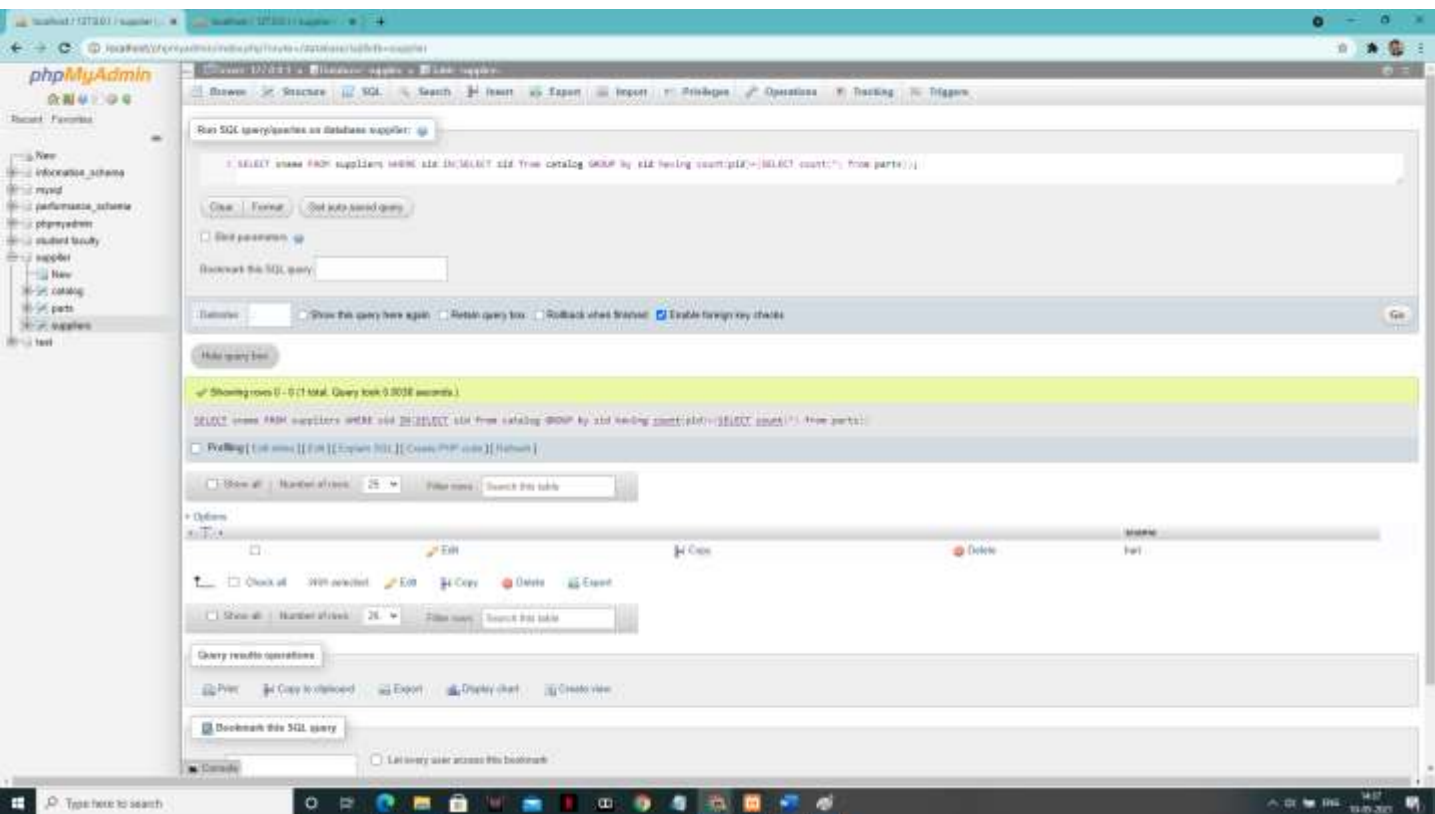
```
Press Ctrl+Enter to execute query
CREATE TABLE catalog(sid INT, pid INT, FOREIGN KEY(sid) REFERENCES suppliers(sid) ON DELETE SET NULL ON UPDATE CASCADE, FOREIGN KEY(pid) REFERENCES parts(pid) ON DELETE SET NULL ON UPDATE CASCADE)
SELECT * FROM 'catalog'
```



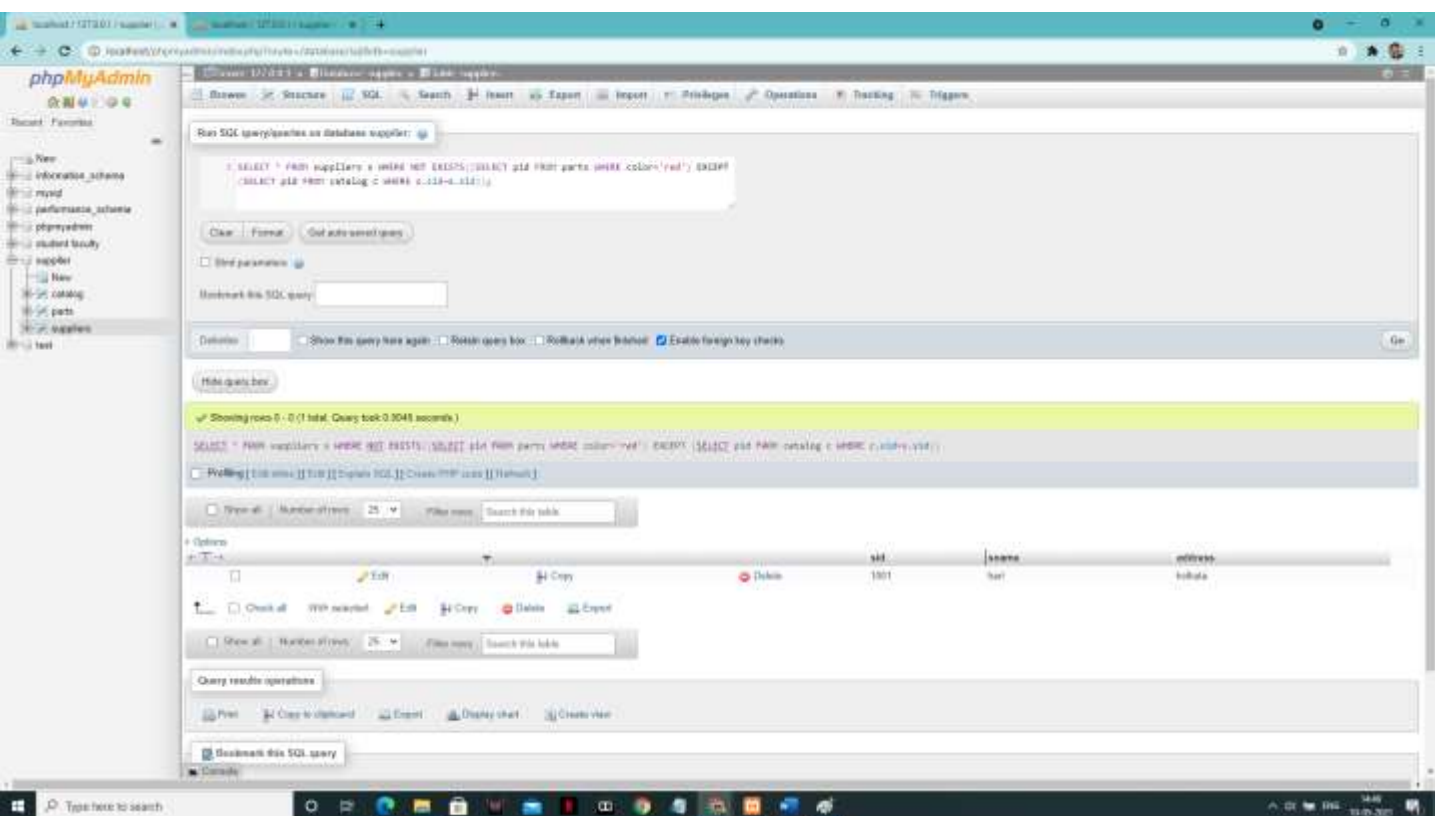
‘CATALOG’ table:



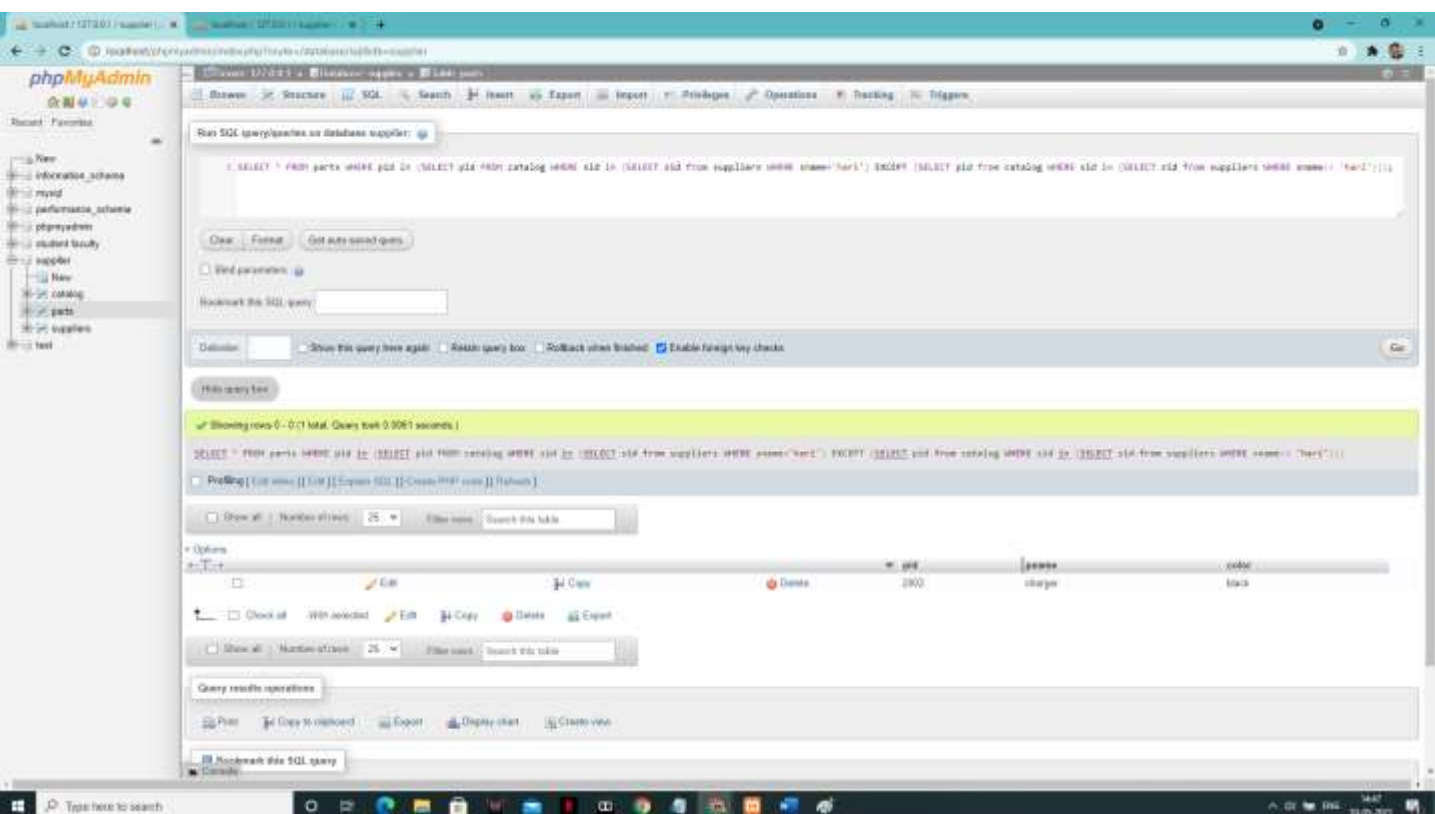
ii. Find the snames of suppliers who supply every part.



