Database Management Systems

Ehsan Noei
e.noei@utoronto.ca

UNIVERSITY OF TORONTO

Nested Queries

- One of the most powerful features of SQL is nested queries.
- A nested query is a query that has another query embedded within it; the embedded query is called a subquery.
- A WHERE clause can itself contain an SQL query.

Example

Sailors (sid: integer, sname: string, rating: integer, age: real)

Boats (bid: integer, bname: string, color: string)

Reserves (sid: integer, bid: integer, day: date)

R1

sid	bid	day
22	101	10/10/96
58	103	11/12/96

S1

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Find names of sailors who've reserved boat #103

```
SELECT  S.sname
FROM    Sailors S, Reserves R
WHERE   S.sid=R.sid AND R.bid=103
```

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	22	101	10/10/96
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	22	101	10/10/96
31	lubber	8	55.5	58	103	11/12/96
58	rusty	10	35.0	22	101	10/10/96
58	rusty	10	35.0	58	103	11/12/96

Find names of sailors who've reserved boat #103

```
SELECT S.sname
FROM Sailors S
WHERE S.sid IN (SELECT R.sid
                FROM Reserves R
                WHERE R.bid=103)
```

R1

sid	bid	day
22	101	10/10/96
58	103	11/12/96

S1

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

IN and NOT IN

```
SELECT S.sname
FROM Sailors S
WHERE S.sid NOT IN (SELECT R.sid
                    FROM Reserves R
                    WHERE R.bid=103)
```

R1

sid	bid	day
22	101	10/10/96
58	103	11/12/96

S1

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Find the name of sailors who've reserved a red boat

```
SELECT  S.sname
FROM  Sailors S, Boats B, Reserves R
WHERE  S.sid=R.sid AND R.bid=B.bid AND B.color = 'red'
```

Find the name of sailors who've reserved a red boat

```
SELECT S.sname
FROM Sailors S
WHERE S.sid IN (SELECT R.sid
                FROM Reserves R
                WHERE R. bid IN (SELECT B.bid
                                 FROM Boats B
                                 WHERE B.color = 'red'
```

Find the name of sailors who have not reserved a red boat

```
SELECT S.sname
FROM Sailors S
WHERE S.sid NOT IN (SELECT R.sid
                    FROM Reserves R
                    WHERE R. bid IN (SELECT B.bid
                                     FROM Boats B
                                     WHERE B.color = 'red'
```

Nested Queries with Correlation

- In the nested queries seen thus far, the inner subquery has been completely independent of the outer query.
- In general, the inner subquery could depend on the row currently being examined in the outer query (in terms of our conceptual evaluation strategy).

Find names of sailors who've reserved boat #103

```
SELECT S.sname
FROM Sailors S
WHERE EXISTS (SELECT *
              FROM Reserves R
              WHERE R.bid=103 AND S.sid=R.sid)
```

- EXISTS is another set comparison operator, like IN.
- Allows test whether a set is nonempty.

EXISTS vs NOT EXISTS

```
SELECT S.sname
FROM Sailors S
WHERE NOT EXISTS (SELECT *
                  FROM Reserves R
                  WHERE R.bid=103 AND S.sid=R.sid)
```

More on Set-Comparison Operators

- We've already seen IN, EXISTS.
- Also available: *op* ANY, *op* ALL $>, <, =, \geq, \leq, \neq$

Find sailors whose rating is greater than that of some sailor called lubber

```
SELECT  *
FROM  Sailors S
WHERE  S.rating > ANY  (SELECT  S2.rating
                        FROM  Sailors S2
                        WHERE S2.sname= 'lubber' )
```

Find sailors whose rating is greater than that of every sailor called lubber

```
SELECT  *
FROM  Sailors S
WHERE  S.rating > ALL  (SELECT  S2.rating
                        FROM  Sailors S2
                        WHERE S2.sname= 'lubber' )
```

