

Join Algorithms I

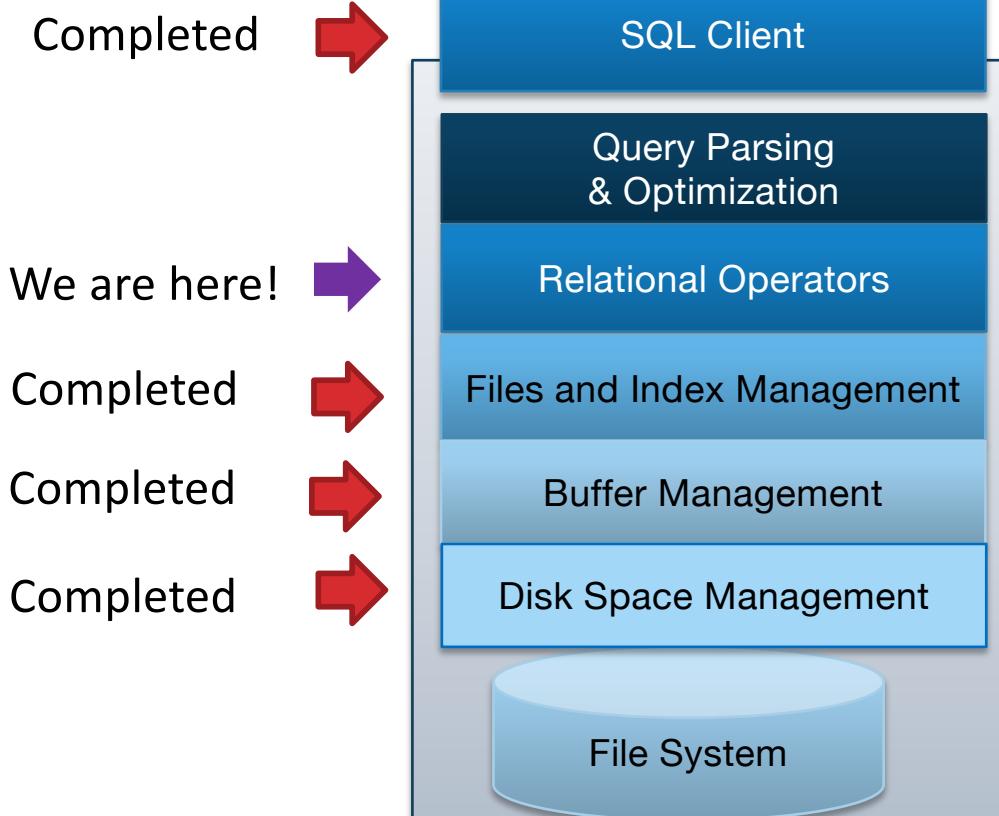
Alvin Cheung

Fall 2022

R&G Chapter 14



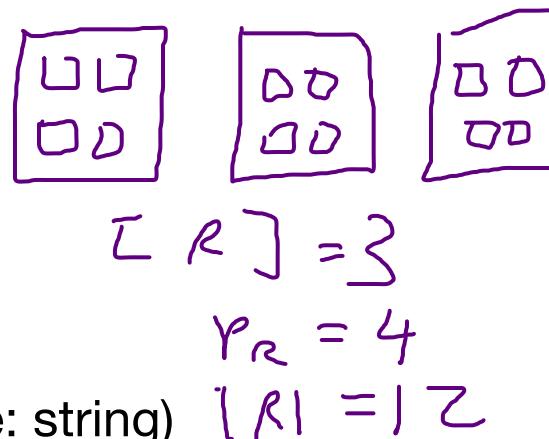
Architecture of a DBMS: What we've learned



Schema & Costing for Examples



- Cost Notation
 - $[R]$: the number of pages to store R
 - p_R : number of records per page of R
 - $|R|$: the cardinality (number of records) of R
 - $|R| = p_R * [R]$
- Reserves (sid: int, bid: int, day: date, rname: string)
 - $[R]=1000$, $p_R=100$, $|R| = 100,000$
- Sailors (sid: int, sname: string, rating: int, age: real)
 - $[S]=500$, $p_S=80$, $|S| = 40,000$



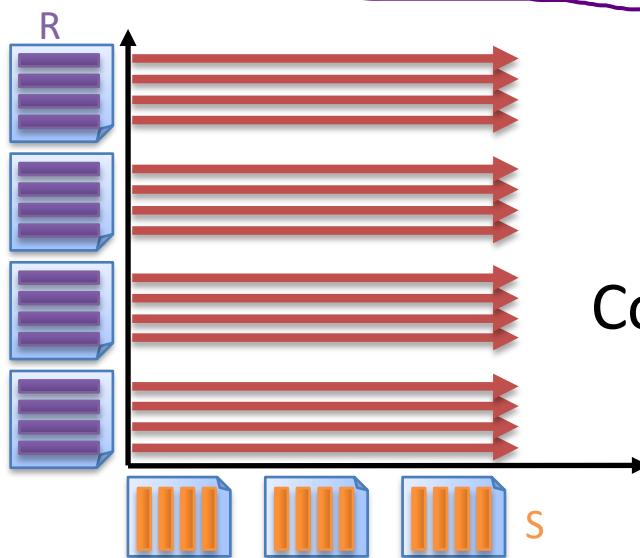
θ = join condition function (ex: $s.id = r.id$)

Simple Nested Loops θ Join



```
foreach record r in R do  
    foreach record s in S do  
        if  $\theta(r, s)$  then add <r, s> to result buffer
```

- We will ignore write cost
- It is the same across approaches
- Tuples are often streamed to subsequent operators rather than written to disk



$[R] = 1000, p_R = 100, |R| = 100,000$
 $[S] = 500, p_S = 80, |S| = 40,000$
Cost: read r pages + read s pages per record
scan R once + scan S once per R tuple

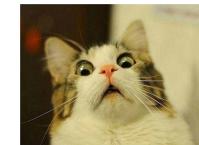
$$\begin{aligned} &= [R] + |R|[S] \\ &= 50,001,000 \end{aligned}$$

[R] : # pages to store R
 p_R : # records per page of R
 $|R|$: # records of R

Changing the Join Order

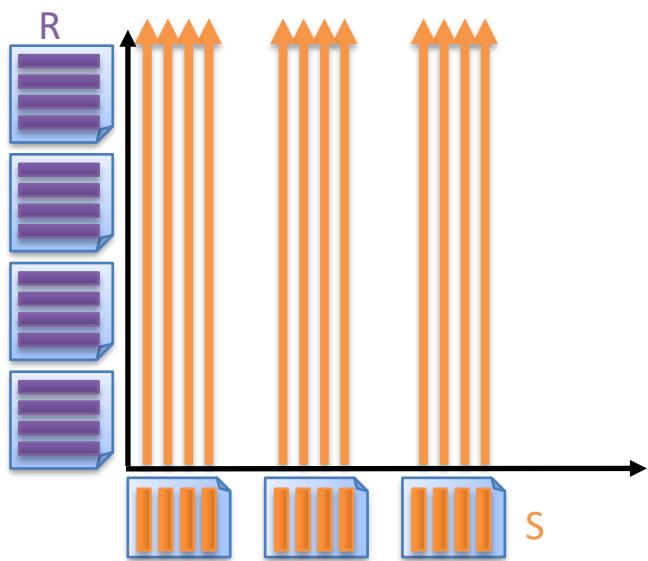


```
foreach record s in S do
    foreach record r in R do
        if θ(r, s) then add <r, s> to result buffer
```



Join orders matter!

Q: Can we improve even more?



$$[R] = 1000, p_R = 100, |R| = 100,000$$

$$[S] = 500, p_S = 80, |S| = 40,000$$

Cost: $\underbrace{\text{scan } S \text{ once}}_{\text{read } S \text{ pages}} + \underbrace{\text{scan } R \text{ once per } S \text{ tuple}}_{\text{read } R \text{ pages per record}}$

$$= [S] + |S|[R]$$

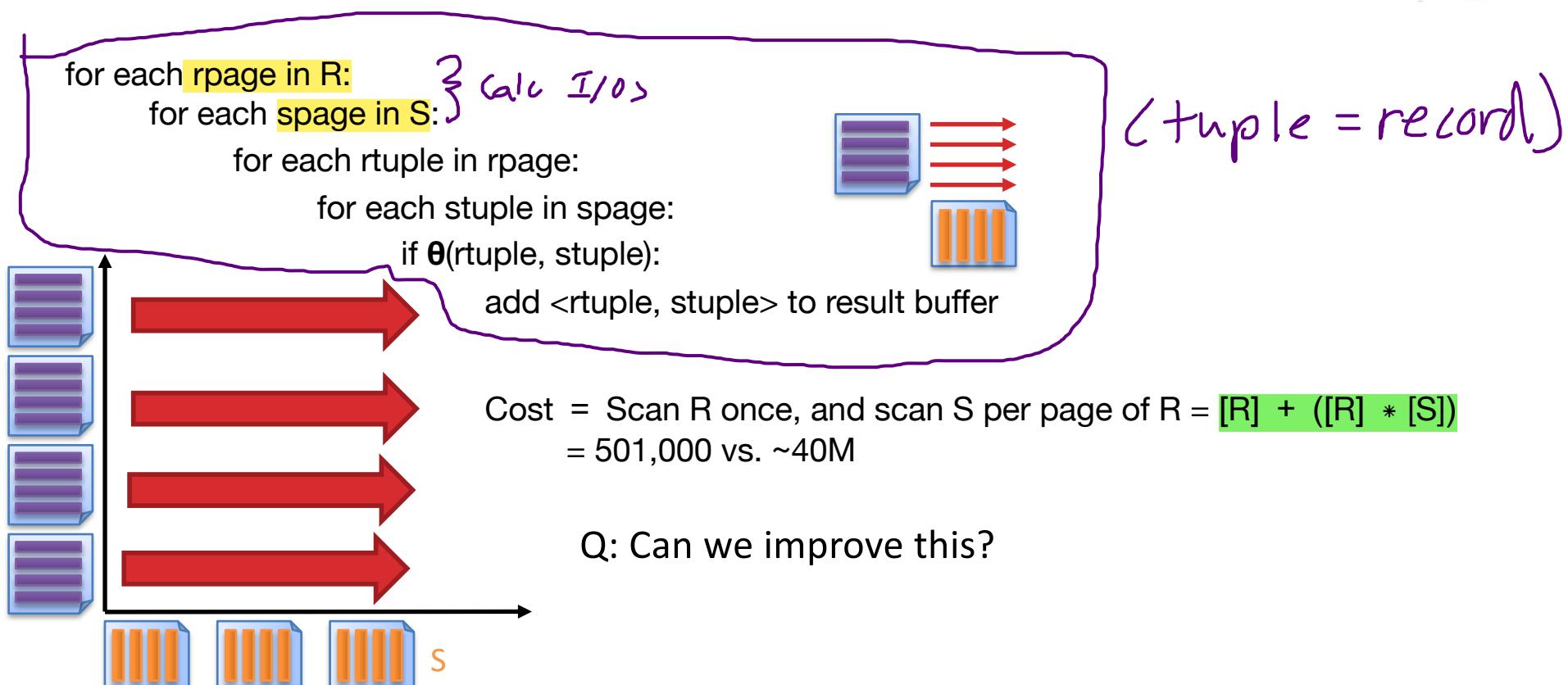
$$= 40,000,500$$

vs. 50,001,000

Page Nested Loop Join



Idea: previous algo was inefficient w.r.t. I/O: operate at granularity of pages!



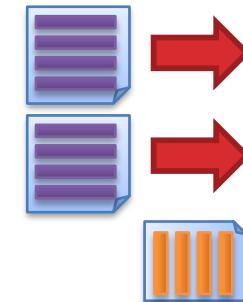
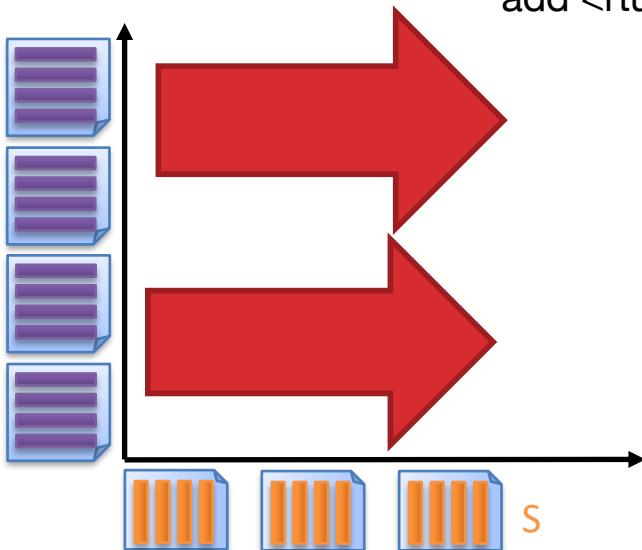
“Block” Nested Loop Join



Idea: Extending even further using a “block” or a “chunk” of S pages at a time

B-2 buffer pages

Calc I/Os { for each rchunk of B-2 pages of R: 1 buffer page for output
 for each spage of S: ← 1 buffer page
 for all matching tuples in spage and rchunk:
 add <rtuple, stuple> to result buffer



Cost = Scan R once, plus scan S as many times as there are chunks
= $[R] + \lceil [R]/(B-2) \rceil * [S]$
= $1000 + \lceil 1000/(B-2) \rceil * 500$
= 6,000 for B=102 (~100x better than Page NL!)

