



# Join Operators

R&G 14.4





# Schema 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)
  - $[S]=500$ ,  $p_S=80$ ,  $|S| = 40,000$



# Simple Nested Loops Join

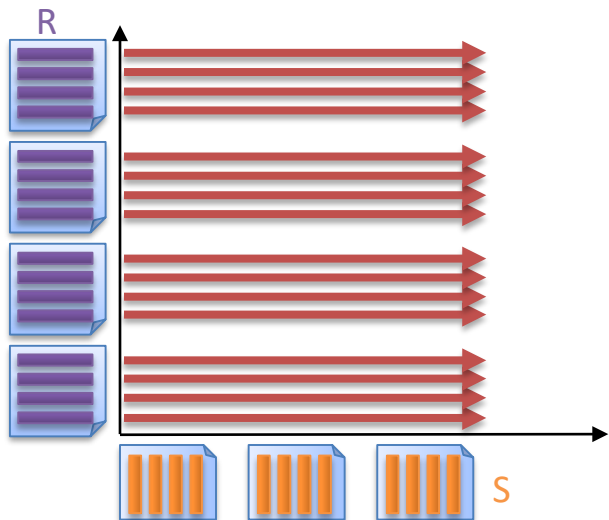


foreach **record**  $r$  in  $R$  do

  foreach **record**  $s$  in  $S$  do

    if  $\theta(r_i, s_j)$  then add  $\langle r_i, s_j \rangle$  to result buffer

*Note: for simplicity we do not present iterator implementations for the join algorithms.*



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

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

Cost:

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

$$= 50,001,000$$



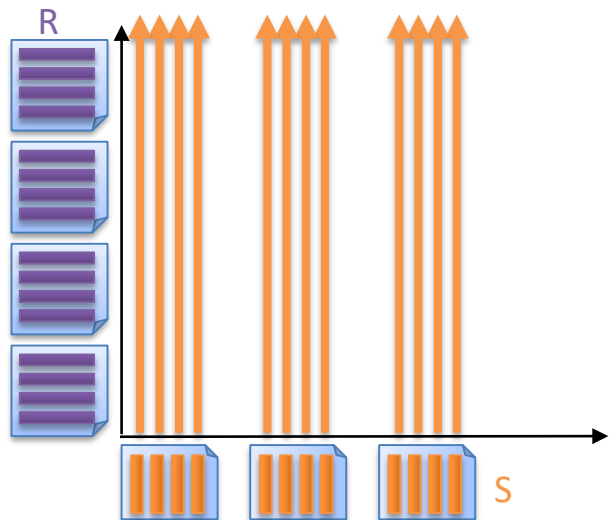
# Changing the Join Order



foreach **record**  $s$  in  $S$  do

  foreach **record**  $r$  in  $R$  do

    if  $\theta(r_i, s_j)$  then add  $\langle r_i, s_j \rangle$  to result buffer



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

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

Cost:

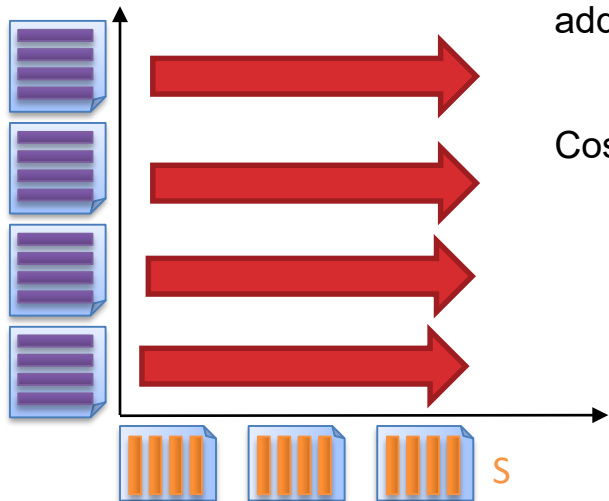
$$\begin{aligned} & [S] + |S| [R] \\ &= 40,000,500 \\ &\text{vs. } 50,001,000 \end{aligned}$$



# Page Nested Loop Join



```
for each rpage in R:
  for each spage in S:
    for each rtuple in rpage:
      for each stuple in spage:
        if join_condition(rtuple, stuple):
          add <rtuple, stuple> to result buffer
```

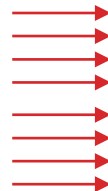


$$\begin{aligned}\text{Cost} &= [R] + ([R] * [S]) \\ &= 501,000\end{aligned}$$



“Chunk”

# ~~“Block”~~ Nested Loop Join

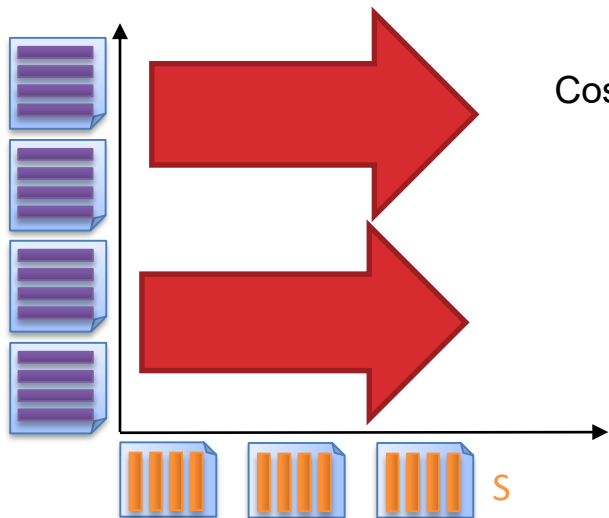


for each rchunk of B-2 pages of R:

for each spage of S:

for all matching tuples in spage and rchunk:

add <rtuple, stuple> to result buffer



$$\begin{aligned}
 \text{Cost} &= [R] + \lceil [R]/(B-2) \rceil * [S] \\
 &= 1000 + \lceil 1000/(B-2) \rceil * 500 \\
 &= 6,000 \text{ for } B=102 \text{ (~100x better than Page NL!)}
 \end{aligned}$$



# Index Nested Loops Join



foreach **tuple**  $r$  in  $R$  do

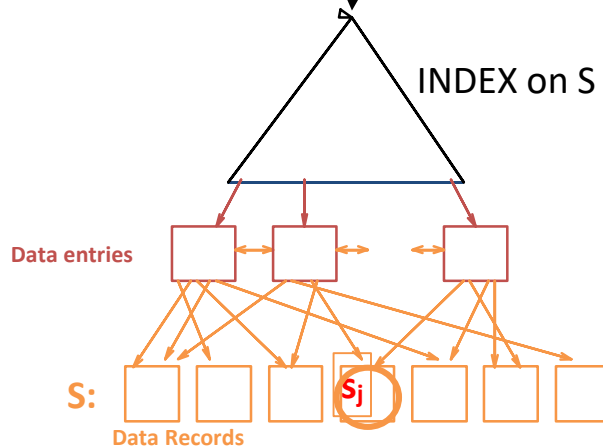
    foreach **tuple**  $s$  in  $S$  where  $r_i == s_j$  do

        add  $\langle r_i, s_j \rangle$  to result buffer

$R$



lookup( $r_i$ )





# Index Nested Loops Join Cost



```
foreach tuple r in R do
    foreach tuple s in S where ri == sj do
        add <ri, sj> 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: up to 1 I/O per matching S tuple





# Index Nested Loops Join Cost, Part 2



- **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)
  - $[S]=500$ ,  $p_S=80$ ,  $|S| = 40,000$
  - Index on sid



# Index Nested Loops Join Cost, Part 3



- Unclustered Cost(R,S) =  $[R] + |R| * (\text{Search} + \# \text{ matching tuples})$
- Clustered Cost(R,S):  $[R] + |R| * (\text{Search} + \# \text{ of matching pages})$
- Here, sid is the primary key for Sailors, so there is exactly one matching sailor for each tuple in R
- Unclustered B+-Tree height 2 (3 I/Os from root to leaf):
  - $R \bowtie S$ :  $1000 + (100,000) * (3 + 1) = 401,000$
- Clustered B+-tree height 2 (3 I/Os from root to leaf)
  - $R \bowtie S$ :  $1000 + (100,000) * (3 + 1) = 401,000$



# Sort-Merge Join



- Requires equality predicate:
  - Equi-Joins & Natural Joins
- Two Stages:
  - Sort tuples in R and S by join key
    - All tuples with same key in consecutive order
    - Input might already be sorted ... why?
  - Join Pass: Merge-scan the sorted partitions and emit tuples that match



# 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	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, 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



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





# 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		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, 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  
  }  
}
```



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 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
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 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



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





# 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	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, 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  
  }  
}
```

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, 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  
  }  
}
```

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 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  
  }  
}
```

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, 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  
  }  
}
```

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, 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



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


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  
  }  
}
```



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








# 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


sid	sname	bid
28	yuppy	103
28	yuppy	104





# 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



sid	sname	bid
28	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		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 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  
  }  
}
```

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  
  }  
}
```

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 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
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 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  
  }  
}
```

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 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  
  }  
}
```

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 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

sid	sname	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	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 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
  }
}
```

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





# 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
}
}
```

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





# 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  
  }  
}
```

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, 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  
  }  
}
```

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, 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  
  }  
}
```

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, 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  
  }  
}
```

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







# 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  
  }  
}
```

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, 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
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, 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
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, 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
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, 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  
  }  
}
```

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, 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  
  }  
}
```

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, 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

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, 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

sid	sname	bid
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		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, 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		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, 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
  }
}
```

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, 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  
  }  
}
```

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, 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
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





# 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	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, 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