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.



- v. Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part)

Showing rows 0 - 7 (8 total. Query took 0.000 seconds)

SELECT parts, s1.sname FROM parts, catalog c1 WHERE parts.p10=c1.p10 AND parts--(SELECT s2.sname FROM catalog c2 WHERE c2.p10=c1.p10)

Options

Options	p10name	s1name	s2name
Options	1001	1001	1001
Options	1001	1001	1001
Options	1001	1001	1001
Options	1001	1001	1001
Options	1001	1001	1001
Options	1001	1001	1001
Options	1001	1001	1001
Options	1001	1001	1001
Options	1001	1001	1001
Options	1001	1001	1001

vi. For each part, find the sname of the supplier who charges the most for that part.

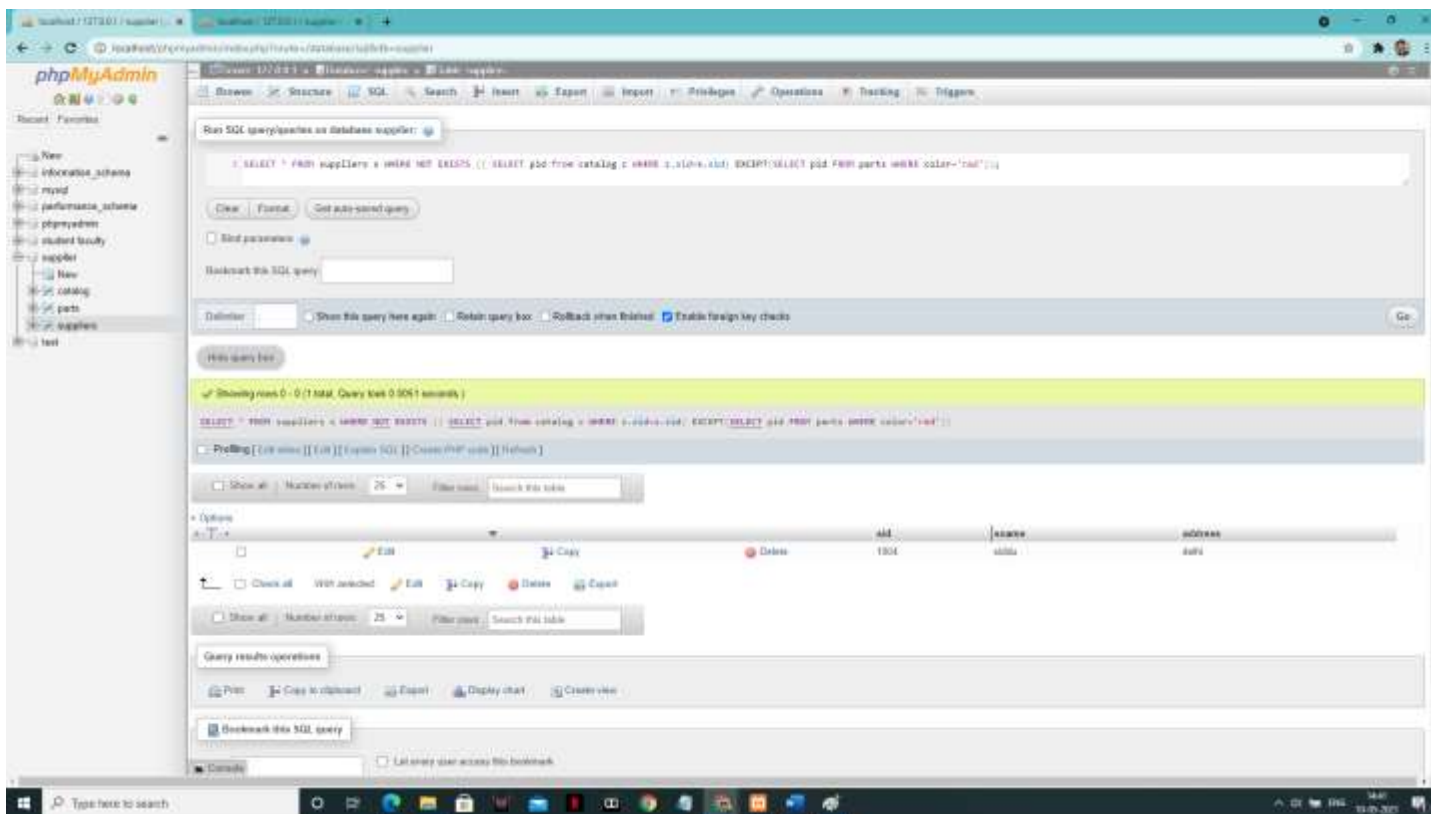
The screenshot shows the phpMyAdmin interface with a SQL query entered in the query box. The query is:

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

The query is executed, and the results are displayed in a table. The table has 9 columns: s_id, s_name, p_name, p_id, p_name, s_name, s_id, p_id, and cost. The results show 10 rows of data.

s_id	s_name	p_name	p_id	p_name	s_name	s_id	p_id	cost
1001	test	test	2001	test	test	1001	2001	10000
1001	test	test	2002	test	test	1001	2002	1000
1001	test	test	2003	test	test	1001	2003	10000
1001	test	test	2004	test	test	1001	2004	2000
1001	test	test	2005	test	test	1001	2005	10000
1002	test	test	2006	test	test	1002	2006	10000
1002	test	test	2007	test	test	1002	2007	10000
1002	test	test	2008	test	test	1002	2008	10000
1002	test	test	2009	test	test	1002	2009	10000
1002	test	test	2010	test	test	1002	2010	10000

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.

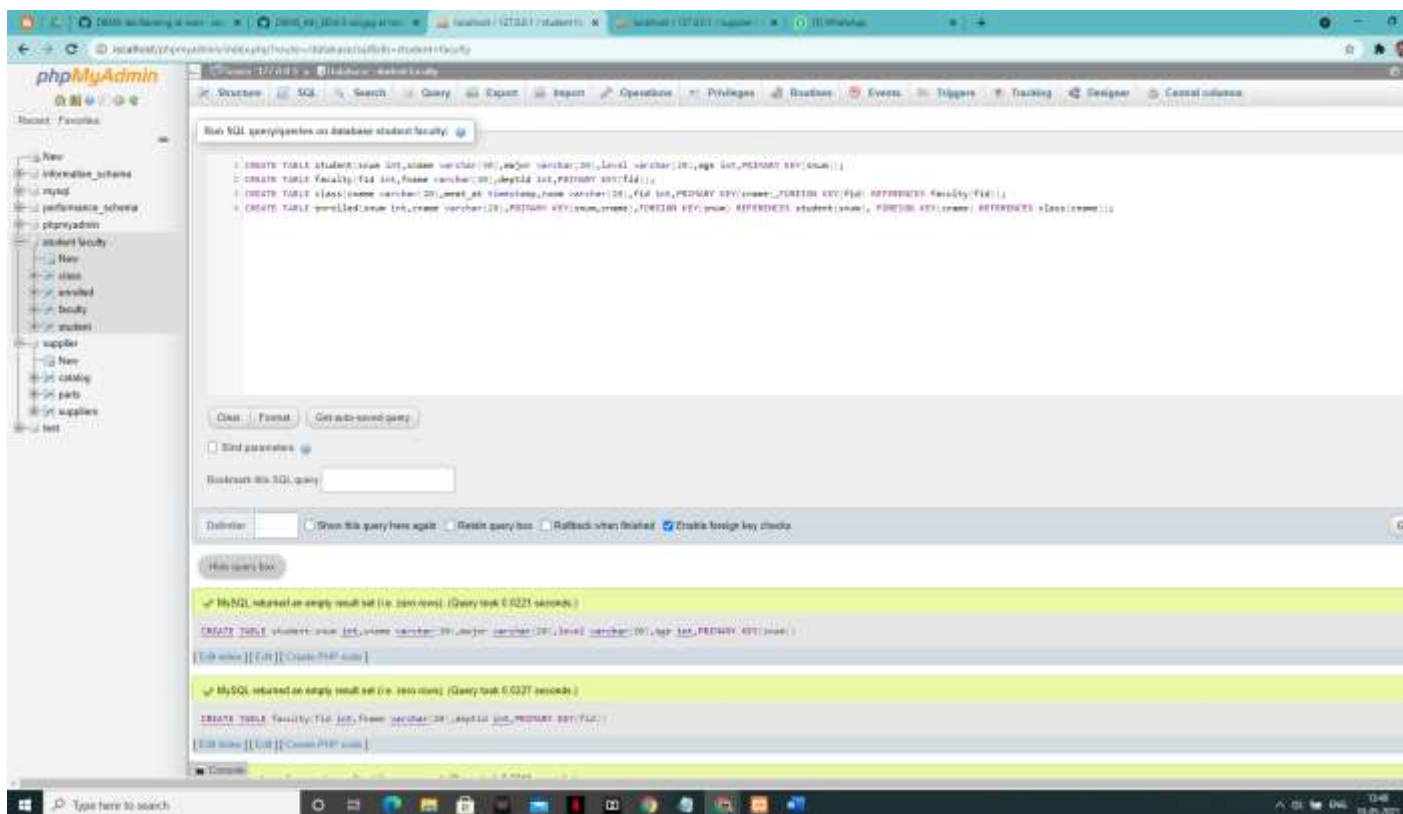
i. Create above mentioned tables

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).