Overall, a frequently used join algorithm, esp. for non-eq. predicates

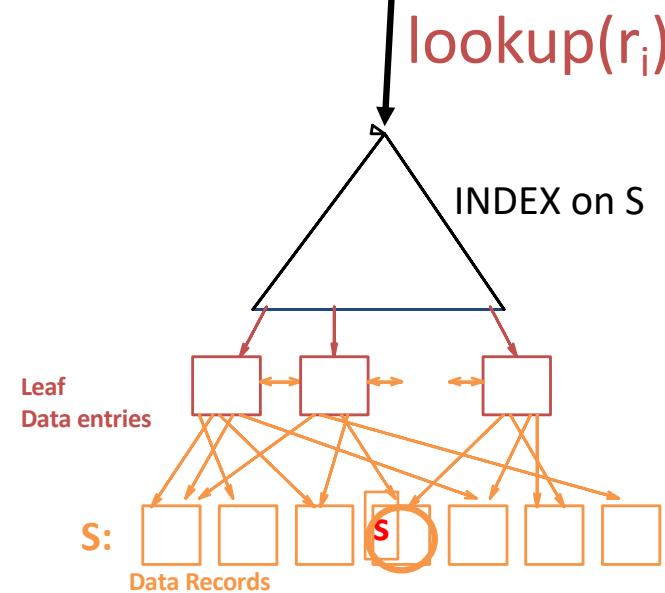
Index Nested Loops Join

Consider when we have equijoin on $r_i = s_j$



for each **tuple** r in R :

for each **tuple** s in S where $r_i == s_i$:
add $\langle r, s \rangle$ to result buffer





Index Nested Loops Join Cost

for each **tuple** r in R :

for each **tuple** s in S where $r_i == s_j$:
 add $\langle r, s \rangle$ to result

$$\text{Cost} = [R] + |R| * \text{cost to find matching } S \text{ tuples}$$

- If index uses Alt. 1 \rightarrow cost to traverse tree from root to leaf. (e.g., 2-4 IOs)
- For Alt. 2 or 3:
 - Cost to lookup RID(s); typically 2-4 IOs for B+Tree.
 - Cost to retrieve records from RID(s)
 - Clustered index: 1 I/O per *page of matching S tuples*.
 - Unclustered index: up to 1 I/O per matching S tuple

Sort-Merge Join



- Requires equality predicate:
 - Equi-Joins & Natural Joins
- Output is sorted on join attribute
- Two Stages:
 - **Sort: sort tuples in R and S by join key**
 - All tuples with same key in consecutive order
 - Input might already be sorted ... say from an earlier sort merge/index scan
 - **Join: Merge-scan the sorted partitions and emit tuples that match**
 - Each tuple in R may match multiple tuples in S
 - Keep track of the start of each block of S tuples with a “mark”
 - That way, we know where to return for the next tuple of R
 - R is “outer loop”, advances forward; S is “inner loop” forward + back to mark
- Will discuss some pseudocode – but may not cover all cases (beware!)

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

Sort-Merge Join

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107



Mark =
lowest index such
that $\text{right}[i] \leq \text{left}[i]$

Sort-Merge Join



sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

sid	sname	bid
28	yuppy	103

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104

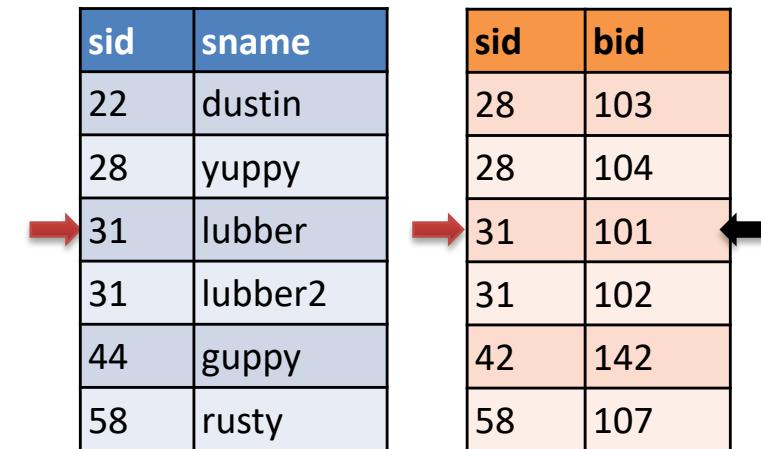
Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104

Sort-Merge Join



sid	sname	bid
28	yuppy	103
28	yuppy	104

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102

Sort-Merge Join



sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join



sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join



sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join



sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join



The diagram illustrates the Sort-Merge Join process. It shows two sorted input tables and their merge result. Red arrows indicate the merging of rows from both tables into the final result table.

Input Table 1 (Left):

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

Input Table 2 (Right):

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

Merge Result (Bottom):

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join



sid	sname	bid
22	dustin	
28	yuppy	
31	lubber	
31	lubber2	
44	guppy	
58	rusty	

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102
58	rusty	107

Sort-merge join pseudocode

Sort-Merge Join, Part 1

```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107



Sort-Merge Join, Part 2

```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

r

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

S

mark



Sort-Merge Join, Part 3

```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

r

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

s

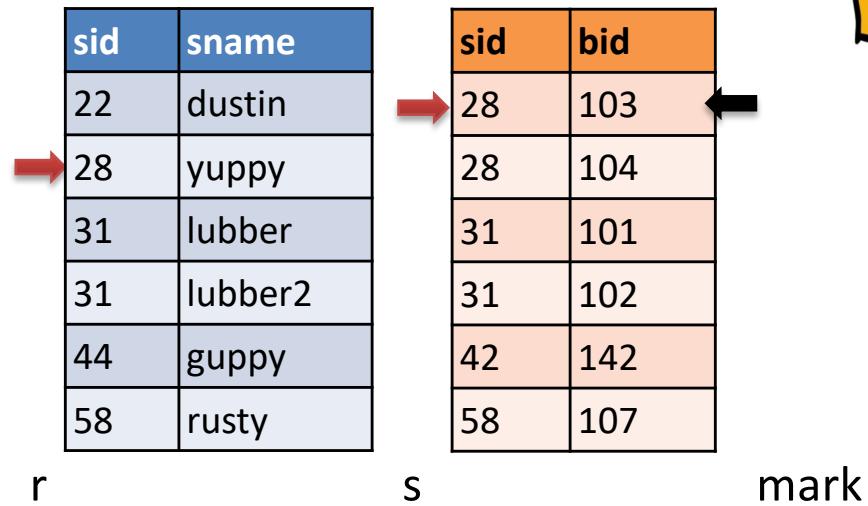
mark



Sort-Merge Join, Part 4



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```



Sort-Merge Join, Part 5



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

r *s* mark

Sort-Merge Join, Part 6

```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
    }
    return result
}
else {
    reset s to mark
    advance r
    mark = NULL
}
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

r

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

s

mark



Sort-Merge Join, Part 7



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
    }
    return result
}
else {
    reset s to mark
    advance r
    mark = NULL
}
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

r

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

s

mark

Sort-Merge Join, Part 8



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

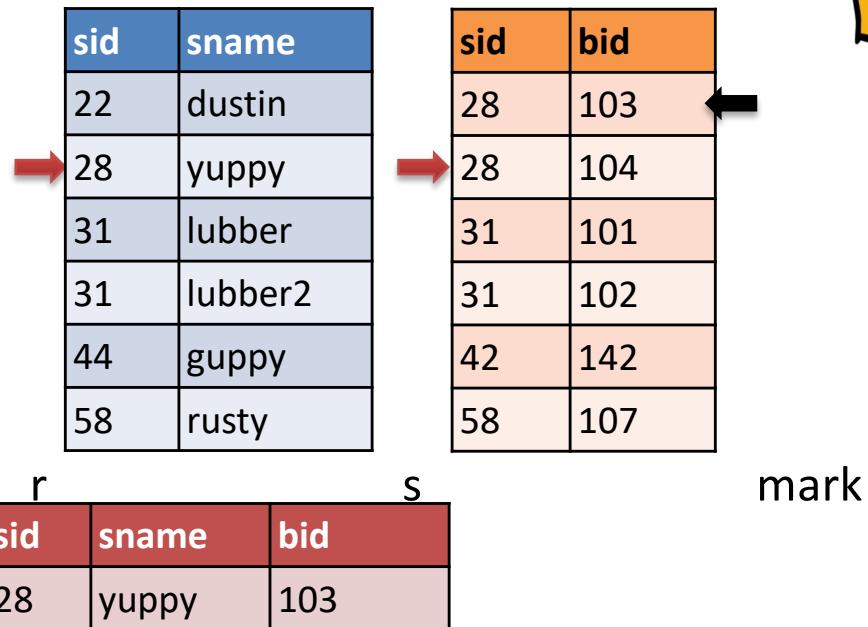
r		S		mark
sid	sname	sid	bid	
22	dustin	28	103	
28	yuppy	28	104	
31	lubber	31	101	
31	lubber2	31	102	
44	guppy	42	142	
58	rusty	58	107	

sid	sname	bid
28	yuppy	103

Sort-Merge Join, Part 9



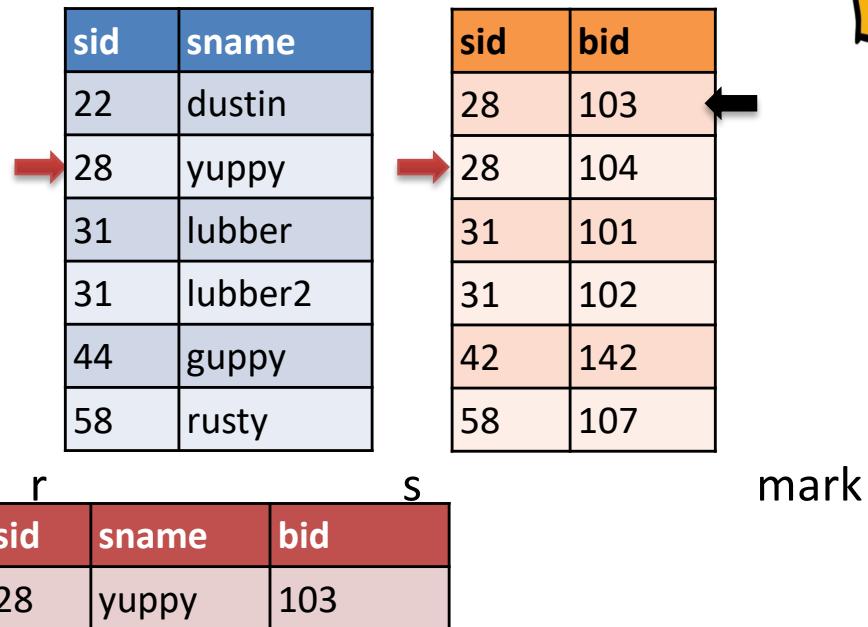
```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```



Sort-Merge Join, Part 10



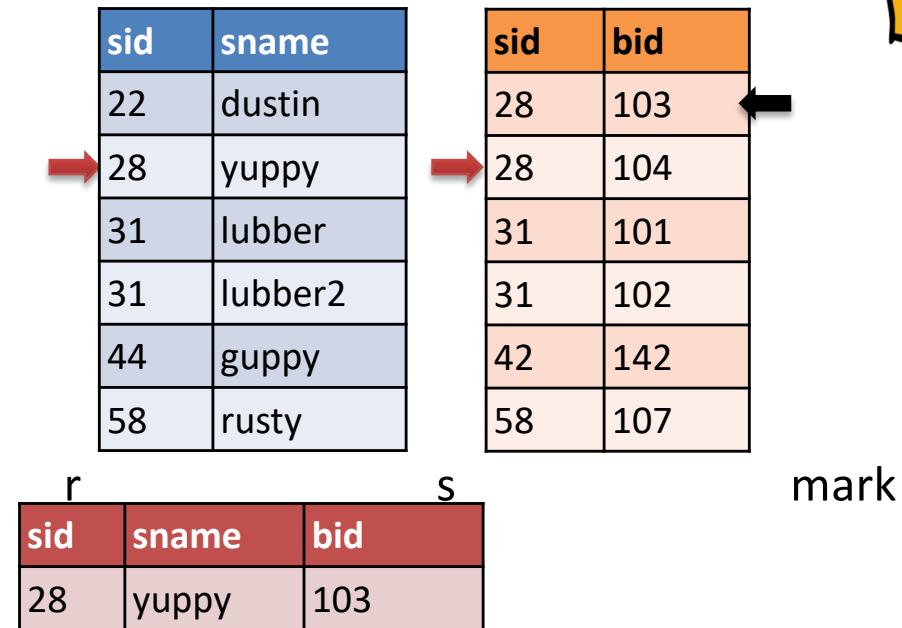
```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```