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





# 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
  }
}
```

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, 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
}
}
```

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



# 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  
  }  
}
```

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 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

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 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

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 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

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 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  
  }  
}
```

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 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  
  }  
}
```

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 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

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 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
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 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  
  }  
}
```

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
  }
}
```

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
  }
}
```

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
  }
}
```

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, 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  
  }  
}
```

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, 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  
  }  
}
```

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, 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  
  }  
}
```

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, 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  
  }  
}
```

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, 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
  }
}
```

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, 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  
}  
}
```

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 65



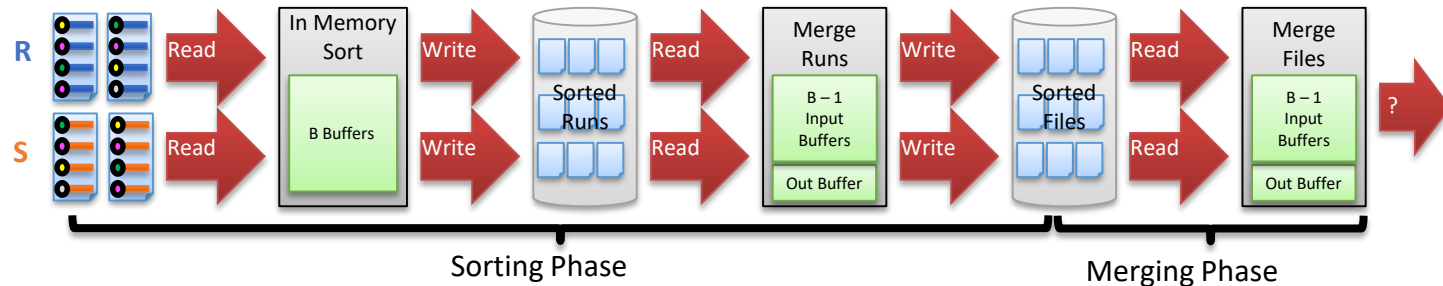
```
do {  
  if (!mark) {  
    while (r < s) { advance r }  
    while (r > s) { advance s }  
    // mark start of "block" of S  
    mark = s  
  }  
  if (r == s) {  
    result = <r, s>  
    advance s  
    return result  
  }  
  else {  
    reset s to mark  
    advance r  
    mark = NULL 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

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
58	rusty	107



# Cost of Sort-Merge Join



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



# 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)
- 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)
- Block NLJ
  - Join:  $[S] + ([S]/(B-2))*[R] = 5,500$
  - Sort:  $2 * [R] * (1 + \log_{B-1} [R]/B) = 4000$





# 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)

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

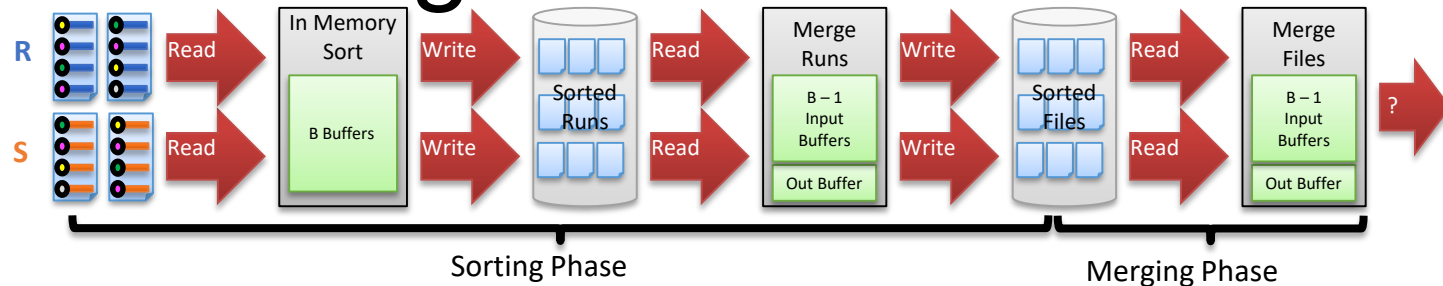
Sort S:  $2*[S]*(2) = 2000$

$R + S = 1500$

Total = 7500



# Sort-Merge Refinement



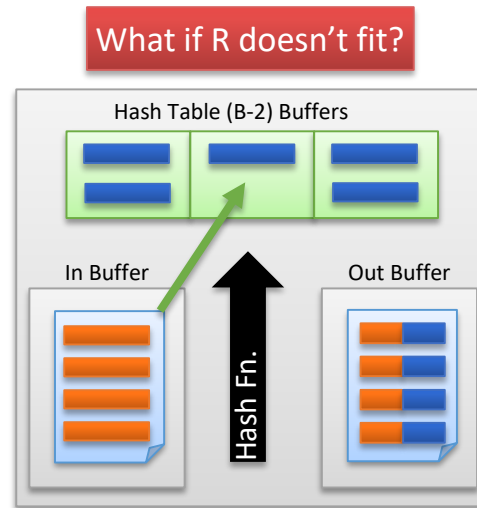
- An important refinement combines last pass of merge-sort with join pass
  - Given enough buffers to accommodate all runs in R and S on the penultimate (second-to-last) pass of sorting
  - Example for 2-pass SMJ (join during the final merging pass of sort)
    - Read R and write out sorted runs (pass 0)
    - Read S and write out sorted runs (pass 0)
    - Merge R-runs and S-runs, while finding  $R \bowtie S$  matches
  - 2-pass Cost =  $3*[R] + 3*[S] = 3000 + 1500 = 4500$



# Naïve in Memory Hash Join



- Requires equality predicate:
  - Works for Equi-Joins & Natural Joins
- Assume  $R$  is smaller relation
  - Require  $R$  to fit in memory
- Simple algorithm:
  - Load all  $R$  into hash table
  - Scan  $S$  and probe  $R$
- Memory requirements?
  - $R < (B-2) * \text{hash\_fill}$





# Properties that help



- $\sigma_{\text{sid}=4 \vee \text{sid}=6} (R \bowtie_{\text{sid}} S) = \sigma_{\text{sid}=4} (R \bowtie_{\text{sid}} S) \cup \sigma_{\text{sid}=6} (R \bowtie_{\text{sid}} S)$
- Can Decompose Into Smaller “Partial Joins”
- $R \bowtie_{\text{sid}} S = \cup ( \sigma_{\text{hash}(\text{sid})}(R) \bowtie_{\text{sid}} \sigma_{\text{hash}(\text{sid})}(S) )$
- Pick a hash function so that  $\sigma_{\text{hash}(\text{sid})}(R)$  fits in memory!



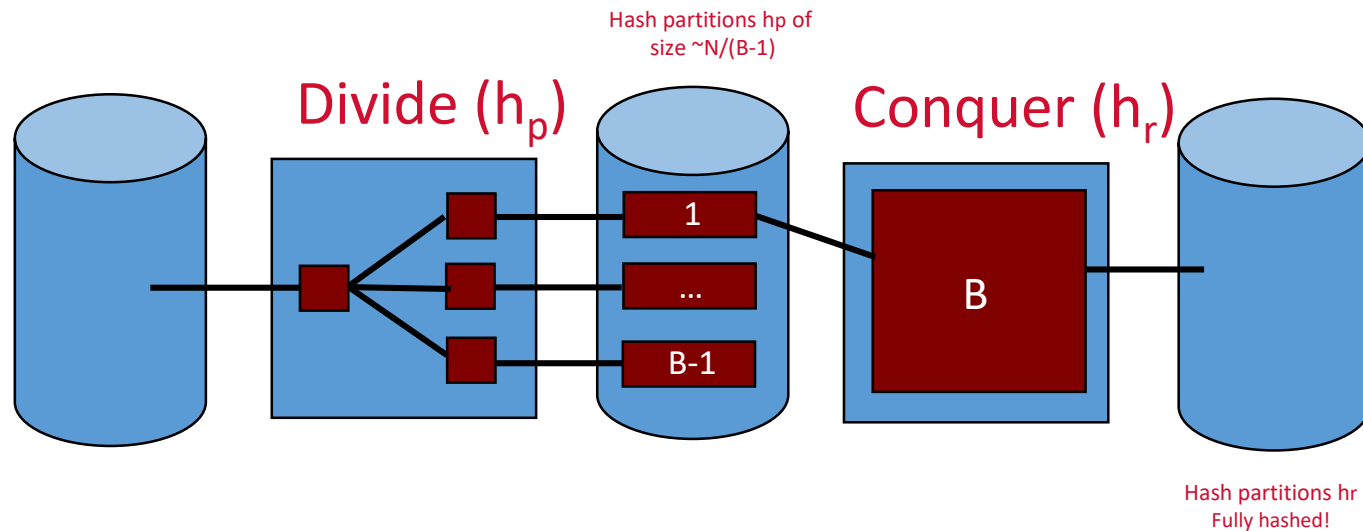
# Grace Hash Join



- Requires equality predicate:
  - **Equi-Joins & Natural Joins**
- Two Stages:
  - **Partition** tuples from **R** and **S** by join key and store on scratch disk
    - all tuples for a given key in same partition
  - **Build & Probe** a separate hash table for each partition (like in Naïve Hash)
    - Assume **partition** of smaller rel. fits in memory
      - Recurse if necessary...

