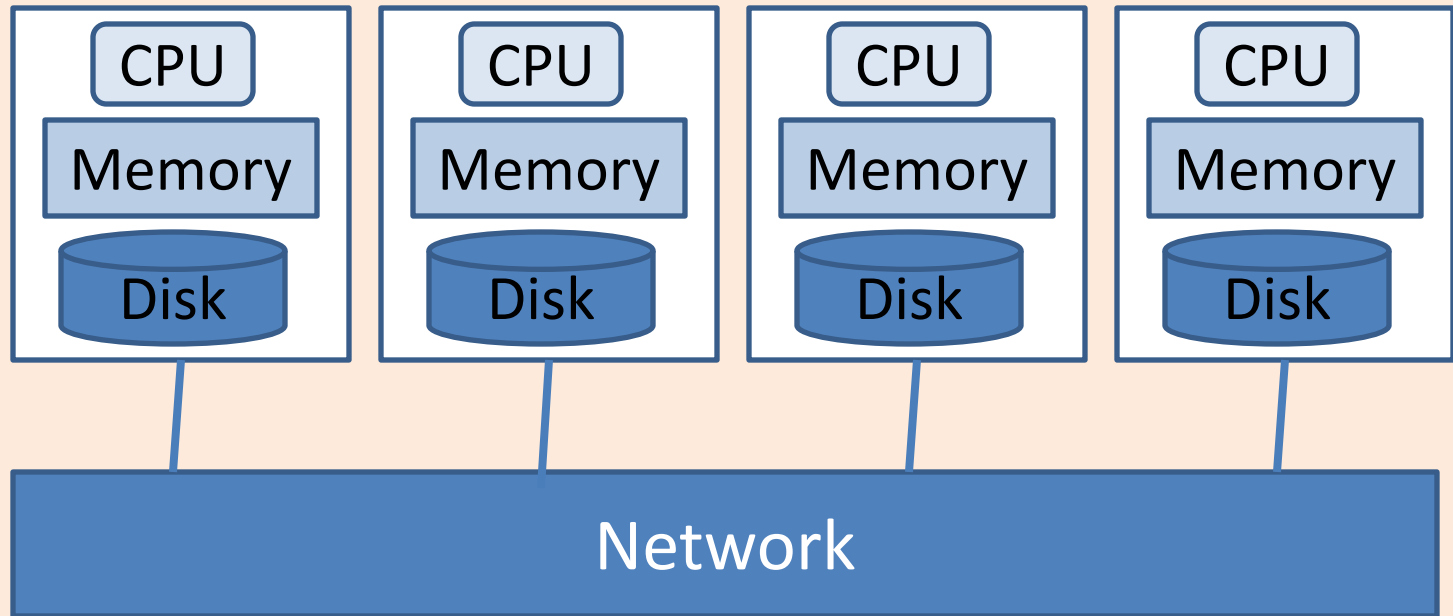


ICS 421 Spring 2010

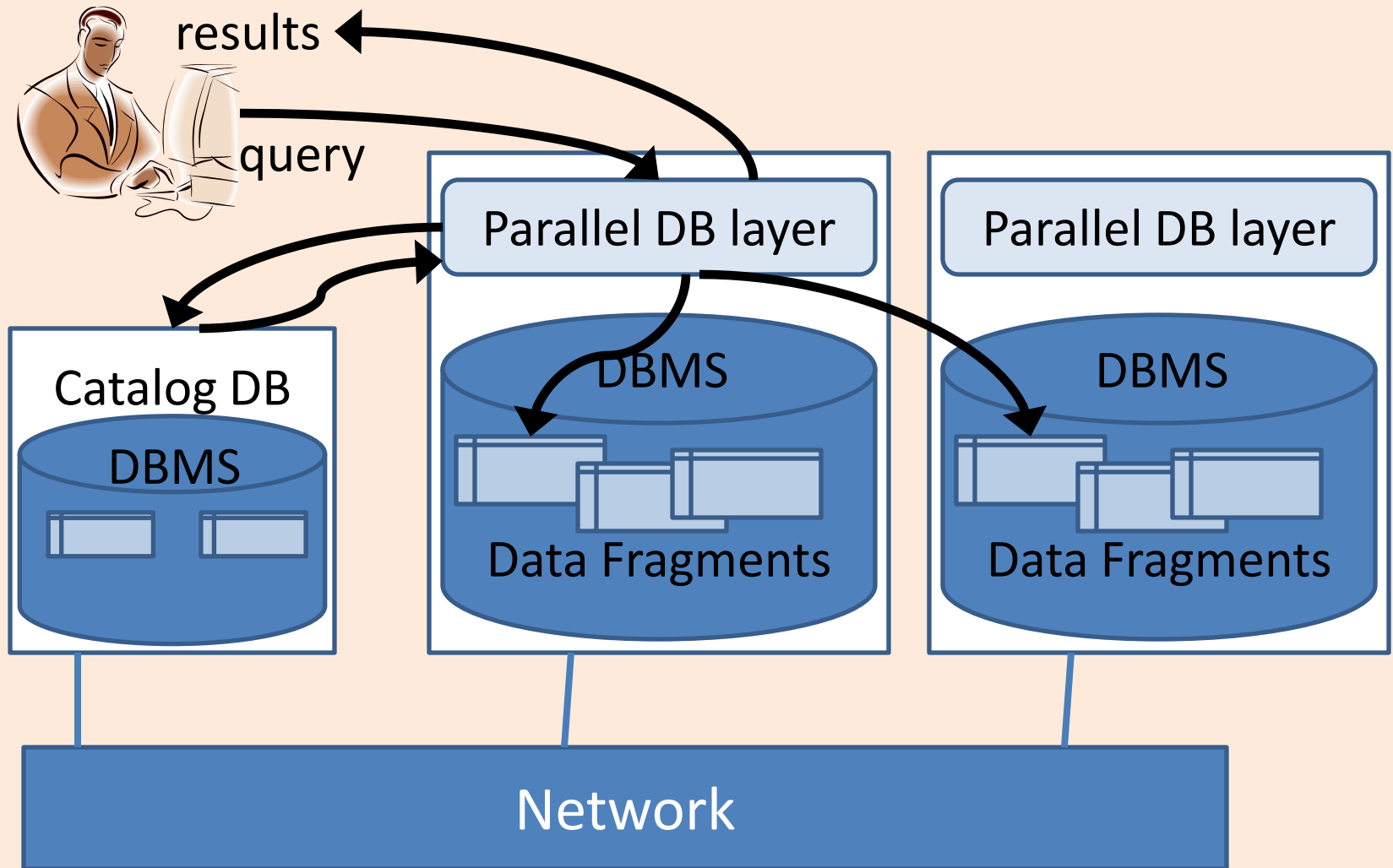
Parallel & Distributed Databases 2

Asst. Prof. Lipyeow Lim
Information & Computer Science Department
University of Hawaii at Manoa

Shared-Nothing Architecture



Logical Parallel DBMS Architecture



Horizontal Fragmentation: Range Partition

sid	sname	rating	age
22	dustin	7	45
29	brutus	1	33
31	lubber	8	55
32	andy	4	23
58	rusty	10	35
64	horatio	7	35

Partition 1

sid	sname	rating	age
29	brutus	1	33