Sort-Merge Join, Part 11



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
    }
    return result
}
else {
    reset s to mark
    advance r
    mark = NULL
}
}
```



Sort-Merge Join, Part 12



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
    }
    return result
}
else {
    reset s to mark
    advance r
    mark = NULL
}
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

r	s	mark
sid	sname	bid
28	yuppy	103

Sort-Merge Join, Part 13



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

r		S		mark
sid	sname	sid	bid	
22	dustin	28	103	
28	yuppy	28	104	
31	lubber	31	101	
31	lubber2	31	102	
44	guppy	42	142	
58	rusty	58	107	

sid	sname	bid
28	yuppy	103
28	yuppy	104

Sort-Merge Join, Part 14



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

r	S	
sid	sname	bid
28	yuppy	103
28	yuppy	104

mark

Sort-Merge Join, Part 15



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

mark

r	sname	bid	s
sid	yuppy	103	
28	yuppy	104	

Sort-Merge Join, Part 16



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

r *S* mark

sid	sname	bid
28	yuppy	103
28	yuppy	104

Sort-Merge Join, Part 17



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

r		s		mark
sid	sname	sid	bid	
22	dustin	28	103	
28	yuppy	28	104	
31	lubber	31	101	
31	lubber2	31	102	
44	guppy	42	142	
58	rusty	58	107	

sid	sname	bid
28	yuppy	103
28	yuppy	104

Sort-Merge Join, Part 18



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

r

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

s

sid	sname	bid
28	yuppy	103
28	yuppy	104

mark

Sort-Merge Join, Part 19



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname		sid	bid
r	s	mark		
sid	sname	bid		
22	dustin		28	103
28	yuppy		28	104
31	lubber		31	101
31	lubber2		31	102
44	guppy		42	142
58	rusty		58	107

Sort-Merge Join, Part 20



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

The diagram illustrates the state of two sorted tables, *r* and *s*, during a sort-merge join. The table *r* (left) has columns sid and sname, with rows: 22 (dustin), 28 (yuppy), 31 (lubber), 31 (lubber2), 44 (guppy), and 58 (rusty). The table *s* (right) has columns sid and bid, with rows: 28 (103), 28 (104), 31 (101), 31 (102), 42 (142), and 58 (107). Red arrows point from the first row of each table to the variable 'mark' at the bottom right, indicating that the join process has just started or is about to begin.

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

r *s* mark

sid	sname	bid
28	yuppy	103
28	yuppy	104

Sort-Merge Join, Part 21



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

The diagram illustrates the state of two sorted tables, *r* and *s*, during a sort-merge join. Red arrows point from the *r* table to the *s* table, indicating the current range of values being compared. The *r* table has columns sid and sname. The *s* table has columns sid and bid. The *mark* variable is shown as a pointer to the current row in the *s* table.

r		s		mark
sid	sname	sid	bid	
22	dustin	28	103	
28	yuppy	28	104	
31	lubber	31	101	
31	lubber2	31	102	
44	guppy	42	142	
58	rusty	58	107	

sid	sname	sid	bid
28	yuppy	103	
28	yuppy	104	

Sort-Merge Join, Part 22



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

r	sname	s
sid		bid
28	yuppy	103
28	yuppy	104

Sort-Merge Join, Part 23



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

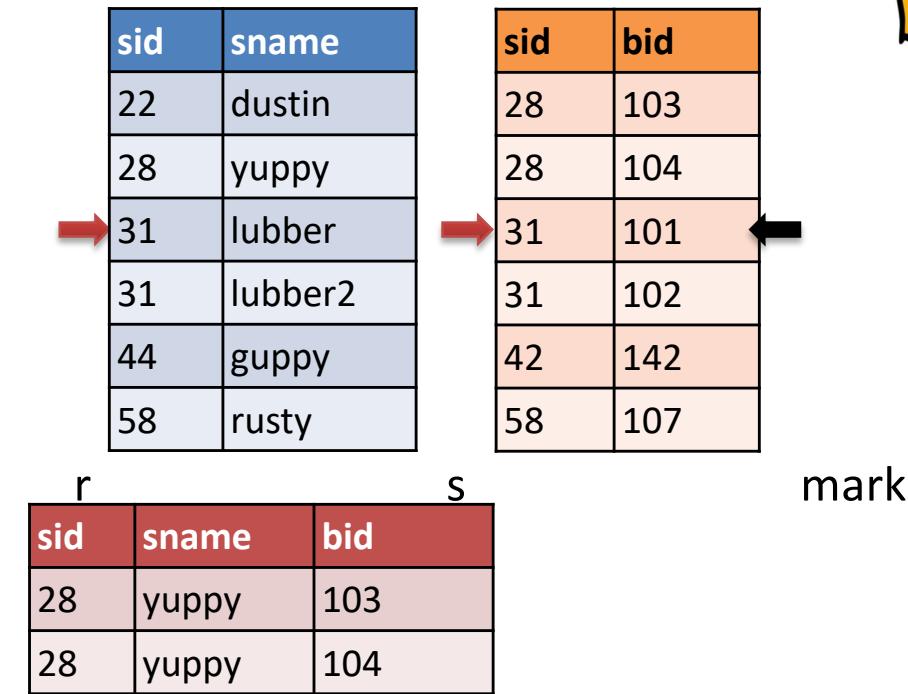
r	S	
sid	sname	bid
28	yuppy	103
28	yuppy	104

mark

Sort-Merge Join, Part 24



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
    }
    return result
}
else {
    reset s to mark
    advance r
    mark = NULL
}
}
```



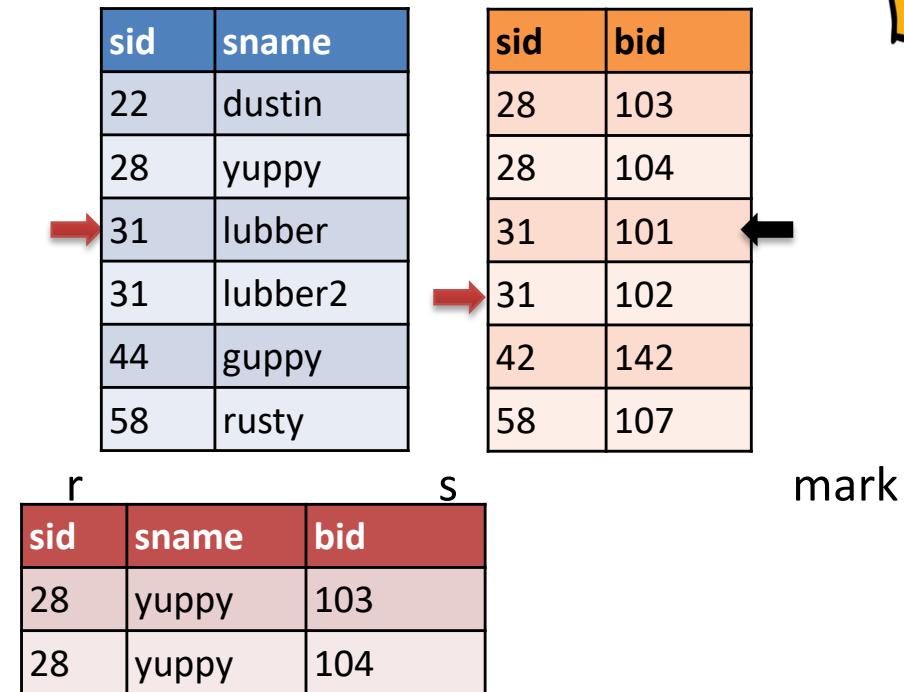
Sort-Merge Join, Part 25



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```



Sort-Merge Join, Part 26



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

r

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

s

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

mark

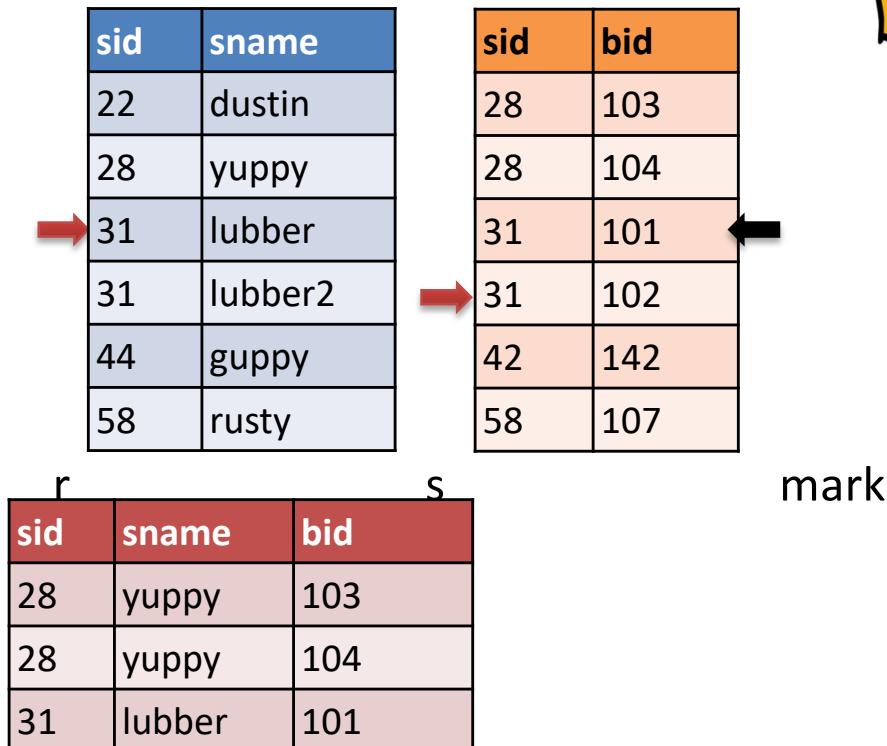
Sort-Merge Join, Part 27



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```



Sort-Merge Join, Part 28



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

r

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

s

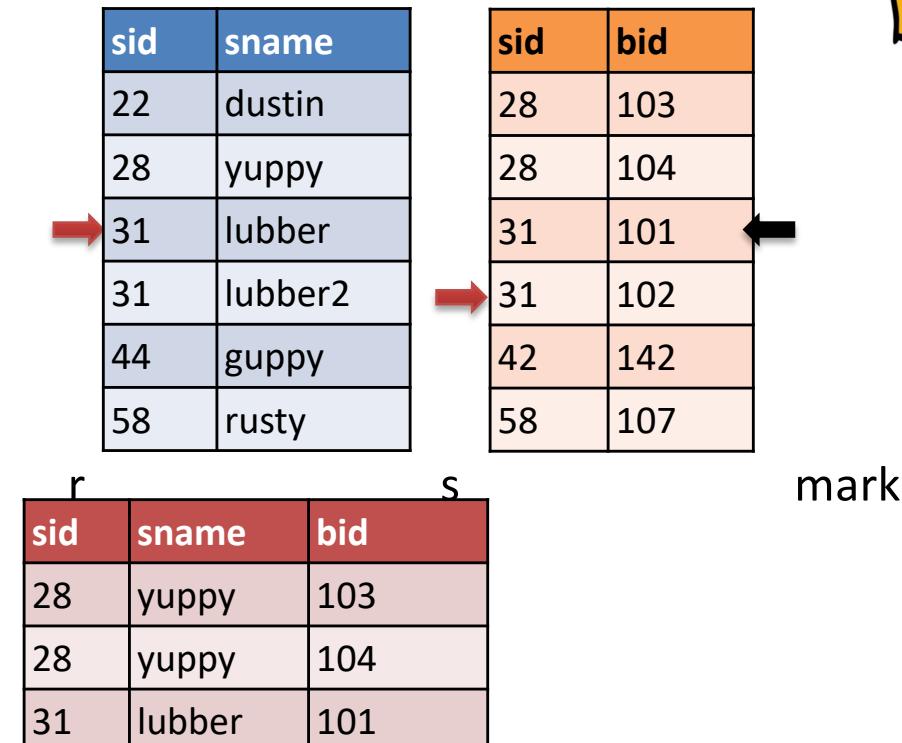
sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

mark

Sort-Merge Join, Part 29



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
    }
    return result
}
else {
    reset s to mark
    advance r
    mark = NULL
}
}
```



Sort-Merge Join, Part 30



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
    }
    return result
}
else {
    reset s to mark
    advance r
    mark = NULL
}
```

The diagram illustrates a sort-merge join between two sorted tables, *r* and *s*. The table *r* (left) has columns sid and sname, with data: 22 (dustin), 28 (yuppy), 31 (lubber), 31 (lubber2), 44 (guppy), 58 (rusty). The table *s* (right) has columns sid and bid, with data: 28 (103), 28 (104), 31 (101), 31 (102), 42 (142), 58 (107). Red arrows point from the last row of *r* to the first row of *s*, indicating the merge process. A black arrow points to the row in *s* with sid 31, which is highlighted in orange. The word "mark" is positioned to the right of the *s* table, with a horizontal line connecting it to the highlighted row in *s*.

