

# Database Management Systems

Vijaya Saradhi

**IIT Guwahati**

Thu, 20<sup>th</sup> Feb 2020

# Re-naming

$\rho(\text{Result}(A1, B1, B2, D1), \text{table1}) \bowtie_{\text{table1.B}=\text{table2.B} \& \text{table1.A} < \text{table2.D}} \text{table2}$

table		table2	
A	B	B	D
1	2	2	3
1	2	4	5
		4	5

Result			
A1	B1	B2	D1
1	2	2	3
1	2	2	3

```
CREATE TABLE Result (
    SELECT  A AS A1, table1.B AS B1,
           table2.B AS B2, D AS D1
    FROM    table1
    JOIN    table2
    ON      table1.B = table2.B
    WHERE   table1.A < table2.D;
);
```

# Distinct

 $\pi_{attr1, attr2}(table1)$ 

table1		
attr1	attr2	attr3
1	2	5
3	4	6
1	2	7
1	2	7

table1	
attr1	attr2
1	2
3	4

```
SELECT DISTINCT attr1 , attr2
FROM      table1
```

# Aggregation Operations - SUM

## Example

table1	
A	B
1	2
3	4
1	2
1	2

SUM(B)
10

```
SELECT SUM(B)
FROM   table1;
```

# Aggregation Operations - Average

## Example

table1	
A	B
1	2
3	4
1	2
1	2

AVG(A)
1.5

```
SELECT AVG(A)
FROM   table1;
```

# Aggregation Operations - MIN

## Example

table1	
A	B
1	2
3	4
1	2
1	2

MIN(A)
1

```
SELECT MIN(A)
FROM   table1;
```

# Aggregation Operations - MAX

## Example

table1	
A	B
1	2
3	4
1	2
1	2

MAX(A)
3

```
SELECT MAX(A)
FROM   table1;
```

# Aggregation Operations - COUNT

## Example

table1	
A	B
1	2
3	4
1	2
1	2

COUNT(A)
4

```
SELECT COUNT(A)
FROM   table1;
```



# Extended Projection

$\pi_{A,B+C \rightarrow X}(table1)$

table1		
A	B	C
0	1	2
0	1	2
3	4	5

A	X
0	3
0	3
3	9

```
SELECT A, (B + C) AS X
FROM   table1;
```

# Extended Projection

 $\pi_{B-A \rightarrow X, C-B \rightarrow Y}(table1)$ 

table1		
A	B	C
0	1	2
0	1	2
3	4	5

X	Y
1	1
1	1
1	1

```
SELECT (B - A) AS X, (C - B) AS Y
FROM   table1;
```

# Sorting

 $\tau_A(table1)$ 

table1		
A	B	C
3	4	5
1	1	2
7	1	2

table1		
A	B	C
1	1	2
3	4	5
7	1	2

```
SELECT A, B, C
FROM   table1;
ORDER BY A;
```

# Right Outer Join

## Right Outer Join

U		
A	B	C
1	2	3
4	5	6
7	8	9

  

V		
B	C	D
2	3	10
2	3	11
6	7	12

$U \overset{\circ}{\bowtie} V$ $R$					
A	B	C	B	C	D
1	2	3	2	3	10
1	2	3	2	3	11
⊥	⊥	⊥	6	7	12

```

SELECT *
FROM U
RIGHT OUTER JOIN V
ON U.B = V.B
AND U.C = V.C;

```

# Left Outer Join

## Left Outer Join

U		
A	B	C
1	2	3
4	5	6
7	8	9

  

V		
B	C	D
2	3	10
2	3	11
6	7	12

$U \overset{\circ}{\bowtie}_L V$					
A	B	C	B	C	D
1	2	3	2	3	10
1	2	3	2	3	11
4	5	6	⊥	⊥	⊥
7	8	9	⊥	⊥	⊥

```

SELECT *
FROM U
LEFT OUTER JOIN V
ON U.B = V.B
AND U.C = V.C;

