CSCI4333 Database Design & Implement

Lecture Sixteen SQL 2

Instructor: Dr. Yifeng Gao

What's contained in an SQL Query?

```
SELECT      target-list
FROM        relation-list
WHERE       qualification
```

Every SQL Query must have:

- *SELECT clause: specifies columns to be retained in result*
- *FROM clause: specifies a cross-product of tables*

The WHERE clause (optional) specifies selection conditions on the tables mentioned in the FROM clause

Table Definitions

We will be using the following relations in our examples:

Sailors(sid, sname, rating, age)

Boats(bid, bname, color)

Reserves(sid, bid, day)

Sailors(S)

sid	*sname*	*rating*	*age*
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

Reserves(R)

sid	*bid*	*day*
22	101	10/10/04
22	102	10/10/04
22	103	10/08/04
22	104	10/07/04
31	102	11/10/04
31	103	11/06/04
31	104	11/12/04
64	101	09/05/04
64	102	09/08/04
74	103	09/08/04

Boats(B)

bid	*bname*	*Color*
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

UNION, INTERSECT, EXCEPT

- UNION: Can be used to compute the union of any two *union-compatible* sets of tuples (which are themselves the result of SQL queries).
- EXCEPT: Can be used to compute the set-difference operation on two *union-compatible* sets of tuples (Note: In ORACLE, the command for set-difference is *MINUS*).
- INTERSECT: Can be used to compute the intersection of any two *union-compatible* sets of tuples.

Illustration of UNION

Sailors who reserves red or 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'
(Sailors who reserves red boat)
UNION
SELECT  S.sname
FROM  Sailors S, Boats B, Reserves R
WHERE  S.sid=R.sid AND R.bid=B.bid
           AND B.color= 'green' ;
(Sailors who reserves green boat)
```

Illustration of EXCEPT

Find the sids of all sailors who have reserved red boats ***but not*** *green boats:*

```
SELECT  S.sid
FROM  Sailors S, Boats B, Reserves R
WHERE  S.sid=R.sid AND R.bid=B.bid AND B.color= ‘red’
EXCEPT
SELECT  S2.sid
FROM  Sailors S2, Boats B2, Reserves R2
WHERE  S2.sid=R2.sid AND R2.bid=B2.bid AND B2.color= ‘green’ ;
```

Use MINUS instead of EXCEPT in Oracle

Illustration of INTERSECT…1

Find sids 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  S2.sid
FROM  Sailors S2, Boats B2, Reserves R2
WHERE  S2.sid=R2.sid AND R2.bid=B2.bid
AND B2.color= 'green' ;
```

Illustration of INTERSECT…1

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

Illustration of INTERSECT…2

*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'
INTERSECT
SELECT  S2.sname
FROM  Sailors S2, Boats B2, Reserves R2
WHERE  S2.sid=R2.sid AND R2.bid=B2.bid AND B2.color= 'green' ;
```

(Is this correct??)

if there are two sailors shared same name

sid	*sname*	*rating*	*age*
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35.0
64	Horatio	7	35.0
71	Zorba	10	16.0
74	Horatio	9	35.0
85	Art	3	25.5
95	Bob	3	63.5

(Semi-)Correct SQL Query for the Previous Example

```
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  S2.sid
FROM  Sailors S2, Boats B2, Reserves R2
WHERE  S2.sid=R2.sid AND R2.bid=B2.bid
            AND B2.color= 'green' ;
```

(This time we have actually extracted the ***sids*** of sailors, and not their names.)
(But the query asks for the names of the sailors.)

Nested Queries

- A **nested** query is a query that has another query embedded within it; this embedded query is called the **subquery**.
- Subqueries generally occur within the WHERE clause (but can also appear within the FROM and HAVING clauses)
- Nested queries are a very powerful feature of SQL. They help us write short and efficient queries.

(Think of nested **for** loops in C++. Nested queries in SQL are similar)

Nested Query 1

Find names of sailors who have reserved boat 103

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

Nested Query 2

Find names of sailors who ***have not*** *reserved boat 103*

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

Revisit a previous query

Find names 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  S2.sid
FROM  Sailors S2, Boats B2, Reserves R2
WHERE  S2.sid=R2.sid AND R2.bid=B2.bid
            AND B2.color= 'green' ;
```

Revisit a previous query

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

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