32	andy	4	23

Partition 2

sid	sname	rating	age
22	dustin	7	45
31	lubber	8	55
58	rusty	10	35
64	horatio	7	35

Range Partition on rating column

- Partition 1: $0 \leq \text{rating} < 5$
- Partition 2: $5 \leq \text{rating} \leq 10$

Range Partition: Query Processing

- Which partitions?
- Better than non-parallel ?

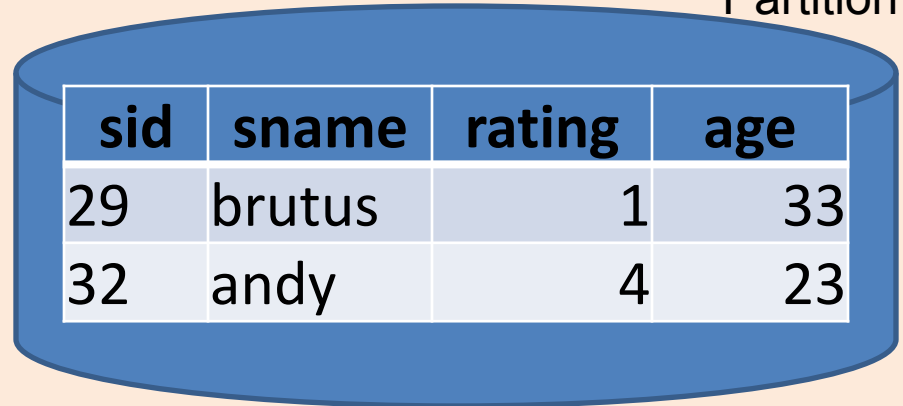
```
SELECT *  
FROM Sailors S
```

```
SELECT *  
FROM Sailors S  
WHERE rating = 2
```

```
SELECT *  
FROM Sailors S  
WHERE age > 30
```

