Name: Brendan Lucas, Div: SE COMP B, Roll No: 8953

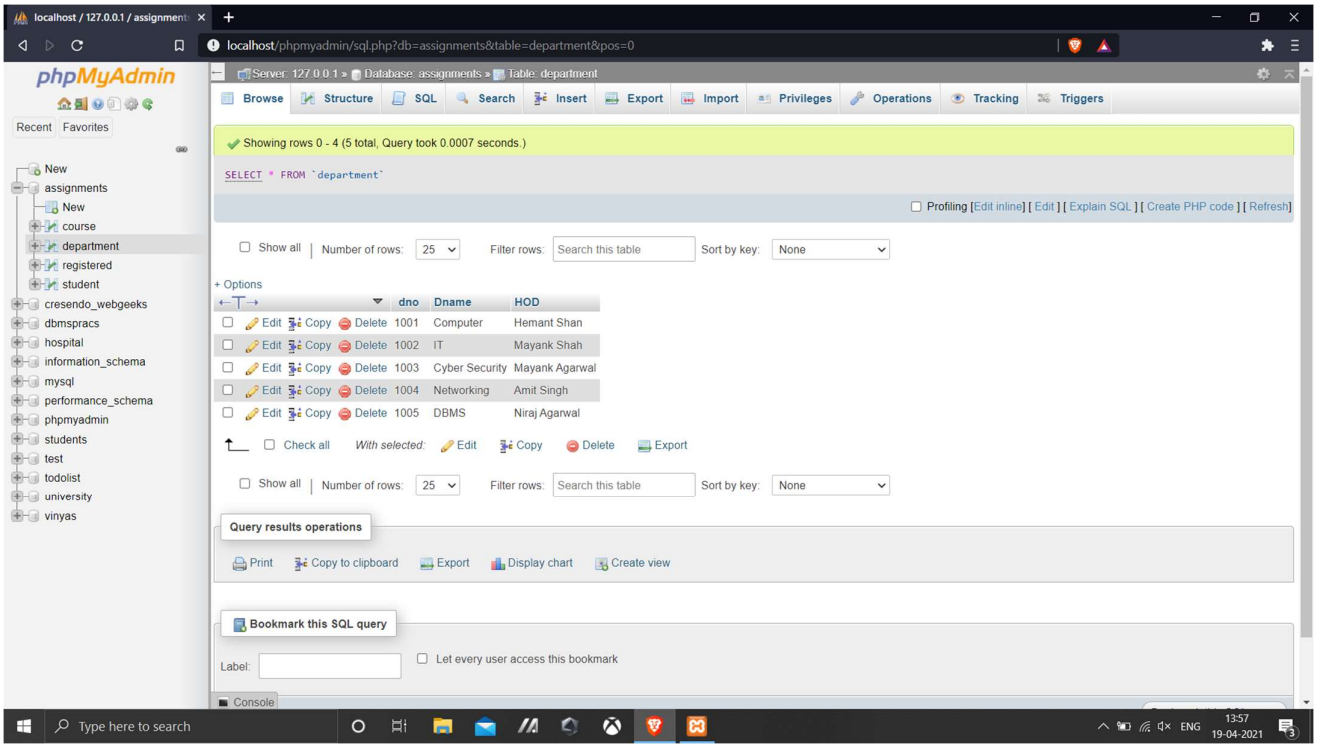
DBMS Assignment No 1.

Q1. Implement the following in MySQL:

A) Create the following Tables and Insert Values:

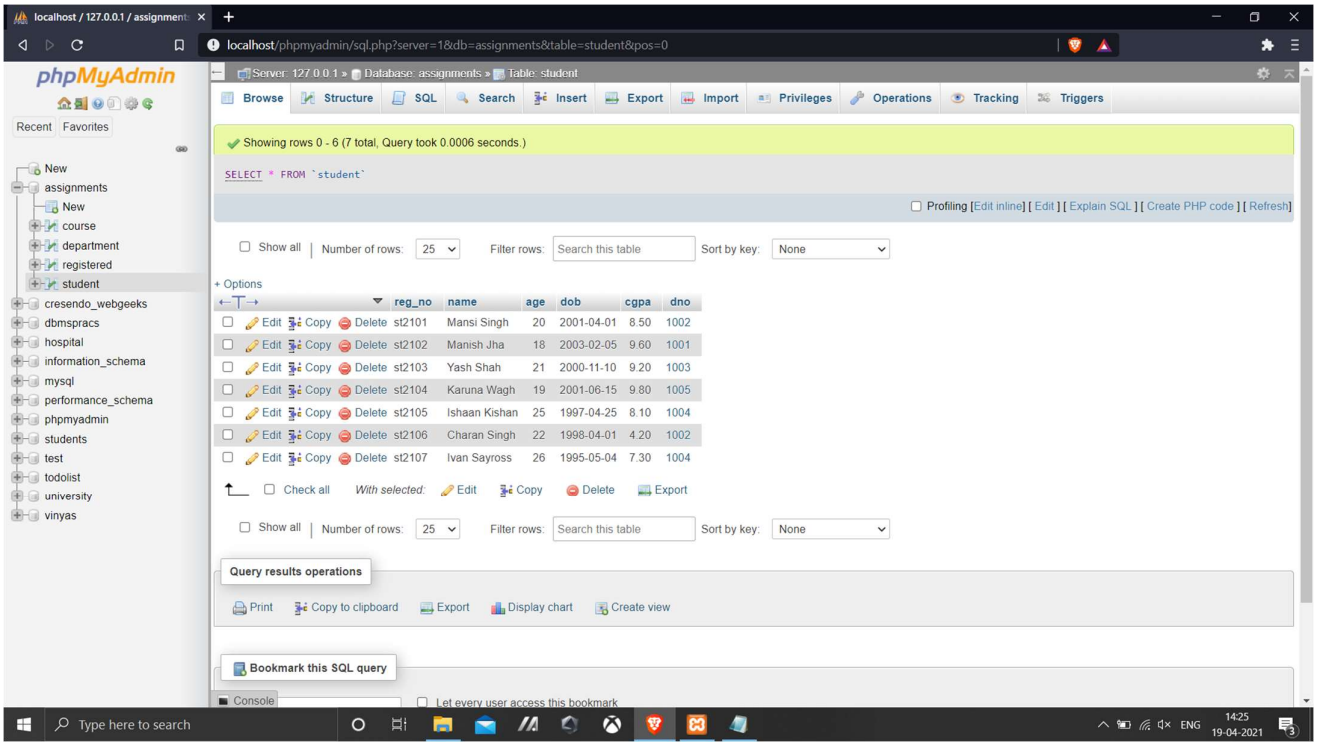
i) Table Department:

```
CREATE TABLE department(
        dno INT PRIMARY KEY NOT NULL,
    Dname VARCHAR(20) NOT NULL,
    HOD VARCHAR(20) NOT NULL
);

INSERT INTO `department` (`dno`, `Dname`, `HOD`) VALUES
('1001','Computer','Hemant Shan'),
('1002','IT','Mayank Shah'),
('1003','Cyber Security','Mayank Agarwal'),
('1004','Networking','Amit Singh'),
('1005','DBMS','Niraj Agarwal');
```



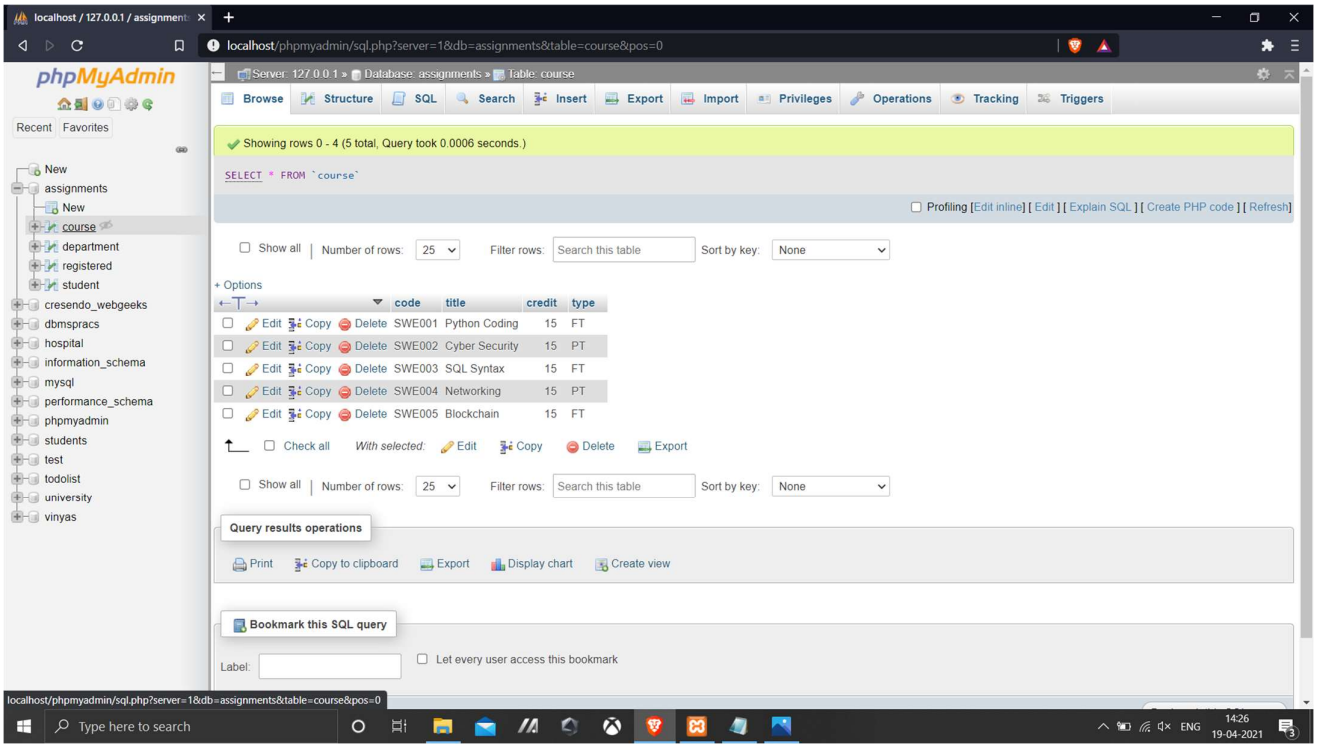
ii) Student Table:

```
CREATE TABLE student(
        reg_no VARCHAR(6) PRIMARY KEY NOT NULL,
    name VARCHAR(20) NOT NULL,
    age SMALLINT CHECK (age>0),
    dob date NOT NULL,
    cgpa decimal(3,2),
    dno INT,
    FOREIGN KEY (dno) REFERENCES department(dno)
);

INSERT INTO `student` (`reg_no`, `name`, `age`, `dob`, `cgpa`, `dno`) VALUES
('st2101', 'Mansi Singh', '20', '2001-04-01', '8.5', '1002'),
('st2102', 'Manish Jha', '18', '2003-02-05', '9.6', '1001'),
('st2103', 'Yash Shah', '21', '2000-11-10', '9.2', '1003'),
('st2104', 'Karuna Wagh', '19', '2001-06-15', '9.8', '1005'),
('st2105', 'Ishaan Kishan', '25', '1997-04-25', '8.1', '1004'),
('st2106', 'Charan Singh', '22', '1998-04-01', '4.2', '1002'),
('st2107', 'Ivan Sayross', '26', '1995-05-04', '7.3', '1004');
```



