

# Advanced SQL 1

15 December 2009  
Lecture 9

## Topics for Today

- Nested Queries
- Aggregates
  
- Next time
  - Table constraints, Assertions
  - Triggers

December 15, 2009

ISE 322: Database Systems

2

## Nested Queries

- A **nested query** is one that has multiple layers of queries
  - WHERE
  - FROM
  - HAVING (soon)
- Why?
  - To do two separate calculations
  - To make "temporary tables"
  - Think of rename in RA

December 15, 2009

ISE 322: Database Systems

3

## Using Nested Queries

- Find the 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)
```

```
Sailors(sid:integer, sname:string, rating:integer, age:real)  
Boats(bid:integer, bname:string, color:string)  
Reserves(sid:integer, bid:integer, day:date)
```

December 15, 2009

ISE 322: Database Systems

4

## Using Nested Queries

- Find the 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)
```

```
Sailors(sid:integer, sname:string, rating:integer, age:real)  
Boats(bid:integer, bname:string, color:string)  
Reserves(sid:integer, bid:integer, day:date)
```

December 15, 2009

ISE 322: Database Systems

5

## Using nested queries

- Find the names of sailors who have 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'))
```

December 15, 2009

ISE 322: Database Systems

6

## Using nested queries

- Each nested query is a new table in memory
- Recomputed (logically) **for each row** in the outer query
  - Nested queries can refer to the outer one

December 15, 2009

ISE 322: Database Systems

7

## Correlated nested queries

- Since inner queries are recalculated we can refer to outer variables and columns inside
- This is **correlated nested query**

December 15, 2009

ISE 322: Database Systems

8

## Using correlated nested queries

- Find the names of sailors who have reserved boat number 103

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

```
Sailors(sid:integer, sname:string, rating:integer, age:real)
Boats(bid:integer, bname:string, color:string)
Reserves(sid:integer, bid:integer, day:date)
```

December 15, 2009

ISE 322: Database Systems

9

## Using set comparison operators

- op ANY
  - op can be: <, <=, >, >=, =, <>
- Semantics:
  - True for value x and set Y if **there exists** y in Y such that (x op y)
  - If Y is empty, it's trivially false
  - IN is the same as = ANY

December 15, 2009

ISE 322: Database Systems

10

## Using op ANY

- Find sailors whose rating is better than some sailor called Horatio

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

```
Sailors(sid:integer, sname:string, rating:integer, age:real)
Boats(bid:integer, bname:string, color:string)
Reserves(sid:integer, bid:integer, day:date)
```

December 15, 2009

ISE 322: Database Systems

11

## Using set comparison operators

- op ALL
  - op can be: <, <=, >, >=, =, <>
- Semantics:
  - true for value x and set Y if there **doesn't exist** y in Y such that !(x op y)
  - If Y is empty, it's trivially true
  - NOT IN is the same as <> ALL

December 15, 2009

ISE 322: Database Systems

12

## Using op ALL

- Find sailors whose rating is better than **every** sailor called Horatio

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

Sailors(sid:integer, sname:string, rating:integer, age:real)  
Boats(bid:integer, bname:string, color:string)  
Reserves(sid:integer, bid:integer, day:date)

December 15, 2009

ISE 322: Database Systems

13

## Using op ALL

- Find the sids for the sailors with the highest rating

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

Sailors(sid:integer, sname:string, rating:integer, age:real)  
Boats(bid:integer, bname:string, color:string)  
Reserves(sid:integer, bid:integer, day:date)

December 15, 2009

ISE 322: Database Systems

14

## So far

- Nested Queries
- Aggregates

December 15, 2009

ISE 322: Database Systems

15

## Aggregate Operators

- SQL lets you combine values and rows
  - COUNT ([DISTINCT] A)
    - Without DISTINCT, COUNT(A) is identical to COUNT(\*)
  - SUM ([DISTINCT] A)
  - AVG ([DISTINCT] A)
  - MAX(A)
  - MIN(A)
- They are **not** found in Relational Algebra
  - They can be written using the G or  $\gamma$  operators which we didn't see
- They don't all make sense for non-numerical types

December 15, 2009

ISE 322: Database Systems

16

## Using Aggregates

- Find the average age of all sailors

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

Sailors(sid:integer, sname:string, rating:integer, age:real)  
Boats(bid:integer, bname:string, color:string)  
Reserves(sid:integer, bid:integer, day:date)

December 15, 2009

ISE 322: Database Systems

17

## Using Aggregates

- Find the average age of all sailors with rating 10

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

Sailors(sid:integer, sname:string, rating:integer, age:real)  
Boats(bid:integer, bname:string, color:string)  
Reserves(sid:integer, bid:integer, day:date)

December 15, 2009

ISE 322: Database Systems

18

## Using aggregates

- Find the name and age of the oldest sailor.
- Maybe:

```
SELECT S.sname, MAX(S.age)    What's wrong here?
FROM Sailors S
```

Sailors(sid:integer, sname:string, rating:integer, age:real)  
Boats(bid:integer, bname:string, color:string)  
Reserves(sid:integer, bid:integer, day:date)

December 15, 2009

ISE 322: Database Systems

19

## Using aggregates

- That's illegal!
- If you use an aggregate operator in SELECT, you must **ONLY** use aggregate operators
  - Unless you use a GROUP BY (soon)

December 15, 2009

ISE 322: Database Systems

20

## Using aggregates

- Find the name and age of the oldest sailor.
- Correct:

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

Sailors(sid:integer, sname:string, rating:integer, age:real)  
Boats(bid:integer, bname:string, color:string)  
Reserves(sid:integer, bid:integer, day:date)

December 15, 2009

ISE 322: Database Systems

21

## Using aggregates

- Count the number of sailors

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

Sailors(sid:integer, sname:string, rating:integer, age:real)  
Boats(bid:integer, bname:string, color:string)  
Reserves(sid:integer, bid:integer, day:date)

December 15, 2009

ISE 322: Database Systems

22

## Using aggregates

- Count the number of different sailor names

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

Sailors(sid:integer, sname:string, rating:integer, age:real)  
Boats(bid:integer, bname:string, color:string)  
Reserves(sid:integer, bid:integer, day:date)

December 15, 2009

ISE 322: Database Systems

23

## Aggregates and Sets

- You can replace some set operators with aggregate operators
  - > **MAX** for > **ALL**
  - ≥ **MIN** for ≥ **ANY**

December 15, 2009

ISE 322: Database Systems

24

## Group By / Having

- Break up the results into groups
  - Grouping conditions – GROUP BY
  - Checks on groups – HAVING

```
SELECT [DISTINCT] select-list
FROM from-list
WHERE qualification
GROUP BY grouping-list
HAVING group-qualification
```

December 15, 2009

ISE 322: Database Systems

25

## Group By Example

- Find the youngest sailor for each rating level

```
SELECT S.rating, MIN(S.age)
FROM Sailors S
GROUP BY S.rating
```

December 15, 2009

ISE 322: Database Systems

26

## Group By rules

- Select list contains
  - Column names
  - Terms `aggop(column-name)` as new
- Each column in select must appear in grouping list

December 15, 2009

ISE 322: Database Systems

27

## Having rules

- If you use a group-qualification
  - One value per group to compare
  - Decides whether include the group
- SQL:1999
  - New set functions per group
  - Any row
  - All rows

December 15, 2009

ISE 322: Database Systems

28

## Group By Having

- If you have a HAVING and no GROUP BY
  - The table is all one group

December 15, 2009

ISE 322: Database Systems

29

## Example

- Find the age of the youngest sailor who is eligible to drink (i.e is at least 18 years old) for each rating level with at least two such sailors

```
SELECT S.rating, MIN(S.age) AS minage
FROM Sailors S
WHERE S.age >= 18
GROUP BY S.rating
HAVING COUNT(*) > 1
```

```
Sailors(sid:integer, sname:string, rating:integer, age:real)
Boats(bid:integer, bname:string, color:string)
Reserves(sid:integer, bid:integer, day:date)
```

December 15, 2009

ISE 322: Database Systems

30

## Example

```
SELECT S.rating, MIN(S.age) AS minage
FROM Sailors S
WHERE S.age >= 18
GROUP BY S.rating
HAVING COUNT(*) > 1
```

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

rating	age
7	45.0
1	33.0
8	55.5
8	25.5
10	35.0
7	35.0
9	35.0
3	25.5
3	63.5
3	25.5

rating	age
1	33.0
3	25.5
3	63.5
3	25.5

rating	minage
7	45.0
7	35.0
8	55.5
8	25.5
9	35.0
10	35.0

December 15, 2009

ISE 322: Database Systems

31

## Conclusion

- Nested Queries
- Aggregates
- Recitation is practice on nested queries, aggregates

December 15, 2009

ISE 322: Database Systems

32