Ayush Goyal

190905522 CSE D 62

DBS Lab-4 (Week 4) – Complex Queries on SQL

Group By:

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

   select title, count(distinct takes.ID) no_of_students

   from student,takes,course

   where student.ID=takes.ID and takes.course_id=course.course_id and student.dept_name=course.dept_name group by title;

```
SQL> select title, count(distinct takes.ID) no_of_students
  2  from student,takes,course
  3  where student.ID=takes.ID and takes.course_id=course.course_id and student.dept_name=course.dept_name group by titl
e;

TITLE                                              NO_OF_STUDENTS
-------------------------------------------------- --------------
Intro. to Biology                                               1
Database System Concepts                                        2
Physical Principles                                             1
Music Video Production                                          1
Genetics                                                        1
Investment Banking                                              1
World History                                                   1
Game Design                                                     2
Image Processing                                                1
Intro. to Digital Systems                                       1
Intro. to Computer Science                                      4

TITLE                                              NO_OF_STUDENTS
-------------------------------------------------- --------------
Robotics                                                        1

12 rows selected.

SQL>
```

2. **Find those departments where the number of students are greater than 10. (Taken as 1 instead of 10 for this question).**

   select dept_name,count(ID) as total_students from student group by dept_name having count(ID) > 1;

```
SQL> select dept_name,count(ID) as total_students from student group by dept_name having count(ID) > 1;

DEPT_NAME            TOTAL_STUDENTS
-------------------- --------------
Elec. Eng.                        2
Physics                           3
Comp. Sci.                        4

SQL>
```

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

select dept_name,count(course_id) from course group by dept_name;

```
SQL> select dept_name,count(course_id) from course group by dept_name;

DEPT_NAME            COUNT(COURSE_ID)
-------------------- ----------------
Elec. Eng.                          1
Physics                             1
Comp. Sci.                          5
Finance                             1
Biology                             3
History                             1
Music                               1

7 rows selected.

SQL>
```

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;

```
SQL> select dept_name,avg(salary) from instructor group by dept_name having avg(salary)>42000;

DEPT_NAME            AVG(SALARY)
-------------------- -----------
Elec. Eng.                 80000
Physics                    91000
Comp. Sci.           77333.3333
Finance                    85000
Biology                    72000
History                    61000

6 rows selected.

SQL>
```

5. **Find the enrolment of each section that was offered in Spring 2009.Ordering the display of Tuples**

select sec_id,count(ID) from takes where semester='Spring' and year=2009 group by sec_id;

```
SQL> select sec_id,count(ID) from takes where semester='Spring' and year=2009 group by sec_id;

SEC_ID    COUNT(ID)
-------- ----------
1                 1
2                 2

SQL>
```

(Use ORDER BY ASC/DESC):

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

   select * from prereq order by course_id asc;

```
SQL> select *
  2  from prereq
  3  order by course_id asc;

COURSE_I PREREQ_I
-------- --------
BIO-301  BIO-101
BIO-399  BIO-101
CS-190   CS-101
CS-315   CS-101
CS-319   CS-101
CS-347   CS-101
EE-181   PHY-101

7 rows selected.

SQL>
```

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

   select * from instructor order by salary desc;

```
SQL> select * from instructor order by salary desc;

ID    NAME                 DEPT_NAME               SALARY
----- -------------------- -------------------- ----------
22222 Einstein             Physics                  95000
83821 Brandt               Comp. Sci.               92000
12121 Wu                   Finance                  90000
33456 Gold                 Physics                  87000
98345 Kim                  Elec. Eng.               80000
76543 Singh                Finance                  80000
45565 Katz                 Comp. Sci.               75000
76766 Crick                Biology                  72000
10101 Srinivasan           Comp. Sci.               65000
58583 Califieri            History                  62000
32343 El Said              History                  60000

ID    NAME                 DEPT_NAME               SALARY
----- -------------------- -------------------- ----------
15151 Mozart               Music                    40000

12 rows selected.

SQL>
```

Derived Relations: (means using sub-queries in the from clause)

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

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

```
SQL> select max(total_salary) from (select dept_name, sum(salary) as total_salary from instructor group by dept_name);

MAX(TOTAL_SALARY)
-----------------
           232000

SQL>
```

