SQL – 2

Joins

Joins support the ability to select data from more than one table in a query.

Basic Joins

- Trivial method to achieve a join is to list each table in the from clause and provide a join condition in the where clause.

  Student DB

  - Show the student names and class id in which they are enrolled.

    ```
    SELECT name, cid
    FROM student, enrolled
    WHERE student.id = enrolled.sid;
    ```

  - List the department, class number (such as CSE 1341) and student id of any class in which there is a student enrolled.

    ```
          SELECT dept, num, sid
          FROM classes, enrolled
          WHERE classes.id = enrolled.cid;
    ```

  Employees Database

  - List the employee number, first name, last name, and the date the title of manager was assigned to any employee.

    ```
          SELECT employees.emp_no, employees.first_name,
          employees.last_name,      titles.title
          FROM employees NATURAL JOIN titles
          WHERE titles.title="MANAGER"
    ```

    OR

    ```
          SELECT employees.emp_no, employees.first_name,
          employees.last_name,      titles.title
          FROM employees, titles
          WHERE titles.title="MANAGER"
          AND employees.emp_no = titles.emp_no;
    ```

  - List the employee name and department name for each manager in the dept_manager table.

    ```
    SELECT dept_name, first_name, last_name
    FROM departments d, dept_manager dm, employees e
    WHERE d.dept_no = dm.dept_no
    ```

```
AND dm.emp_no = e.emp_no;
```

OR

```
SELECT dept_name, first_name, last_name
FROM departments NATURAL JOIN dept_manager NATURAL JOIN
employees;
```

- What is the name of the latest/last manager for each department?

```
select first_name, last_name
from dept_manager NATURAL JOIN employees
where to_date like "9999%";
```

Outer Joins

Left Outer Join

- Retains rows from the left table that do not satisfy the join condition.
- In MySQL, LEFT JOIN and LEFT OUTER JOIN are identical commands

Student Database Example

- List each class and the students enrolled in it. If a class has no students, return null. Expected Output:

```
+------+------+------+---------+------+------+
| id   | dept | num  | prof    | sid  | cid  |
+------+------+------+---------+------+------+
|    1 | HIST | 1311 | Jackson | NULL | NULL |
|    2 | CSE  | 1341 | Hahsler | 123  |    2 |
|    2 | CSE  | 1341 | Hahsler | 312  |    2 |
|    3 | PLSC | 1303 | Jackson | 214  |    3 |
|    4 | ART  | 1313 | Scott   | NULL | NULL |
+------+------+------+---------+------+------+
```

```
SELECT *
FROM classes c LEFT OUTER JOIN enrolled e ON c.id = e.cid;
```

Right Outer Join

- Retains rows from the right table that do not satisfy the join condition.
- Use join command RIGHT JOIN or RIGHT OUTER JOIN.

Full Outer Join

- Retains rows from both tables in the query that do not match the join condition
- Not available in MySQL. Use a UNION with LEFT JOIN and RIGHT JOIN to achieve the same effect.

Subqueries

- One SELECT embedded in another SELECT statement
- 3 Types:
    - Scalar subquery – returns a single value

- Row subquery – Multiple columns, but only a single row
- Table subquery – returns one or more columns with multiple rows.

Employees DB example:

Equality subquery:
List all of the titles that Margareta Markovitch ever had assigned to her. Use a subquery.

```
select * from titles
where emp_no = ( select emp_no
                 from employees
                 where last_name = 'Markovitch'
                     and first_name = 'Margareta')
```

Exists and Not Exists

- If any values are returned for EXIST subquery, the result is true. If any values are returned for NOT EXISTS subquery, the result is false.

Give the first name and last name of employees that are managers.

```
select *
from employees e
where exists (select *
              from titles t
              where e.emp_no = t.emp_no
              and t.title = 'manager')
```

Database Updates

- Modifying the db can be done with the following 3 commands:
  - Insert – add new data (covered in 1st handout)
  - Update – modify existing data
  - Delete – remove data

Delete

```
DELETE FROM tableName
[WHERE searchCondition];
```

Note: Omitting the where clause will delete all of the data in the table.

Delete class with ID 5.

```
DELETE FROM classes.id WHERE id = 5;
```

Update

```
UPDATE tableName
SET col1 = val1 [, col2 = val2, …]
[WHERE searchCondition];
```

Change the prof for CSE 1341 to Smith.

```
UPDATE classes
SET prof = 'Smith'
WHERE dept='CSE' and num='3333';
```

Give all students a 10% GPA boost!

```
UPDATE student
SET gpa = 1.1 * gpa;
```