```

# Grouping

 $\gamma_{rating}(Sailors)$ 

Sailors			
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

$\gamma_{rating}(Sailors)$			
sid	sname	rating	age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
58	Rusty	10	35.0
74	Horatio	9	35.0
85	Art	3	25.5

# Grouping

## Group by rating

```
SELECT *
FROM   Sailors
GROUP BY rating;
```

## Output

Sailors			
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

$\gamma_{rating}(Sailors)$			
sid	sname	rating	age
22	Dustin	7	45.0
29	Brutus	1	33.0
31	Lubber	8	55.5
58	Rusty	10	35.0
74	Horatio	9	35.0
85	Art	3	25.5

# Grouping

Group by rating such that each group has at least two sailors

```
SELECT *
FROM   Sailors
GROUP BY rating
HAVING COUNT(rating) > 1;
```

## Output

Sailors			
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

$\gamma_{rating}(Sailors)$			
sid	sname	rating	age
22	Dustin	7	45.0
31	Lubber	8	55.5
58	Rusty	10	35.0
85	Art	3	25.5



# Grouping

Group by rating such that each group has at least two sailors where sailor age  $\geq 30$

```
SELECT *
FROM   Sailors
WHERE  age >= 30
GROUP BY rating
HAVING COUNT(rating) > 1;
```

## Output

Sailors			
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

$\gamma_{rating}(Sailors)$			
sid	sname	rating	age
22	Dustin	7	45.0
31	Lubber	8	55.5
58	Rusty	10	35.0

# All six clauses of SELECT

Group by rating such that each group has at least two sailors where sailor age  $\geq 20$  sort by sailor names

```
SELECT *
FROM   Sailors
WHERE  age >= 20
GROUP BY rating
HAVING COUNT(rating) > 1
ORDER BY sname;
```

## Output

Sailors			
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

$\gamma_{rating}(Sailors)$			
sid	sname	rating	age
85	Art	3	25.5
22	Dustin	7	45.0
31	Lubber	8	55.5
58	Rusty	10	35.0

# Set Operator - Union

table1  $\cup$  table2

table1		table2	
A	B	B	D
1	2	2	3
1	2	4	5
		4	5

table1 $\cup$ table2	
A	B
1	2
2	3
4	5

```
(SELECT      *
FROM      table1)
```

```
UNION
```

```
(SELECT      *
FROM      table2);
```

# Set Operator - Intersection

table1  $\cap$  table2

table1		table2	
A	B	B	D
1	2	2	3
1	2	4	5
		4	5

table1 $\cap$ table2	
A	B

```

SELECT      *
FROM        table1
WHERE (a, b)

IN

(SELECT      *
FROM        table2);

```

# Set Operator - Difference

table1 – table2

table1		table2	
A	B	B	D
1	2	2	3
1	2	4	5
		4	5

table1 – table2	
A	B
1	2
2	3
4	5

```

SELECT  *
FROM    table1
WHERE   (a, b)

NOT IN

(SELECT  *
FROM    table2);

```

# Queries on Example Database

Q1: Find the names of the **Sailors** who have reserved **Boat** 103

$\pi_{sname}(\sigma_{bid=103}(Reserves) \bowtie Sailors)$

# Queries on Example Database

Q1: Find the names of the Sailors who have reserved Boat 103

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

# Queries on Example Database

Q1: Find the names of the Sailors who have reserved Boat 103

```
SELECT  sname
FROM    Sailors
JOIN    Reserves
ON      Sailors.sid = Reserves.sid
WHERE   (Reserves.bid = 103)
```



# Queries on Example Database

Q1: Find the names of the Sailors who have reserved Boat 103

```
SELECT S.sname
FROM Sailors AS S
JOIN Reserves AS R
ON S.sid = R.sid
WHERE (R.bid = 103)
```

# Queries on Example Database

Q1: Find the names of the Sailors who have reserved Boat 103

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

# Queries on Example Database

Q2: Find the names of the Sailors who reserved a red boat

```
SELECT S.sname
FROM Sailors AS S, Reserves AS R, Boats AS B
WHERE (S.sid = R.sid
      AND R.bid = B.bid
      AND B.color = 'red')
```

