

Notes
February 2, 2009

Nested SQL queries:

```
SELECT ...  
FROM ...  
WHERE (SELECT ...  
      FROM ...  
      WHERE ...)
```

Example:

Find the 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)
```

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

You may use variables defined in the main query in nested queries. This is called “correlated nested queries.”

	[NOT] EXISTS	(-)	<i>True if subquery returns one or more tuples</i>
(-)	[NOT] IN	(-)	<i>True if the main query is a subset of the subquery</i>
	[NOT] UNIQUE	(-)	<i>True if subquery has no duplicates</i>
(-) op	[NOT] ANY	(-)	
(-) op	[NOT] ALL	(-)	

Where (-) are ~~queries~~. EXISTS and UNIQUE are unary, the others are binary.

op ∈ {<, >, =, ≠, ≤, ≥}

values of attributes

Q:

Find the names of sailors with the highest rating.

```
SELECT S.sname  
FROM Sailors S  
WHERE S.rating ≥ ALL (SELECT S1.rating  
                    FROM Sailors S1)
```

Q:

Find the names of sailors who reserved all boats.

Solution 1: Find sailor S such that there is no boat B that is not associated with S in a reservation.
This type of operation is division in relational algebra.

```
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))
```

Solution 2: Set1 = Boats
Set2 = Set of all tuples in ~~Set1~~ associated with current x_i

if (Set1 – Set2 = {}) then x_i qualifies

the "Boats" part of Reserves

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

Aggregates:

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

It does NOT make any difference when we use DISTINCT for MAX and MIN

Note: A query using aggregates always returns a single tuple! Use of multiple aggregates will return a relation with multiple attributes, but still only one tuple.

GROUP BY Syntax:

```
SELECT [DISTINCT] target_list
FROM relation_list
WHERE qualification
GROUP BY grouping_list
HAVING group_qualification
```

Evaluation:

1. Cross product
2. Select qualifying tuples
3. Group tuples based on grouping_list
4. Ignore groups where group_qualification returns false
5. Project the columns needed