Database Management Systems (CSN-351)
SQL (Contd. 3)

BTech 3rd Year (CS) + Minor + Audit

Instructor: **R**anita **B**iswas
Department of Computer Science and Engineering
Indian Institute of Technology Roorkee
Roorkee, Uttarakhand - 247 667, India

Test for the Absence of Duplicate Tuples

course(<u>*course_id*</u>, *title*, *dept_name*, *credits*)

section(<u>*course_id*</u>, <u>*sec_id*</u>, <u>*semester*</u>, <u>*year*</u>, *building*, *room_number*, *time_slot_id*)

- Find all courses that were offered exactly once in 2009.

Test for the Absence of Duplicate Tuples

course(<u>*course_id*</u>, *title*, *dept_name*, *credits*)

section(<u>*course_id*</u>, <u>*sec_id*</u>, <u>*semester*</u>, <u>*year*</u>, *building*, *room_number*, *time_slot_id*)

- Find all courses that were offered exactly once in 2009.

```
select T.course_id
from course as T
where unique (select R.course_id
              from section as R
              where T.course_id= R.course_id and
                    R.year = 2009);
```

Test for the Presence of Duplicate Tuples

course(<u>*course_id*</u>, *title*, *dept_name*, *credits*)

section(<u>*course_id*</u>, <u>*sec_id*</u>, <u>*semester*</u>, <u>*year*</u>, *building*, *room_number*, *time_slot_id*)

- Find all courses that were offered at least twice in 2009.

Test for the Presence of Duplicate Tuples

course(<u>*course_id*</u>, *title*, *dept_name*, *credits*)

section(<u>*course_id*</u>, <u>*sec_id*</u>, <u>*semester*</u>, <u>*year*</u>, *building*, *room_number*, *time_slot_id*)

- Find all courses that were offered at least twice in 2009.

```
select T.course_id
from course as T
where not unique (select R.course_id
                  from section as R
                  where T.course_id= R.course_id and
                        R.year = 2009);
```

Subqueries in the From Clause

instructor(<u>ID</u>, *name*, *dept_name*, *salary*)

- Find the average instructors' salaries of those departments where the average salary is greater than 42,000.

Subqueries in the From Clause

instructor(<u>*ID*</u>, *name*, *dept_name*, *salary*)

- Find the average instructors' salaries of those departments where the average salary is greater than 42,000.

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

Subqueries in the From Clause

instructor(*ID*, *name*, *dept_name*, *salary*)

- Find the average instructors' salaries of those departments where the average salary is greater than 42,000.

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

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

Subqueries in the From Clause (contd.)

instructor(<u>ID</u>, *name*, *dept_name*, *salary*)

- Find the maximum across all departments of the total salary at each department.

Subqueries in the From Clause (contd.)

instructor(<u>*ID*</u>, *name*, *dept_name*, *salary*)

- Find the maximum across all departments of the total salary at each department.

```
select max (tot_salary)
from (select dept_name, sum(salary)
        from instructor
        group by dept_name) as dept_total (dept_name, tot_salary);
```

The **lateral** Clause

instructor(<u>*ID*</u>, *name*, *dept_name*, *salary*)

- Find the names of each instructor, along with their salary and the average salary in their department.

The **lateral** Clause

instructor(<u>*ID*</u>, *name*, *dept_name*, *salary*)

- Find the names of each instructor, along with their salary and the average salary in their department.

```
select name, salary, avg_salary
from instructor I1, lateral (select avg(salary) as avg_salary
                             from instructor I2
                             where I2.dept_name= I1.dept_name);
```

The **with** Clause

department(*dept_name*, *building*, *budget*)

- Find the departments with the maximum budget.

The **with** Clause

department(*dept_name*, *building*, *budget*)

- Find the 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;
```

The **with** Clause (contd.)

instructor(<u>ID</u>, *name*, *dept_name*, *salary*)

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

The **with** Clause (contd.)

instructor(*ID*, *name*, *dept_name*, *salary*)

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

```
with dept_total (dept_name, value) as
        (select dept_name, sum(salary)
        from instructor
        group by dept_name),
dept_total_avg(value) as
        (select avg(value)
        from dept_total)
select dept_name
from dept_total, dept_total_avg
where dept_total.value >= dept_total_avg.value;
```

Scalar Subqueries

instructor(<u>ID</u>, *name*, *dept_name*, *salary*)

department(<u>*dept_name*</u>, *building*, *budget*)

- List all departments along with the number of instructors in each department.

Scalar Subqueries

instructor(<u>ID</u>, *name*, *dept_name*, *salary*)

department(<u>*dept_name*</u>, *building*, *budget*)

- List all departments along with the number of instructors in each department.