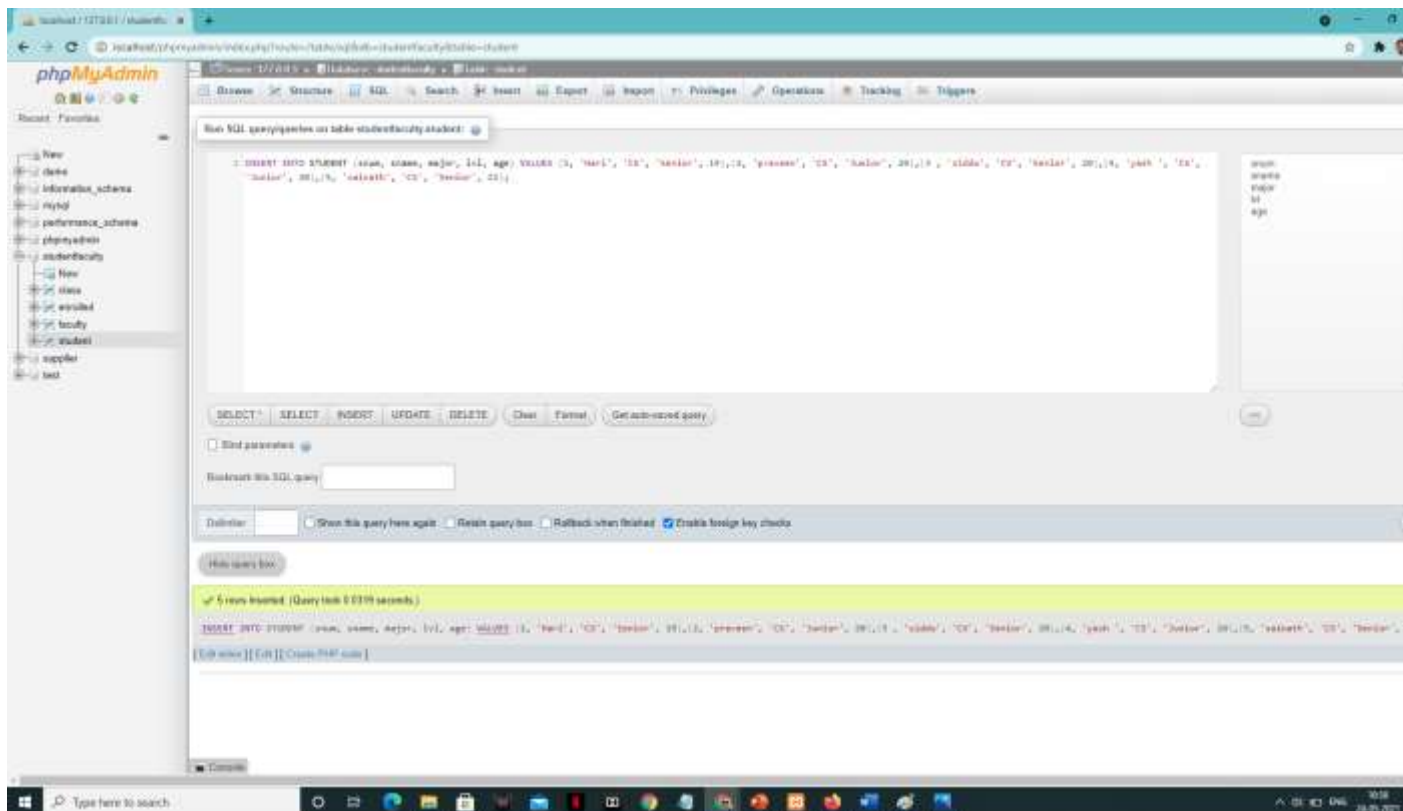
10) Create the above tables by properly specifying the primary keys and the foreign keys.

Create table:-

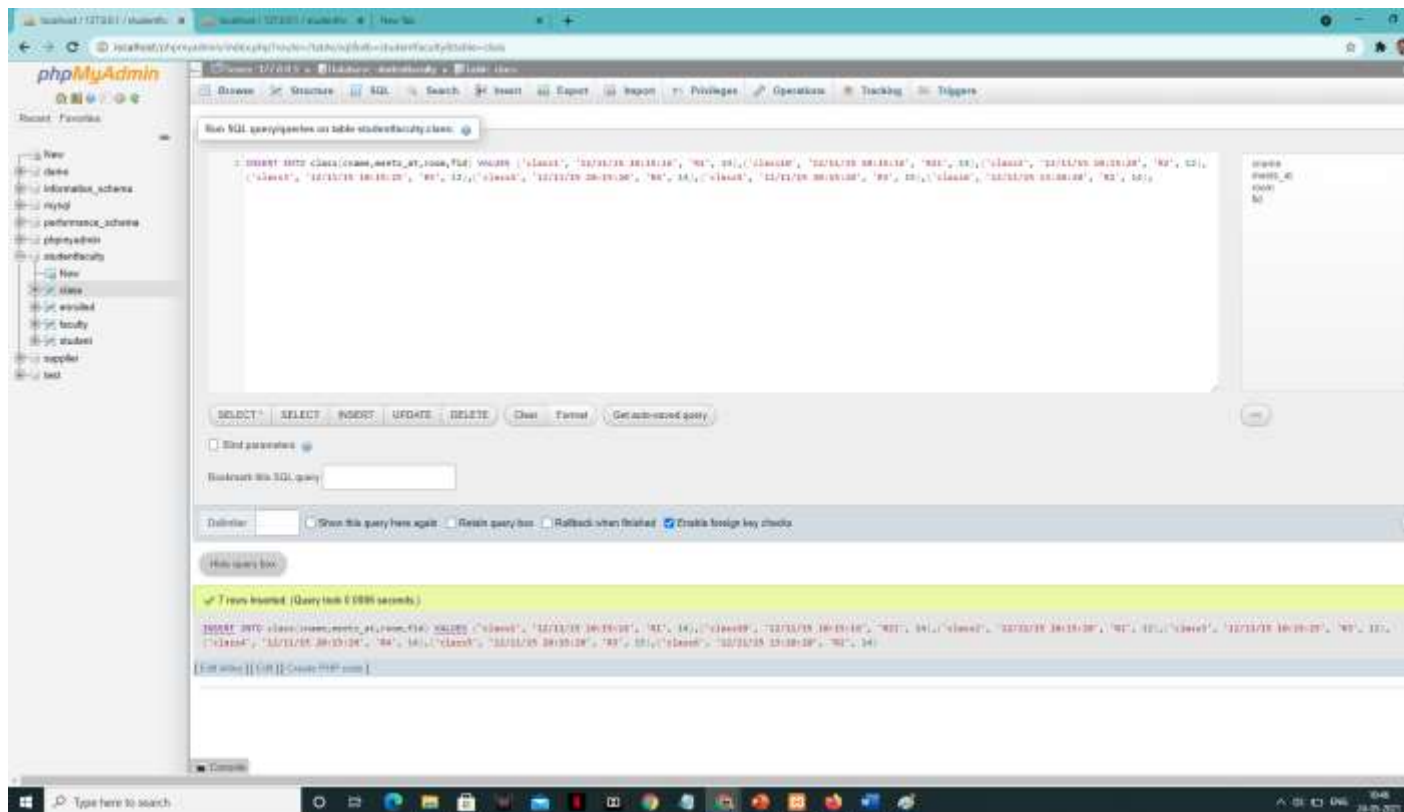


4) Enter tuples for each relation.

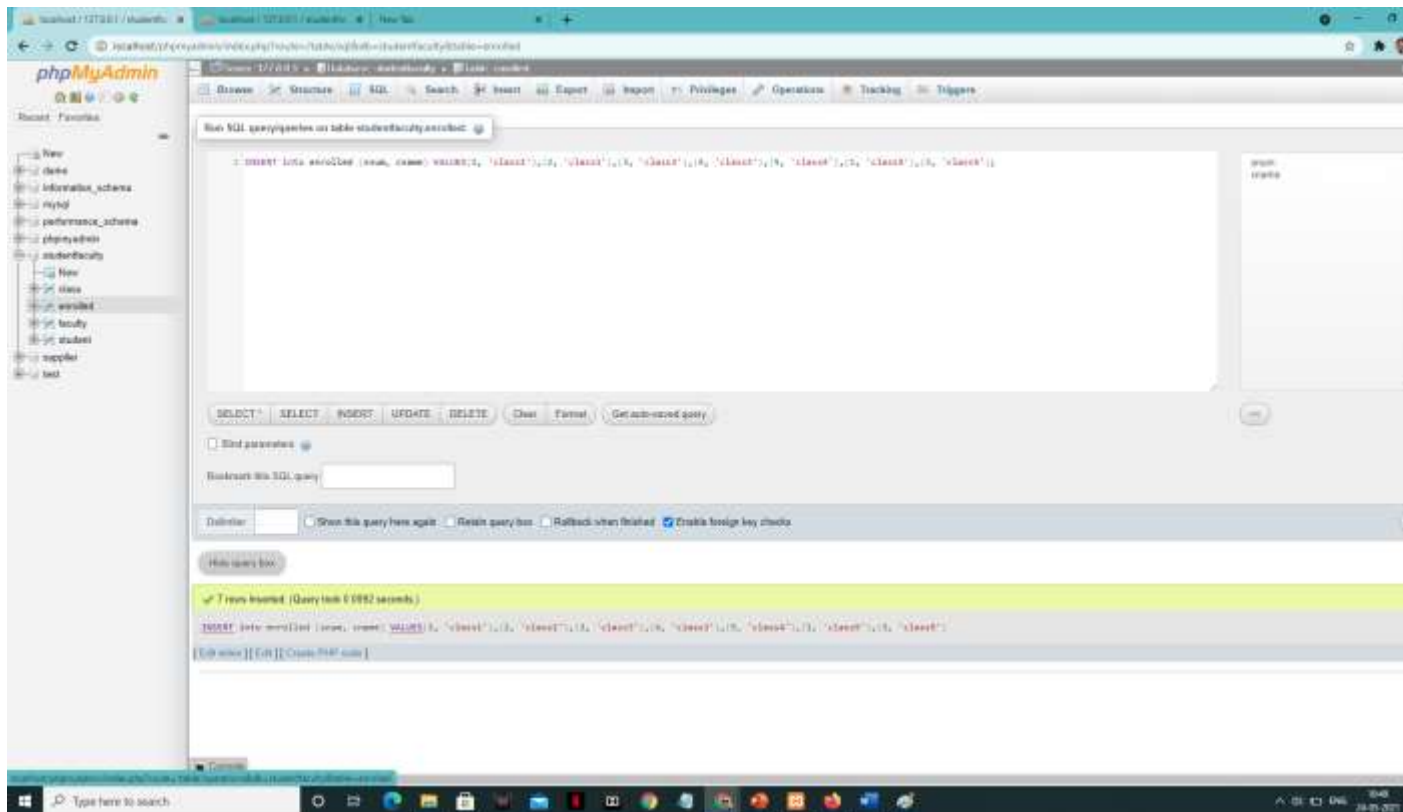
‘student’ table:



