SQL Queries, Deletes, and Updates

Tom Kelliher, CS 318

Feb. 13, 15, 2002

1 Administrivia

Announcements

We've seen insert already — no need to review.

Just touching on the basics, refer to the text and PostgreSQL docs for more advanced material.

Collect homework.

Assignment

Read 10.1–4.

From Last Time

Relational algebra.

Outline

1. SQL queries.
2. SQL deletes.

3. SQL updates.

Coming Up

PostgreSQL and PHP.

2 SQL Queries

1. Example query:

```
SELECT S.Name
FROM Student S
WHERE S.Status = 'Senior';
```

2. Query evaluation strategy I:

   (a) Form Cartesian product of relations mentioned in `FROM` clause.

   (b) Apply `WHERE` clause individually to resulting tuples.

   (c) Apply `SELECT` clause.

3. Example using a natural join on two relations. Retrieve names of all students who have ever taken CS 318.

```
SELECT S.Name
FROM Student S, Transcript T
WHERE S.Id = T.StuId AND T.CrsCode = 'CS318';
```

   Use of tuple variables.

4. Example using a self-join. Retrieve ID and salary difference of all employees earning at least twice as much as their supervisors.

```
SELECT E1.Id, E1.Salary - E2.Salary
FROM Employee E1, Employee E2
WHERE E1.Supervisor = E2.Id AND E1.Sal >= 2 * E2.Sal;
```

5. Retrieve *distinct* names of all students who have taken any CS course.

```
SELECT DISTINCT S.Name
FROM Student S, Transcript T
WHERE S.Id = T.StuID AND T.CrsCode LIKE 'CS%';
```

Alternative for retrieving students who have taken topics courses:

```
WHERE S.Id = T.StuID AND T.CrsCode IN ('CS318', 'CS319');
```

6. Retrieve IDs of students who did not take any courses in F2001. Version 1 using set difference:

```
(SELECT S.Id
 FROM Student S)
EXCEPT
(SELECT T.StuId
 FROM Transcript T
 WHERE T.Semester = 'F2001');
```

Version 2 using `NOT IN`:

```
SELECT S.Id
FROM Student S
WHERE S.Id NOT IN (SELECT T.StuId
                   FROM Transcript T
                   WHERE T.Semester = 'F2001');
```

Sub-query.

Version 3 using `NOT EXISTS`:

```
SELECT S.Id
FROM Student S
WHERE NOT EXISTS (SELECT *
                  FROM Transcript T
                  WHERE S.Id = T.StuId AND T.Semester = 'F2001');
```

Correlated sub-query.

7. Find potential student graders for this semester's courses

```
SELECT P.Id, TR.CrsCode, TR.StuId
FROM Transcript TR, Professor P
WHERE TR.Semester <> 'S2002'
      AND TR.CrsCode IN (SELECT TE.CrsCode
                         FROM Teaching TE
                         WHERE P.Id = TE.ProfId
                               AND TE.Semester = 'S2002');
```

8. Retrieve name of student with largest ID number.

```
SELECT S1.Name
FROM Student S1
WHERE S1.Id >ALL (SELECT S2.Id
                  FROM Student S2);
```

Alternative using aggregate function `MAX`:

```
SELECT S1.Name
FROM Student S1
WHERE S1.Id > (SELECT MAX(S2.Id)
               FROM Student S2);
```

These will both return empty relations. Why?

9. How many students do we have?

```
SELECT COUNT(S.Id)
FROM Student S;
```

10. What's the average salary in Accounting?

```
SELECT AVG(E.Sal)
FROM Employee E
WHERE E.Dept = 'Accounting';
```

How would you retrieve the ID of the employee with the highest salary?

11. Categorizing tuples. Retrieve count and average calories of foods with less than 100 calories, categorized by food type, assuming there are at least 20 foods such foods in the food type. List by food type.

```
SELECT F.FoodType, Count(F.Name) as Count,
       AVG(F.Calories) AS AvgCalories
FROM Food F
WHERE F.Calories < 100
GROUP BY F.FoodType
HAVING COUNT(F.Name) >= 20;
```

Query evaluation strategy II:

(a) Form Cartesian product of relations mentioned in `FROM` clause.

(b) Apply `WHERE` clause individually to resulting tuples.

(c) Apply `GROUP BY` clause to partition tuples into groups.

(d) Apply `HAVING` clause to winnow out groups.

(e) Apply `SELECT` clause.

3 SQL Deletes

1. Delete the student whose ID is `123456789`.

```
DELETE FROM Student
WHERE Id = '123456789';
```

2. Delete all students who are in CS 318 this semester.

```
DELETE FROM Student
WHERE Id IN (SELECT T.StuId
             FROM Transcript T
             WHERE T.CrsCode = 'CS318' AND T.Semester = 'S2002');
```

4 SQL Updates

Give a 10% raise to all faculty who teach the most ornery student, whose ID is `123456789`.

```
UPDATE Professor
SET Salary = 1.1 * Salary
WHERE ID IN (SELECT TE.ProfID
             FROM Teaching TE, Transcript TR
             WHERE TE.CrsCode = TR.CrsCode AND TR.StuId = '123456789');
```