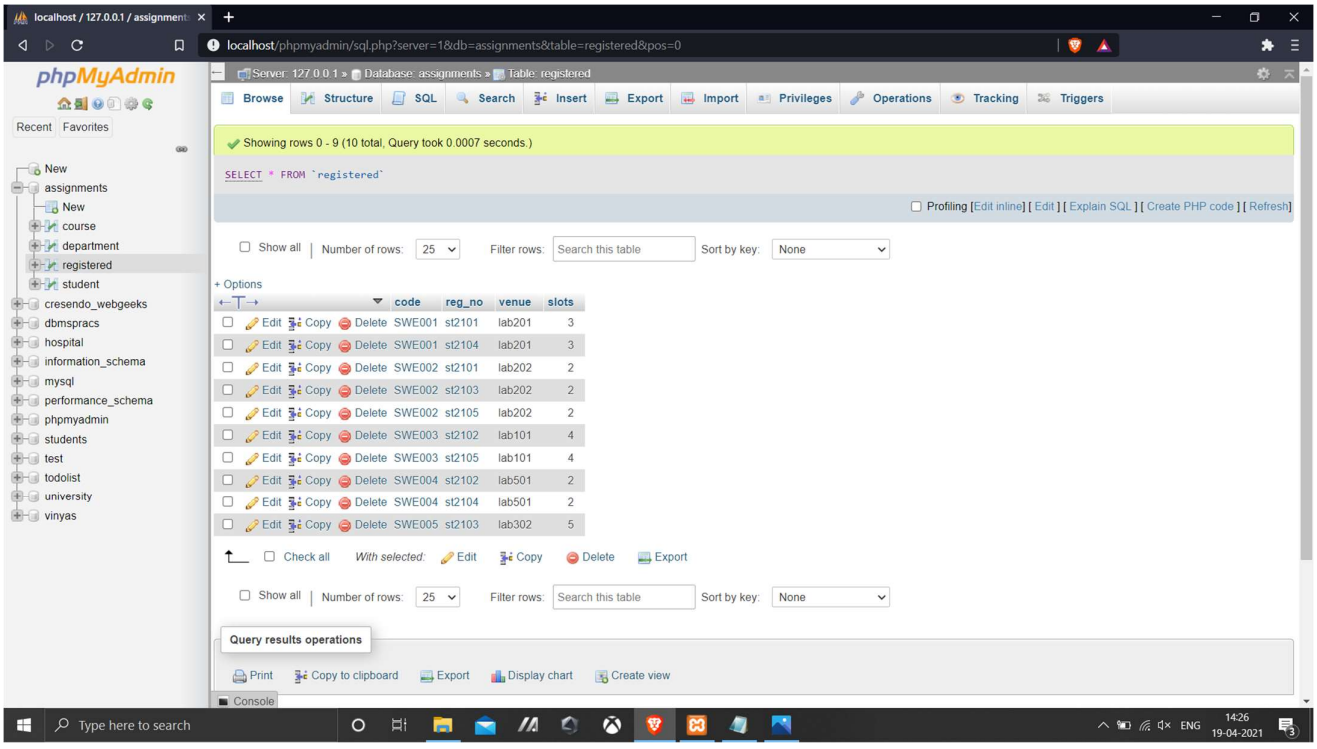
iii) Courses Table:

```
CREATE TABLE course(
        course_code VARCHAR(6) PRIMARY KEY NOT NULL,
    title VARCHAR(20) NOT NULL,
    credit SMALLINT CHECK (credit>0),
    type VARCHAR(2) DEFAULT "FT"
);

INSERT INTO `course` (`code`, `title`, `credit`, `type`) VALUES
('SWE001', 'Python Coding', '15', 'FT'),
('SWE002', 'Cyber Security', '15', 'PT'),
('SWE003', 'SQL Syntax', '15', 'FT'),
('SWE004', 'Networking', '15', 'PT'),
('SWE005', 'Blockchain', '15', 'FT');
```



iv) Registered Table:

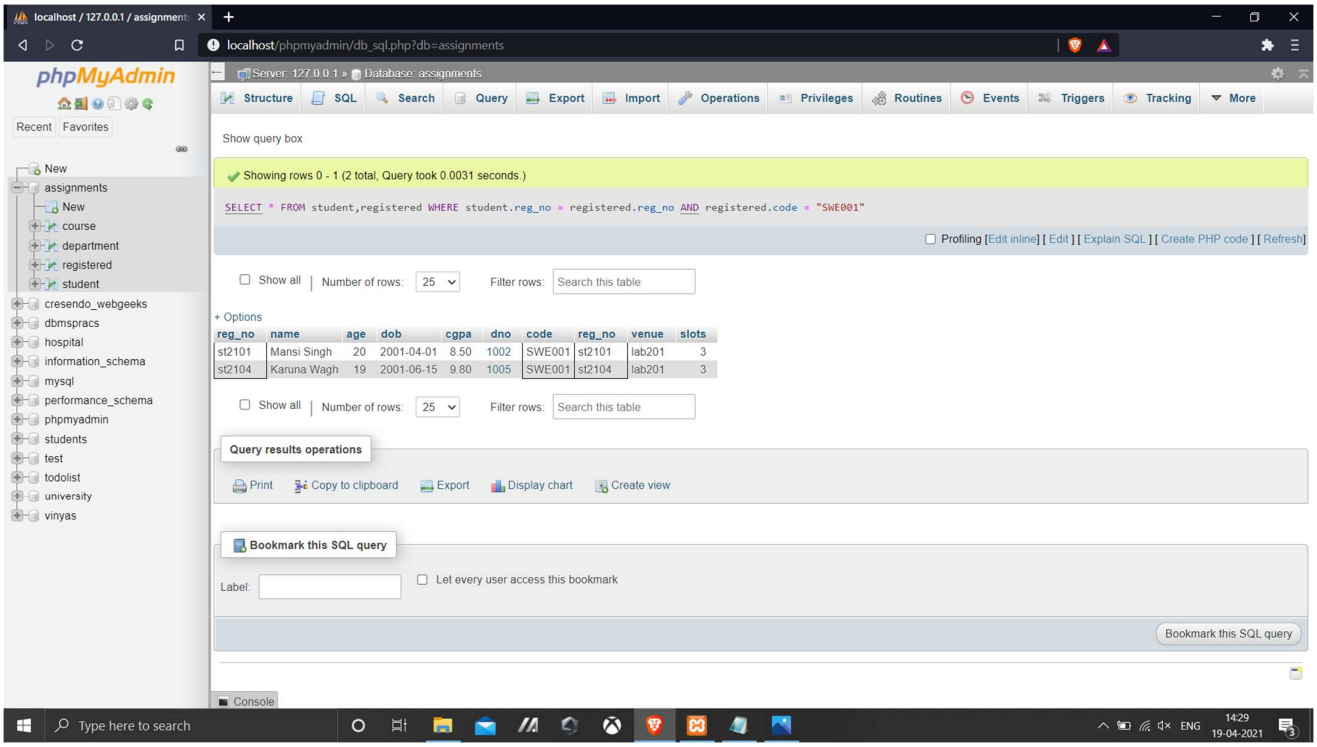
```
CREATE TABLE registered(
        code VARCHAR(6) NOT NULL,
        reg_no VARCHAR(6) NOT NULL,
   venue VARCHAR(6) NOT NULL,
   slots SMALLINT NOT NULL CHECK (slots>0),
   PRIMARY KEY (code,reg_no),
   FOREIGN KEY (code) REFERENCES course(code),
   FOREIGN KEY (reg_no) REFERENCES student(reg_no)
);

INSERT INTO `registered` (`code`, `reg_no`, `venue`, `slots`) VALUES
('SWE001', 'st2101', 'lab201', '3'),
('SWE002', 'st2101', 'lab202', '2'),
('SWE003', 'st2102', 'lab101', '4'),
('SWE004', 'st2102', 'lab501', '2'),
('SWE005', 'st2103', 'lab302', '5'),
('SWE002', 'st2103', 'lab202', '2'),
('SWE001', 'st2104', 'lab201', '3'),
('SWE004', 'st2104', 'lab501', '2'),
('SWE003', 'st2105', 'lab101', '4'),
('SWE002', 'st2105', 'lab202', '2');
```