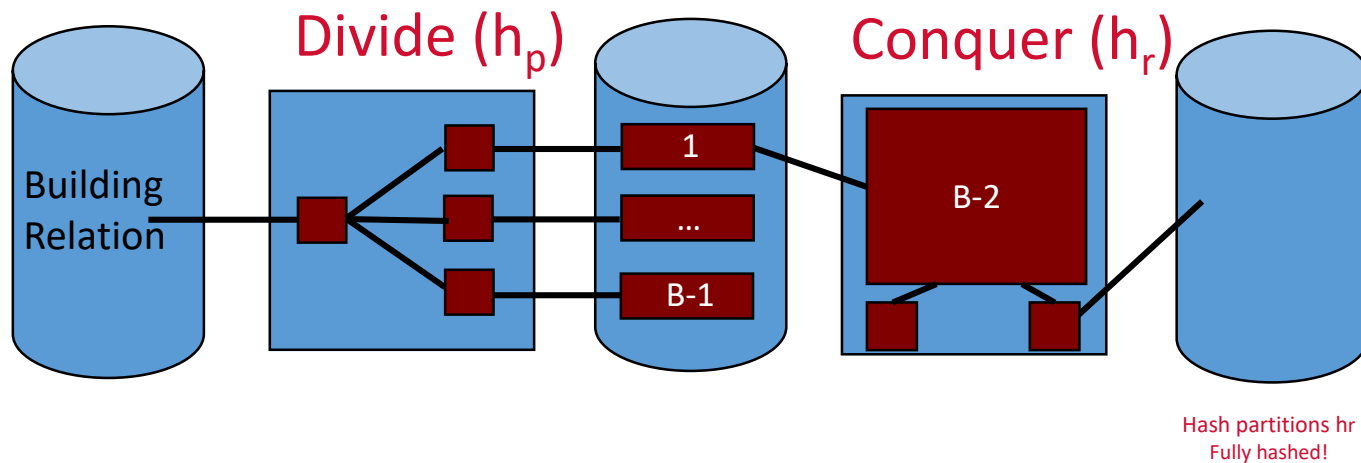


# Remember External Hashing?



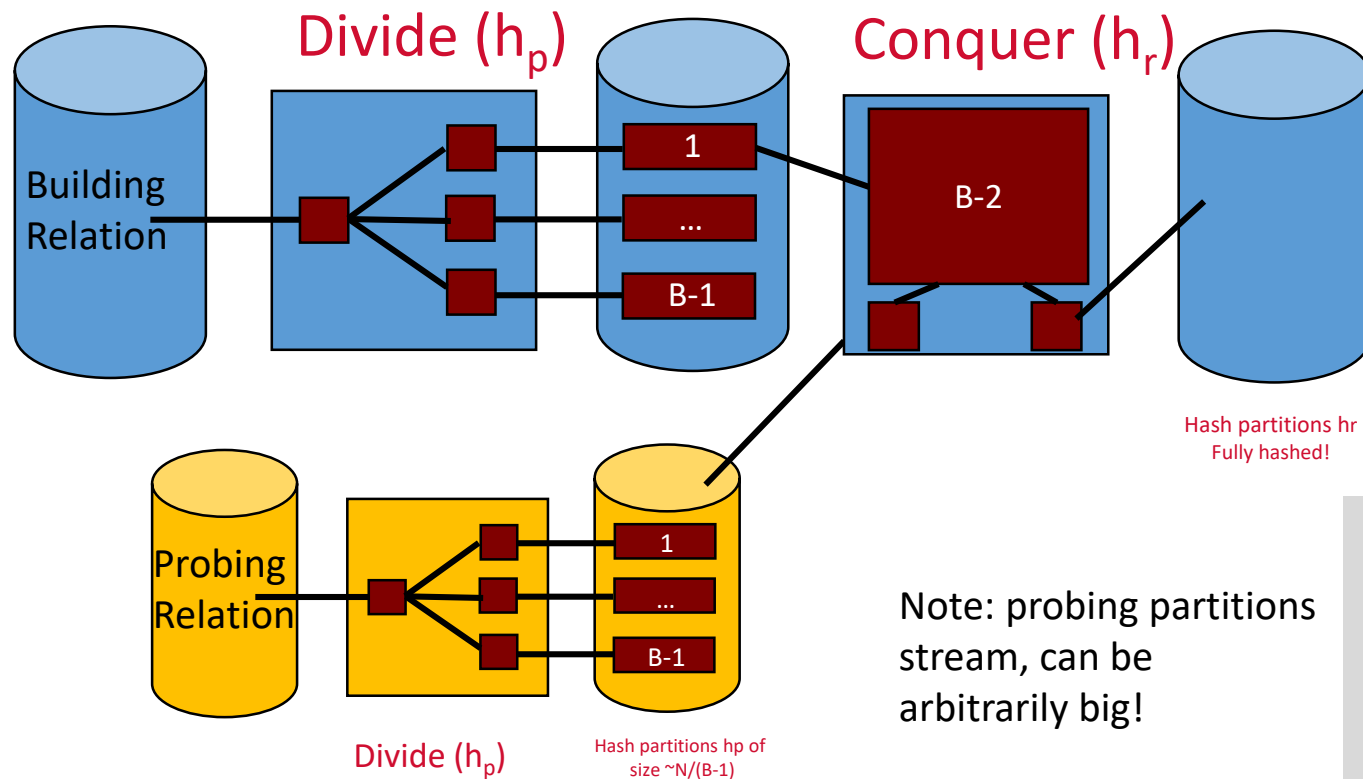


# Sketch of Grace Hash Join





# Sketch of Grace Hash Join, cont.







# PseudoCode, Grace Hash

For Cur in {R, S}

For page in Cur

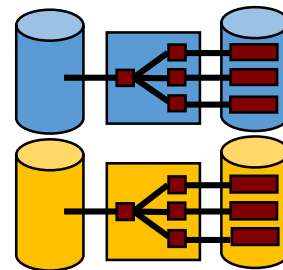
Read page into input buffer

For tup on page

Place tup in output buf  $\text{hash}_p(\text{tup.joinkey})$

If output buf full then flush to disk partition

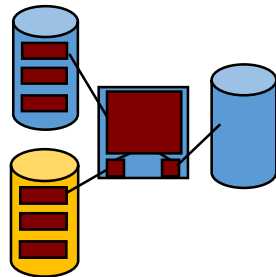
Flush output bufs to disk partitions





# PseudoCode, Grace Hash, cont.

```
For Cur in {R, S}
  For page in Cur
    Read page into input buffer
    For tup on page
      Place tup in output buf  $\text{hash}_p(\text{tup.joinkey})$ 
      If output buf full then flush to disk partition
    Flush output bufs to disk partitions
For  $i$  in  $[0..(B-1))$ 
  For page in  $R_i$ 
    For tup on page
      Build tup into memory  $\text{hash}_r(\text{tup.joinkey})$ 
  For page in  $S_i$ 
    Read page into input buffer
    For tup on page
      Probe memory  $\text{hash}_r(\text{tup.joinkey})$  for matches
      Send all matches to output buffer
      Flush output buffer if full
```





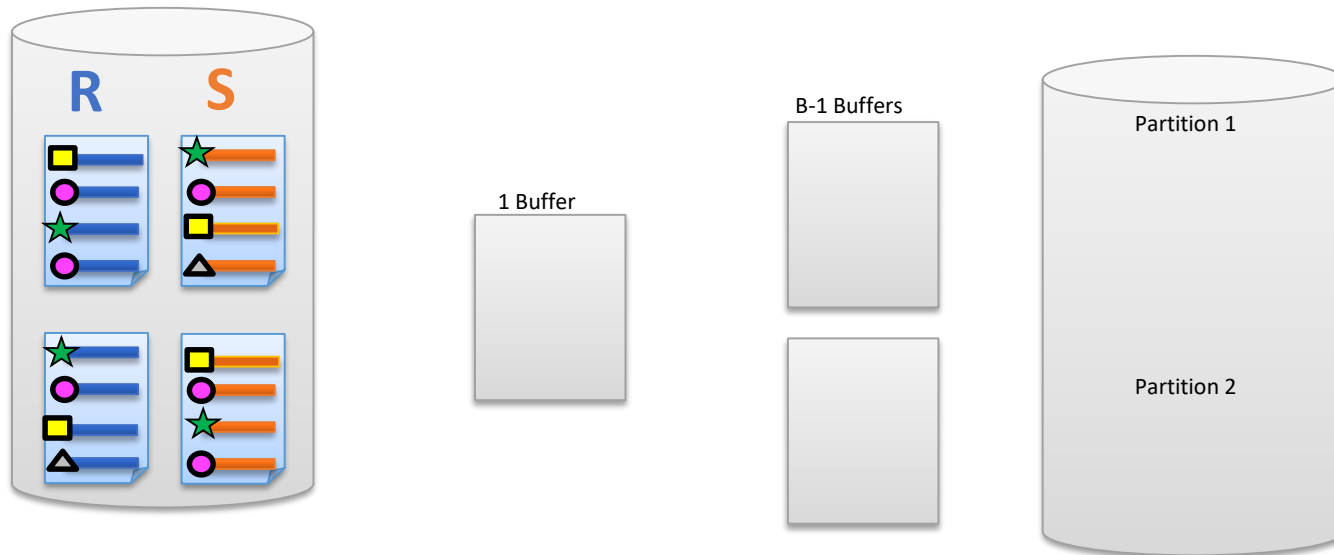
# Grace Hash Join



- An animation
- Two phases:
  - Partition (divide)
  - Build & Probe hash tables (conquer)