	<i>r</i>	<i>s</i>	mark	
	sid	sname	sid	bid
1	22	dustin	28	103
2	28	yuppy	28	104
3	31	lubber	31	101
4	31	lubber2	31	102
5	44	guppy	42	142
6	58	rusty	58	107

	<i>r</i>	<i>s</i>		
	sid	sname	sid	bid
1	28	yuppy	28	103
2	28	yuppy	28	104
3	31	lubber	31	101

Sort-Merge Join, Part 31



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

r	sname	s
sid		bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102

Sort-Merge Join, Part 32



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

r	sid	sname	s	sid	bid	mark
	28	yuppy		28	103	
	28	yuppy		28	104	
	31	lubber		31	101	
	31	lubber		31	102	

Sort-Merge Join, Part 33



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

r	s	
sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102

Sort-Merge Join, Part 34



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of s
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

The diagram illustrates a relationship between two tables. On the left, a table with columns **sid** and **sname** contains rows for Dustin, Yuppy, Lubber, Lubber2, Guppy, and Rusty. On the right, a table with columns **sid** and **bid** contains rows for Dustin (with bid 103), Yuppy (with bid 104), Lubber (with bid 101), Lubber2 (with bid 102), Guppy (with bid 142), and Rusty (with bid 107). Red arrows point from the **sid** column of the left table to the **sid** column of the right table, indicating that each student ID in the left table corresponds to a specific row in the right table.

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

r	sname	s
sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102

Sort-Merge Join, Part 35



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

<i>sid</i>	<i>sname</i>
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

<i>sid</i>	<i>bid</i>
28	103
28	104
31	101
31	102
42	142
58	107

<i>r</i>	<i>sid</i>	<i>sname</i>	<i>bid</i>	<i>S</i>	mark
	28	yuppy	103		
	28	yuppy	104		
	31	lubber	101		
	31	lubber	102		

Sort-Merge Join, Part 36



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

r		s	mark
sid	sname	bid	
28	yuppy	103	
28	yuppy	104	
31	lubber	101	
31	lubber	102	

Sort-Merge Join, Part 37



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

r	sid	sname	bid	s	mark
	28	yuppy	103		
	28	yuppy	104		
	31	lubber	101		
	31	lubber	102		

Sort-Merge Join, Part 38



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

→

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

←

r	sname	s
sid	bid	
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102

Sort-Merge Join, Part 39



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

→

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

←

r	sname	s
sid	bid	
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102

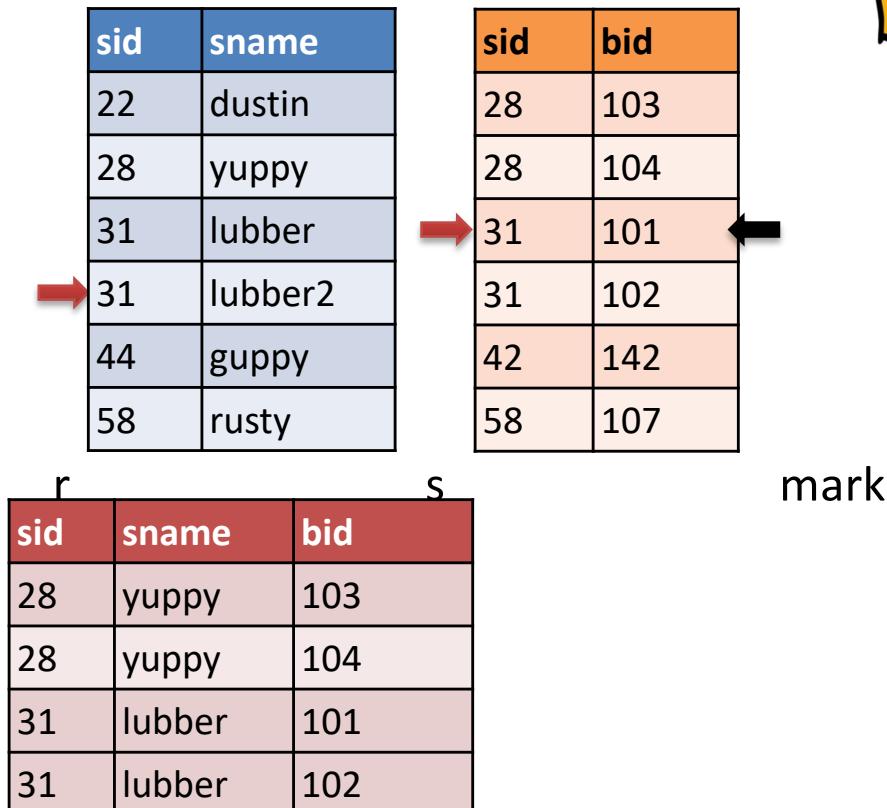
Sort-Merge Join, Part 40



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```



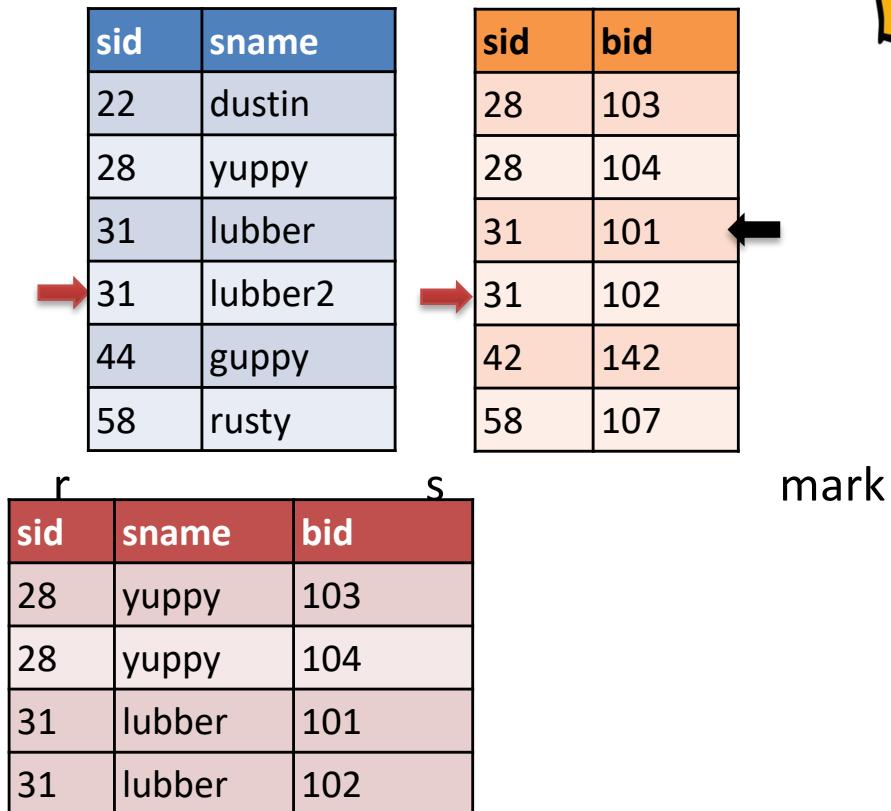
Sort-Merge Join, Part 41



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
    }
    return result
}
else {
    reset s to mark
    advance r
    mark = NULL
}
}

```



Sort-Merge Join, Part 42



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

r	sname	s	mark
sid	sid	sid	
28	yuppy	103	
28	yuppy	104	
31	lubber	101	
31	lubber	102	
31	lubber2	101	

Sort-Merge Join, Part 43



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

r	sid	sname	bid	S	mark
	28	yuppy	103		
	28	yuppy	104		
	31	lubber	101		
	31	lubber	102		
	31	lubber2	101		

Sort-Merge Join, Part 44



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

r	sname	s
sid		bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101

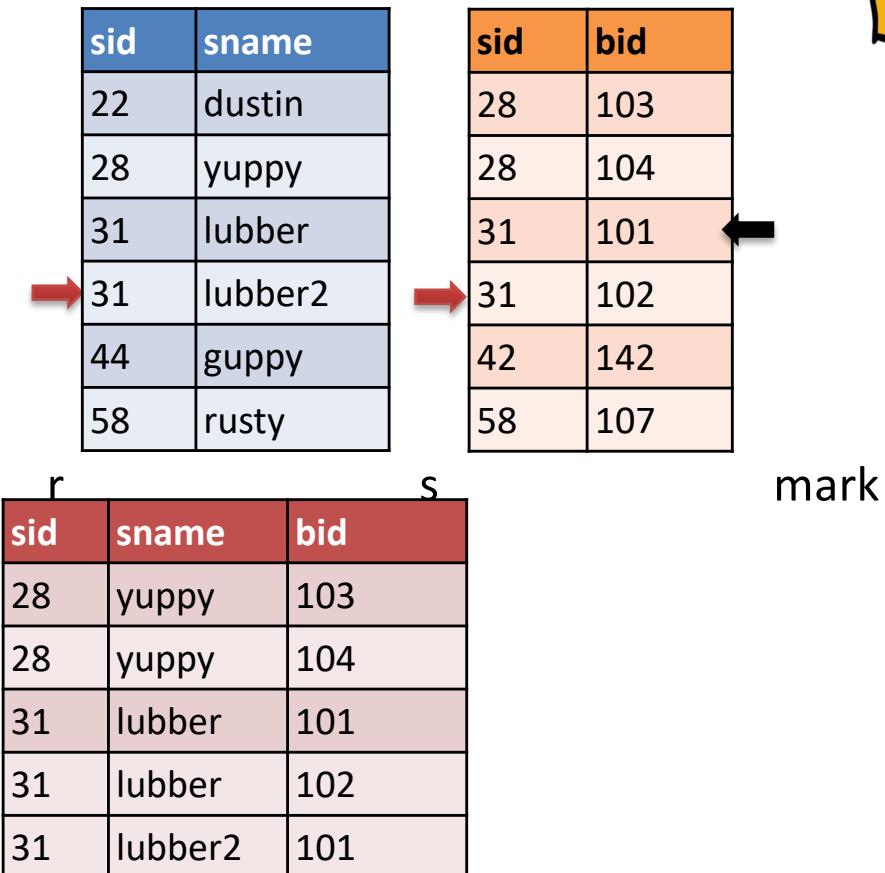
Sort-Merge Join, Part 45



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```



Sort-Merge Join, Part 46



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
    }
    return result
}
else {
    reset s to mark
    advance r
    mark = NULL
}
}

```

mark

<i>r</i>	<i>sname</i>	<i>bid</i>	<i>s</i>
22	dustin		
28	yuppy	103	28
31	lubber	104	31
31	lubber2	101	31
44	guppy	102	42
58	rusty	142	58
		107	

Sort-Merge Join, Part 47



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

<i>sid</i>	<i>sname</i>	<i>sid</i>	<i>bid</i>
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

<i>r</i>	<i>sid</i>	<i>sname</i>	<i>bid</i>	<i>s</i>	mark
	28	yuppy	103		
	28	yuppy	104		
	31	lubber	101		
	31	lubber	102		
	31	lubber2	101		
	31	lubber2	102		

Sort-Merge Join, Part 48



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join, Part 49



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

r	sname	s	mark
sid	bid		
28	yuppy	103	
28	yuppy	104	
31	lubber	101	
31	lubber	102	
31	lubber2	101	
31	lubber2	102	