# Queries on Example Database

Q2: Find the names of the Sailors who reserved a red boat

```
SELECT S.sname
FROM Sailors AS S
JOIN Reserves AS R
JOIN Boats AS B
ON (S.sid = R.sid
    AND R.bid = B.bid)
WHERE B.color = 'red';
```

# Queries on Example Database

Q3: Find the colors of boats reserved by Lubber

```
SELECT S.sname
FROM Sailors AS S, Reserves AS R, Boats AS B
WHERE (S.sid = R.sid
      AND R.bid = B.bid
      AND S.sname = 'Lubber')
```

# Queries on Example Database

Q3: Find the colors of boats reserved by Lubber

```
SELECT S.sname
FROM Sailors AS S
JOIN Reserves AS R
JOIN Boats AS B
ON (S.sid = R.sid AND
    R.bid = B.bid)
WHERE S.sname = 'Lubber';
```

# Queries on Example Database

Q4: Find the names of Sailors who have reserved at least one boat

```
SELECT S.sname
FROM   Sailors AS S, Reserves AS R
WHERE  (S.sid = R.sid);
```

# Queries on Example Database

Q4: Find the names of Sailors who have reserved at least one boat

```
SELECT S.sname
FROM Sailors AS S
JOIN Reserves AS R
ON (S.sid = R.sid);
```



# Queries on Example Database

Q5: Find the names of Sailors who have reserved a red or a green Boat

```
(SELECT S.sname
FROM   Sailors AS S, Reserves AS R, Boats AS B
WHERE  (S.sid = R.sid AND R.bid = B.bid AND B.color = 'red')
;

UNION

(SELECT S.sname
FROM   Sailors AS S, Reserves AS R, Boats AS B
WHERE  (S.sid = R.sid AND R.bid = B.bid AND B.color = 'green'
      '));
```

# Queries on Example Database

Q5: Find the names of Sailors who have reserved a red or a green Boat

```
(SELECT S.sname
FROM   Sailors AS S
JOIN   Reserves AS R
JOIN   Boats AS B
ON     (S.sid = R.sid AND
       R.bid = B.bid)
WHERE  B.color = 'red');
```

UNION

```
(SELECT S.sname
FROM   Sailors AS S
JOIN   Reserves AS R
JOIN   Boats AS B
ON     (S.sid = R.sid AND
       R.bid = B.bid)
WHERE  B.color = 'green');
```

# Queries on Example Database

Q6: Find the names of Sailors who have reserved a red AND a green Boat

```
(SELECT R.sid  
FROM Reserves AS R  
JOIN Boats AS B  
ON (R.bid = B.bid)  
WHERE B.color = 'red' AND R.sid
```

IN

```
(SELECT R.sid  
FROM Reserves AS R  
JOIN Boats AS B  
ON (R.bid = B.bid)  
WHERE B.color = 'green');
```

# Queries on Example Database

Q6: Find the names of Sailors who have reserved a red AND a green Boat

```
SELECT S1.sname
FROM Sailors AS S1
JOIN (SELECT R.sid
      FROM Reserves AS R
      JOIN Boats AS B
      ON (R.bid = B.bid)
      WHERE B.color = 'red' AND R.sid
      IN
      (SELECT R.sid
      FROM Reserves AS R
      JOIN Boats AS B
      ON (R.bid = B.bid)
      WHERE B.color = 'green')) AS S2
ON S1.sid = S2.sid
```

# Queries on Example Database

Q7: Find the names of **Sailors** who have **reserved** at least two boats

```
CREATE TABLE Temp1 AS
  (SELECT      S.sid , S.sname , R.bid
   FROM        Sailors AS S, Reserves AS R
   WHERE       S.sid = R.sid);

SELECT T1.sname
FROM      Temp1 AS T1
JOIN      Temp1 AS T2
ON        T1.sid = T2.sid
WHERE     (T1.bid <> T2.bid)
```