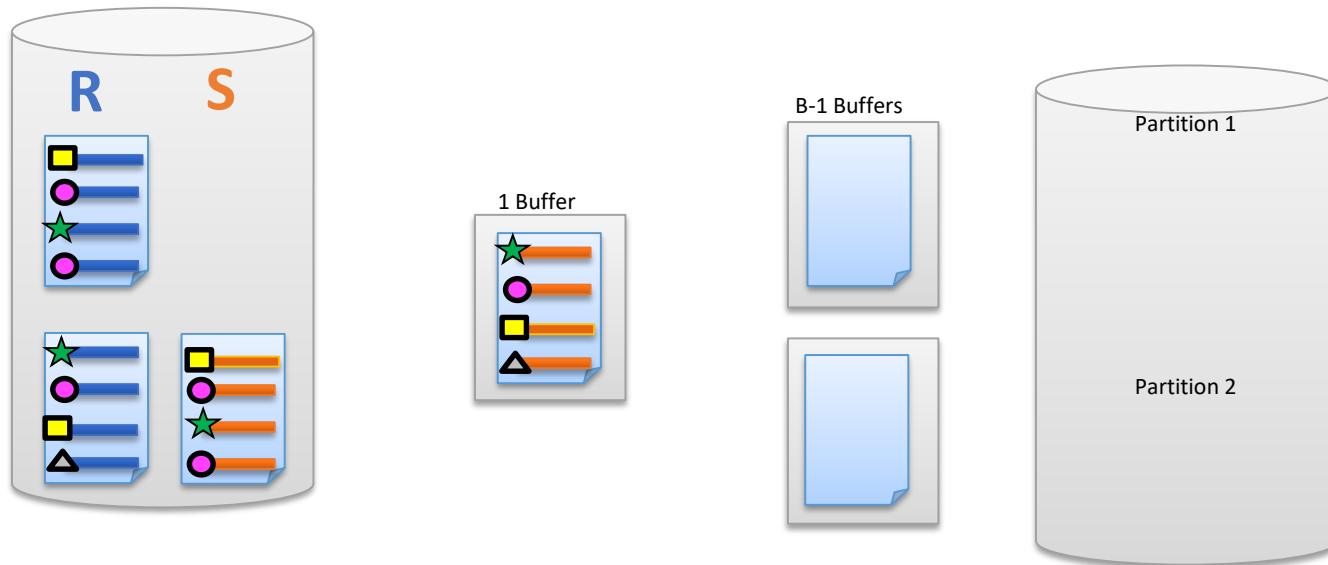


# Grace Hash Join: *Partition*



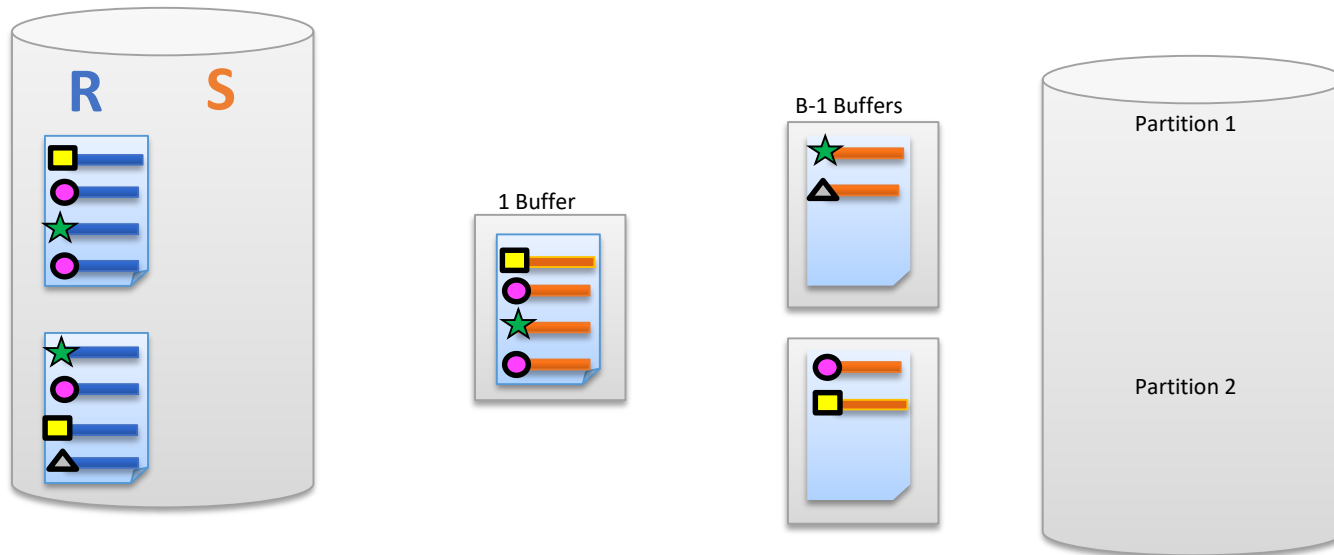


# Grace Hash Join: *Partition, Part 2*



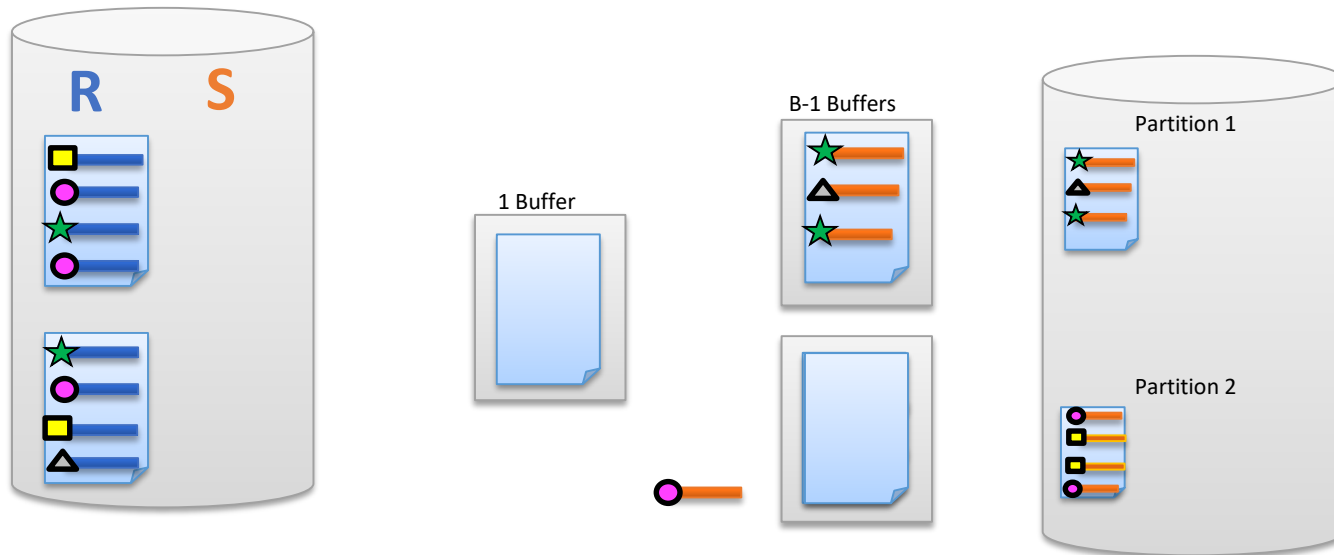


# Grace Hash Join: *Partition, Part 3*



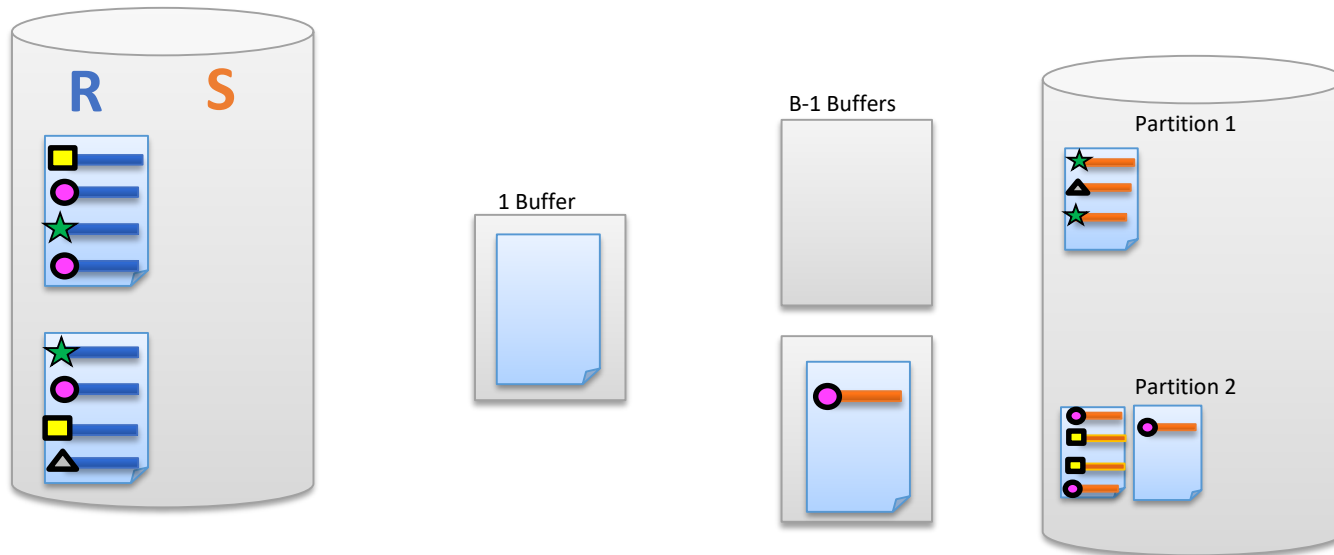


# Grace Hash Join: *Partition Part 4*





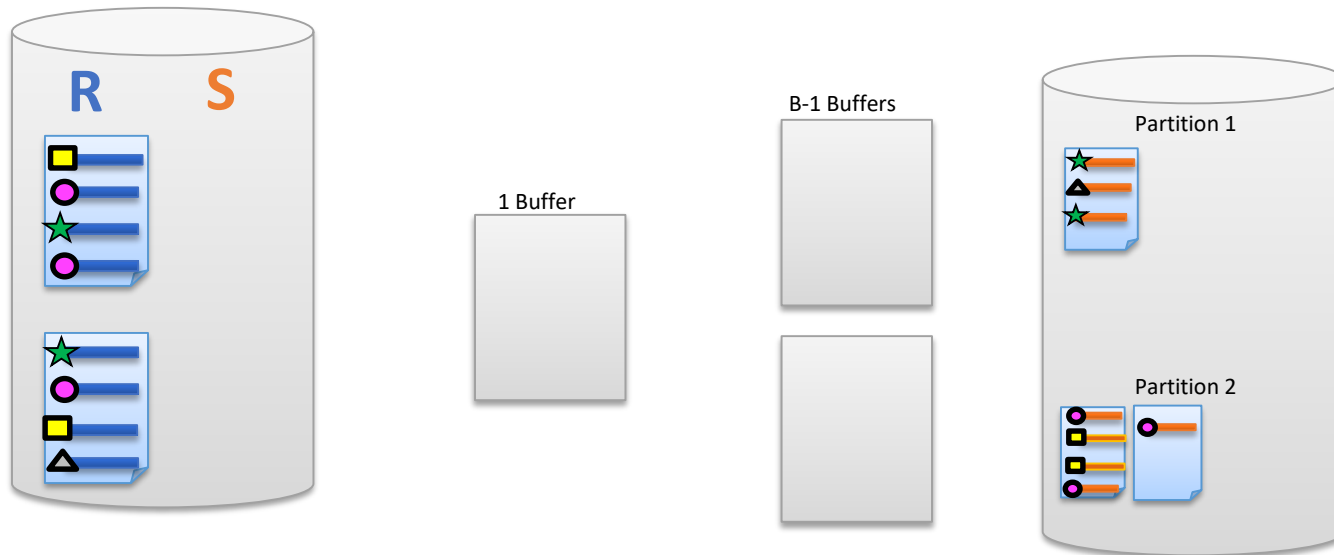
# Grace Hash Join: *Partition Part 5*





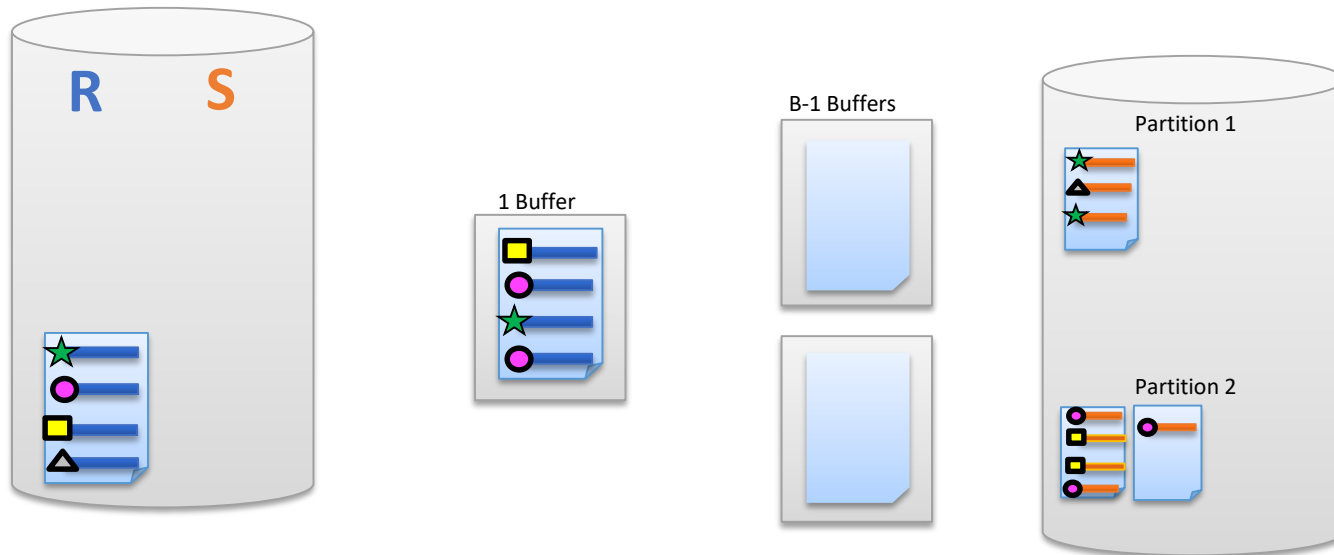