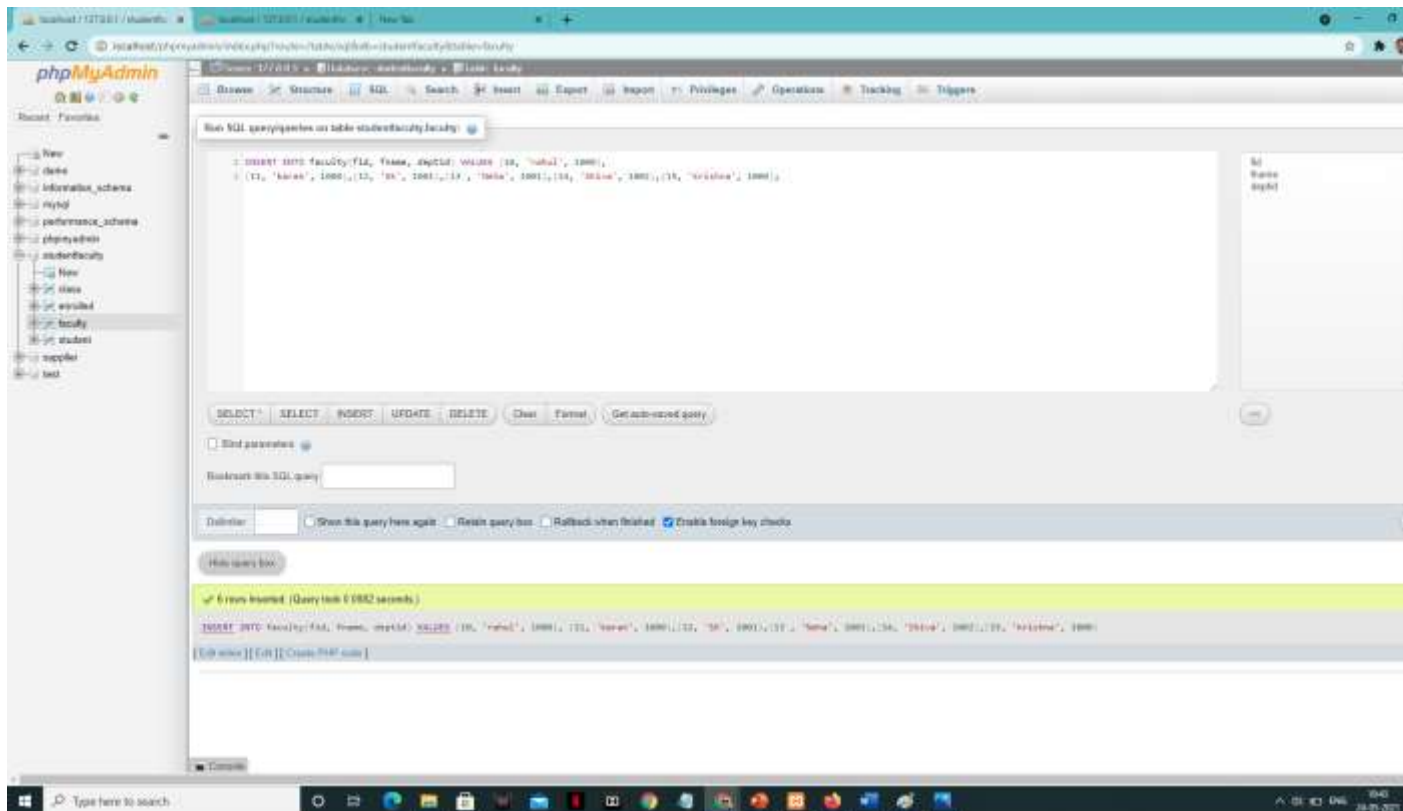
‘class’ table:



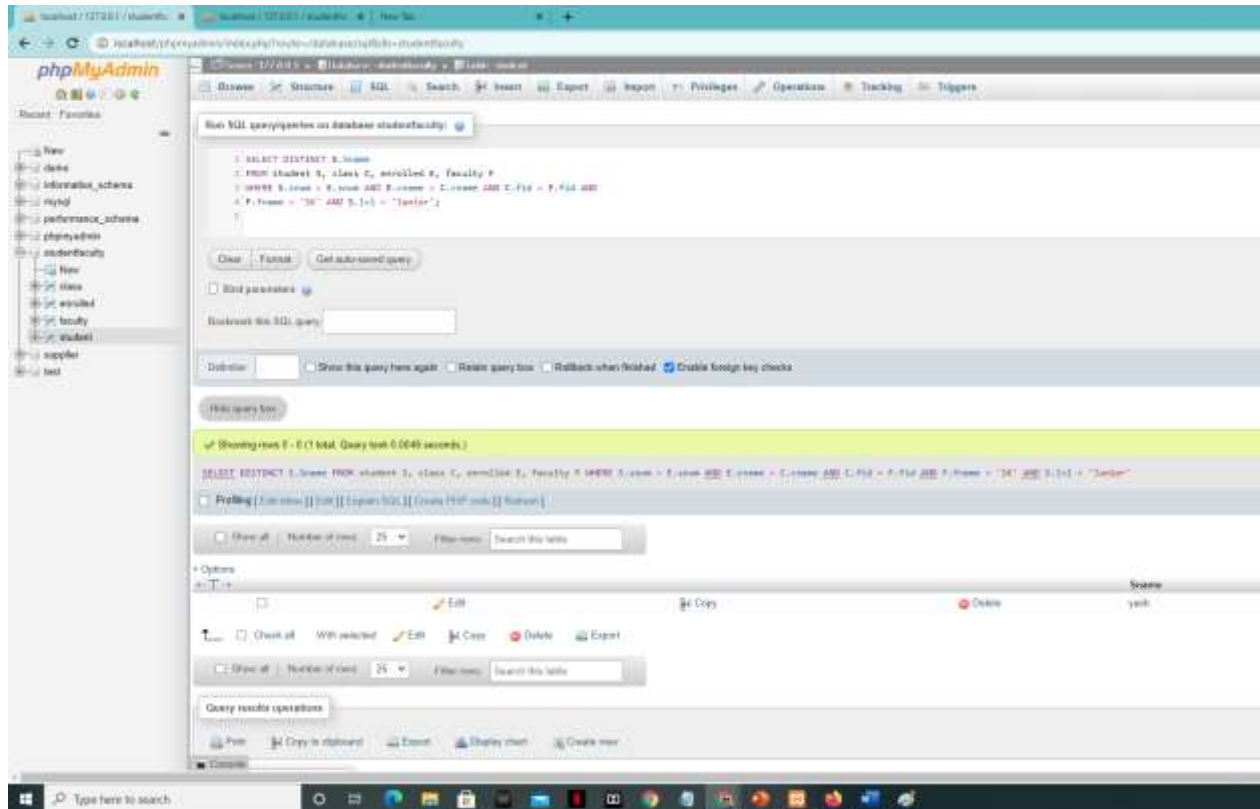
‘enrolled’ table:



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=2 (SELECT * FROM enrolled GROUP BY class HAVING COUNT(class)=6)
```

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

Query result:

class	enrolled_at	room	RM
class16	2012-11-16 10:10:16	R21	14

Query result operations: Print, Copy to clipboard, Export, Display chart, Create view

Bookmark this SQL query

3:-

4:-

5:-

phpMyAdmin interface showing a SQL query execution result for the 'student' database.

Run SQL query/queries on database student:

```
SELECT DISTINCT s.sname FROM PROFI Faculty s WHERE s.sname IN (SELECT COUNT(s.sname) FROM class c, enrolled e WHERE c.sname = s.sname AND c.FID = e.FID);
```

Showing rows 0 - 5 (5 total, Query took 0.0071 seconds):

sname
adithi
adithi
adithi
adithi
adithi

The screenshot shows the phpMyAdmin interface with a SQL query executed. The query is: `SELECT DISTINCT s.sname FROM PROFI Faculty s WHERE s.sname IN (SELECT COUNT(s.sname) FROM class c, enrolled e WHERE c.sname = s.sname AND c.FID = e.FID);`. The result shows 5 rows, all with the value 'adithi' in the 'sname' column. The interface includes a sidebar with database structure, a top navigation bar, and a bottom status bar.

6:-

phpMyAdmin interface showing a SQL query execution result for the 'student' database.

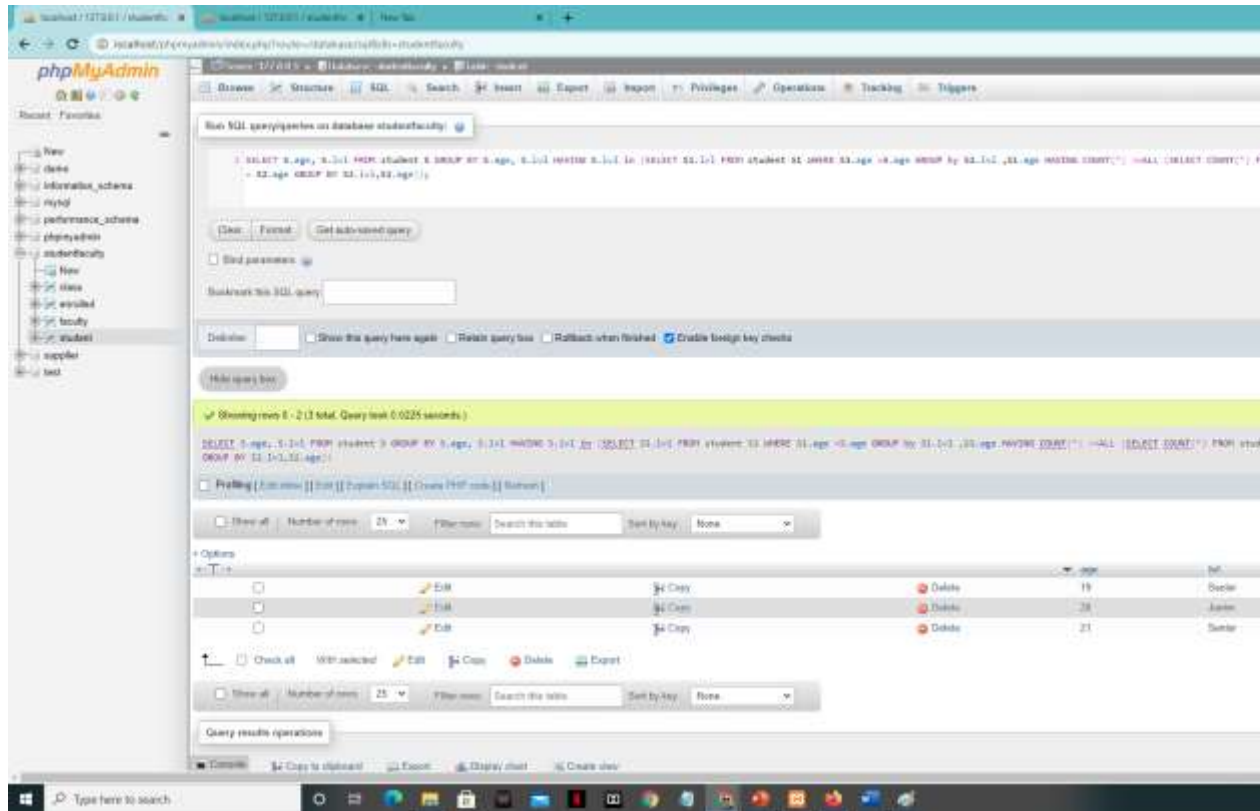
Run SQL query/queries on database student:

