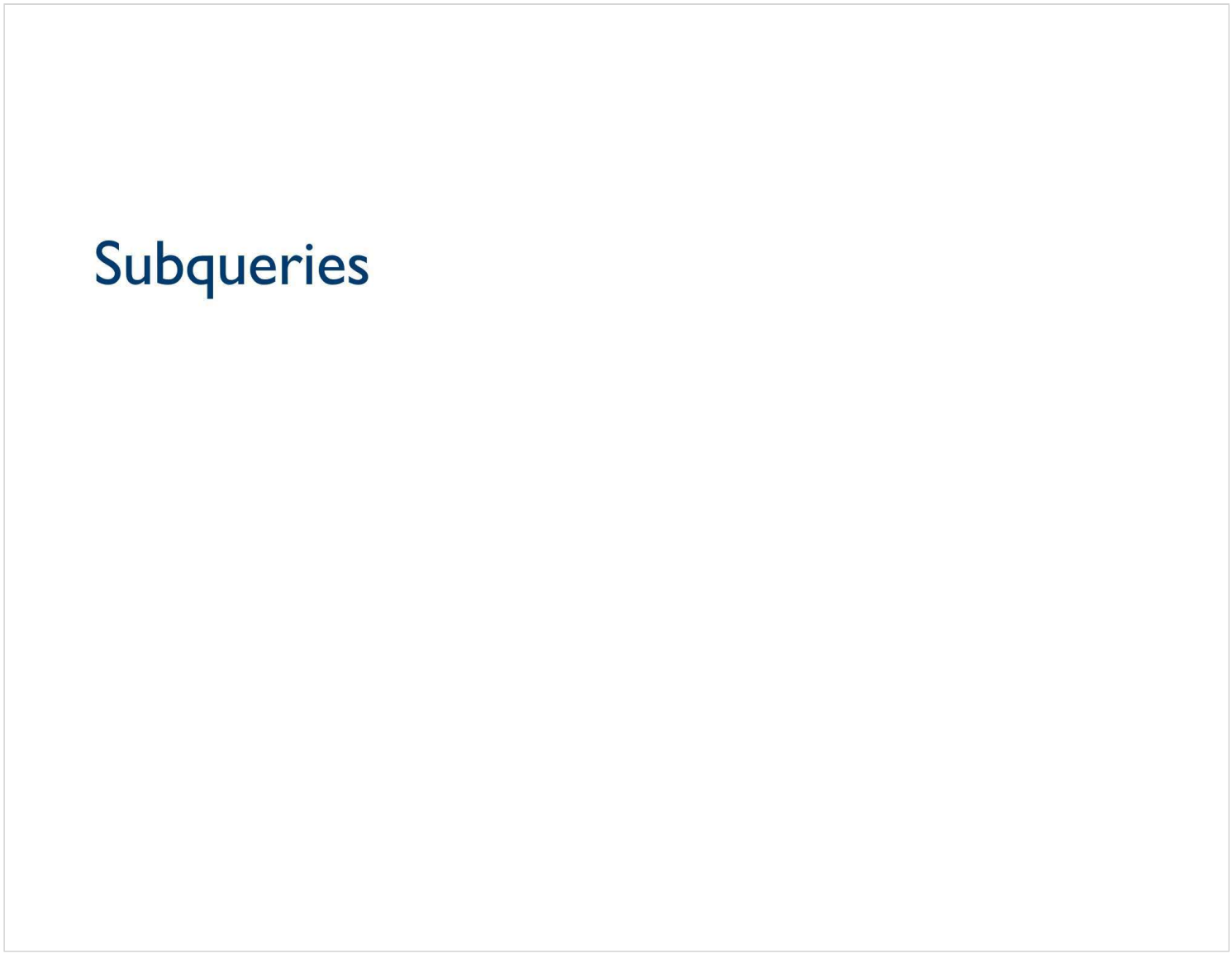
Week4 - Subqueries

January 29, 2025 5:17 PM



Where can a subquery go?

- Relational algebra syntax is so elegant that it's easy to see where subqueries can go.
- In SQL, a bit more thought is required . . .

Subqueries in a FROM clause

- In place of a relation name in the FROM clause, we can use a subquery.
- The subquery must be parenthesized.
- Must name the result, so you can refer to it in the outer query.

```
select <   >
from A, ( select . . . ) as B
```

csc 343 ⇒ csc343

Worksheet, Q1:

```
SELECT sid, dept||cnum as course, grade
FROM Took,
  (SELECT *
   FROM Offering
   WHERE instructor='Horton') Hoffering
WHERE Took.oid = Hoffering.oid;
```

oID dept ins
1 Horton
2 Horton

- This FROM is analogous to:

  Took × $\rho_{Hoffering}$ («*subquery*»)

- Can you suggest another version?

using join

Subquery as a value in a WHERE

- If a subquery is guaranteed to produce exactly one tuple, then the subquery can be used as a value.
- Simplest situation: that one tuple has only one component.

Select
From
Where (select q.) <, =, >

if subquery is scalar, then simple logical operations are valid.

Worksheet, Q2:

```
SELECT sid, surname
FROM Student
WHERE cgpa >
    (SELECT cgpa
     FROM Student
     WHERE sid = 99999);
```

→ 3.4

- We can't do the analogous thing in RA:

$$\pi_{\text{sid, surname}}\ \sigma_{\text{cgpa} > (\textit{«subquery»})}\text{Student}$$

Special cases

- What if the subquery returns `NULL`? later
- What if the subquery could return more than one value?

1. List of values

Quantifying over multiple results

- When a subquery can return multiple values, we can make comparisons using a quantifier.
- Example:

```
SELECT sid, surname
FROM Student
WHERE cgpa >
    (SELECT cgpa
     FROM Student
     WHERE campus = 'StG');
```

c = 3.5 T
c = 2.0 F
c = 3.2

{ 3.4 , 3.1 , 2.1 }

- We can require that
  - cgpa **> all** of them, or
  - cgpa **> at least one** of them.

The Operator ANY

holds true for any of the list of values

- Syntax:

  x «*comparison*» ANY («*subquery*»)

  or equivalently

  x «*comparison*» SOME («*subquery*»)

- Semantics:
  Its value is true iff the comparison holds for at least one tuple in the subquery result, i.e.,
  ∃ y ∈ «*subquery results*» | x «*comparison*» y

- x can be a *list* of attributes,
  but this feature is not supported by psql.

The Operator ALL

holds true only if true for all values in the list of values

- Syntax:

  x «comparison» ALL («subquery»)

- Semantics:
  Its value is true iff the comparison holds for every tuple in the subquery result, i.e.,
  ∀ y ∈ «subquery results» | x «comparison» y

- x can be a list of attributes,
  but this feature is not supported by psql.

- Example: any-all

The Operator IN

{1, 2, 3}

a in ( )

- Syntax:

  x IN (*«subquery»*)

- Semantics:

  Its value is true iff x is in the set of rows generated by the subquery.

- x can be a list of attributes, and psql does support this feature.

Worksheet, Q3:

1. grade >80 , Lakemeyer course
2. grade >80, Lakemeyer course in this course
3. grade >80, Lakemeyer course taken together

```
SELECT sid, dept||cnum AS course, grade
FROM Took NATURAL JOIN Offering
WHERE
   grade >= 80 AND
   (cnum, dept) IN (
       SELECT cnum, dept
       FROM Took NATURAL JOIN Offering
                 NATURAL JOIN Student
       WHERE surname = 'Lakemeyer');
```

CSE 343, CSE 148