# Grace Hash Join: *Partition Part 6*



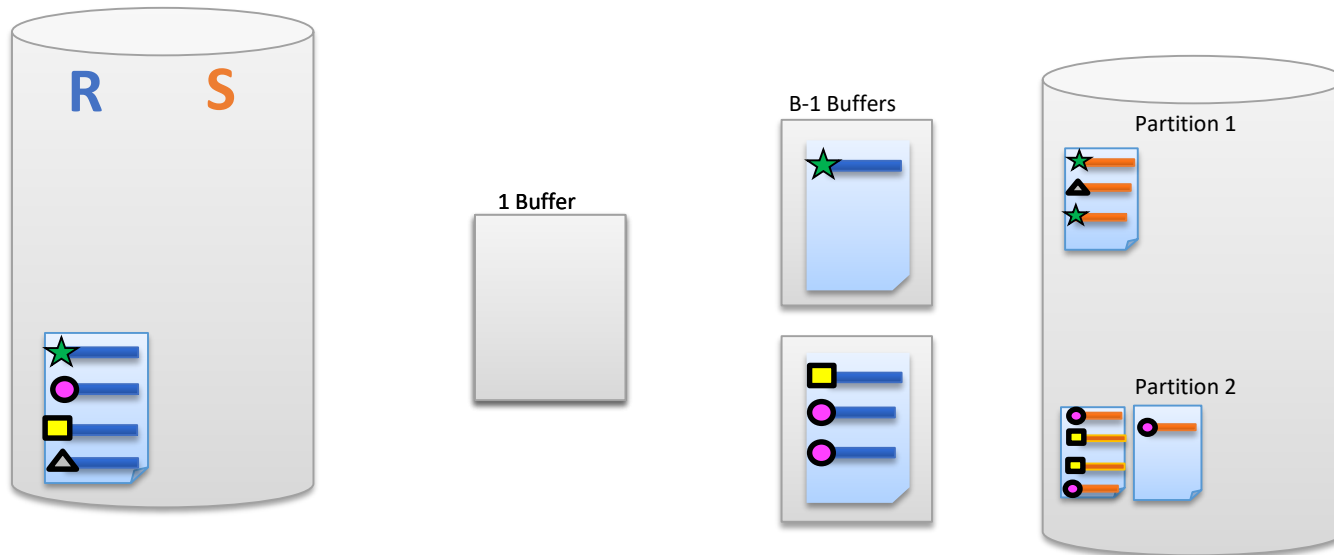


# Grace Hash Join: *Partition Part 7*



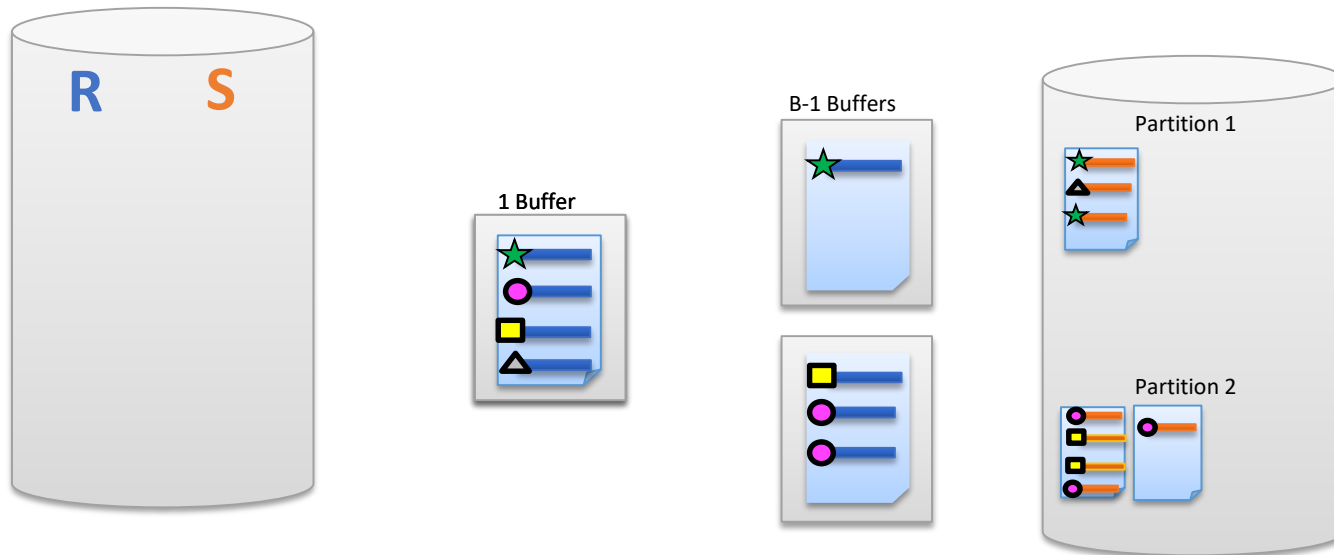


# Grace Hash Join: *Partition Part 8*



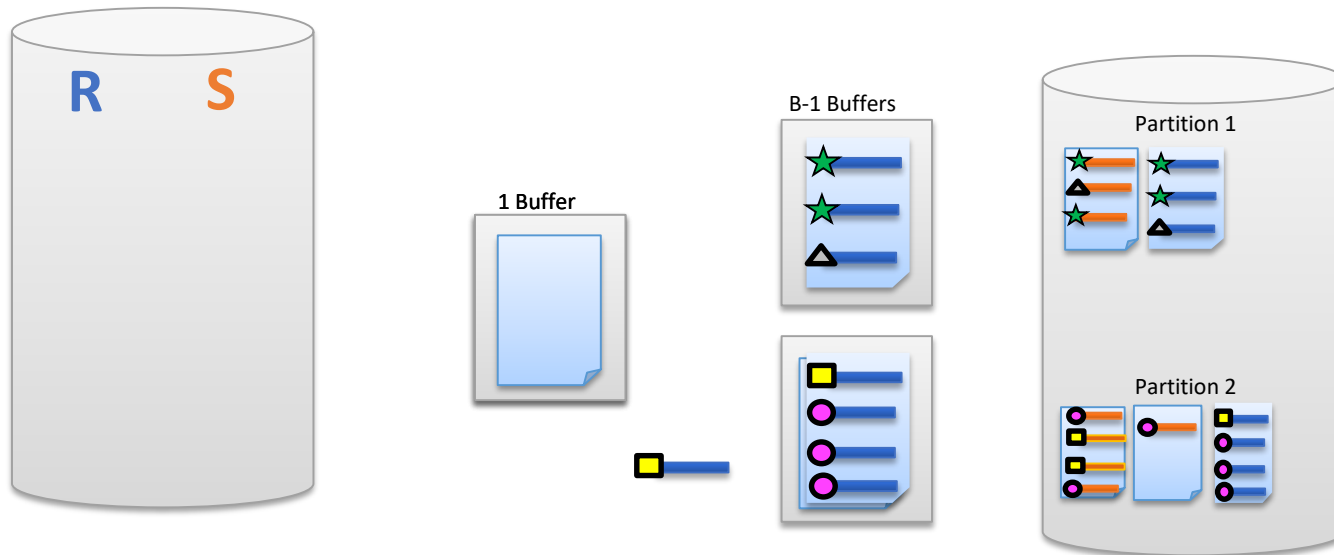


# Grace Hash Join: *Partition 9*



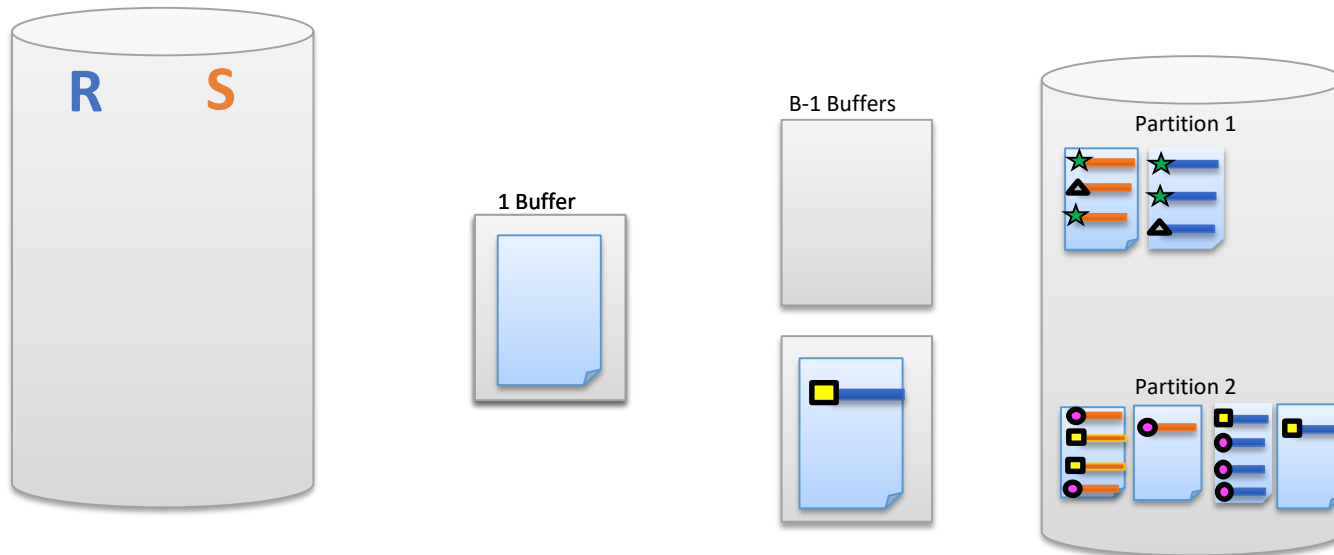


# Grace Hash Join: *Partition Part 10*





# Grace Hash Join: *Partition Part 11*

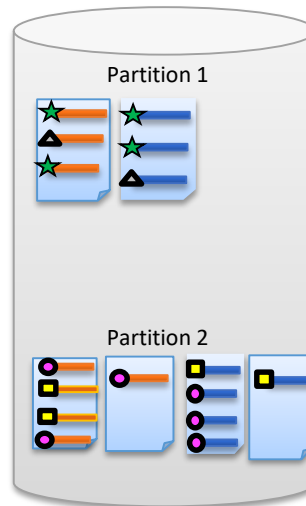




# Grace Hash Join: *Partition Part 12*

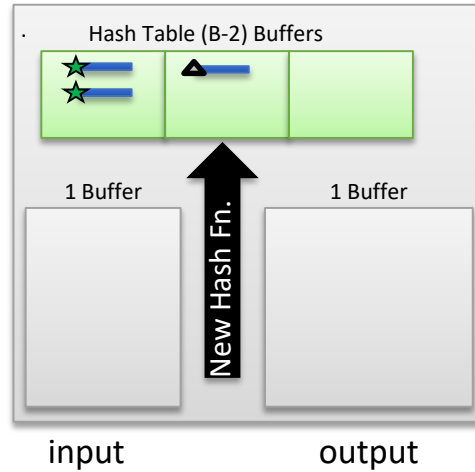