```
select dept_name,
        (select count(*)
         from instructor
         where department.dept_name = instructor.dept_name)
        as num_instructors
from department;
```

Deletion

delete from r
where P;

Deletion

delete from r
where P;

- Delete all tuples from the *instructor* relation.

Deletion

delete from r
where P;

- Delete all tuples from the *instructor* relation.

delete from *instructor*;

Deletion

delete from r
where P;

- Delete all tuples from the *instructor* relation.

delete from *instructor*;

- Delete all tuples in the *instructor* relation pertaining to instructors in the Finance department.

Deletion

delete from r
where P;

- Delete all tuples from the *instructor* relation.

delete from *instructor*;

- Delete all tuples in the *instructor* relation pertaining to instructors in the Finance department.

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

Deletion (contd.)

- Delete all instructors with a salary between 13,000 and 15,000.

Deletion (contd.)

- Delete all instructors with a salary between 13,000 and 15,000.

delete from *instructor*
where *salary* **between** 13000 **and** 15000;

Deletion (contd.)

- Delete all instructors with a salary between 13,000 and 15,000.

```
delete from instructor
where salary between 13000 and 15000;
```

- Delete all tuples in the *instructor* relation for those instructors associated with a department located in the Watson building.

Deletion (contd.)

- Delete all instructors with a salary between 13,000 and 15,000.

```
delete from instructor
where salary between 13000 and 15000;
```

- Delete all tuples in the *instructor* relation for those instructors associated with a department located in the Watson building.

```
delete from instructor
where dept_name in (select dept_name
                    from department
                    where building = 'Watson');
```

Deletion (contd.)

- Delete the records of all instructors with salary below the average at the university.

Deletion (contd.)

- Delete the records of all instructors with salary below the average at the university.

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

Insertion

insert into *course*
values ('CS-437', 'Database Systems', 'Comp. Sci.', 4);

Insertion

insert into *course*
values ('CS-437', 'Database Systems', 'Comp. Sci.', 4);

insert into *course* (*course_id*, *title*, *dept_name*, *credits*)
values ('CS-437', 'Database Systems', 'Comp. Sci.', 4);

insert into *course* (*title*, *course_id*, *credits*, *dept_name*)
values ('Database Systems', 'CS-437', 4, 'Comp. Sci.');

Insertion (contd.)

- Make each student in the Music department who has earned more than 144 credit hours, an instructor in the Music department, with a salary of 18,000.

Insertion (contd.)

- Make each student in the Music department who has earned more than 144 credit hours, an instructor in the Music department, with a salary of 18,000.

```
insert into instructor
        select ID, name, dept_name, 18000
        from student
        where dept_name = 'Music' and tot_cred > 144;
```

Insertion (contd.)

```
insert into student
        select *
        from student;
```

Insertion (contd.)

```
insert into student
        select *
        from student;

insert into student
        values ('3003', 'Green', 'Finance', null);
```

Updates

- Increase salaries of all instructors by 5 percent.

Updates

- Increase salaries of all instructors by 5 percent.

update *instructor*
set *salary*= *salary* * 1.05;

Updates

- Increase salaries of all instructors by 5 percent.

```
update instructor
set salary= salary * 1.05;
```

- Increase salaries of all instructors whose salary is less than 70,000 by 5 percent.

Updates

- Increase salaries of all instructors by 5 percent.

```
update instructor
set salary= salary * 1.05;
```

- Increase salaries of all instructors whose salary is less than 70,000 by 5 percent.

```
update instructor
set salary = salary * 1.05
where salary < 70000;
```

Updates (contd.)

- Give a 5 percent salary raise to instructors whose salary is less than average.

Updates (contd.)

- Give a 5 percent salary raise to instructors whose salary is less than average.

```
update instructor
set salary = salary * 1.05
where salary < (select avg (salary)
                from instructor);
```

Updates (contd.)

- Give a 5 percent salary raise to instructors whose salary is less than average.

```
update instructor
set salary = salary * 1.05
where salary < (select avg (salary)
                from instructor);
```

- All instructors with salary over 100,000 receive a 3 percent raise, whereas all others receive a 5 percent raise.

Updates (contd.)

- Give a 5 percent salary raise to instructors whose salary is less than average.

```
update instructor
set salary = salary * 1.05
where salary < (select avg (salary)
                from instructor);
```

- All instructors with salary over 100,000 receive a 3 percent raise, whereas all others receive a 5 percent raise.

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

```
update instructor
set salary = salary * 1.05
where salary <= 100000;
```

Updates (contd.)

```
update instructor
set salary = case
                when salary <= 100000 then salary * 1.05
                else salary * 1.03
             end
```

Updates (contd.)

```
update instructor
set salary = case
                when salary <= 100000 then salary * 1.05
                else salary * 1.03
             end
```

```
case
    when pred_1 then result_1
    when pred_2 then result_2
    ...
    when pred_n then result_n
    else result_0
end
```

Updates (contd.)

- Set the *tot_cred* attribute of each *student* tuple to the sum of the credits of courses successfully completed by the student.

Updates (contd.)

- Set the *tot_cred* attribute of each *student* tuple to the sum of the credits of courses successfully completed by the student.

```
update student S
set tot_cred = (
    select sum(credits)
    from takes natural join course
    where S.ID= takes.ID and
                takes.grade <> 'F' and
                takes.grade is not null);
```

Updates (contd.)

- Set the *tot_cred* attribute of each *student* tuple to the sum of the credits of courses successfully completed by the student.

```
update student S
set tot_cred = (
    select sum(credits)
    from takes natural join course
    where S.ID= takes.ID and
                takes.grade <> 'F' and
                takes.grade is not null);
```

```
select case
        when sum(credits) is not null then sum(credits)
        else 0
        end
```