B) Execute Queries for the following:

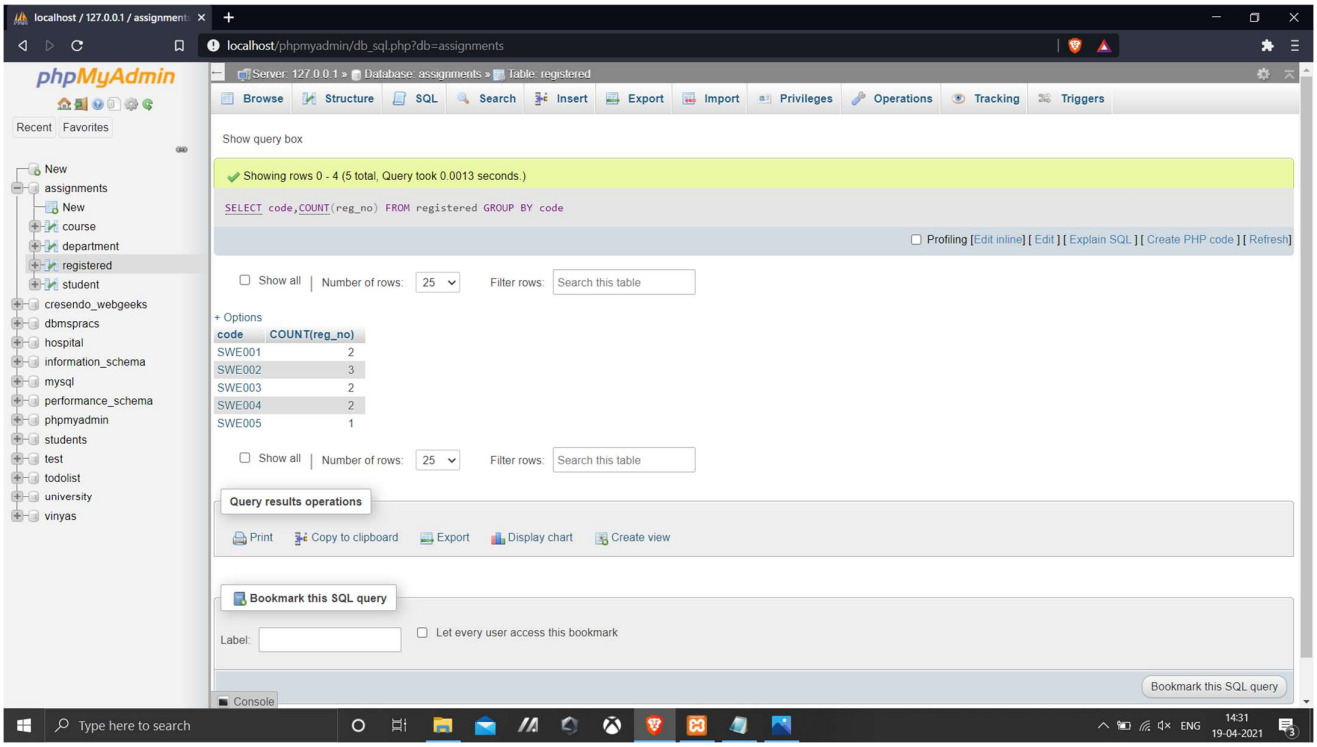
i) List the student details who registered Course code ‘SWE1001’:

```
SELECT *
FROM student,registered
WHERE student.reg_no = registered.reg_no
AND registered.code = "SWE001";
```



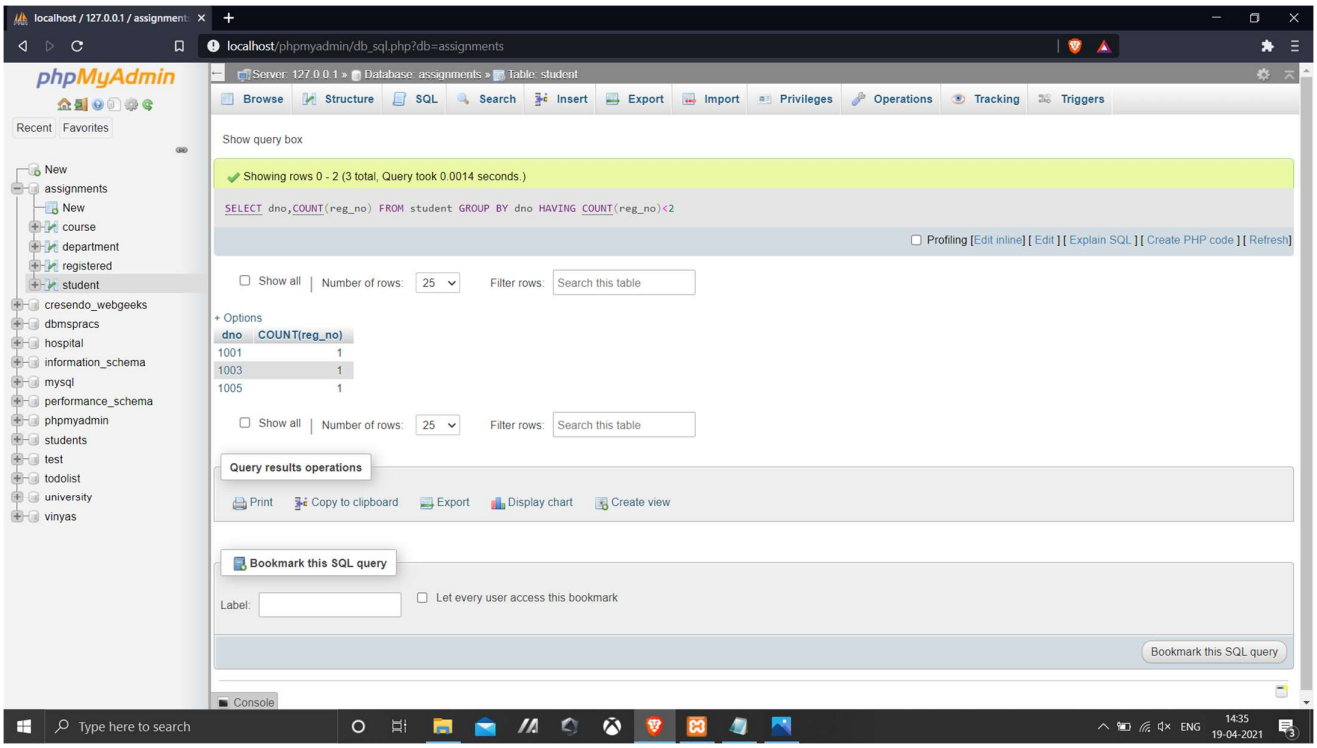
ii) List the number of students in each course.:

```
SELECT code,COUNT(reg_no)
FROM registered
GROUP BY code;
```



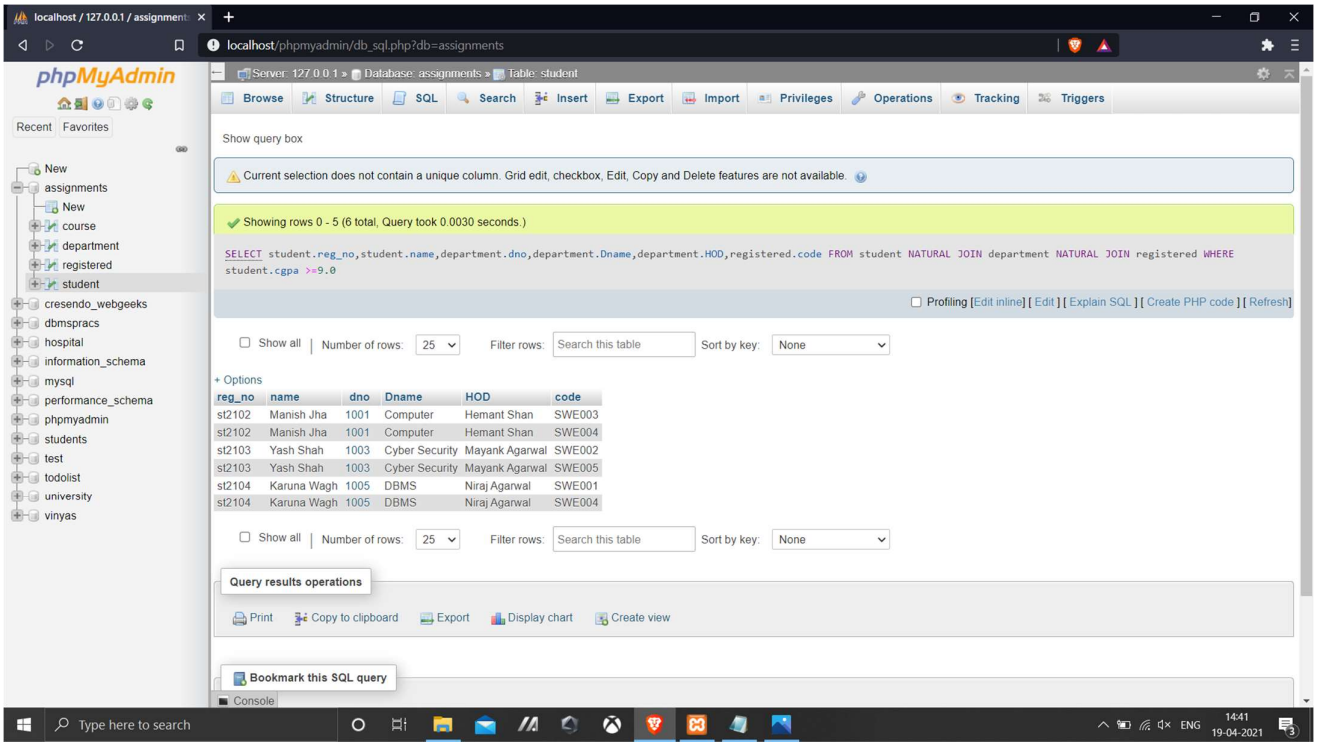
iii) List the department no which are having less than 2 students.:

```
SELECT dno,COUNT(reg_no)
FROM student
GROUP BY dno
HAVING COUNT(reg_no)<2;
```