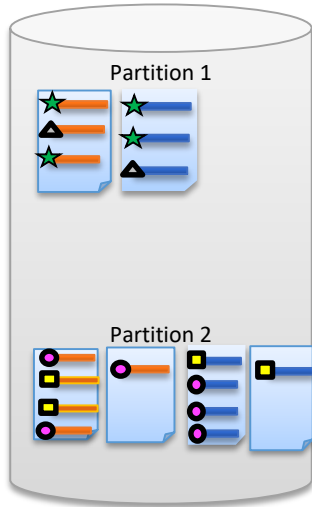


- Each key is assigned to one partition
  - e.g., **green star** keys only in Partition 1 →
- Sensitive to key Skew
  - **Fuchsia circle** Key
- Each partition could be on a different disk or even different machine





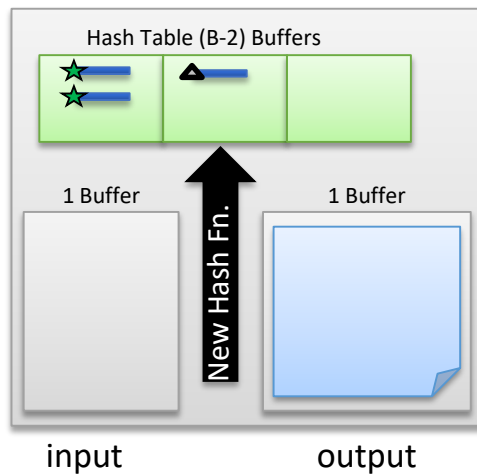
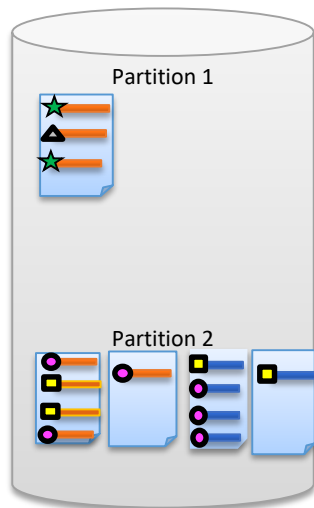
# Grace Hash Join: **Build** & Probe





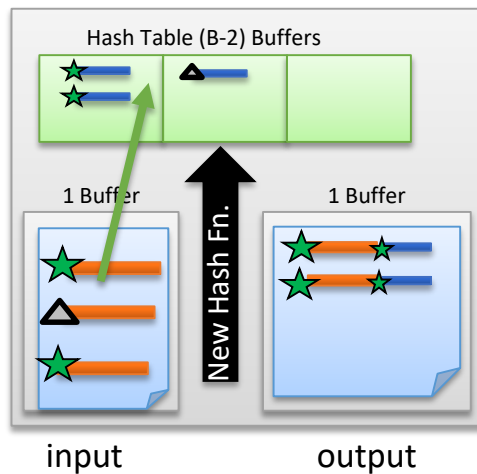
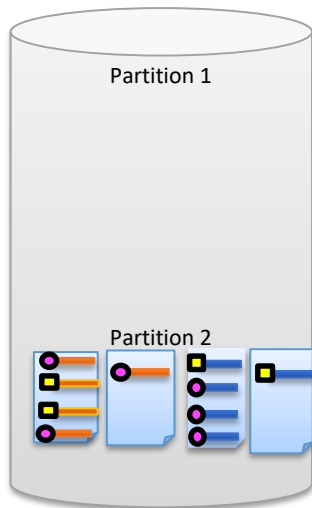