```
SELECT DISTINCT s.sname FROM PROFI student s WHERE s.sname NOT IN (SELECT s.sname FROM PROFI enrolled e);
```

Showing rows 0 - 0 (0 total, Query took 0.0034 seconds):

The screenshot shows the phpMyAdmin interface with a SQL query executed. The query is: `SELECT DISTINCT s.sname FROM PROFI student s WHERE s.sname NOT IN (SELECT s.sname FROM PROFI enrolled e);`. The result shows 0 rows. The interface includes a sidebar with database structure, a top navigation bar, and a bottom status bar.

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)

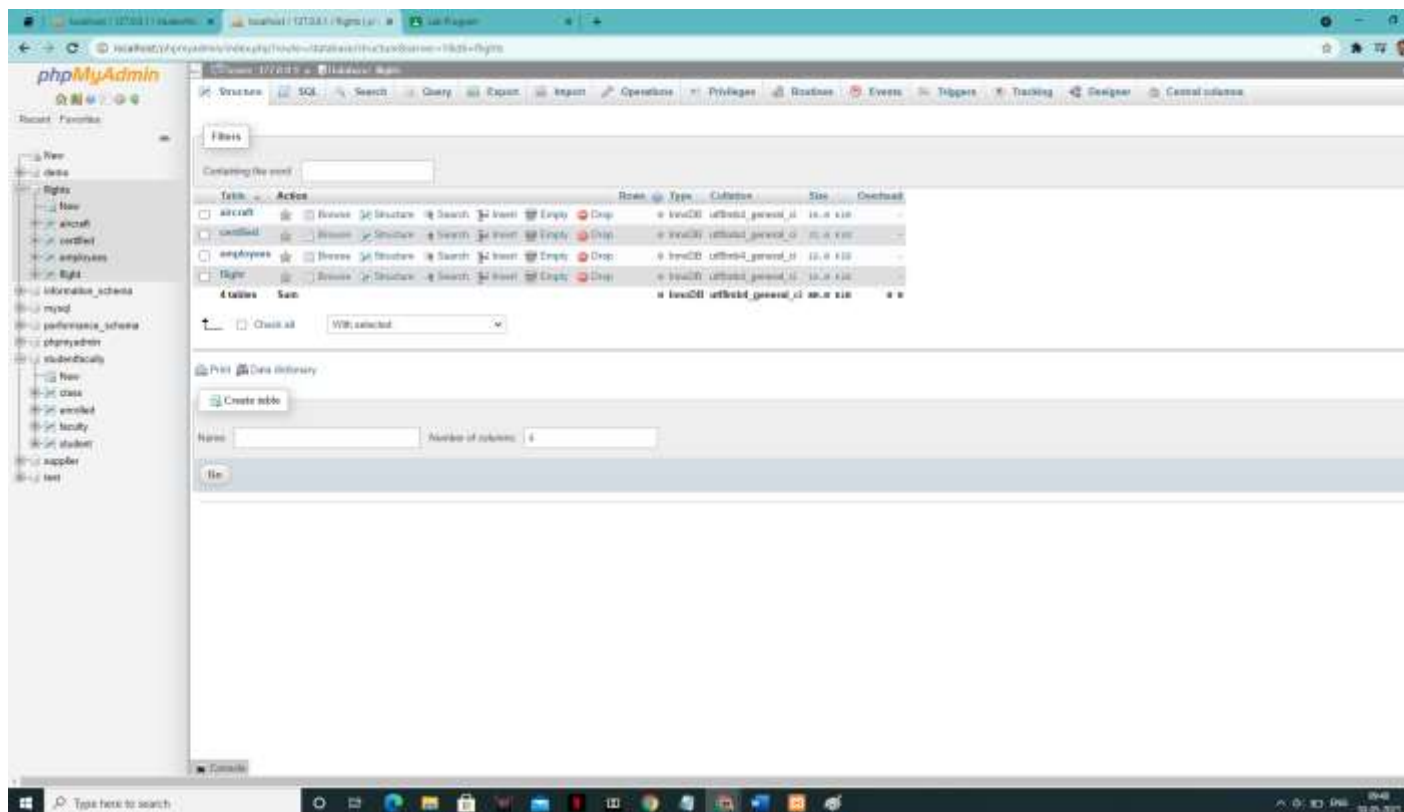
Note that the Employees relation describes pilots and other kinds of employees as well; Every pilot is certified

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

Write each of the following queries in SQL.

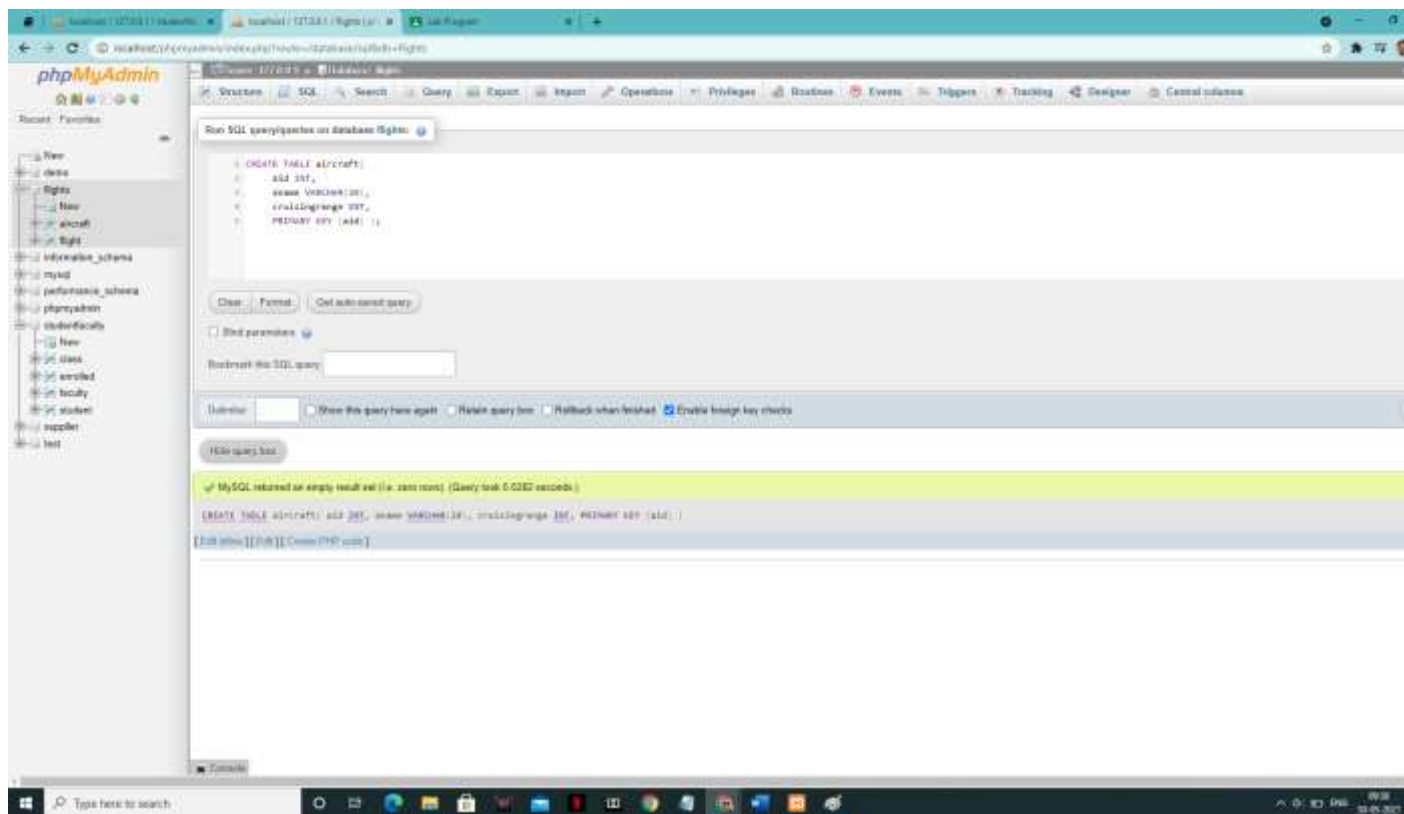
- i. Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80,000.**
- ii. For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruising range of the aircraft for which she or he is certified.**