Sort-Merge Join, Part 50



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

r	sid	sname	s	bid	mark
	28	yuppy	28	103	
	28	yuppy	28	104	
	31	lubber	31	101	
	31	lubber	31	102	
	31	lubber2	31	101	
	31	lubber2	31	102	

Sort-Merge Join, Part 51



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

r	sname	S
sid		bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join, Part 52



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

r *S* mark

sid	sname	bid	
22	dustin		
28	yuppy		
31	lubber		
31	lubber2		
44	guppy		
58	rusty		

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join, Part 53



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

mark

	r		S
	sid	sname	bid
	22	dustin	
	28	yuppy	
	31	lubber	
	31	lubber2	
→	44	guppy	
	58	rusty	

	sid	sname	bid
	28	yuppy	103
	28	yuppy	104
	31	lubber	101
	31	lubber	102
	31	lubber2	101
	31	lubber2	102

Sort-Merge Join, Part 54



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

r	sname	S	mark
sid	bid	sid	bid
28	yuppy	103	
28	yuppy	104	
31	lubber	101	
31	lubber	102	
31	lubber2	101	
31	lubber2	102	

Sort-Merge Join, Part 55



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of s
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

sid	sname	sid	bid
22	dustin	28	103
28	yuppy	28	104
31	lubber	31	101
31	lubber2	31	102
44	guppy	42	142
58	rusty	58	107

sid	sname	bid	mark
28	yuppy	103	
28	yuppy	104	
31	lubber	101	
31	lubber	102	
31	lubber2	101	
31	lubber2	102	

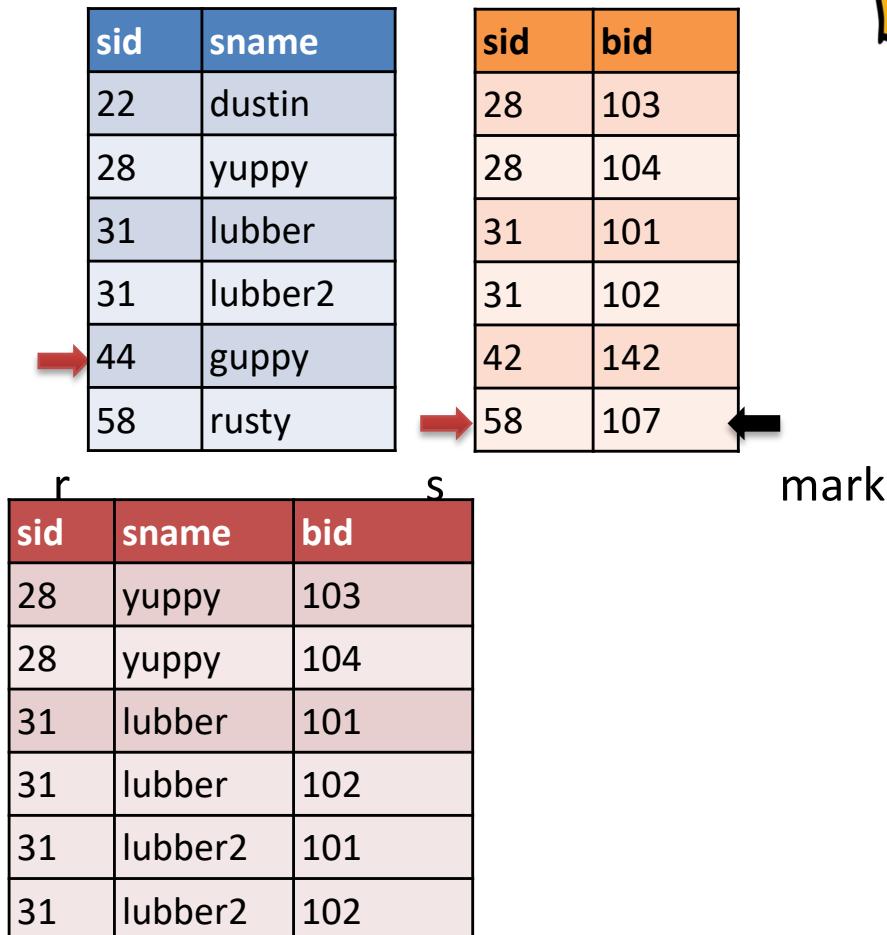
Sort-Merge Join, Part 56



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```



Sort-Merge Join, Part 57



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

r *s* mark

sid	sname	
22	dustin	
28	yuppy	
31	lubber	
31	lubber2	
44	guppy	
58	rusty	

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join, Part 57



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

r *s* mark

sid	sname	
22	dustin	
28	yuppy	
31	lubber	
31	lubber2	
44	guppy	
58	rusty	

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

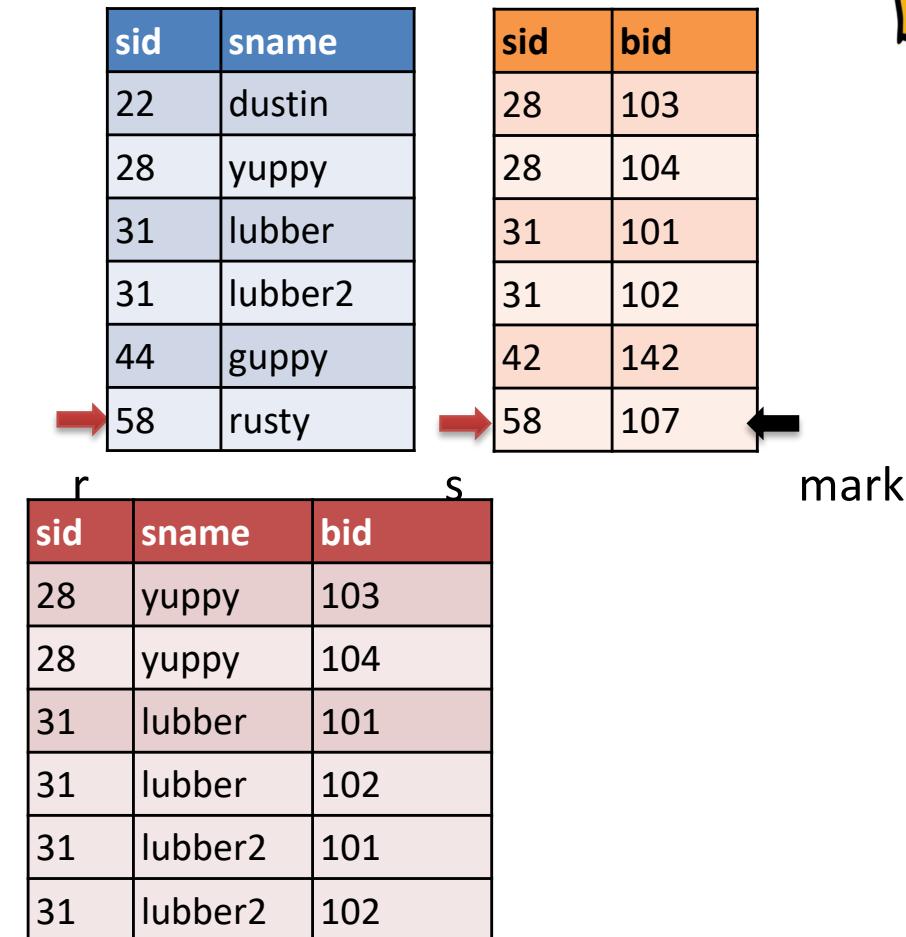
sid	sname	bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join, Part 58

```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```



Sort-Merge Join, Part 59



```
do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}
```

The diagram illustrates the state of two sorted tables, *r* and *s*, during a sort-merge join. Red arrows point from the bottom of table *r* and the top of table *s* to a central column labeled "mark".

Table r:

sid	sname
22	dustin
28	yuppy
31	lubber
31	lubber2
44	guppy
58	rusty

Table s:

sid	bid
28	103
28	104
31	101
31	102
42	142
58	107

Join Result:

r sid	r sname	s bid
28	yuppy	103
28	yuppy	104
31	lubber	101
31	lubber	102
31	lubber2	101
31	lubber2	102

Sort-Merge Join, Part 60



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```

mark

<i>sid</i>	<i>sname</i>	<i>bid</i>
22	dustin	
28	yuppy	
31	lubber	
31	lubber2	
44	guppy	
58	rusty	

→ →

<i>r</i>	<i>sid</i>	<i>sname</i>	<i>bid</i>	<i>s</i>
	28	yuppy	103	
	28	yuppy	104	
	31	lubber	101	
	31	lubber	102	
	31	lubber2	101	
	31	lubber2	102	

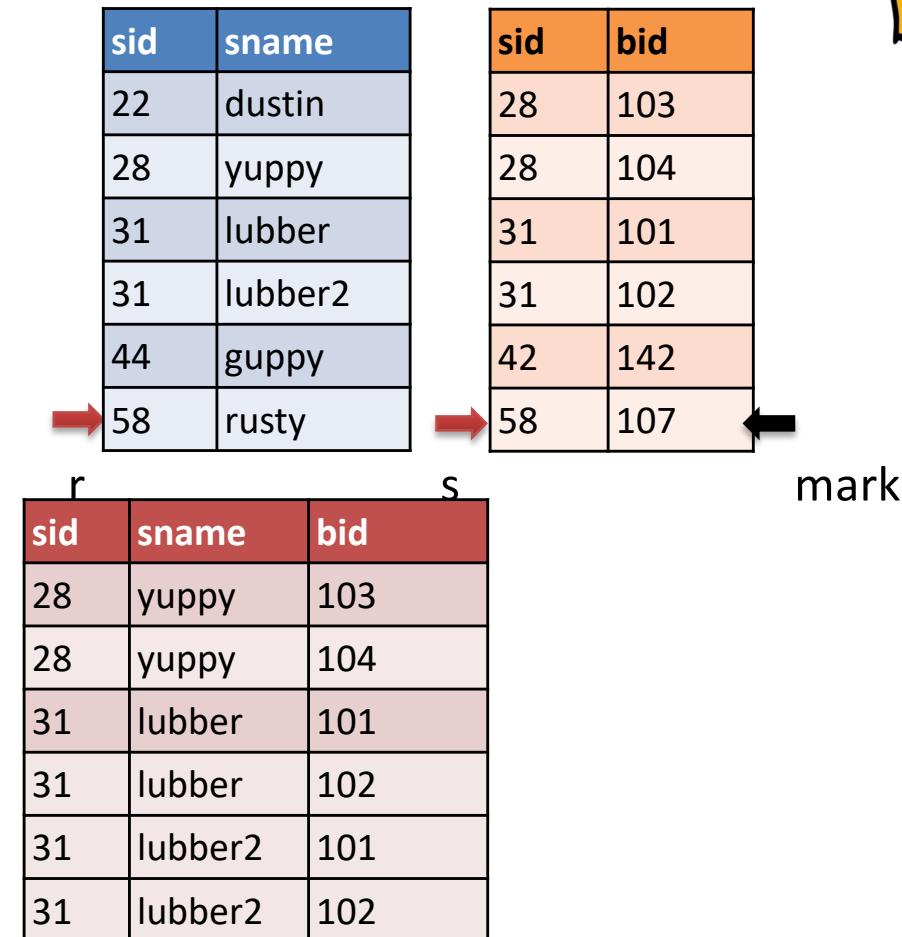
Sort-Merge Join, Part 61



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```



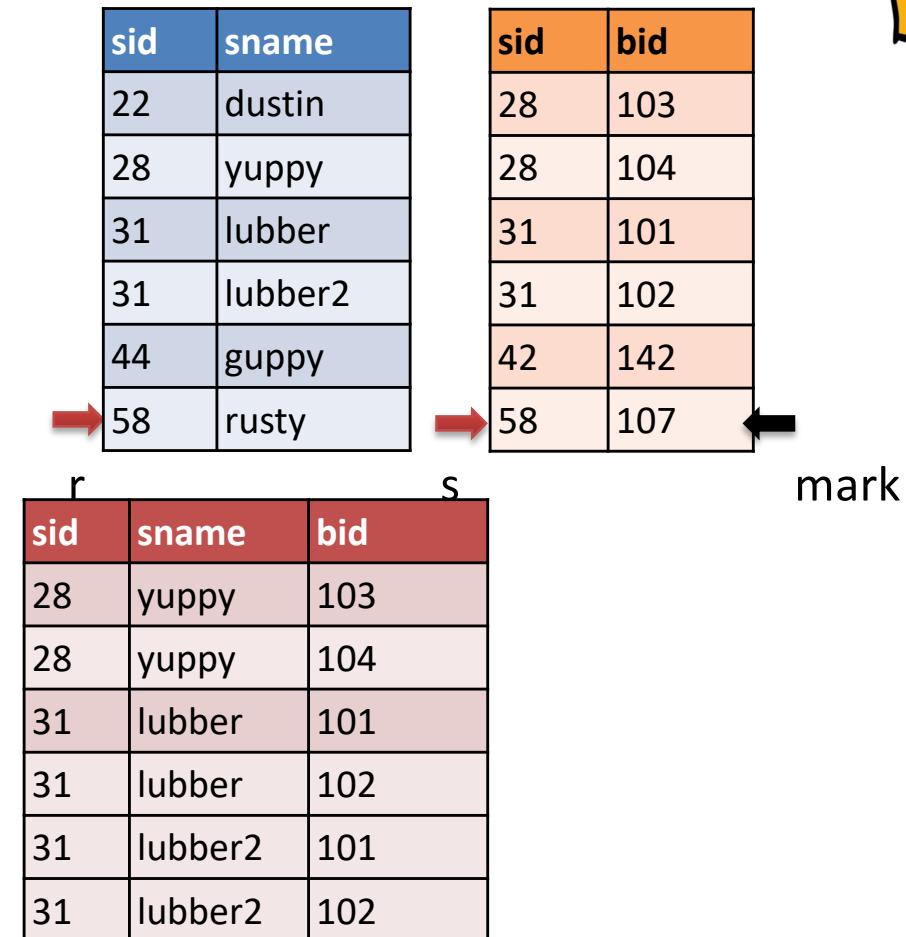
Sort-Merge Join, Part 62



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```



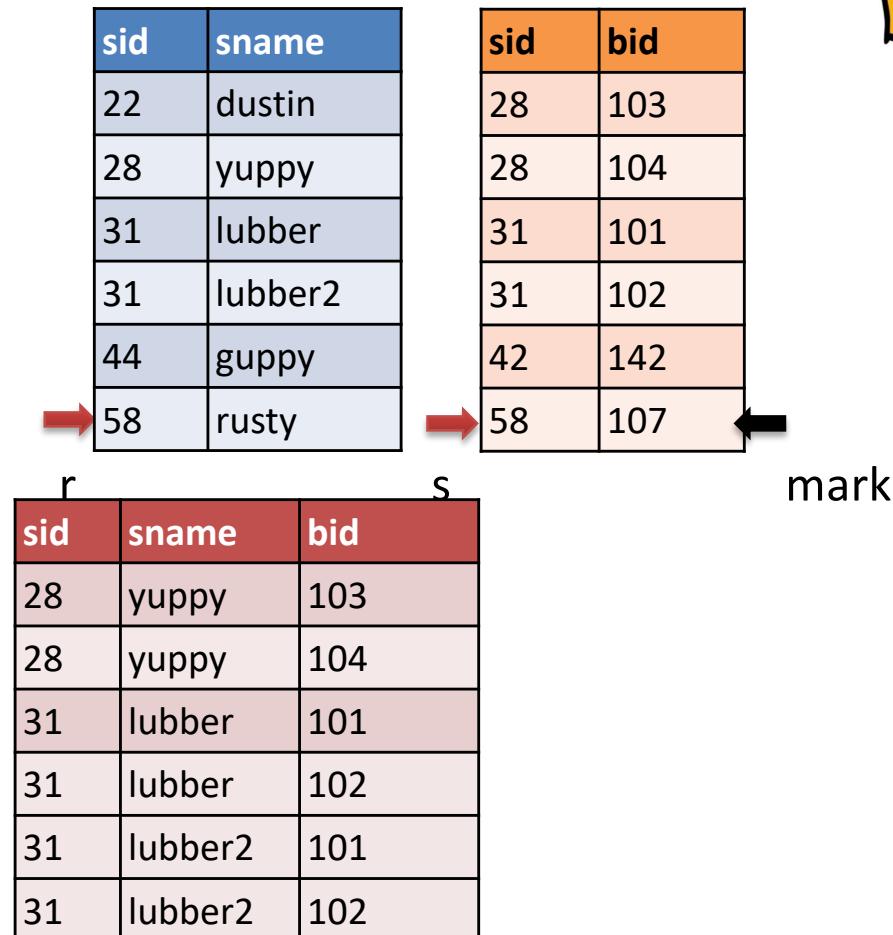
Sort-Merge Join, Part 63



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
    }
    return result
}
else {
    reset s to mark
    advance r
    mark = NULL
}
}