# Queries on Example Database

Q7: Find the names of Sailors who have reserved at least two boats

```
SELECT T1.sname
FROM (SELECT S.sid , S.sname , R.bid
      FROM Sailors AS S, Reserves AS R
      WHERE S.sid = R.sid) AS T1
JOIN (SELECT S.sid , S.sname , R.bid
      FROM Sailors AS S, Reserves AS R
      WHERE S.sid = R.sid) AS T2
ON T1.sid = T2.sid
WHERE (T1.bid <> T2.bid);
```

# Queries on Example Database

Q8: Find the **sids** of **Sailors** with age over 20 who have not **reserved** a red **boat**

```
SELECT S1.sid
FROM   Sailors AS S1
WHERE  S1.age >= 20 AND S1.sid
      NOT IN
(
  SELECT S2.sid
FROM    Sailors AS S2
JOIN    Reserves AS R2
JOIN    Boats AS B2
ON      (S2.sid = R2.sid AND R2.bid = B2.bid)
WHERE   (B2.color = 'red'))
)
```

## Example 09 - Expressions in SELECT

Compute increments of ratings who have sailed two different boats on same day

```
SELECT      S1.name, S1.rating + 1 as rating
FROM        Sailors AS S1
JOIN        Reserves AS R1
JOIN        Reserves AS R2
WHERE       S1.sid = R1.sid
AND         S1.sid = R2.sid
AND         R1.day = R2.day
AND         R1.bid <> R2.bid
```



## Example 10 - Regular Expressions

Find the ages of sailors whose name begins and ends with B and has at least three characters

```
SELECT      S1.age
FROM        Sailors AS S1
WHERE       S1.sname LIKE 'B_%B'
```

### LIKE regular expression

- % denote wild-card symbol. It matches zero or more characters
- \_ denote matching for exactly one arbitrary character
- B\_ denote any string starting with B followed by exactly one character
- B\_%B denote any string starting with B followed by exactly one character followed by zero or more characters and ending with B

## Example 11(a) - Union

Find the names of sailors who have reserved a red or green boat

```
(SELECT S1.sname
FROM   Sailors AS S1
JOIN   Reserves AS R1
JOIN   Boats AS B1
ON     S1.sid = R1.sid
AND    R1.bid = B1.bid
WHERE  B1.color = 'red')
```

UNION

```
(SELECT S1.sname
FROM   Sailors AS S1
JOIN   Reserves AS R1
JOIN   Boats AS B1
ON     S1.sid = R1.sid
AND    R1.bid = B1.bid
WHERE  B1.color = 'green')
```

## Example 11(b) - Union

Find the names of sailors who have reserved a red or green boat

```
SELECT S1.sname
FROM   Sailors AS S1
JOIN   Reserves AS R1
JOIN   Boats AS B1
ON     S1.sid = R1.sid
AND    R1.bid = B1.bid
WHERE  B1.color = 'red' OR B1.color = 'green'
```

## Example 12 - Intersection

Find the names of sailors who have reserved a red **and** a green boat

```
SELECT S1.sname
FROM   Sailors AS S1
JOIN   Reserves AS R1
JOIN   Boats AS B1
ON     S1.sid = R1.sid
AND    R1.bid = B1.bid
WHERE  B1.color = 'red'
AND    (S1.sid, S1.sname, S1.rating, S1.age)
```

IN

```
(SELECT *
FROM   Sailors AS S1
JOIN   Reserves AS R1
JOIN   Boats AS B1
ON     S1.sid = R1.sid
AND    R1.bid = B1.bid
WHERE  B1.color = 'green')
```

## Example 13 - Difference

Find sids of all sailors who have reserved red boat but not green boat

```
SELECT S1.sid
FROM Sailors AS S1
JOIN Reserves AS R1
JOIN Boats AS B1
ON S1.sid = R1.sid
AND R1.bid = B1.bid
WHERE B1.color = 'red'
AND (S1.sid, S1.sname, S1.rating, S1.age)
```

