

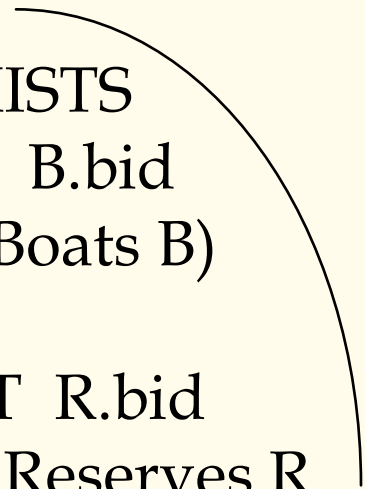


Division in SQL

Find sailors who've reserved all boats.

Solution 1:

```
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))
```





Division in SQL

Solution 2:

Let's do it the hard way, without EXCEPT:

```
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))
```

Sailors S such that ...

there is no boat B without ...

a Reserves tuple showing S reserved B



Aggregate Operators

- Significant extension of relational algebra.
 - COUNT (*)
 - COUNT ([DISTINCT] A)
 - SUM ([DISTINCT] A)
 - AVG ([DISTINCT] A)
 - MAX (A)
 - MIN (A)
- A is single column



Examples of Aggregate Operators

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

```
SELECT COUNT (DISTINCT S.rating)  
FROM Sailors S  
WHERE S.sname='Bob'
```

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

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

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



Find name and age of the oldest sailor(s)

- The first query is illegal! (We'll look into the reason a bit later, when we discuss **GROUP BY**.)
- The third query is equivalent to the second query, and is allowed in the SQL/92 standard, but is not supported in some systems.

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

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

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



Motivation for Grouping

- So far, we've applied aggregate operators to all (qualifying) tuples. Sometimes, we want to apply them to each of several *groups* of tuples.
- Consider: *Find the age of the youngest sailor for each rating level.*
 - In general, we don't know how many rating levels exist, and what the rating values for these levels are!
 - Suppose we know that rating values go from 1 to 10; we can write 10 queries that look like this (!):

For $i = 1, 2, \dots, 10$:

```
SELECT MIN (S.age)
FROM Sailors S
WHERE S.rating =  $i$ 
```



Queries With GROUP BY and HAVING

```
SELECT    [DISTINCT] target-list
FROM      relation-list
WHERE     qualification
GROUP BY  grouping-list
HAVING    group-qualification
```

- The *target-list* contains
 - (i) attribute names
 - (ii) terms with aggregate operations (e.g., MIN (*S.age*)).
- The attribute list (i) must be a subset of *grouping-list*. Intuitively, each answer tuple corresponds to a *group*, and these attributes must have a single value per group. (A *group* is a set of tuples that have the same value for all attributes in *grouping-list*.)