```



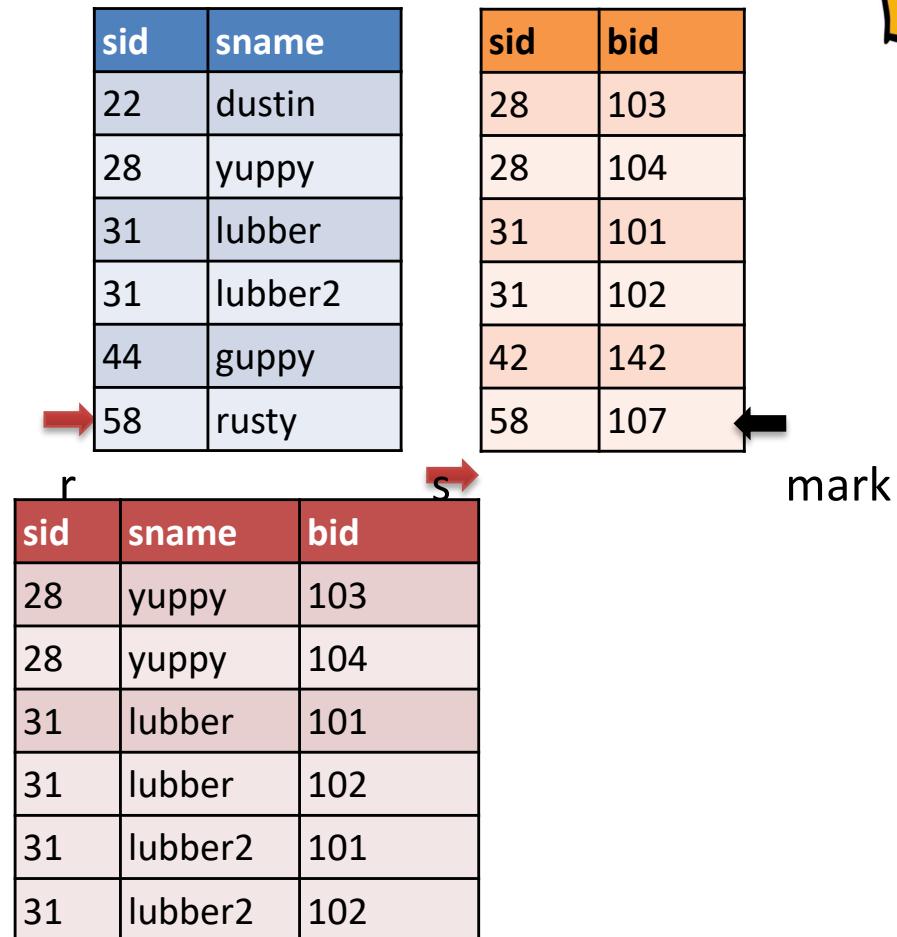
Sort-Merge Join, Part 64



```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
    }
    return result
}
else {
    reset s to mark
    advance r
    mark = NULL
}
}

```



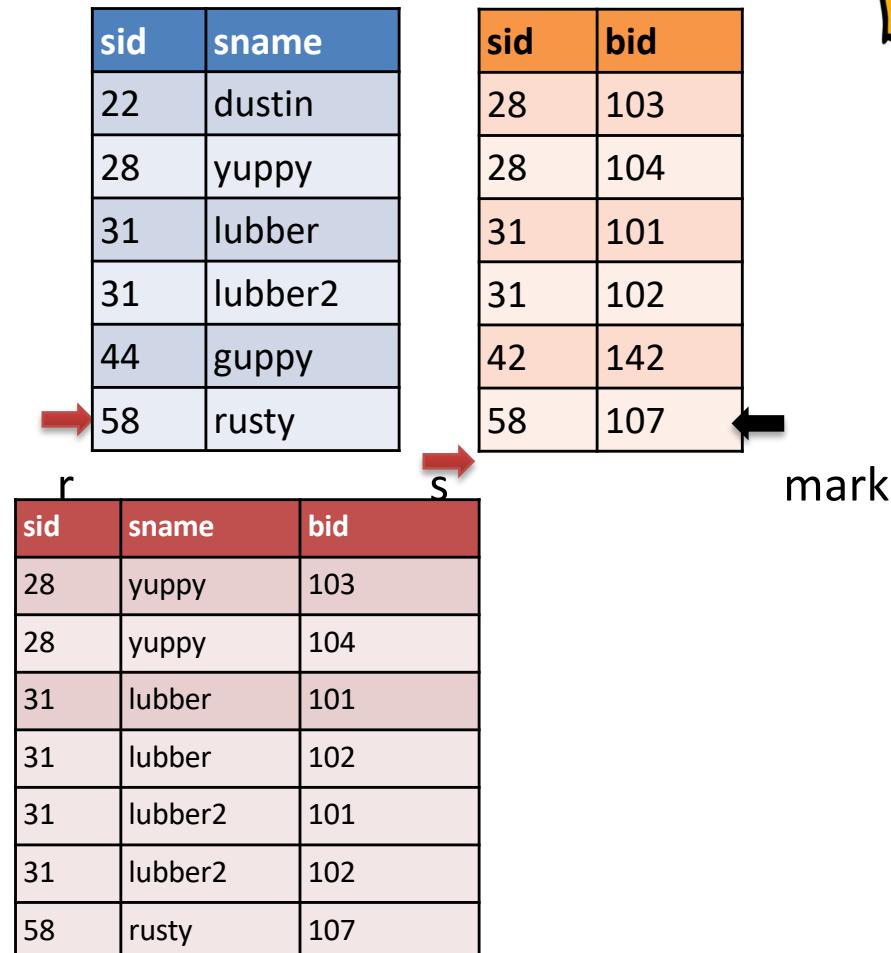
Sort-Merge Join, Part 65



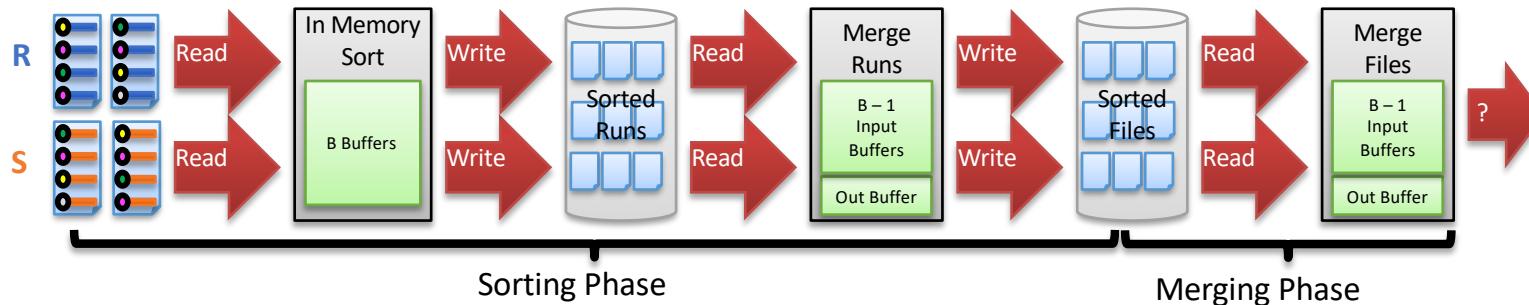
```

do {
    if (!mark) {
        while (r < s) { advance r }
        while (r > s) { advance s }
        // mark start of "block" of S
        mark = s
    }
    if (r == s) {
        result = <r, s>
        advance s
        return result
    }
    else {
        reset s to mark
        advance r
        mark = NULL
    }
}

```



Cost of Sort-Merge Join



- Best case cost: $\text{Sort } R + \text{Sort } S + ([R] + [S])$
 - But in worst case, last term could be $|R| * |S|$ (very unlikely!)
 - Q: what is worst case?
- Question: To sort both R and S in two passes each, how big does the buffer have to be?
- Suppose buffer $B > \sqrt{(\max([R], [S]))}$
 - Both R and S can be sorted in 2 passes
 - Cost is then $4 * 1000 + 4 * 500 + (1000 + 500) = 7500$

$$\begin{aligned} [R] &= 1000, p_R = 100, |R| = 100,000 \\ [S] &= 500, p_S = 80, |S| = 40,000 \end{aligned}$$

Alternative: Join First, Sort Later



```
SELECT sid, bid, sname, rname  
FROM R, S  
WHERE R.sid = S.sid  
ORDER BY sid
```

$[R]=1000, p_R=100, |R| = 100,000$
 $[S]=500, p_S=80, |S| = 40,000$
 $B = 102$

- Reserves (*sid*: int, *bid*: int, *day*: date, *rname*: string)
- Sailors (*sid*: int, *sname*: string, *rating*: int, *age*: real)
- Special case: every reservation matches exactly one sailor
 - Output has $|R|$ tuples
- Block NLJ
 - Join: $[S] + ([S]/(B-2))^*[R]$
 - Sort: ?