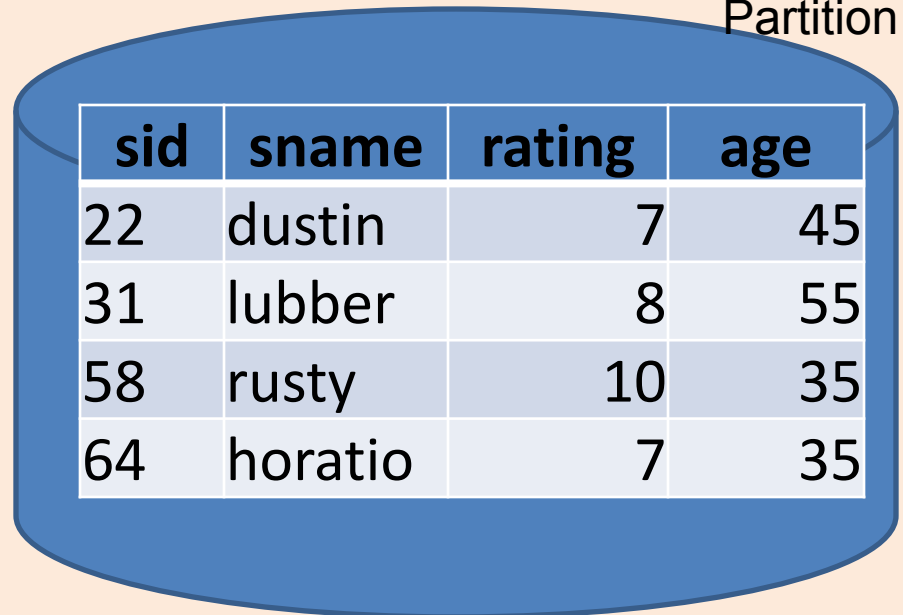
```
SELECT *  
FROM Sailors S  
WHERE rating < 2 and age < 30
```

Partition 1



sid	sname	rating	age
29	brutus	1	33
32	andy	4	23

Partition 2



sid	sname	rating	age
22	dustin	7	45
31	lubber	8	55
58	rusty	10	35
64	horatio	7	35

Horizontal Fragmentation: Hash Partition

sid	sname	rating	age
22	dustin	7	45
29	brutus	1	33
31	lubber	8	55
32	andy	4	23
58	rusty	10	35
64	horatio	7	35

Partition 1

sid	sname	rating	age
31	lubber	8	55
32	andy	4	23
58	rusty	10	35

Partition 2

sid	sname	rating	age
22	dustin	7	45
29	brutus	1	33
64	horatio	7	35

- Hash partitioning using hash function
 - Partition = rating mod 2

Hash Partition: Query Processing

- Which partitions?
- Better than non-parallel ?

```
SELECT *  
FROM Sailors S
```

```
SELECT *  
FROM Sailors S  
WHERE rating = 2
```

```
SELECT *  
FROM Sailors S  
WHERE age > 30
```

```
SELECT *  
FROM Sailors S  
WHERE rating < 2 and age < 30
```

Partition 1

sid	sname	rating	age
31	lubber	8	55
32	andy	4	23
58	rusty	10	35

Partition 2

sid	sname	rating	age
22	dustin	7	45
29	brutus	1	33
64	horatio	7	35

Vertical Fragmentation/Partition

sid	sname	rating	age
22	dustin	7	45
29	brutus	1	33
31	lubber	8	55
32	andy	4	23
58	rusty	10	35
64	horatio	7	35

sid	sname	rating
22	dustin	7
29	brutus	1
31	lubber	8
32	andy	4
58	rusty	10
64	horatio	7