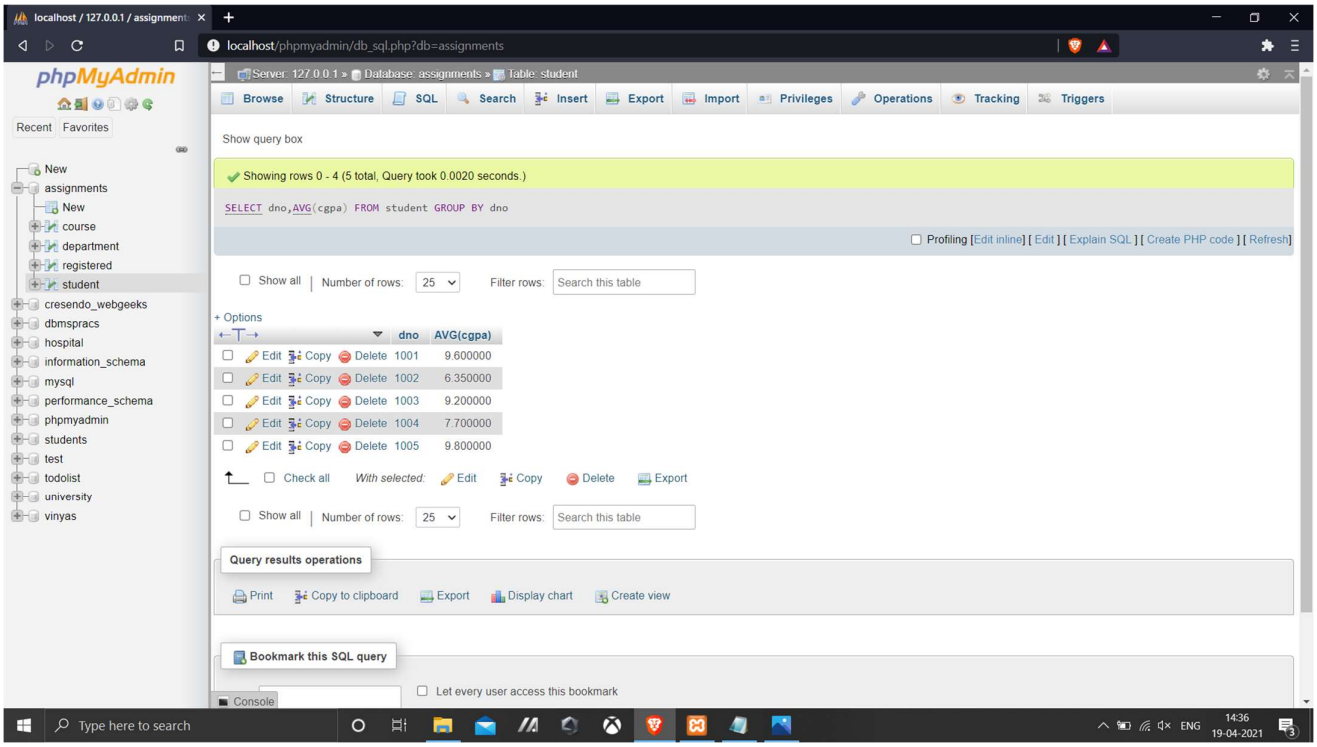
iv) List the student name, department name, HOD and the courses registered by him/her who is having more than 9.0 CGPA:

```
SELECT
student.reg_no,student.name,department.dno,department.Dname,department.HOD,registered
.code
FROM student NATURAL JOIN department NATURAL JOIN registered
WHERE student.cgpa >=9.0 ;
```



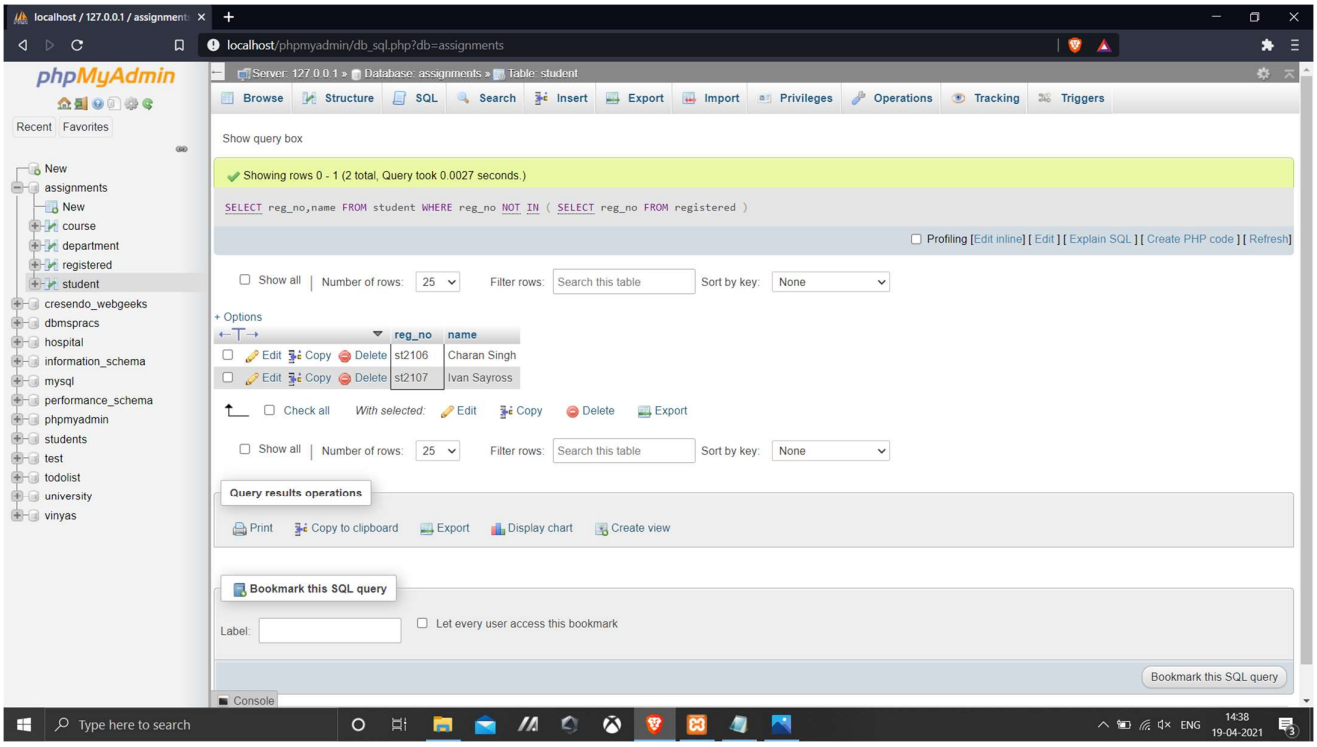
v) Retrieve the average CGPA of each department:

```
SELECT dno,AVG(cgpa)
FROM student
GROUP BY dno;
```



vi) Retrieve the student register numbers who are not registered for any course:

```
SELECT reg_no,name
FROM student
WHERE reg_no NOT IN (
    SELECT reg_no
    FROM registered;
);
```

