

Feb 2

ex. Find names of sailors who have reserved boat #103

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

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

Note that we used S for Sailors in the nested query.

EXISTS, and UNIQUE evaluates the query and return either true or false

EXISTS	()	Unary
NOT EXISTS	()	Unary
UNIQUE	()	Unary
NOT UNIQUE	()	Unary
() IN ()		Binary
() NOT IN	()	Binary

Operators { >, <, =, ≥, ≤ }

OP ANY (set) Binary

OP ALL (set) Binary

S.rating > any ()

EX. Find names of sailors with the highest rating

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

```
SELECT S.sname
FROM Sailors S
WHERE S.rating ≥ ANY (SELECT S1.ratings → ≥ ANY, returns everything
                     FROM Sailors S1)
```

```
SELECT S.sname
FROM Sailors S
WHERE S.rating > ANY (SELECT S1.ratings
                     FROM Sailors S1)
```

Division

Find the names of sailors who reserved all boats.

Sol.1) In English words:

Find sailors such that there is NO boat B that is NOT associated with S in a reservation

```

SELECT S.sname
FROM Sailors S
WHERE ( NOT EXIST
      (SELECT B.bid
       FROM Boat B
       WHERE ( NOT EXIST
             (SELECT R.bid
              FROM Reservation R
              WHERE R.bid = B.bid AND R.sid = s.sid)
            )
      )
    )
  
```

Sol.2) Define two sets

R	
X	Y
X ₁	Y ₁
X ₂	Y ₂
X _i	Y _i
X _n	Y _n

T
Y
Y ₁
Y ₂
Y _i
Y _n

Set1: All tuple in T: {y₁, y₂, ..., y_n}

Set 2: Tuples in R₂ that is associated with X_i in R

For X_i:

If (SET1 – set2 == Φ) then x_i qualifies

```

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

Aggregates:

COUNT

SUM

AVG

MAX

MIN

Returns one single value (still considered a relation)

All can reference only one single column with exception to COUNT (COUNT * doesn't have to be tied to a single column)

Ex.

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

```
SELECT COUNT (*) FROM Sailors, Reservations
```

If Sailors had N count and Reservations had M, then the SELECT query will return N*M

You can specify to count duplicates or exclude them using the keyword DISTINCT →
COUNT (DISTINCT A)

Ex. Find names of sailors with highest rating

```
SELECT S.sname
```

```
FROM Sailors S
```

```
WHERE S.ratings = (SELECT MAX(S1.rating) FROM Sailors S1)
```

Or

```
WHERE S.ratings IN (SELECT MAX(S1.rating) FROM Sailors S1)
```

If aggregates are used in the SELECT clause, exactly one tuple will be returned.

You **can** do:

```
SELECT MAX(rating), AVG (rating) FROM Sailors
```

You **cannot** do:

```
SELECT S.sname, MAX(rating) FROM Sailors
```

GROUP BY

Syntax:

```
SELECT      [DISTINCT] target_list
```

```
FROM        relation_list
```

```
WHERE       qualifications
```

```
GROUP BY   grouping_list
```

```
HAVING     group_qualifications
```

Evaluation without using GROUP BY

- 1- Cross Product
- 2- SELECT tuples based on qualifications
- 3- Project the columns needed

Evaluation without using GROUP BY

- 1- Cross Product
- 2- Group tuples based on Group_list
- 3- SELECT tuples based on qualifications
- 4- Ignore those groups where grouping_qualifications return false
- 5- Project the columns needed