SQL

Subqueries

Orlando Karam
okaram@spsu.edu

SOUTHERN POLYTECHNIC STATE UNIVERSITY

Subqueries

- We can use a SELECT statement (subquery) instead of a table in another SELECT statement
  - –WITH can make it easier
  - –RECURSIVE WITH, 11g, gives more power
- We can use a subquery inside IN or EXISTS
- We can use a subquery with >ALL,=ANY etc
- Especially useful for negation queries
- In *Correlated* subqueries, the subquery refers to the outside row; these are harder to optimize

Example - IN

- We can use a SELECT statement instead of a list of values with IN (or NOT IN)
  - –IN is a binary operator, so types need to match
  - –Example: List names of students *minoring* in the area with code 'CS'
    - SELECT S.Name
    - FROM Student
    - WHERE Id IN (
      - –SELECT Student
      - –FROM IsMinoring
      - –WHERE Area='CS'
    -)

Example - Exists

- EXISTS is a unary predicate, that takes a list of values and says whether it is empty
  - –But we can use a subquery
  - –Normally leads to *correlated* subqueries, since we refer to the outside table
  - –Example:
    - SELECT Name
    - FROM Student S
    - WHERE EXISTS (
    - SELECT *
    - FROM IsMinoring M
    - WHERE S.Id=M.Student AND Area='CS'
    -);

You try

- Name of students who are minoring in ‘IT’
- Name of students who are NOT minoring in anything
-

Examples - >ALL etc

- For all the comparison operators (=,> etc) we can use ALL or ANY after them, and compare one value with a list
  - =ANY is equivalent to IN
- Example: Name of student(s) who are the oldest
  - SELECT Name
  - FROM Student
  - WHERE Age >=ALL (
    - SELECT Age
    - FROM Student
  -)

You try it

- Name of youngest student(s)
- Name of major with last code alphabetically

Example - WITH

- Can give a local name to the subquery with the WITH clause
- Example:
  - –WITH MinoringInCS AS (SELECT Student FROM IsMinoring WHERE Area='CS')
  - –SELECT Name
  - –FROM Student
  - –WHERE Id IN (SELECT * FROM MinoringInCS);

Subqueries vs JOIN

- JOINS allow you to return data from more than one table
  - –OUTER JOINs let you get rows that don't actually match
- When using subqueries, you only get data from only the table(s) mentioned in the main SELECT
- Negation is easier to understand with subqueries (right ? :)
- OUTER JOINs could be used (weirdly) instead of subqueries