**Lab 4: Dipesh Singh – 190905520**

**Question 1:**  **Find the number of students in each course.**

select course\_id,

    title,

    count(\*)

from takes

    natural join course

group by (course\_id, title);

**Question 2: Find those departments where the average number of students are greater than 10.**

with cnt\_stu(dept\_name, cnt) as (

    select dept\_name,

        count(\*)

    from student

    group by dept\_name

)

select dept\_name,

    cnt

from cnt\_stu

where cnt > 100;

**Question 3: Find the total number of courses in each department.**

select dept\_name,

    count(\*)

from course

group by (dept\_name);

**Question 4: Find the names and average salaries of all departments whose average salary is greater than 42000.**

with avg\_sal(dept\_name, sal\_avg) as (

    select dept\_name,

        avg(salary)

    from instructor

    group by dept\_name

)

select dept\_name,

    sal\_avg

from avg\_sal

where sal\_avg > 42000;

**Question 5: Find the enrolment of each section that was offered in Spring 2009.**

select sec\_id,

    semester,

    year,

    count(distinct ID)

from takes

group by (sec\_id, semester, year)

having semester = 'Spring'

    and year = 2009;

**Question 6: List all the courses with prerequisite courses, then display course id in increasing order.**

select \*

from prereq

order by course\_id asc;

**Question 7: Display the details of instructors sorting the salary in decreasing order.**

select \*

from instructor

order by salary desc;

**Question 8: Find the maximum total salary across the departments.**

select max(sal)

from (

        select dept\_name,

            sum(salary) as sal

        from instructor

        group by dept\_name

    );

**Question 9: Find the average instructors’ salaries of those departments where the average salary is greater than 42000.**

select dept\_name,

    average

from (

        select dept\_name,

            avg(salary) as average

        from instructor

        group by dept\_name

    )

where average > 42000;

**Question 10: Find the sections that had the maximum enrolment in Spring 2010.**

select sec\_id,

    total

from (

        select sec\_id,

            count(distinct ID) as total

        from takes

        where semester = 'Fall'

            and year = 2009

        group by (sec\_id)

    )

where total >= all(

        select count(distinct ID) as total

        from takes

        where semester = 'Spring'

            and year = 2010

        group by (sec\_id)

    );

**Question 11: Find the names of all instructors who teach all students that belong to ‘CSE’ department.**

select distinct name

from instructor

    natural join teaches

where course\_id in (

        select distinct course\_id

        from takes

            natural join course

        where dept\_name = 'Comp. Sci.'

    );

**Question 12: Find the average salary of those department where the average salary is greater than 50000 and total number of instructors in the department are more than 5.**

select dept\_name,

    total,

    avg\_sal

from(

        select dept\_name,

            count(\*) as total,

            avg(salary) as avg\_sal

        from instructor

        group by dept\_name

    )

where total > 5

    and avg\_sal > 50000;

**Question 13: Find all departments with the maximum budget.**

with max\_bug(val) as (

    select max(budget)

    from department

)

select dept\_name,

    budget

from department,

    max\_bug

where budget = val;

**Question 14: Find all departments where the total salary is greater than the average of the total salary at all departments.**

with tot(dept\_name, total) as (

    select dept\_name,

        sum(salary) as tot

    from instructor

    group by dept\_name

),

avge(val) as (

    select avg(total)

    from tot

)

select dept\_name,

    total

from tot,

    avge

where total > val;

**Question 15: Find the sections that had the maximum enrolment in Fall 2009**

with totl(sec\_id, cnt) as (

    select sec\_id,

        count(distinct ID)

    from takes

    where semester = 'Fall'

        and year = 2009

    group by sec\_id

),

mx(val) as (

    select max(cnt)

    from totl

)

select sec\_id,

    cnt

from totl,

    mx

where cnt = val;

**Question 16: Select the names of those departments where the total credits earned by all the students is greater than the total credits earned by all the students in the Finance Department**

with tot\_credits(dept\_name, credits) as (

    select dept\_name,

        sum(tot\_cred)

    from student

    group by dept\_name

),

fin\_tot(val) as (

    select sum(tot\_cred)

    from student

    where dept\_name = 'Finance'

)

select dept\_name,

    credits

from tot\_credits,

    fin\_tot

where credits > val;

**Question 17: Delete all the instructors of Finance department.**

savepoint q17;

delete from instructor

where dept\_name = 'Finance';

**Question 18: Delete all courses in CSE department.**

delete from takes

where course\_id in (

        select course\_id

        from takes

            natural join course

        where dept\_name = 'Comp. Sci.'

    );

delete from teaches

where course\_id in (

        select course\_id

        from teaches

            natural join course

        where dept\_name = 'Comp. Sci.'

    );

delete from prereq

where course\_id in (

        select course\_id

        from prereq

            natural join course

        where dept\_name = 'Comp. Sci.'

    )

    or prereq\_id in (

        select c.course\_id

        from prereq p,

            course c

        where dept\_name = 'Comp. Sci.'

            and p.prereq\_id = c.course\_id

    );

delete from course

where dept\_name = 'Comp. Sci.';

**Question 19: Transfer all the students from CSE department to IT department.**

update student

set dept\_name = 'Physics'

where dept\_name = 'Comp. Sci.';

**Question 20: Increase salaries of instructors whose salary is over $100,000 by 3%, and all others receive a 5% raise**

update instructor

set salary = case

        when salary > 100000 then salary \* 1.03

        else salary \* 1.05

    end;

    rollback;