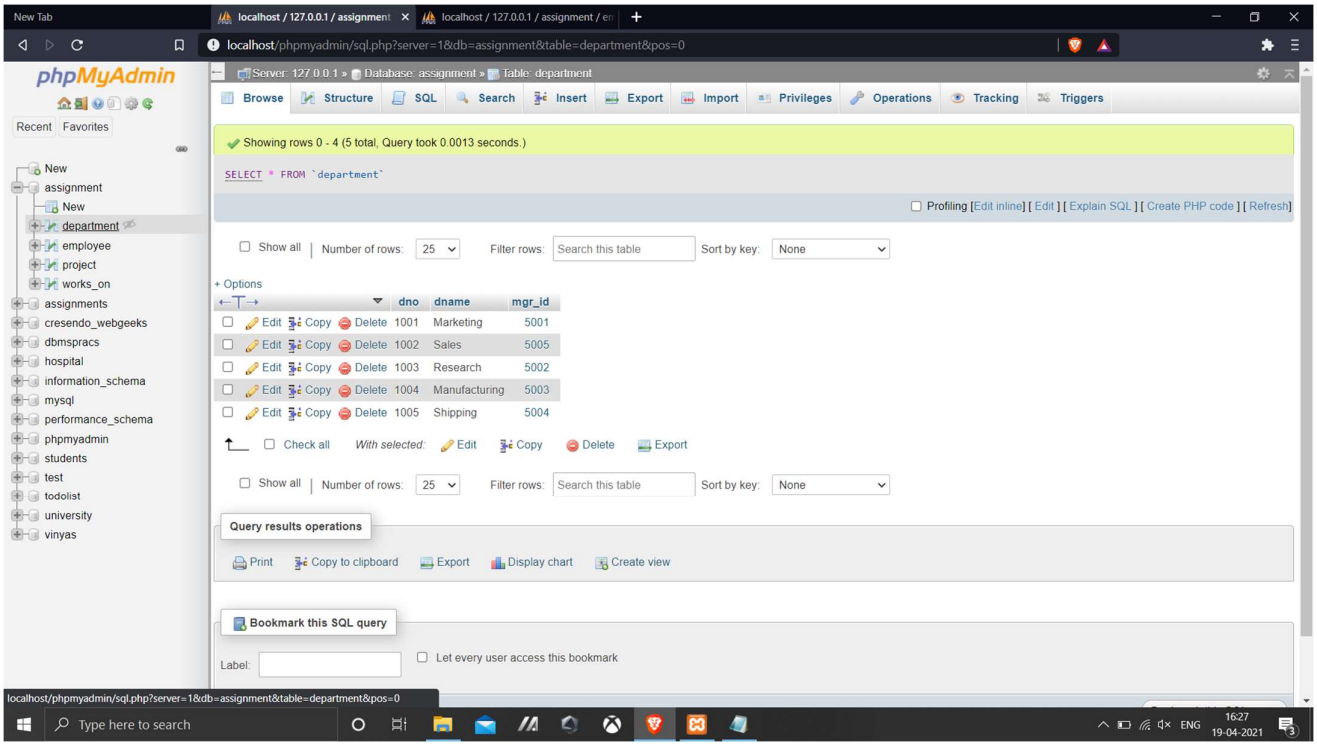


Q2. Implement the following in MySQL:

A) Create the following Tables and Insert Values:

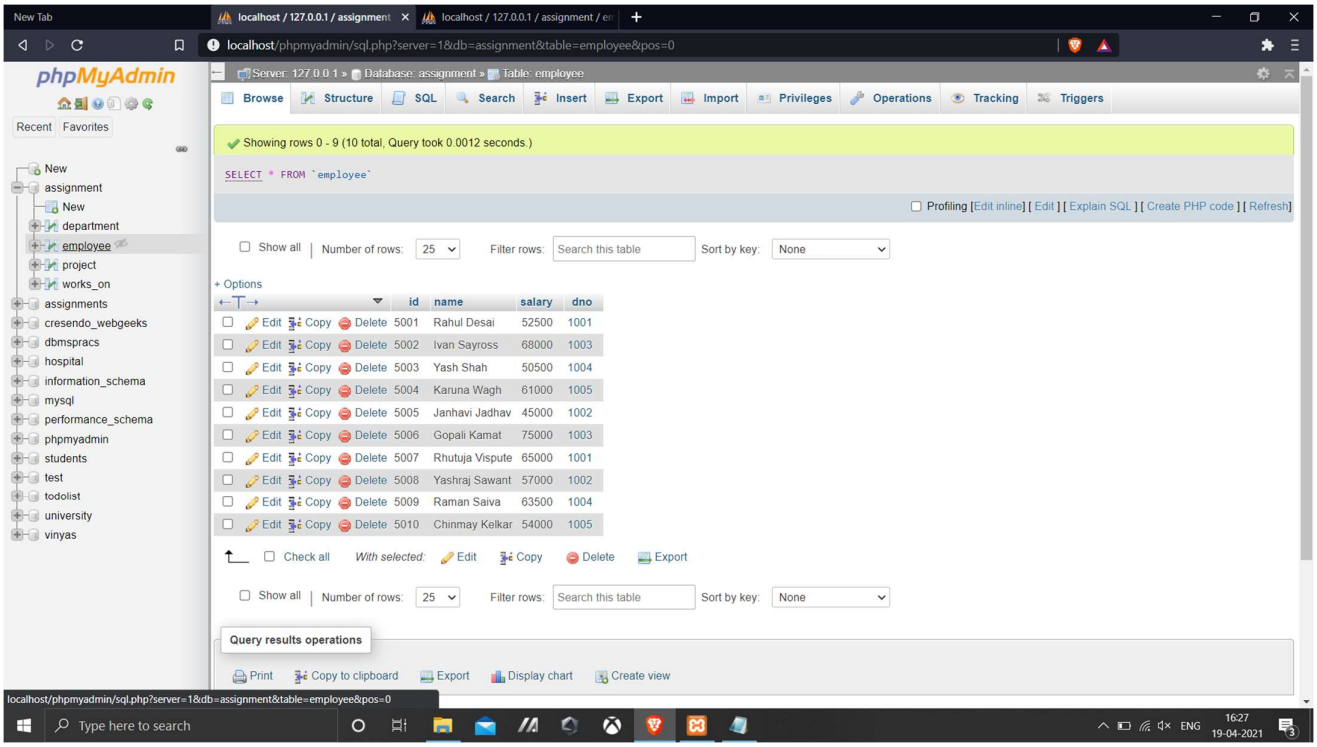
i) Table Department:

```
CREATE TABLE department(
    dno INT PRIMARY KEY NOT NULL,
    dname VARCHAR(20),
    mgr_id int
);

INSERT INTO `department` (`dno`, `dname`, `mgr_id`) VALUES
('1001', 'Marketing', '5001'),
('1002', 'Sales', '5005'),
('1003', 'Research', '5002'),
('1004', 'Manufacturing', '5003'),
('1005', 'Shipping', '5004');

ALTER TABLE department ADD FOREIGN KEY (mgr_id) REFERENCES employee(id);
```



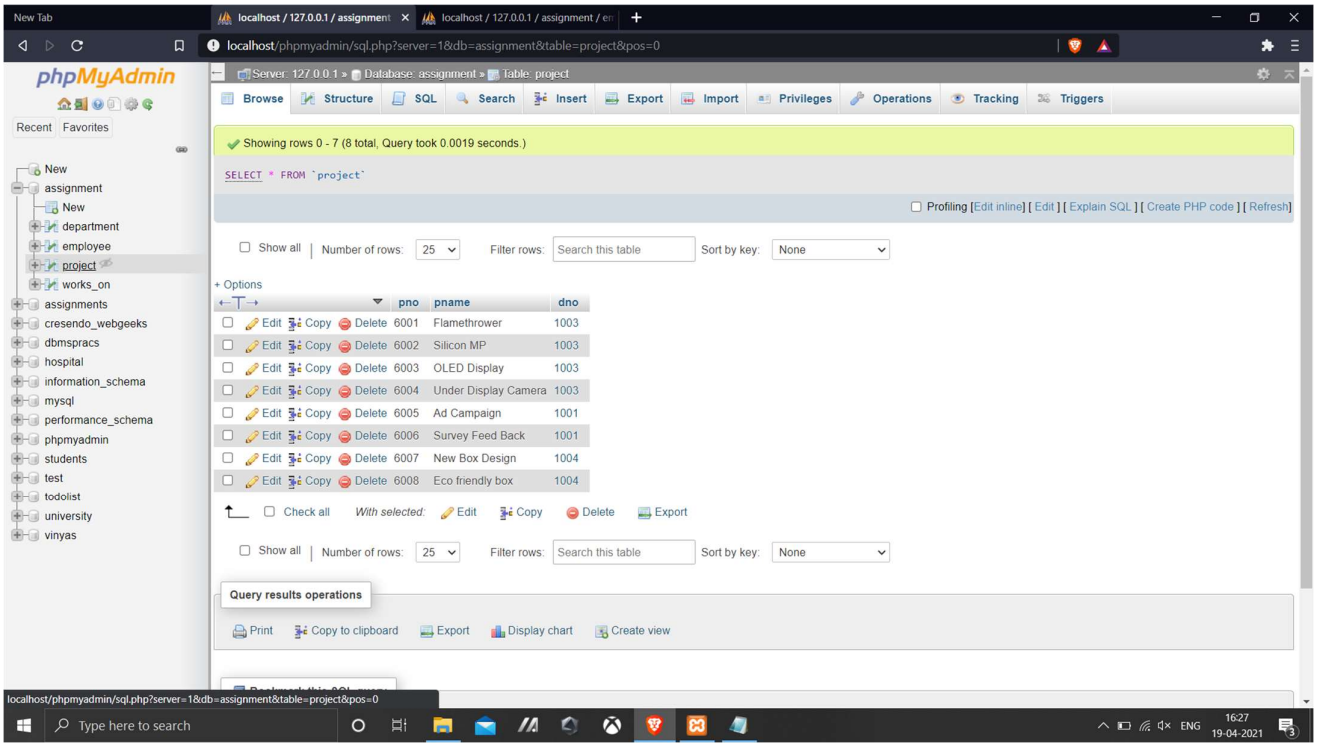
ii) Table Employee:

```
CREATE TABLE employee(
    id INT PRIMARY KEY NOT NULL,
    name VARCHAR(20),
    salary INT,
    dno INT,
    FOREIGN KEY (dno) REFERENCES department(dno)
);

INSERT INTO `employee` (`id`, `name`, `salary`, `dno`) VALUES
('5001', 'Rahul Desai', '52500', '1001'),
('5002', 'Ivan Sayross', '68000', '1003'),
('5003', 'Yash Shah', '50500', '1004'),
('5004', 'Karuna Wagh', '61000', '1005'),
('5005', 'Janhavi Jadhav', '45000', '1002'),
('5006', 'Gopali Kamat', '75000', '1003'),
('5007', 'Rhutuja Vispute', '65000', '1001'),
('5008', 'Yashraj Sawant', '57000', '1002'),
('5009', 'Raman Saiva', '63500', '1004'),
('5010', 'Chinmay Kelkar', '54000', '1005');
```