# Grace Hash Join: *Build & Probe Part 2*

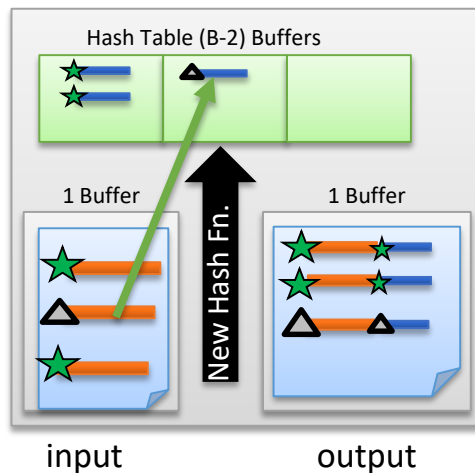
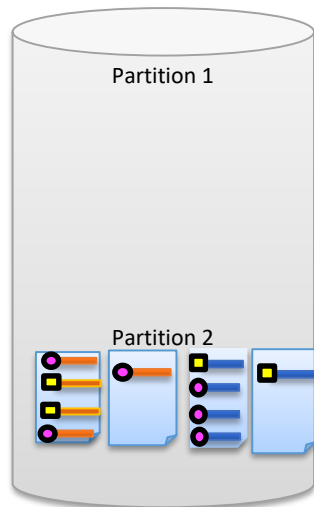




# Grace Hash Join: *Build & Probe Part 3*

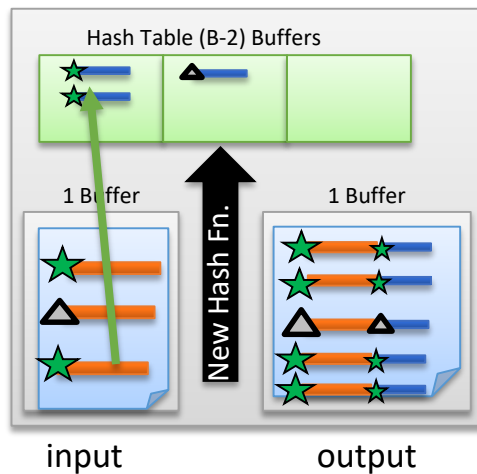
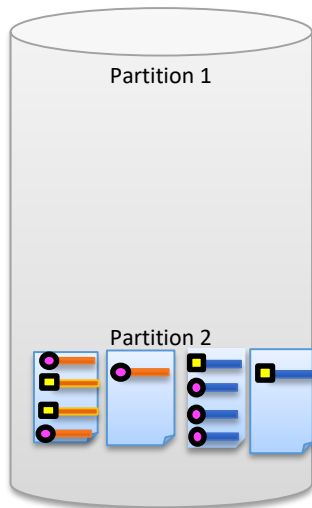


# Grace Hash Join: *Build & Probe Part 4*



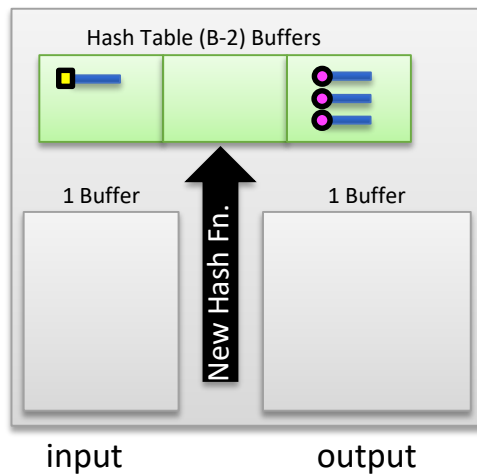
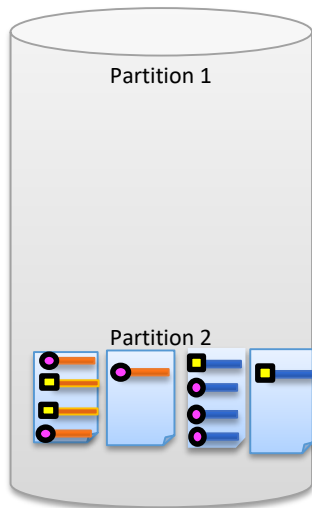


# Grace Hash Join: *Build & Probe Part 5*



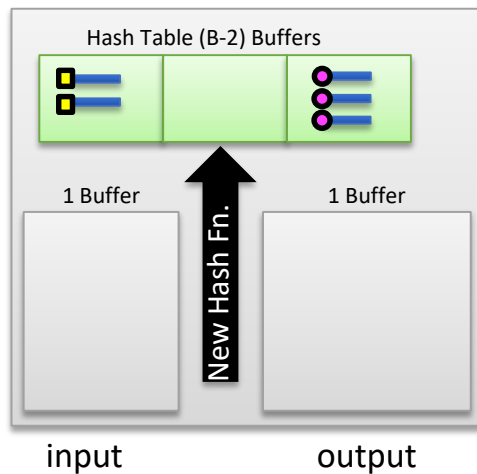
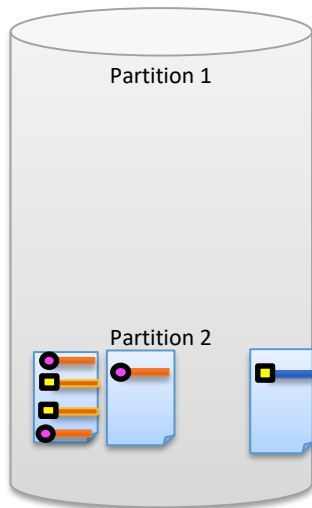


# Grace Hash Join: **Build** & *Probe Part 6*



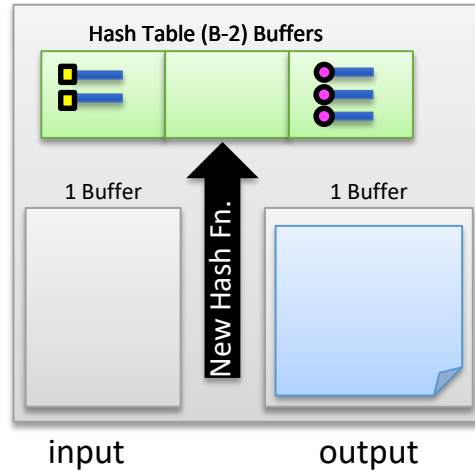
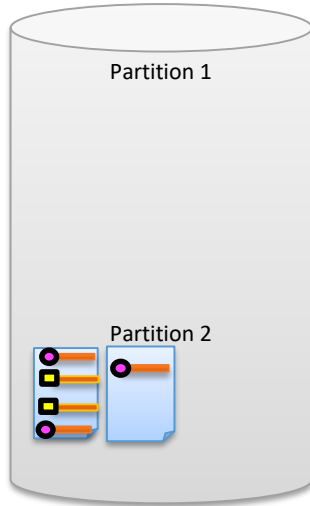


# Grace Hash Join: **Build** & *Probe Part 7*



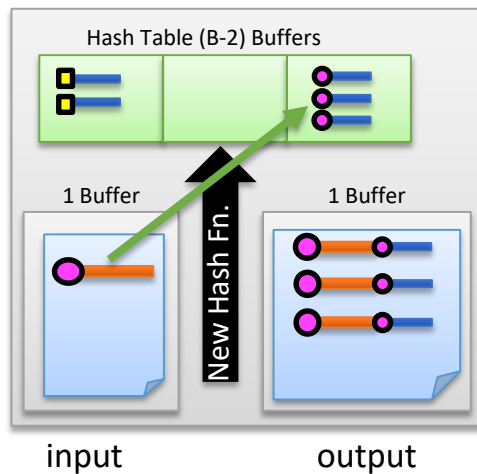


# Grace Hash Join: *Build & Probe Part 8*



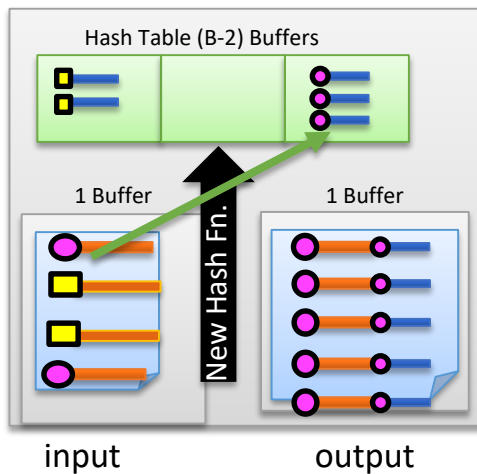
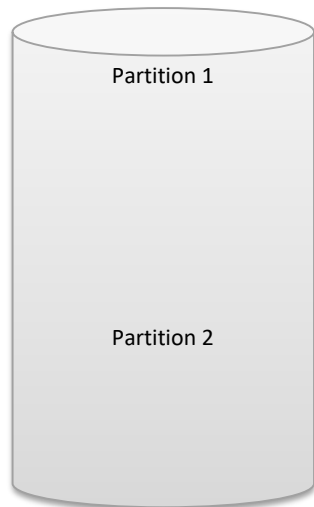


