LAB 4 :

Implement the following Queries on UNIVERSITY Database.

Group By:

1. Find the number of students in each course.

```
SELECT COUNT(ID),course_id FROM takes GROUP BY
course_id;
```

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

```
SELECT COUNT(ID),course_id FROM takes GROUP BY course_id
HAVING COUNT(ID) ⩾ 2;
```

3. Find the total number of courses in each department.

```
SELECT COUNT(ID), dept_name FROM instructor GROUP BY
dept_name;
```

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

```
SELECT dept_name, AVG(salary) FROM instructor GROUP BY
dept_name HAVING AVG(salary) > 42000;
```

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

```
select sec_id,count( distinct id) from takes where
semester='Spring' and year=2009 group by sec_id
```

Ordering the display of Tuples (Use ORDER BY ASC/DESC):

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

```
SELECT course_id, prereq_id FROM prereq ORDER BY
course_id ASC;
```

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

```
select * from instructor order by salary DESC;
```

Derived Relations:

8. Find the maximum total salary across the departments.

```
select max(total_salary) from (Select sum(salary) as
total_salary from instructor group by dept_name);
```

9. Find the average instructors' salaries of those departments where the average salary is greater than 42000.

```
select dept_name,avg(salary) from (Select dept_name,
avg(salary) as avg_salary from instructor group by
dept_name) where avg_salary>42000;
```

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

```
select max(total_student) from (select count(distinct
ID) as total_student from takes group by sec_id,
semester, year having semester='Spring' and year=2010);
```

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

```
select distinct t.Name from Instructor t, teaches s,
takes m, student n where t.id=s.id and
s.course_id=m.course_id and m.id = n.id and n.dept_name
= 'Comp. Sci';
```

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, avg(salary) From instructor group by
dept_name having avg(salary)>50000 and count(id)⩾2;
```

With Clause:

13. Find all departments with the maximum budget.

```
with max_budget(value) as (Select max(budget) from
department) Select budget from department, max_budget
where department.budget=max_budget.value;
```

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

```
with dept_total(dept_name,salary) as (select
dept_name,sum(salary) from instructor group by
dept_name),dept_total_average(salary) as (select
avg(salary) from dept_total) select dept_name from
dept_total, dept_total_average ⩾ dept_total_avg.salary;
```

15. Find the sections that had the maximum enrolment in Fall 2009

```
with max_enroll(value) as(Select max(count(id)) From
takes group by sec_id, year, semester having year=2009
and semester ='Fall') enrollment(sec_id, value) as
(select sec_id, count(id) from takes group by sec_id,
year, semester having year=2009 and semester='Fall')
Select sec_id from enrollment.value;
```

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.

Use ROLLBACK (and SAVEPOINT) to undo the effect of any modification on database before COMMIT :

17. Delete all the instructors of Finance department.

```
delete from instructor where dept_name = 'Finance';
```

18. Delete all courses in CSE department.

```
delete from course where dept_name="Comp.Sci"
```

19. Transfer all the students from CSE department to IT department.

```
update student
set dept_name ='IT';
where dept_name = 'Comp. Sci';
```

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

```
update instructor set salary=salary*1.03 where
salary>100000;

update instructor set salary=salary*1.03 where
salary⩽100000;
```

21. Add all instructors to the student relation with tot_creds set to 0.

```
Insert into student
select ID,name,dept_name,0 from instructor;
```

22. Delete all instructors whose salary is less than the average salary of instructors.

```
delete from instructor where salary < (Select
avg(salary) from instructor);

SQL> commit;

Commit complete.
```