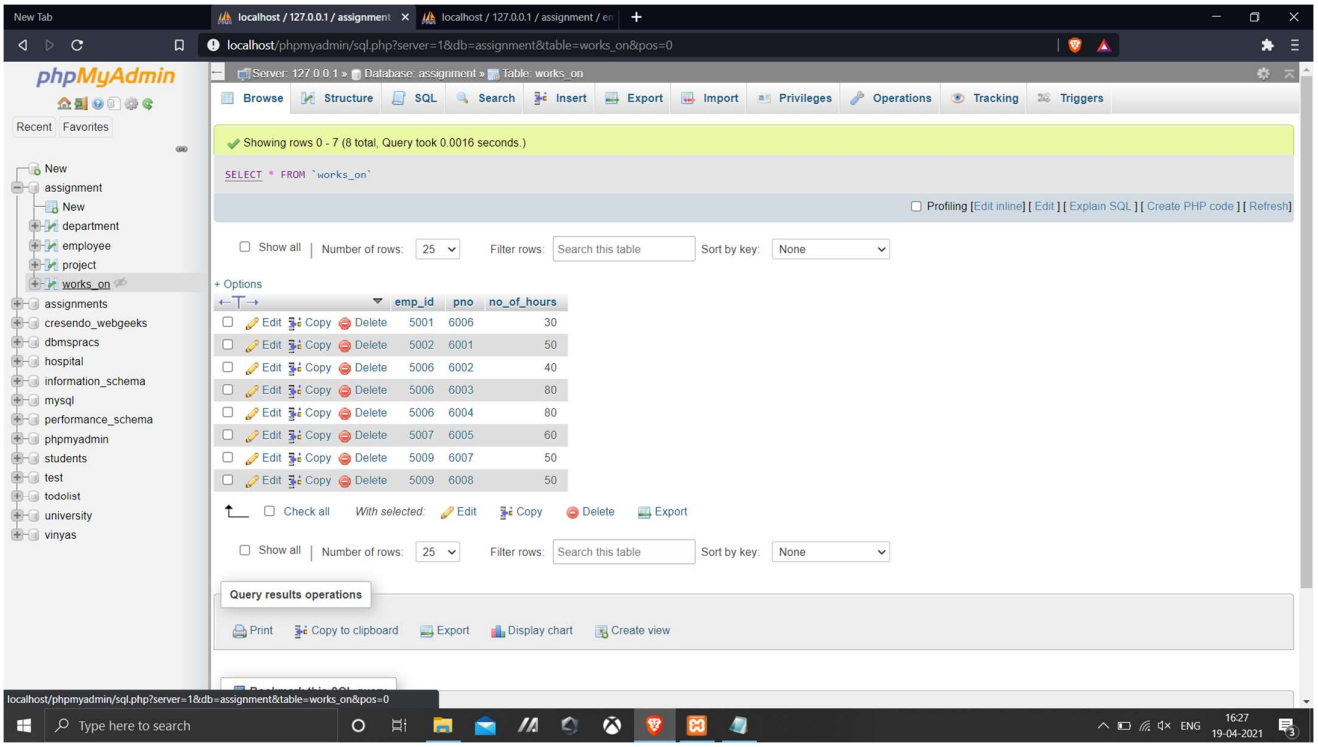
iii) Table Projects:

```
CREATE TABLE project(
   pno INT PRIMARY KEY NOT NULL,
   pname VARCHAR(20),
   dno INT,
   FOREIGN KEY (dno) REFERENCES department(dno)
);

INSERT INTO `project` (`pno`, `pname`, `dno`) VALUES
('6001', 'Flamethrower', '1003'),
('6002', 'Silicon MP', '1003'),
('6003', 'OLED Display', '1003'),
('6004', 'Under Display Camera', '1003'),
('6005', 'Ad Campaign', '1001'),
('6006', 'Survey Feed Back', '1001'),
('6007', 'New Box Design', '1004'),
('6008', 'Eco friendly box', '1004');
```



i) Table Works_on:

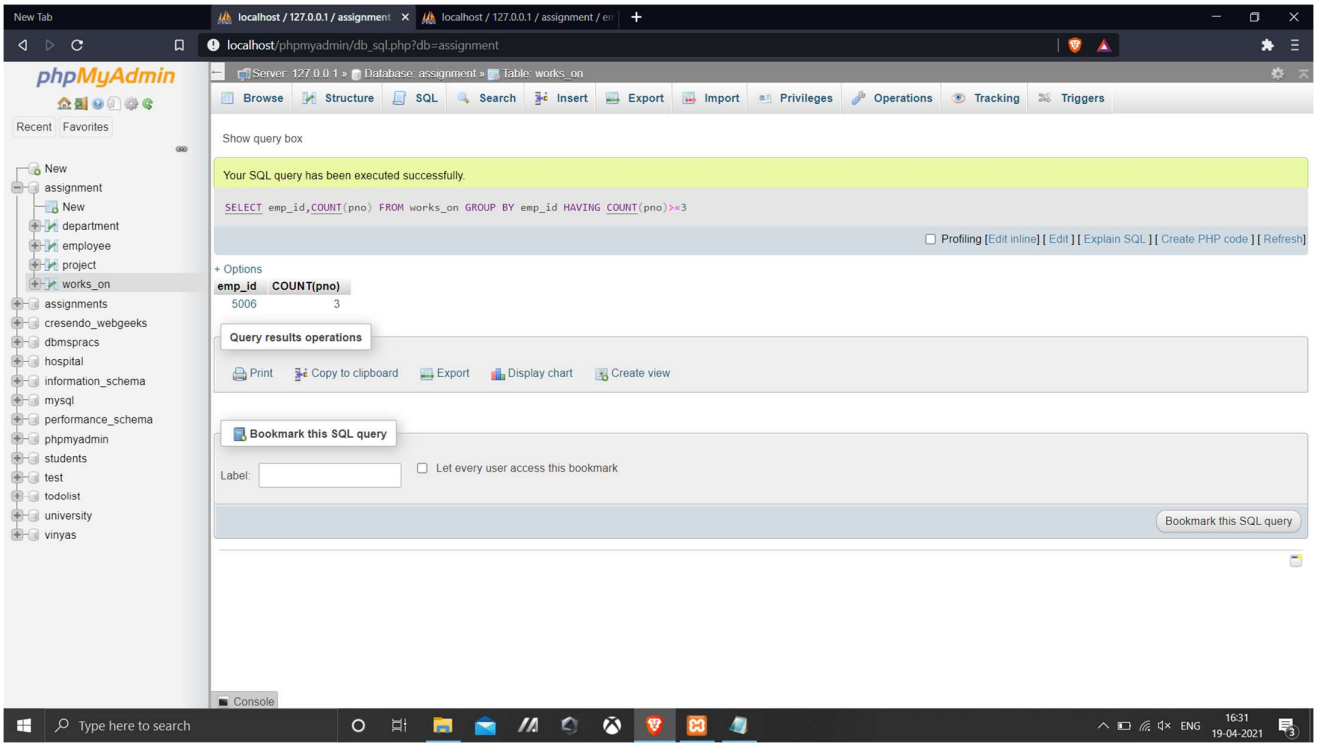
```
CREATE TABLE works_on(
    emp_id INT NOT NULL,
    pno INT NOT NULL,
    no_of_hours INT NOT NULL,
    FOREIGN KEY (pno) REFERENCES project(pno),
    FOREIGN KEY (emp_id) REFERENCES employee(id),
    PRIMARY KEY (emp_id,pno)
);

INSERT INTO `works_on` (`emp_id`, `pno`, `no_of_hours`) VALUES
('5006', '6002', '40'),
('5006', '6004', '80'),
('5006', '6003', '80'),
('5002', '6001', '50'),
('5007', '6005', '60'),
('5001', '6006', '30'),
('5009', '6008', '50'),
('5009', '6007', '50');
```



B) Execute Queries for the following:

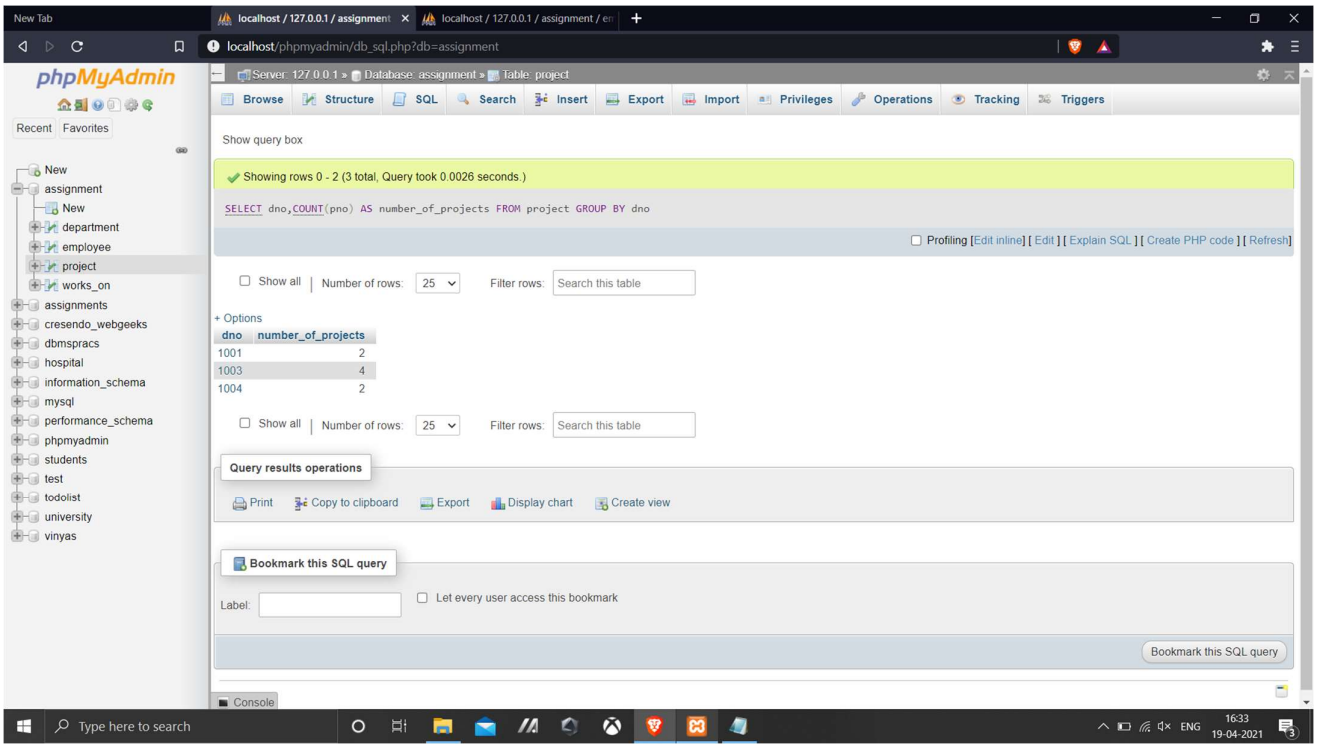
i) List the employee id who is working on more than 3 projects.:

```
SELECT emp_id,COUNT(pno)
FROM works_on
GROUP BY emp_id
HAVING COUNT(pno)>=3;
```



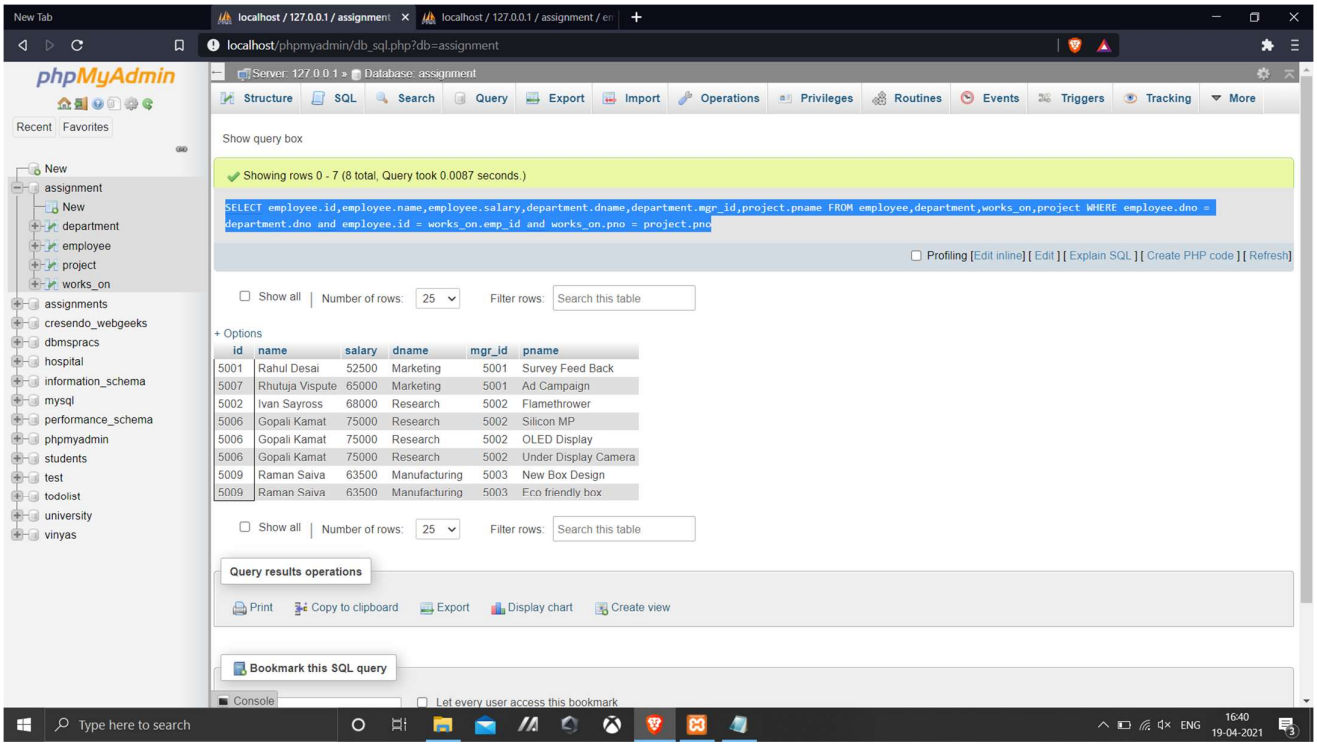
ii) Retrieve the number of projects in each department.

```
SELECT dno,COUNT(pno) AS number_of_projects
FROM project
GROUP BY dno;
```



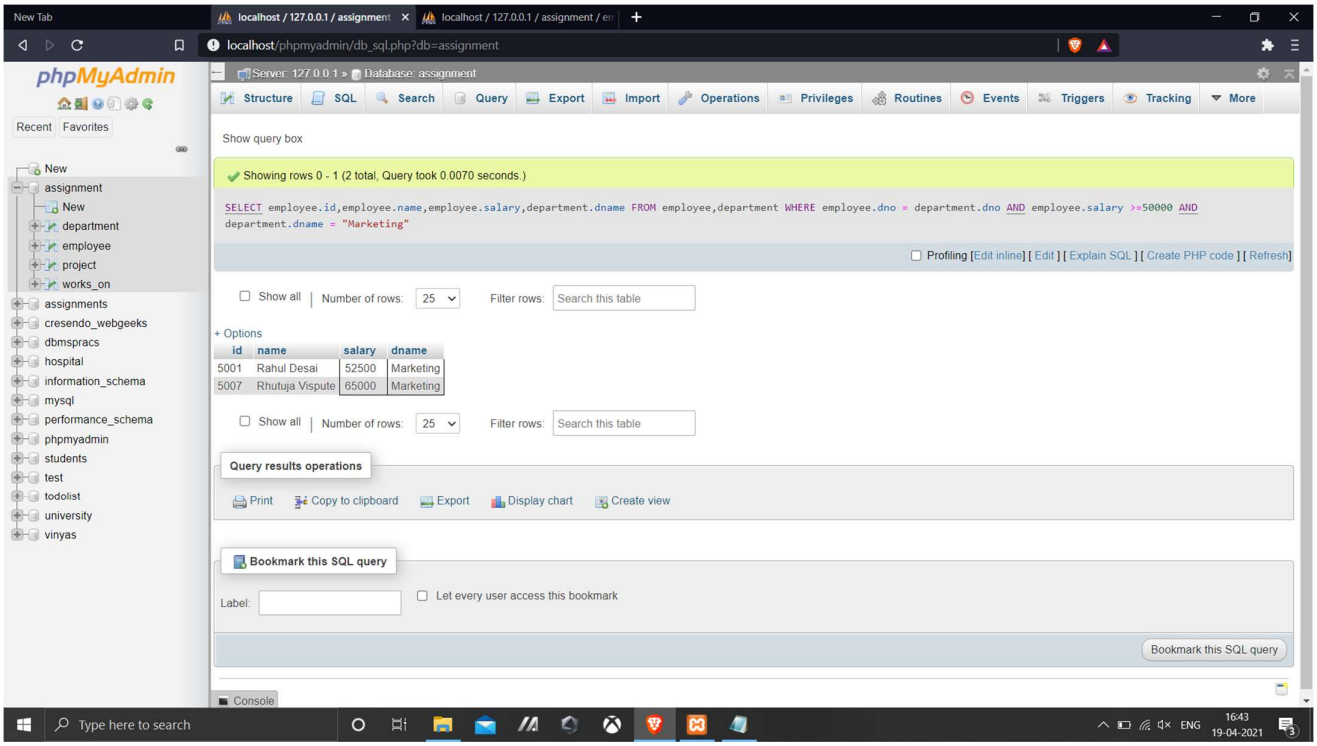
iii) Retrieve the employee id, name, salary, dname, manager id and the project name he/she has been working on:

```
SELECT
employee.id,employee.name,employee.salary,department.dname,department.mgr_id,project.pname
FROM employee,department,works_on,project
WHERE employee.dno = department.dno and employee.id = works_on.emp_id and works_on.pno = project.pno
```