- iii. Find the names 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:-

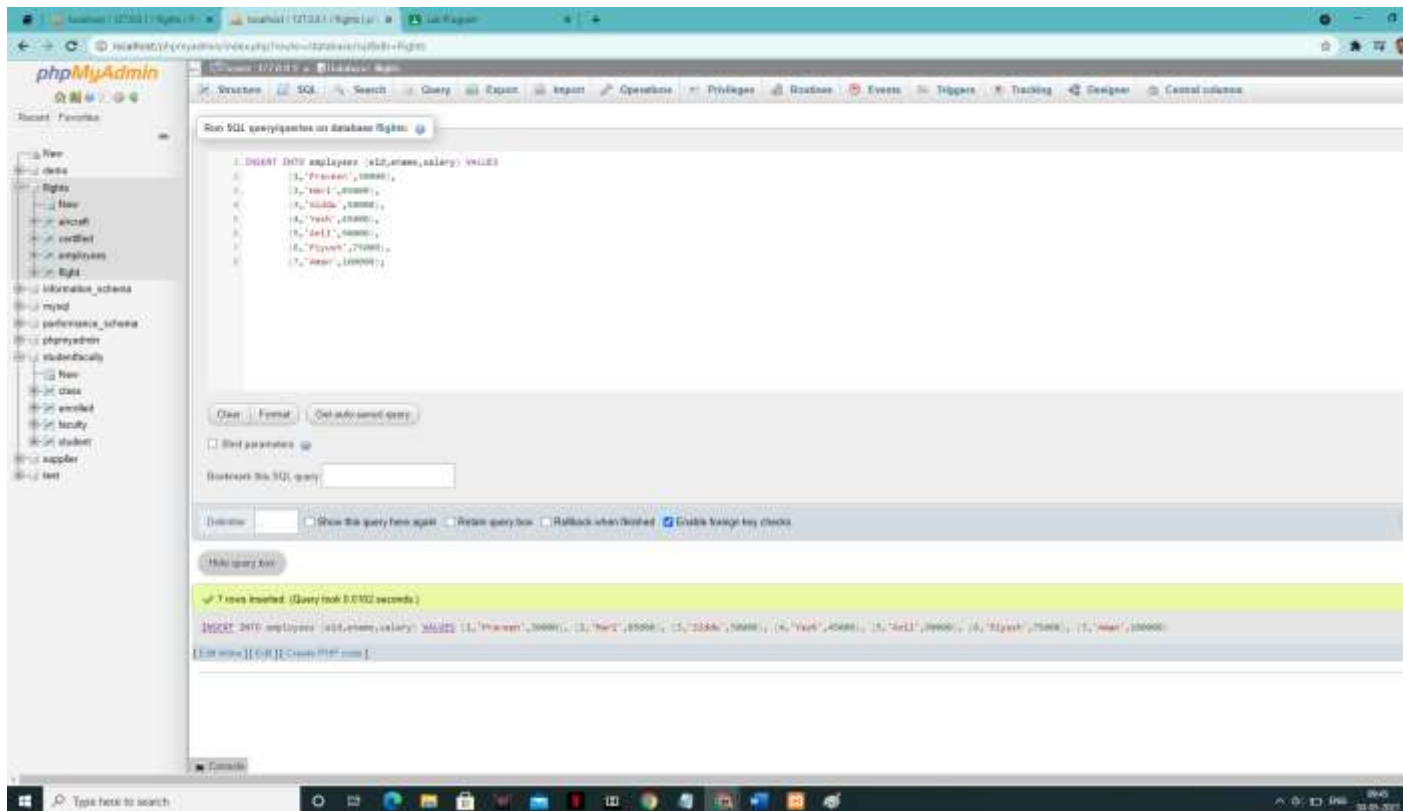
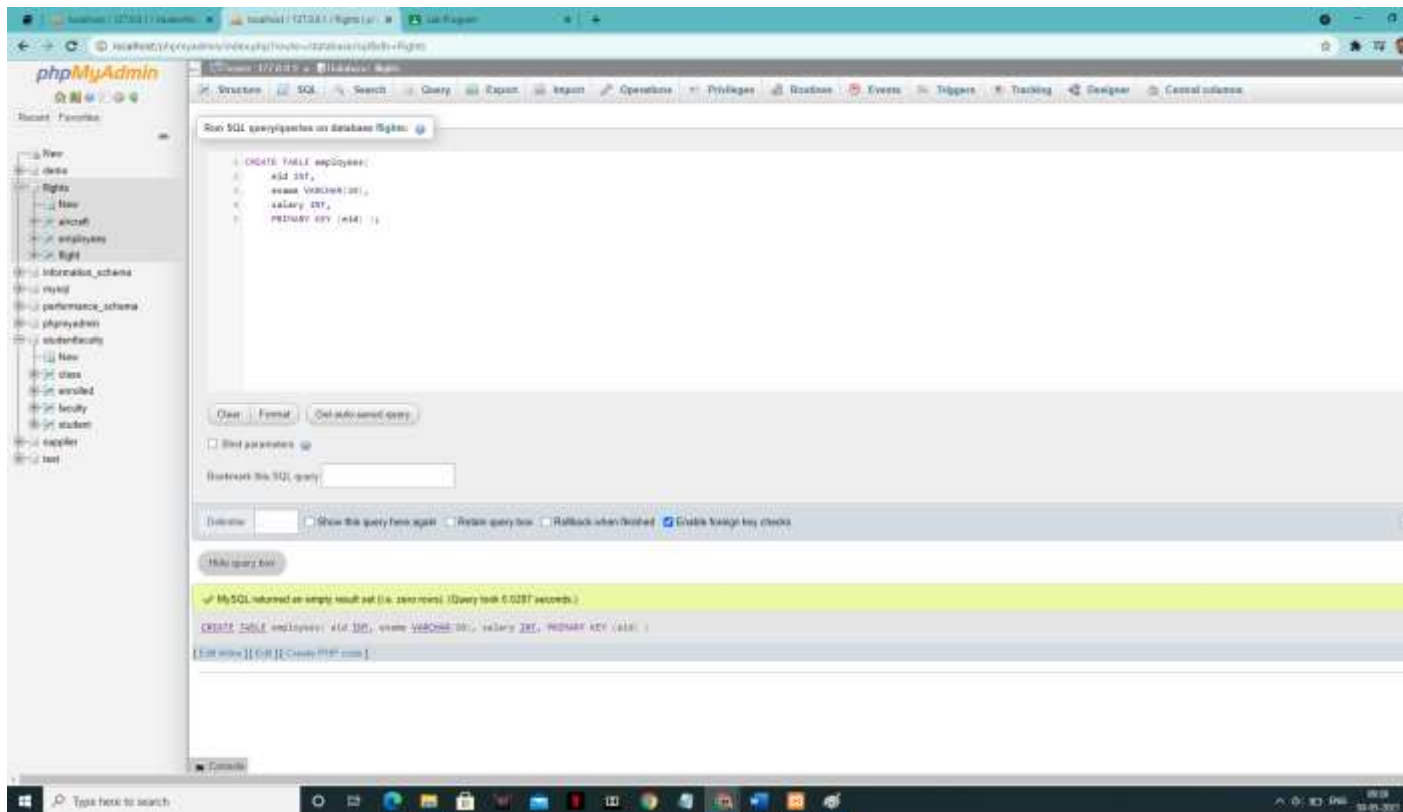


2) Enter tuples for each relation.

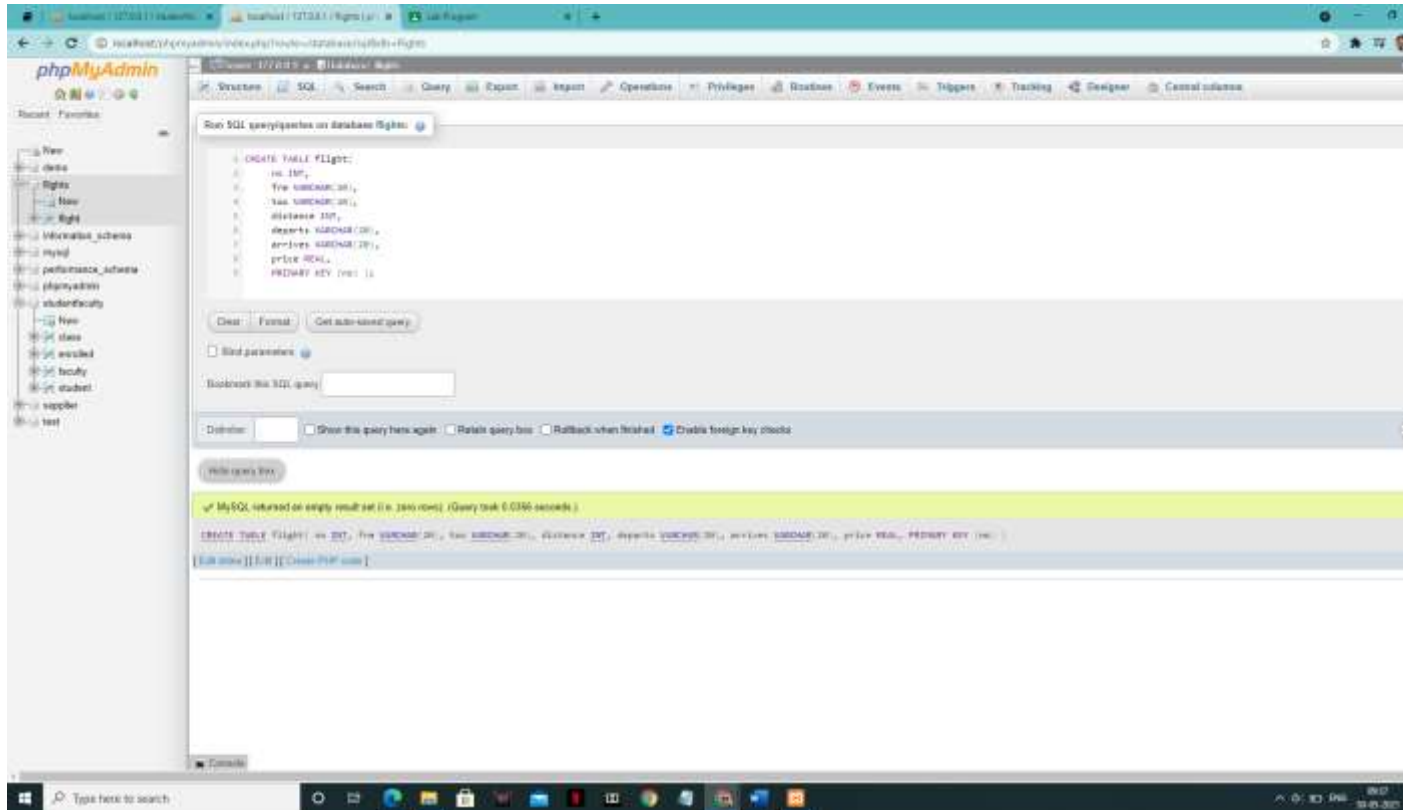
‘AIRCRAFT’ table:

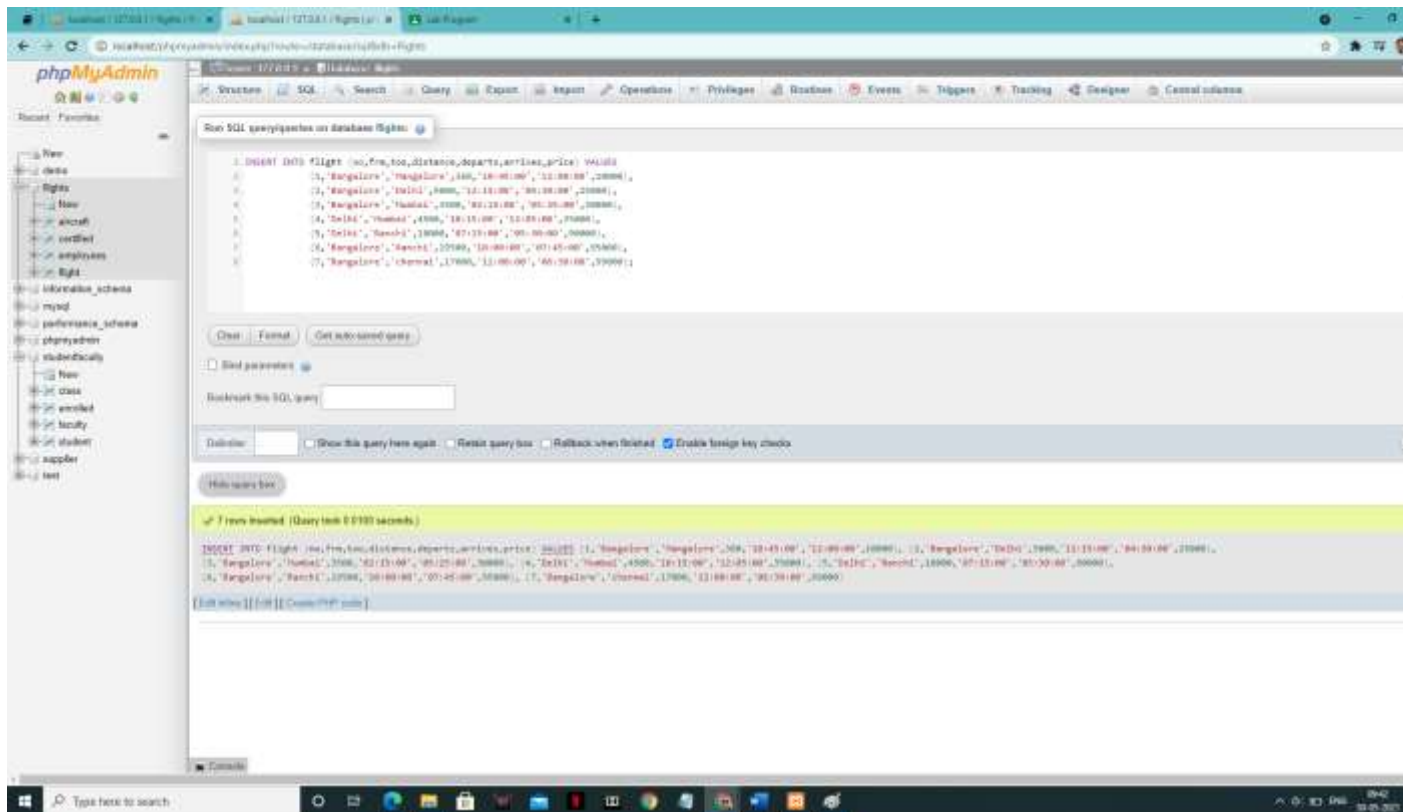




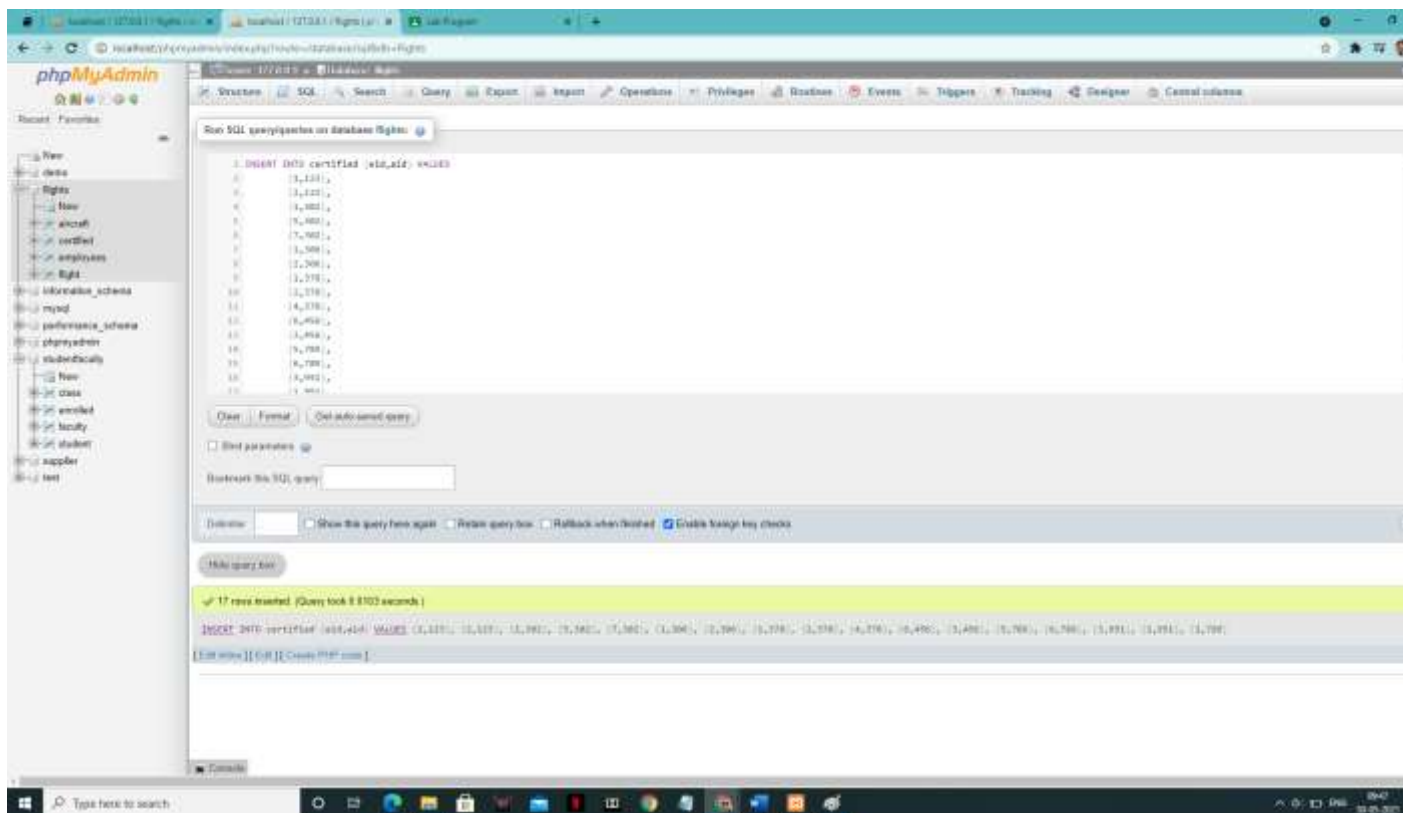


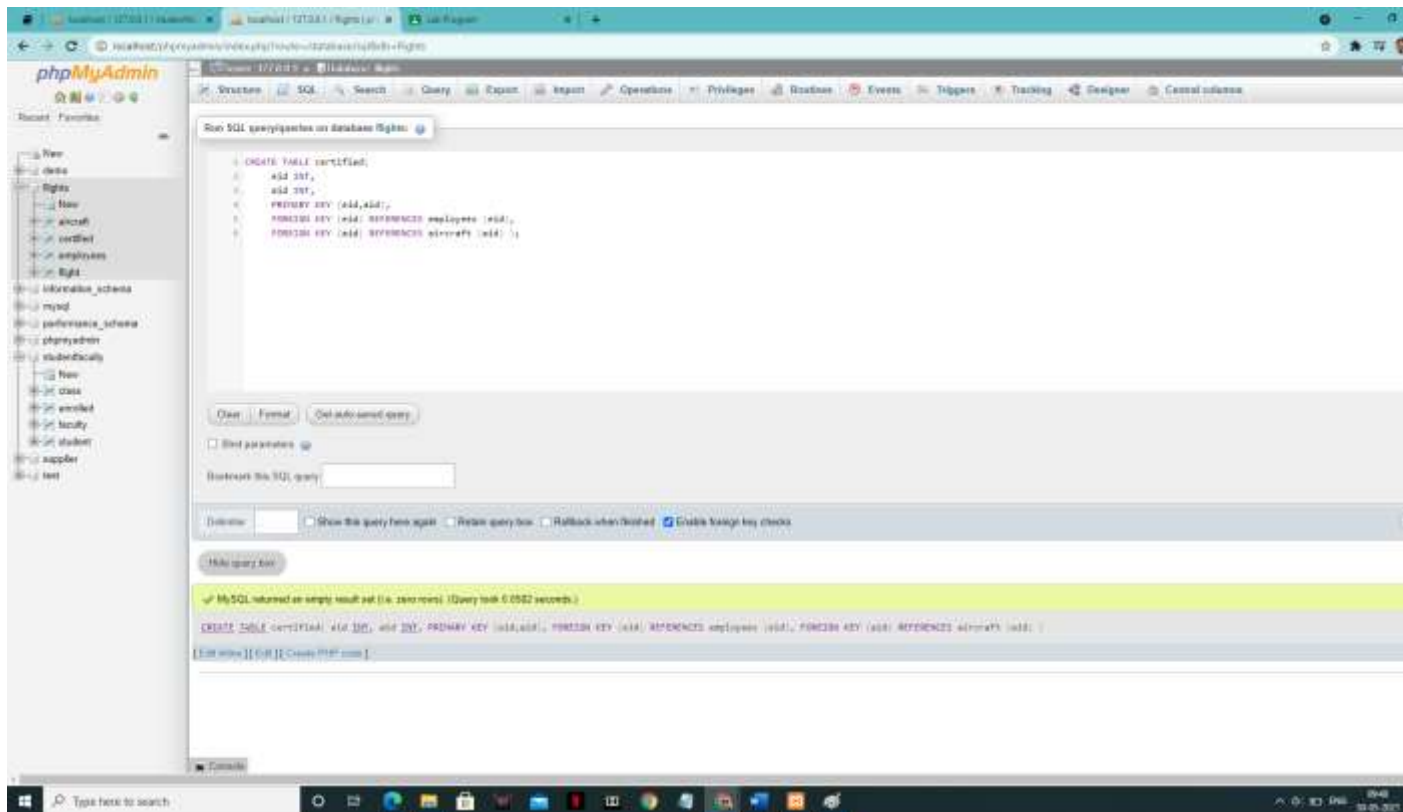
‘FLIGHT’ table:





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: ☐ Enable 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		Alban
<input type="checkbox"/>	✎ Edit	📄 Copy	🗑 Delete		Barney
<input type="checkbox"/>	✎ Edit	📄 Copy	🗑 Delete		Jack
<input type="checkbox"/>	✎ Edit	📄 Copy	🗑 Delete		Archie III
<input type="checkbox"/>	✎ Edit	📄 Copy	🗑 Delete		Archie III

🔍 Check all | With selected: ☐ Edit ☐ Copy ☐ Delete ☐ Export

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

Query results operations

Console

2:-



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>1000 GROUP BY a.airc,a.empno
```