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;

```
SQL> select dept_name, avg_salary from(select dept_name,avg(salary) as avg_salary from instructor group by dept_name) wh
ere avg_salary>42000;

DEPT_NAME            AVG_SALARY
-------------------- ----------
Elec. Eng.                80000
Physics                   91000
Comp. Sci.            77333.3333
Finance                   85000
Biology                   72000
History                   61000

6 rows selected.

SQL>
```

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

    select sec_id, enroll from(select sec_id,count(ID) as enroll from takes where semester='Spring' and year=2010 group by sec_id) where enroll >=all(select count(ID) as enroll from takes where semester='Spring' and year=2010 group by sec_id);

```
SQL> select sec_id, enroll from(select sec_id,count(ID) as enroll from takes where semester='Spring' and year=2010 group
 by sec_id) where enroll >=all(select count(ID) as enroll from takes where semester='Spring' and year=2010 group by sec_
id);

SEC_ID       ENROLL
-------- ----------
1                 7

SQL>
```

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 student natural join takes where dept_name='Comp. Sci.');

```
SQL> select distinct name from instructor natural join teaches where course_id in (select distinct course_id from studen
t natural join takes where dept_name='Comp. Sci.');

NAME
--------------------
Srinivasan
Brandt
Katz

SQL>
```

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,average_salary from(select dept_name,count(*) as total,avg(salary) as average_salary from instructor group by dept_name having count(ID)>2 and avg(salary)>5000);

```
SQL> select dept_name,total,average_salary from(select dept_name,count(*) as total,avg(salary) as average_salary from in
structor group by dept_name having count(ID)>2 and avg(salary)>5000);

DEPT_NAME                TOTAL AVERAGE_SALARY
-------------------- ---------- --------------
Comp. Sci.                    3     77333.3333

SQL>
```

With Clause:

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

with budg(val) as (select max(budget) from department)
select dept_name,budget from budg,department where budg.val=department.budget;

```
SQL> with budg(val) as (select max(budget) from department)
  2  select dept_name,budget from budg,department where budg.val=department.budget;

DEPT_NAME                BUDGET
-------------------- ----------
Finance                  120000

SQL>
```

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;

```
SQL> 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;

DEPT_NAME                 TOTAL
-------------------- ----------
Physics                  182000
Comp. Sci.               232000
Finance                  170000

SQL>
```

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;

```
SQL> with totl(sec_id, cnt) as (select sec_id, count(distinct ID) from takes where semester='Fall' and year=2009 group b
y sec_id), mx(val) as (select max(cnt) from totl) select sec_id, cnt from totl, mx where cnt=val;

SEC_ID          CNT
-------- ----------
1                 7

SQL>
```

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 t1(dept_name,total_cred) as
(select dept_name,sum(tot_cred) from student group by dept_name),
t2(value) as (select total_cred from t1
where dept_name='Finance')
select dept_name from t1,t2
where total_cred>value;

```
SQL> with t1(dept_name,total_cred) as    (select dept_name,sum(tot_cred) from student group by dept_name),    t2(value)
as (select total_cred from t1    where dept_name='Finance')    select dept_name from t1,t2    where total_cred>value;

DEPT_NAME
--------------------
Comp. Sci.
Elec. Eng.
Biology

SQL>
```

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

17. Delete all the instructors of Finance department.

savepoint Q17;
delete from instructor where dept_name = 'Finance';

```
SQL> savepoint Q17;

Savepoint created.

SQL> delete from instructor where dept_name='Finance';

2 rows deleted.

SQL> rollback to Q17;

Rollback complete.

SQL>
```

18. Delete all courses in CSE department.

savepoint Q18;
delete from course where dept_name='Comp. Sci.';

```
SQL> savepoint Q18;

Savepoint created.

SQL> delete from course where dept_name='Comp. Sci.';

5 rows deleted.

SQL>
```

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

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

```
SQL> rollback to Q18;

Rollback complete.

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

0 rows updated.

SQL>
```

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;
```

```
SQL> savepoint Q20;

Savepoint created.

SQL> update instructor
  2  set salary=case
  3  when salary>100000
  4  then salary*1.03
  5  else
  6  salary*1.05
  7       end;

12 rows updated.

SQL> rollback Q20;
rollback Q20
         *
ERROR at line 1:
ORA-02181: invalid option to ROLLBACK WORK


SQL> rollback;

Rollback complete.

SQL>
```

THE END