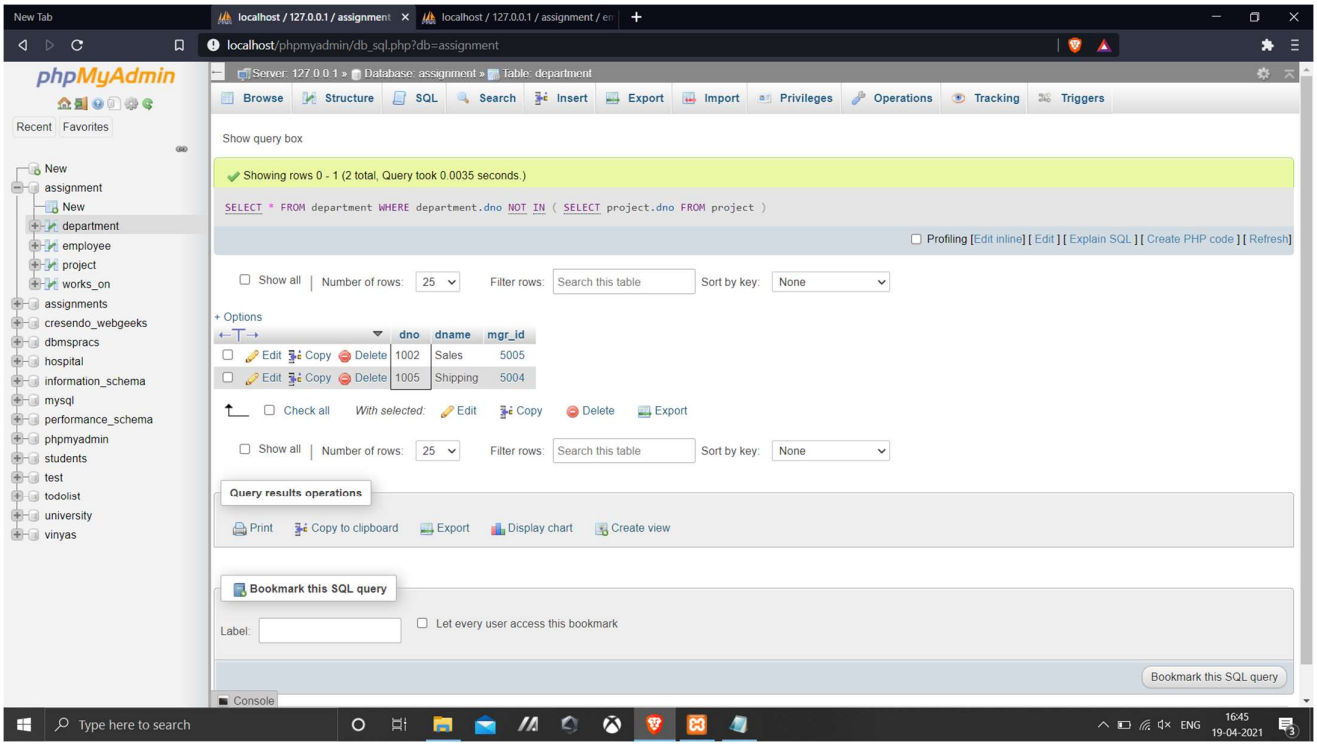
iv) Retrieve the employee details who are working on Department ‘Marketing’ and getting the salary more than 50000:

```
SELECT employee.id,employee.name,employee.salary,department.dname
FROM employee,department
WHERE employee.dno = department.dno AND employee.salary >=50000 AND
department.dname = "Marketing"
```



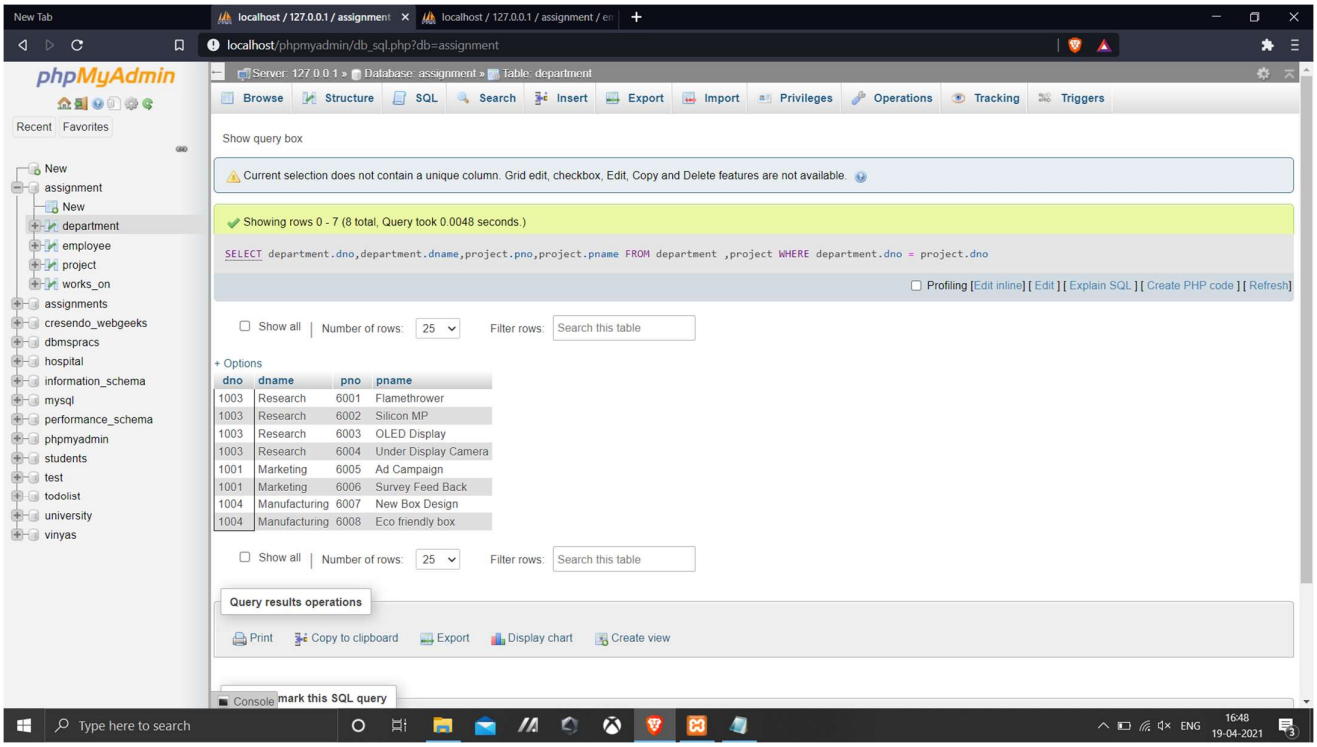
v) Retrieve the department numbers which does not have any project.:

```
SELECT *
FROM department
WHERE department.dno NOT IN
(
 SELECT project.dno FROM project
);
```



vi) Retrieve all the department numbers, department name and controlling project number, project name if any:

```
SELECT department.dno, department.dname, project.pno, project.pname
FROM department ,project
WHERE department.dno = project.dno;
```



Q3

(1) List the student details who registered in course code = 'SWE1001'.

Ans

$$\Pi_{*}\left(\sigma_{r.code = 'SWE1001'}\left(\rho_{s}(student) \bowtie \rho_{r}(Registered)\right)\right)$$

(2) List the number of students in each course

Ans

$$code\ \mathcal{G}_{count(regno)}(Registered)$$

OR.

$$\gamma_{code,\ count(regno)}(Registered)$$

(3) List the department no which are having less than 2 students.

$$\gamma_{dno,\ count(regno)<2}(Registered)$$

OR

$$dno\ \mathcal{G}_{count(regno)<2}(Registered)$$

(4) List the student name, department name, HOD, and course registered by student having more than 9.0 CGPA

$$\Pi_{s.sname,\ d.dname,\ d.HOD,\ r.code}\Big(\sigma_{s.cgpa>9.0}\left(\rho_{d}(department) \bowtie \rho_{s}(student) \bowtie \rho_{r}(registered)\right)\Big)$$

15) Retrieve the average CGPA of each department

Ans

$$\gamma_{dno,\ avg(CGPA)}(Student)$$

OR

$$dno\ (\gamma_{avg(CGPA)}(Student))$$

16) Retrieve the student register number who are not registered for any course

Ans

$$[\pi_{Regno}(Student)] - [\pi_{Regno}(Registered)]$$

Q4

(1) List the employee id who is working on more than 3 projects

Ans

$$\gamma_{empid,\ count(Pno)>=3}(Works\_on)$$

OR.

$$empid\ \mathcal{G}_{count(Pno)>=3}(works\_on)$$

(2) Retrieve the number of projects in each department

Ans

$$\gamma_{dno,\ count(Pno)}(Project)$$

OR

$$dno\ \mathcal{G}_{count(Pno)}(Project)$$

(3) Retrieve the employee id, name, salary, d_name, manager_id and project name working on.

Ans

$$\Pi_{e.id,\ e.name,\ e.salary,\ d.dname,\ d.mgr\_id,\ p.pname}\Big(\rho_e(employee) \underset{e.id=w.empid}{\bowtie} \rho_w(works\_on) \underset{w.pno=p.pno}{\bowtie} \rho_p(Project) \bowtie_{pdno=d.dno} (department)\Big).$$

4 Retrieve the employee details who are working on Department 'Marketing' and getting Salary more than 50000.

Ans

$$\Pi_{*}\left(\sigma_{d.dname='Marketing' \wedge s.Salary>=50000}\left(\rho_{d}(department) \bowtie \rho_{s}(salary)\right)\right)$$

5 Retrieve the department numbers which does not have any project.

Ans

$$\left[\Pi_{dno}(Department)\right] - \left[\Pi_{dno}(Project)\right]$$

(6) Retrive all the department numbers, department name and controlling project numbers, project name if any.

Ans

$$\Pi_{d.dno,\ d.dname,\ P.Pno,\ P.Pname}\left(\rho_{p}(Project) \bowtie \rho_{d}(Department)\right)$$