Join First, Sort Later Part 2

```
SELECT sid, bid, sname, rname  
FROM R, S
```

```
WHERE R.sid = S.sid
```

```
ORDER BY sid
```

[R]=1000, p_R=100, |R| = 100,000

[S]=500, p_S=80, |S| = 40,000

B = 102

- Reserves (sid: int, bid: int, day: date, rname: string)
- Sailors (sid: int, sname: string, rating: int, age: real)
- Special case: every reservation matches exactly one sailor
 - Output has |R| tuples
- Block NLJ
 - Join: [S] + ([S]/(B-2))*[R] = 5500
 - Sort: 4 * [R] (2 passes are enough) = 4000
 - Total: 5500 + 4000 = 9500



Sort First, Join Later

```
SELECT sid, bid, sname, rname  
FROM R, S  
WHERE R.sid = S.sid  
ORDER BY sid
```



$[R]=1000, p_R=100, |R| = 100,000$
 $[S]=500, p_S=80, |S| = 40,000$
 $B = 102$

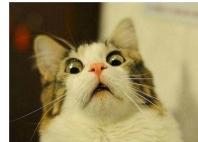
- **Reserves** (sid: int, bid: int, day: date, rname: string)
- **Sailors** (sid: int, sname: string, rating: int, age: real)
- Special case: every reservation matches exactly one sailor
 - Output has $|R|$ tuples

Sort R: $2*[R]^*(2) = 4000$

Sort S: $2*[S]^*(2) = 2000$

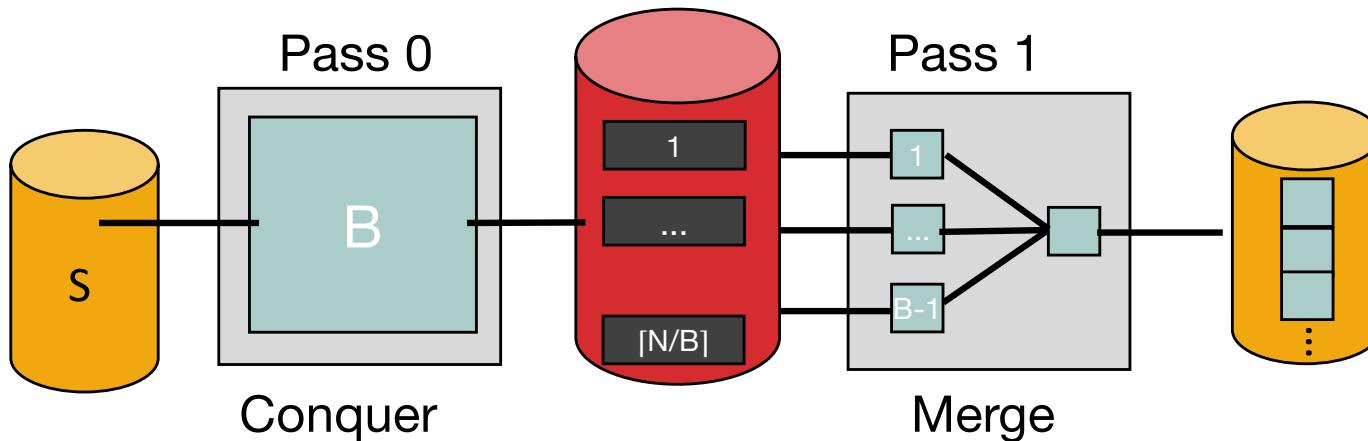
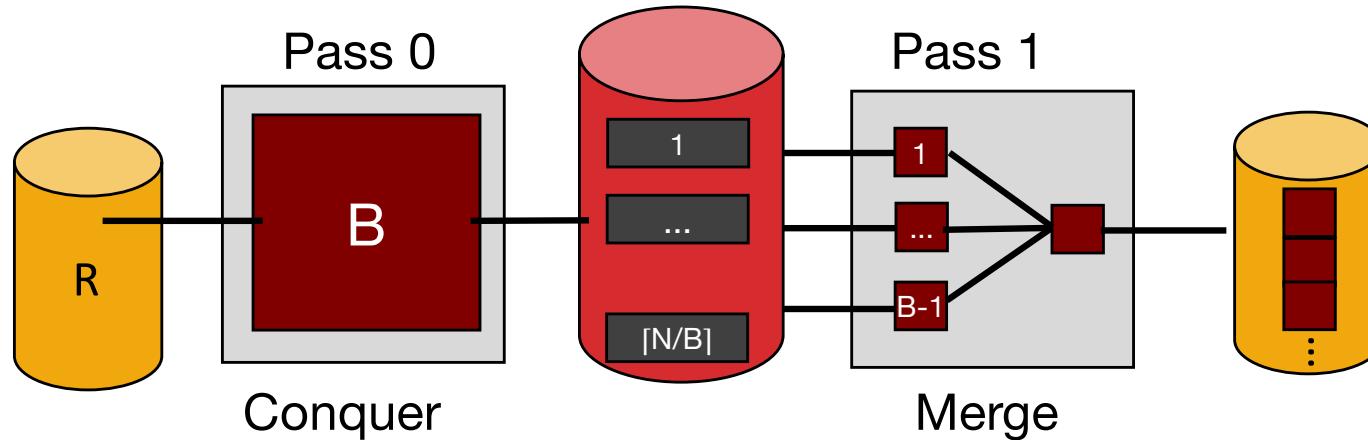
$R + S = 1500$

Total = 7500

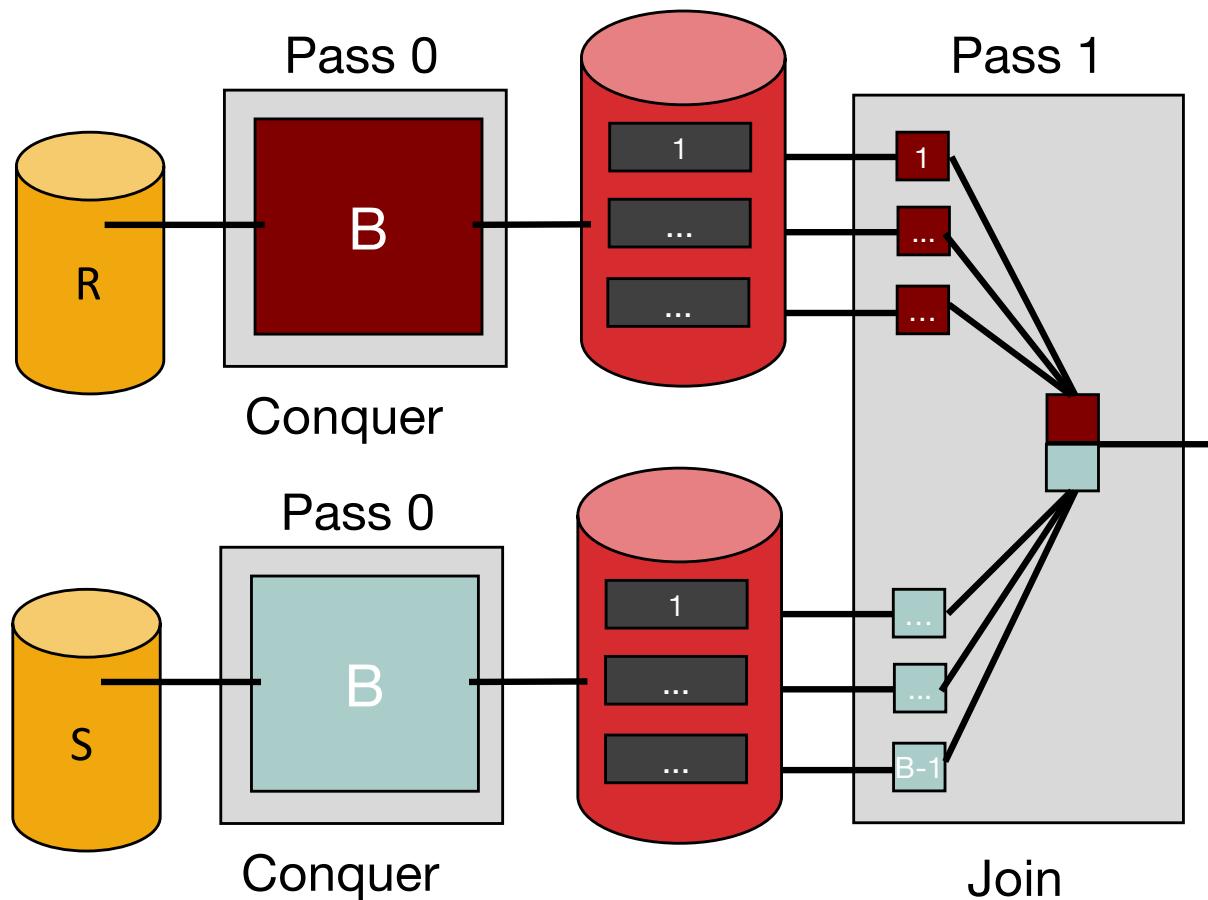


Operator order matters!