NOT IN

```
(SELECT *
FROM Sailors AS S1
JOIN Reserves AS R1
JOIN Boats AS B1
ON S1.sid = R1.sid
AND R1.bid = B1.bid
WHERE B1.color = 'green')
```

## Example 14 - Union

Find all sids of sailors who have a rating of 10 or reserved boat 104

```
(SELECT S1.sid  
FROM   Sailors AS S1  
WHERE  S1.rating = 10)
```

UNION

```
(SELECT R1.sid  
FROM   Reserves AS R1  
WHERE  R1.bid = 104)
```

## Example 15 - Nested Queries

Find the names of sailors who have reserved boat 103

```
SELECT  S1.sname
FROM    Sailors AS S1
WHERE   S1.sid
IN      (SELECT R1.sid
        FROM    Reserves AS R1
        WHERE   R1.bid = 103)
```

## Example 16 - Nested Queries

Find the names of sailors who have NOT reserved boat 103

```
SELECT  S1.sname
FROM    Sailors AS S1
WHERE   S1.sid
NOT IN
      (SELECT R1.sid
       FROM   Reserves AS R1
       WHERE  R1.bid = 103)
```



## Example 17 - Nested Queries

Find the names of Sailors who have reserved a red boat

```
SELECT  S1.sname
FROM    Sailors AS S1
WHERE   S1.sid
IN
    (SELECT R1.sid
     FROM   Reserves AS R1
     WHERE  R1.bid
     IN
        (SELECT B1.bid
         FROM   Boats AS B1
         WHERE  B1.color = 'red')
    )
```

## Example 18 - Correlated Nested Queries

### Correlated nested queries

- The inner sub-query has been completely independent of the outer query
- In general, the inner sub-query could dependent on the row currently being examined in the outer query
- Such queries are known as **Correlated** nested queries

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

```
SELECT  S1.sname
FROM    Sailors AS S1
WHERE   EXISTS
        (SELECT *
         FROM   Reserves AS R1
         WHERE  R1.bid = 103
         AND    S1.sid = R1.sid
        )
```

## Example 18 - Correlated Nested Queries

### Correlated nested queries

- For each Sailor row S1 test whether the set of Reserves row R1 such that  $R1.bid = 103$  AND  $S1.sid = R1.sid$
- If the above test is nonempty, sailor S1.sid has reserved boat 103
- Retrieve all such tuples
- The sub-query clearly depends on the current row of S1
- The sub-query must be evaluated for each row in S1

## Example 19 - Complex Correlated Nested Queries

Find the names of sailors who have reserved both red and green boat

```
SELECT  S1.sname
FROM    Sailors AS S1
WHERE   S1.sid
IN
  (SELECT      R1.sid
   FROM        Reserves AS R1
   JOIN        Boats AS B1
   ON          R1.bid = B1.bid
   WHERE       B1.color = 'red'

   AND        R1.bid
   IN
     (SELECT      R1.sid
    FROM          Reserves AS R1
    JOIN          Boats AS B1
    ON            R1.bid = B1.bid
    WHERE         B1.color = 'green'
```

## Example 20(a) - Complex Correlated Nested Queries

Find the names of sailors who have reserved all boats

```
SELECT  S1.sname
FROM    Sailors AS S1
WHERE
NOT EXISTS
    (SELECT      B1.bid
      FROM        Boats AS B1

      NOT IN

      (SELECT      R1.bid
        FROM        Reserves AS R1
        WHERE      R1.sid = S1.sid)
    )
```

## Example 20(b) - Complex Correlated Nested Queries

Find the names of sailors who have reserved all boats

```
SELECT  S1.sname
FROM    Sailors AS S1
WHERE

    NOT EXISTS

        (SELECT      B1.bid
          FROM        Boats AS B1
          WHERE

              NOT EXISTS

                  (SELECT      R1.bid
                    FROM        Reserves AS R1
                    WHERE        R1.sid = S1.sid
                    AND          R1.bid = B1.bid
                  )
        )
```