Partition 1

Partition 2

sid	age
22	45
29	33
31	55
32	23
58	35
64	35

- Vertical partitioning
 - Use sid as row identifier

Vertical Partition: Query Processing

- Which partitions?
- Better than non-parallel ?

```
SELECT *  
FROM Sailors S
```

```
SELECT sname  
FROM Sailors S
```

```
SELECT *  
FROM Sailors S  
WHERE rating = 2
```

```
SELECT sid  
FROM Sailors S  
WHERE age > 30
```

```
SELECT sid  
FROM Sailors S  
WHERE rating < 2 and age < 30
```

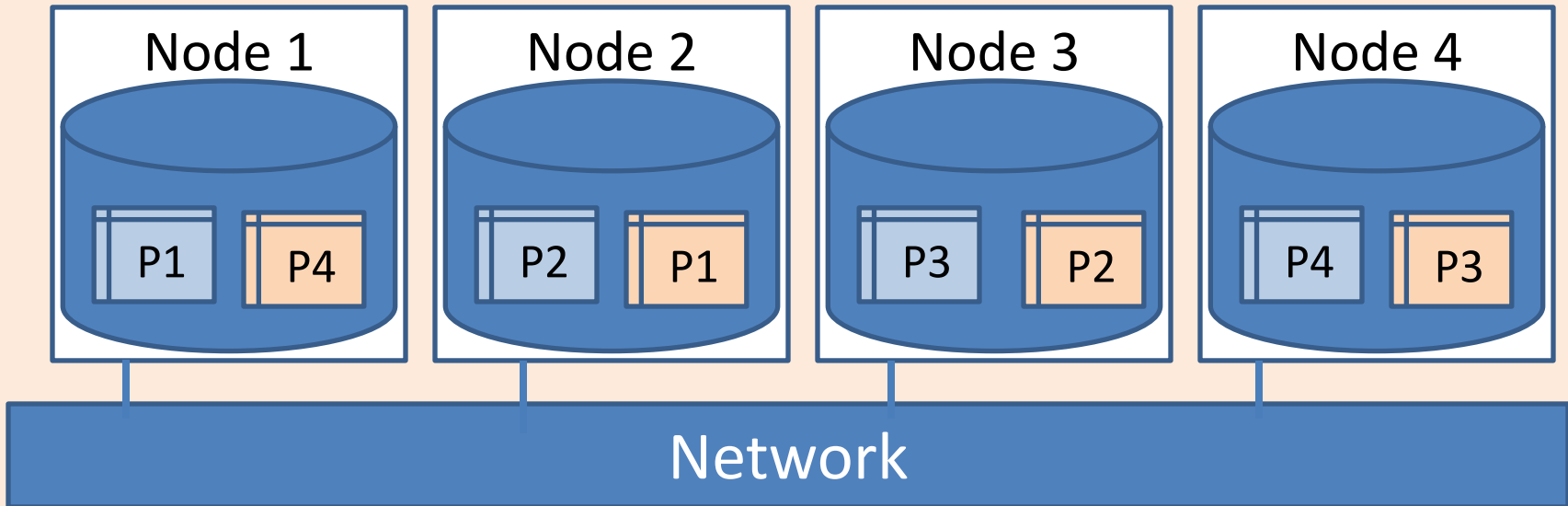
sid	sname	rating
22	dustin	7
29	brutus	1
31	lubber	8
32	andy	4
58	rusty	10
64	horatio	7

Partition 1

Partition 2

sid	age
22	45
29	33
31	55
32	23
58	35
64	35

Fragmentation & Replication



- Suppose table is fragmented into 4 partitions on 4 nodes
- Replication stores another partition on each node
 - What happens when 1 node fails ? 2 nodes ?
 - What happens when a row needs to be updated ?

What about joins ?

```

SELECT R.sid, R.bid
FROM Sailors S,
Reserves R
WHERE S.sid=R.sid AND
rating > 8
    
```

- Sailors: hash
 - part = rating mod 2
- Reserves: hash
 - part = sid mod 2
- Where to perform join ?
- What data to ship ?

Partition 1

Sailors				Reserves		
sid	sname	rating	age	sid	bid	day
31	lubber	8	55	31	101	...
32	andy	4	23	29	103	...
58	rusty	10	35			

Partition 2

Sailors				Reserves		
sid	sname	rating	age	sid	bid	day
22	dustin	7	45	64	105	...
29	brutus	1	33	58	103	...
64	horatio	7	35			