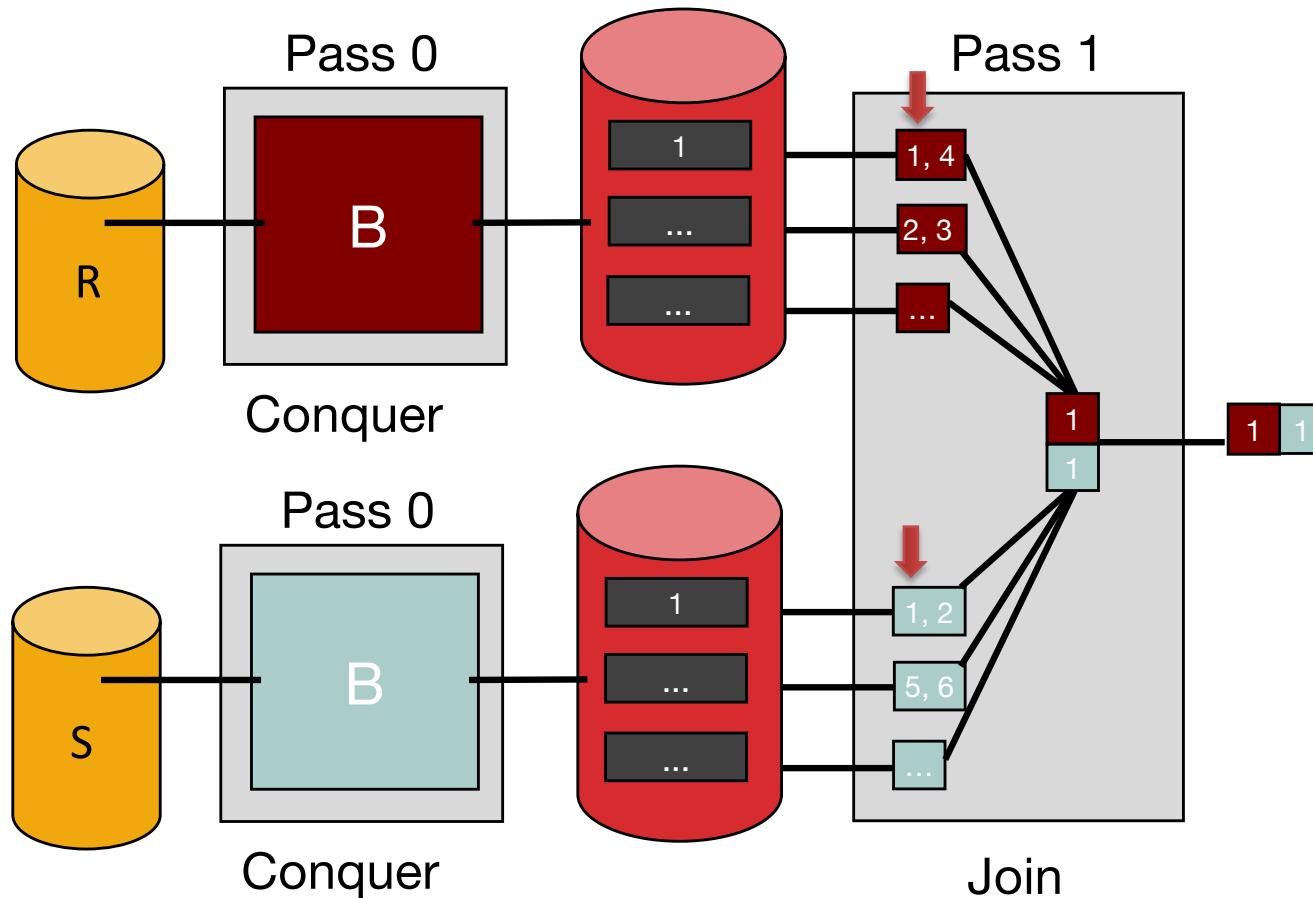
Recall: 2-Pass External Merge Sort



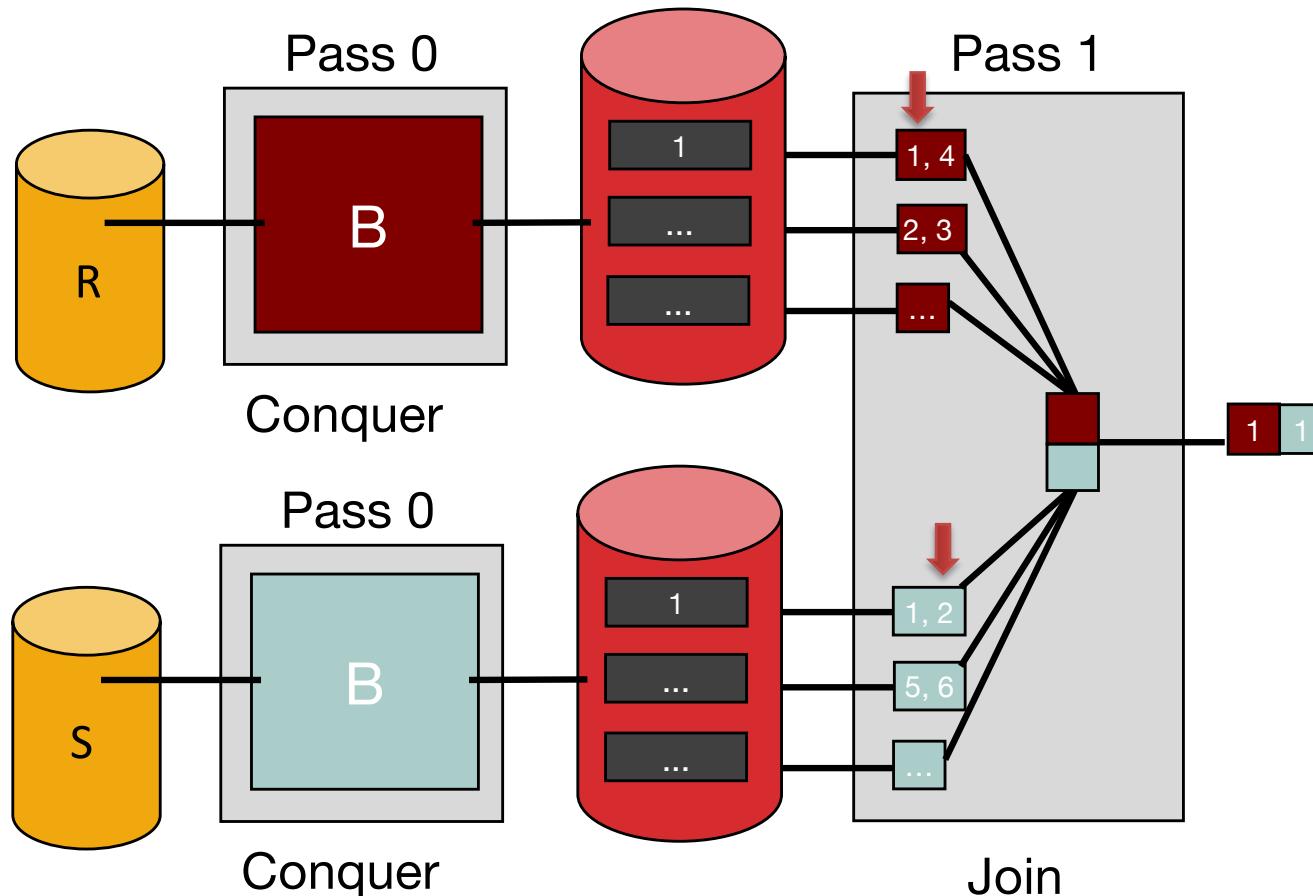
Combining Merge Sort and Join



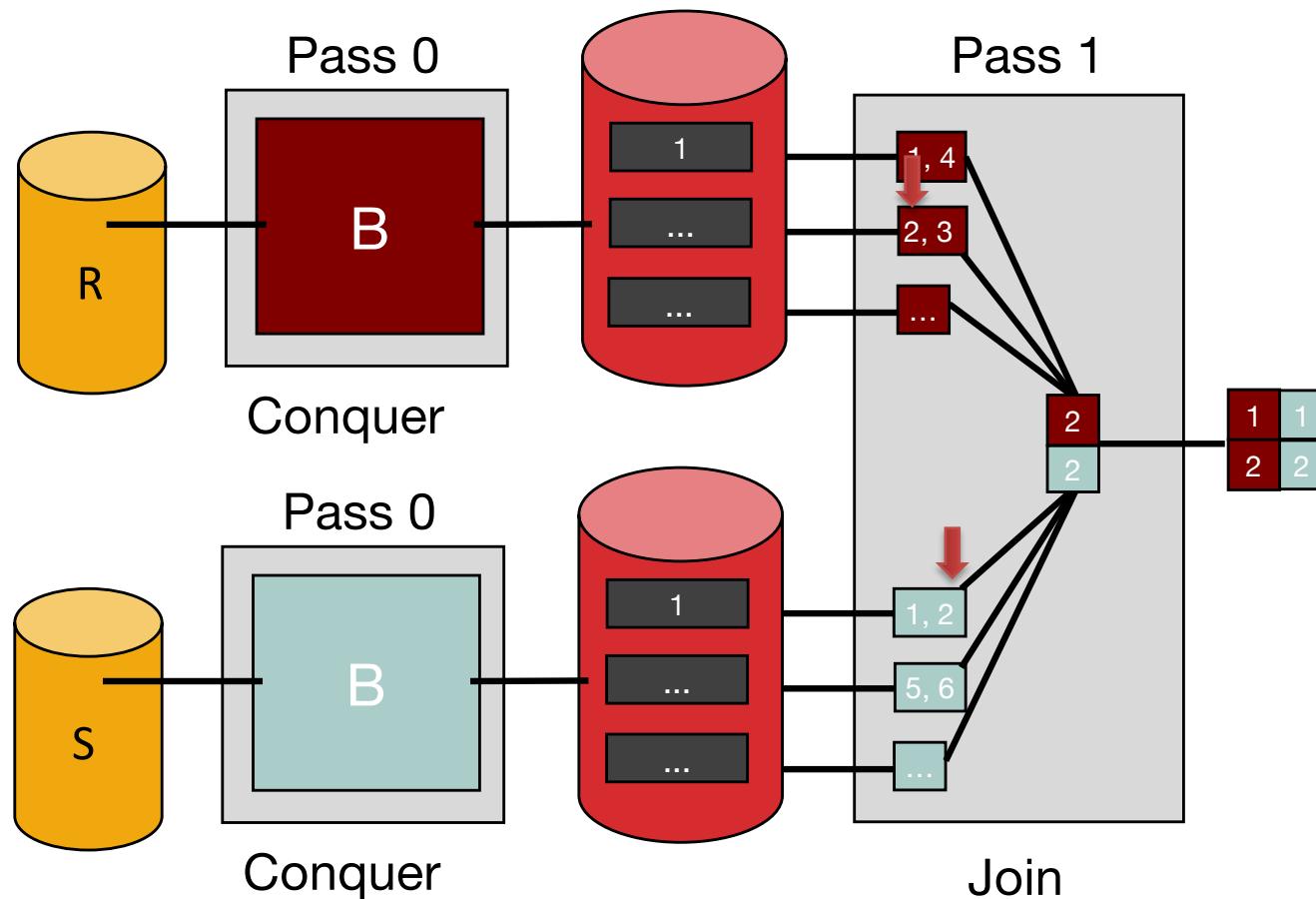
Combining Merge Sort and Join



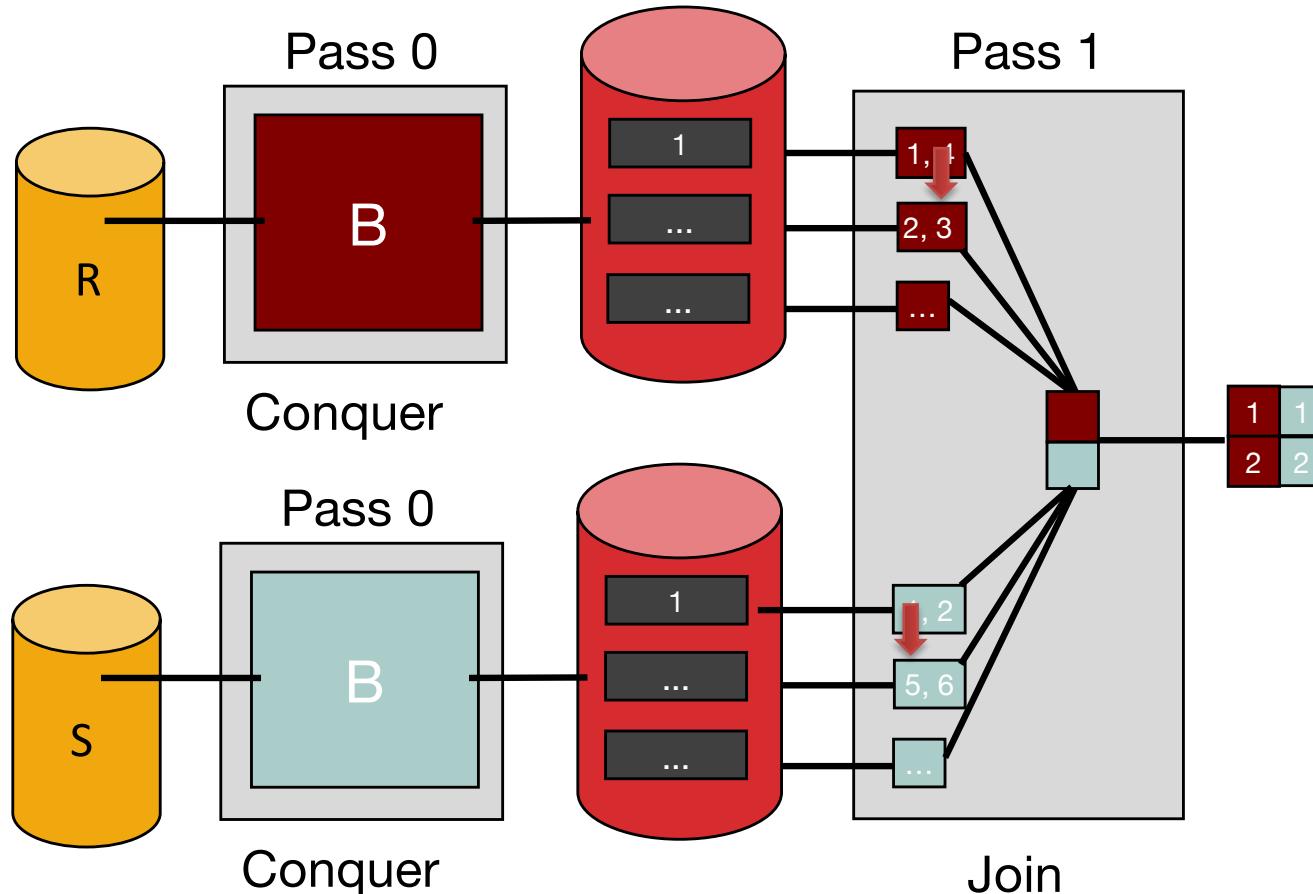
Combining Merge Sort and Join



Combining Merge Sort and Join



Combining Merge Sort and Join



- Need enough buffers for 1 page from each run in R and S in the last merge pass
- 2-pass Cost = $3*[R] + 3*[S] = 3000+1500 = 4500$
Even less than sort-merge join!
- In general, we need # runs in last merge pass for R + # runs in last merge pass for S $\leq B - 1$

Summary: Join algorithms



- Page / Block nested loop join
 - Order of relations matters!
- Index nested loop join
 - Need index built on at least one of the inputs
- Sort-merge join
 - Similar to external sort algorithm
 - Output is sorted