Find the Sailor's with the highest rating.

```
SELECT S.sid
FROM Sailors S
WHERE S.rating >= ALL ( SELECT S2.rating
                        FROM Sailors S2 )
```

IN, NOT IN, =ANY, <>ALL

- IN is equivalent to =ANY
- NOT IN is equivalent to <>ALL

MySQL

- MySQL does not support INTERSECT and EXCEPT
- But supports UNION

- Alternatives?
  - USE IN, NOT IN, EXISTS, NOT EXISTS

Find sid's of sailors who've reserved a red and a green boat.

```
SELECT  S.sid
FROM  Sailors S, Boats B, Reserves R
WHERE  S.sid=R.sid AND
R.bid=B.bid
            AND B.color= 'red'
INTERSECT
SELECT  S.sid
FROM  Sailors S, Boats B, Reserves R
WHERE  S.sid=R.sid AND
R.bid=B.bid
            AND B.color= 'green'
```

Find names of sailors who've reserved a red and a green boat.

```
SELECT  S.sname
FROM  Sailors S, Boats B, Reserves R
WHERE  S.sid=R.sid AND R.bid=B.bid AND B.color= 'red'
        AND S.sid IN (SELECT  S2.sid
                      FROM  Sailors S2, Boats B2, Reserves R2
                      WHERE  S2.sid=R2.sid AND R2.bid=B2.bid
                             AND  B2.color= 'green' )
```

EXCEPT?

```
SELECT B.bid
FROM Boats B
EXCEPT
SELECT R. bid
FROM Reserves R
```

```
SELECT B.bid
FROM Boats B
WHERE B.bid NOT IN (SELECT R. bid
                    FROM Reserves R)
```

```
SELECT B.bid
FROM Boats B
WHERE NOT EXISTS (SELECT R. bid
                  FROM Reserves R
                  WHERE B.bid = R.bid)
```

Find the names of sailors who have reserved all boats.

```
SELECT S.sname
FROM Sailors S
WHERE NOT EXISTS (( SELECT B.bid
                    FROM Boats B )
                    EXCEPT
                    (SELECT R. bid
                    FROM Reserves R
                    WHERE R.sid = S.sid ))
```

Find the names of sailors who have reserved all boats.

```
SELECT S.sname
FROM Sailors S
WHERE NOT EXISTS ( SELECT B.bid
                   FROM Boats B
                   WHERE NOT EXISTS (
                         SELECT R. bid
                         FROM Reserves R
                         WHERE R.bid = B.bid
                         AND R.sid = S.sid ))
```

Aggregate Operators

COUNT (*)
COUNT ([DISTINCT] A)
SUM ([DISTINCT] A)
AVG ([DISTINCT] A)
MAX (A)
MIN (A)

Find the average age of all sailors.

```
SELECT  AVG (S.age)
FROM  Sailors S
```

Find the average age of sailors with a rating of 10.

```
SELECT  AVG (S.age)
FROM  Sailors S
WHERE  S.rating=10
```

Find the name and age of the oldest sailor.

```
SELECT  S.sname
FROM  Sailors S
WHERE  S.rating= (SELECT  MAX(S2.rating)
                  FROM  Sailors S2)
```

Count the number of sailors.

```
SELECT  COUNT (*)
FROM  Sailors S
```

Count the number of different sailor names

```
SELECT  COUNT (DISTINCT S.sname)
FROM  Sailors S
```

Find the names of sailors who are older than the oldest sailor with a rating of 10.

```
SELECT  S.sname
FROM  Sailors S
WHERE  S.age > ( SELECT MAX ( S2.age )
                 FROM Sailors S2
                 WHERE S2.rating = 10 )
```

Find the names of sailors who are older than the oldest sailor with a rating of 10.

```
SELECT  S.sname
FROM  Sailors S
WHERE  S.age > ALL ( SELECT S2.age
                     FROM Sailors S2
                     WHERE S2.rating = 10 )
```