The result shows 3 rows:

airc	empno	salary
332	Swamp	73333.3333
336	arct1	17100.0000
370	Albus DB	53333.3333

5:-

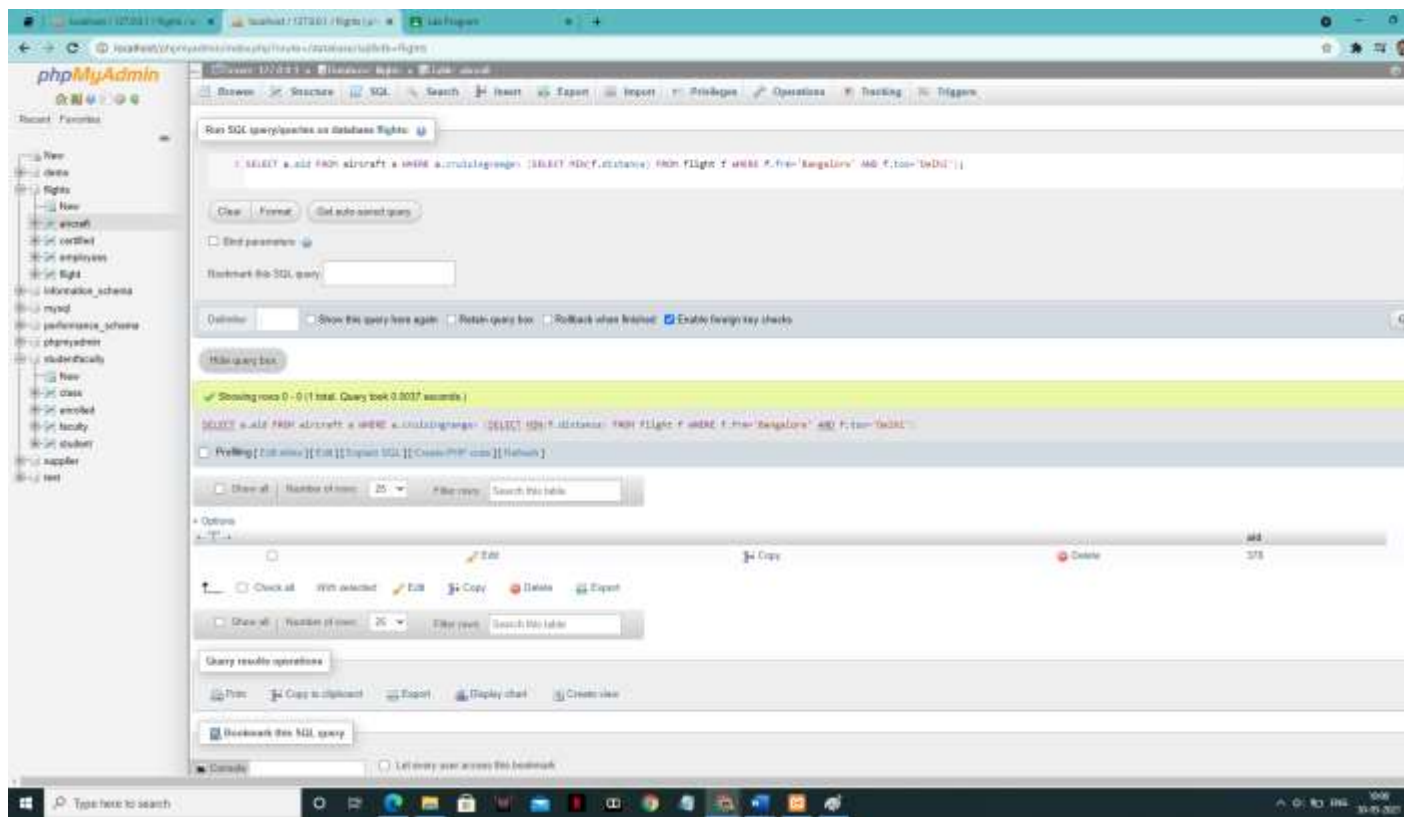
The screenshot shows the phpMyAdmin interface with a SQL query executed. The query is:

```
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'
```

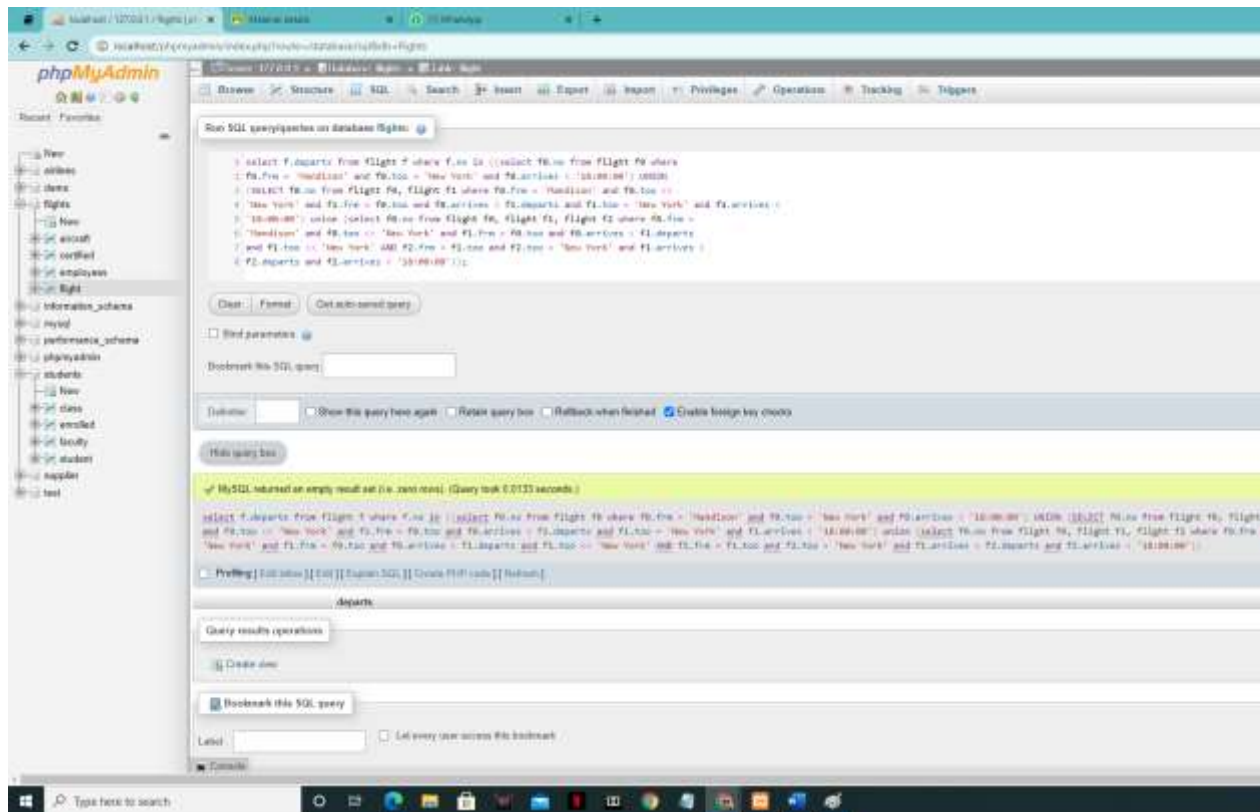
The result shows 3 rows:

empno
Swamp
Arct
Arct

6:-



7:-



8:-