# Grace Hash Join: *Build & Probe Part 9*



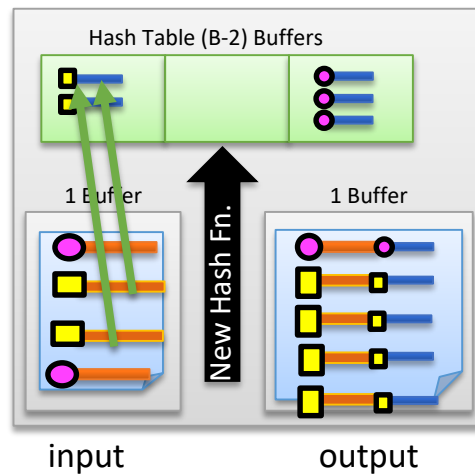
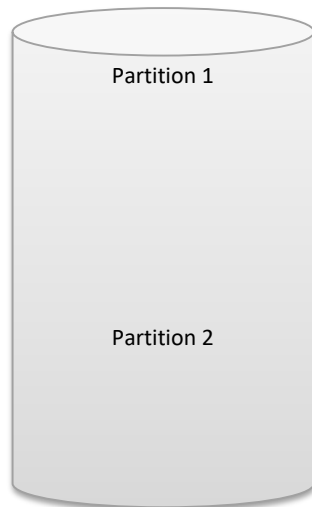


# Grace Hash Join: *Build & Probe Part 10*

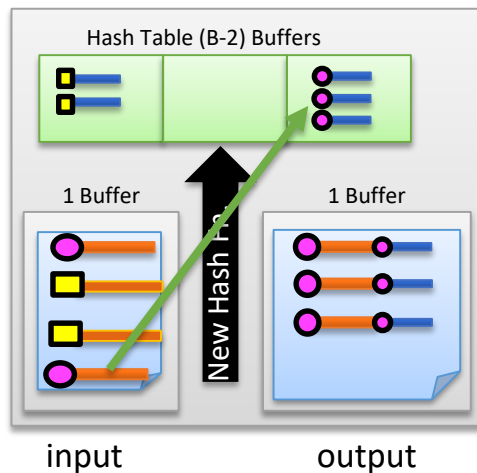
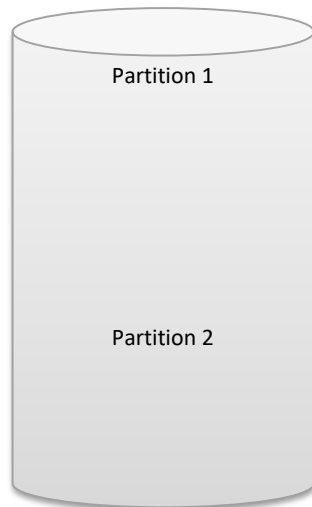




# Grace Hash Join: *Build & Probe Part 11*

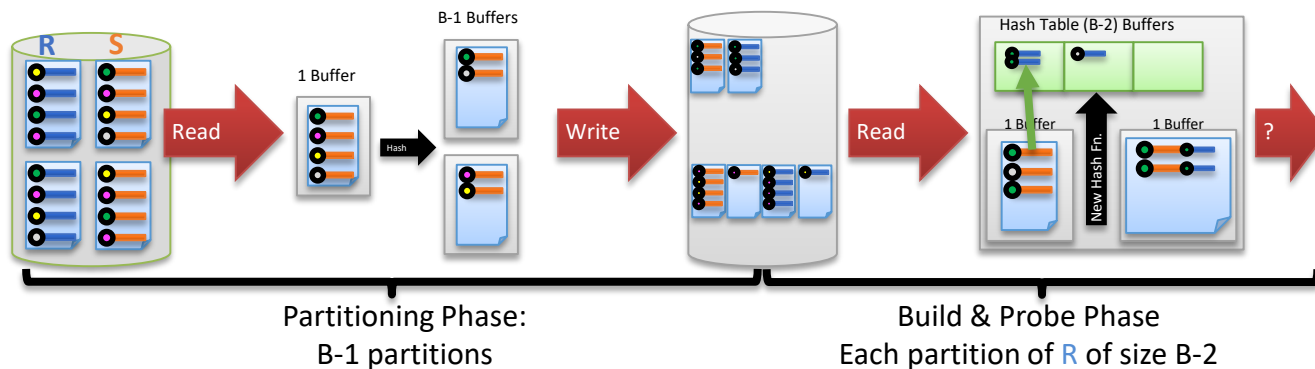


# Grace Hash Join: *Build & Probe Part 12*





# Summary of Grace Hash Join

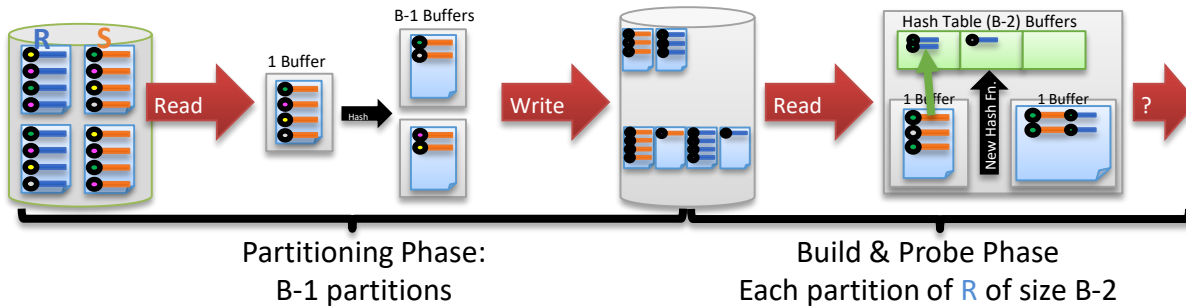


## What is the Cost?



# Cost of Hash Join

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

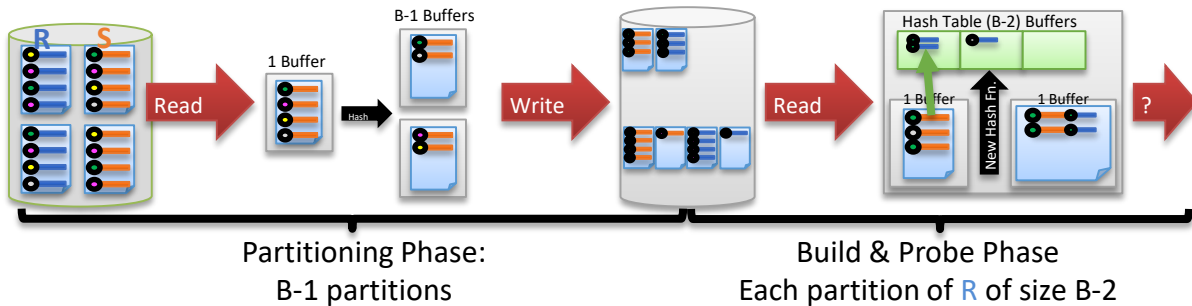


- Partitioning phase: read+write both relations  
①  $2([R]+[S])$  I/Os
- Matching phase: read both relations, forward output  
①  $[R]+[S]$
- Total cost of 2-pass hash join =  $3([R]+[S])$ 
  - $3 * (1000 + 500) = 4500$



# Cost of Hash Join Part 2

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



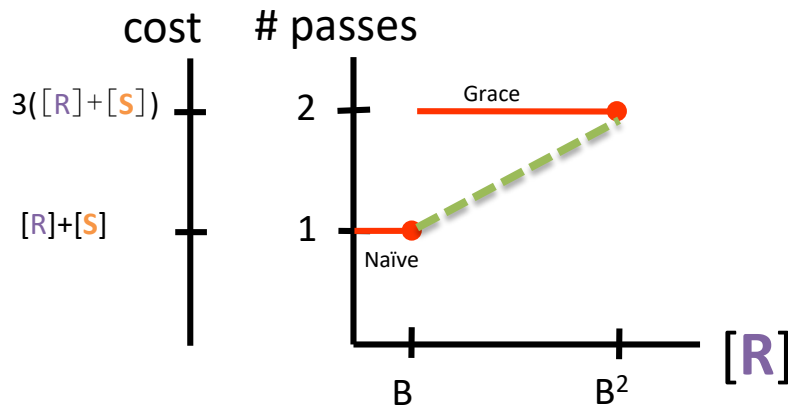
- **Memory Requirements?**
- Build hash table on  $R$  with uniform partitioning
  - **Partitioning Phase** divides  $R$  into  $(B-1)$  runs of size  $[R] / (B-1)$
  - **Matching Phase** requires each  $[R] / (B-1) < (B-2)$
  - $R < (B-1)(B-2) \approx B^2$
- Note: no constraint on size of  $S$  (probing relation)!



# Cost of Hash Join Part 3

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

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



- **Naïve Hash Join:** *requires*  $[R] < B$ 
  - Put all of  $R$  in hash table
  - 1/3 the I/O cost of Grace!
- **Grace Hash Join:** 2-passes for  $[R] < B^2$
- **Hybrid Hash Join:** an algorithm that adapts between the two
  - Tricky to tune



# Hash Join vs. Sort-Merge Join



- Sorting pros:
  - Good if input already sorted, or need output sorted
  - Not sensitive to data skew or bad hash functions
- Hashing pros:
  - For join: # passes depends on size of smaller relation
    - E.g. if smaller relation is  $< B$ , naïve/hybrid hashing is great
  - Good if input already hashed, or need output hashed





# Recap



- Nested Loops Join
  - Works for arbitrary  $\Theta$
  - Make sure to utilize memory in blocks
- Index Nested Loops
  - For equi-joins
  - When you already have an index on one side
- Sort/Hash
  - For equi-joins
  - No index required
  - Hash better if one relation is much smaller than other
- No clear winners – may want to implement them all
- Be sure you know the cost model for each
  - You will need it for query optimization!