### Example Table

#### Table Sailors

<table>
<thead>
<tr>
<th>sid</th>
<th>sname</th>
<th>rating</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>Dustin</td>
<td>7</td>
<td>45</td>
</tr>
<tr>
<td>28</td>
<td>Yuppy</td>
<td>9</td>
<td>35</td>
</tr>
<tr>
<td>31</td>
<td>Lubber</td>
<td>8</td>
<td>55</td>
</tr>
<tr>
<td>36</td>
<td>Guppy</td>
<td>6</td>
<td>36</td>
</tr>
<tr>
<td>44</td>
<td>rusty</td>
<td>5</td>
<td>35</td>
</tr>
</tbody>
</table>

#### Table Reserves

<table>
<thead>
<tr>
<th>sid</th>
<th>bid</th>
<th>day</th>
<th>rname</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>103</td>
<td>12/04/96</td>
<td>guppy</td>
</tr>
<tr>
<td>28</td>
<td>103</td>
<td>11/03/96</td>
<td>ruppy</td>
</tr>
<tr>
<td>31</td>
<td>102</td>
<td>10/12/96</td>
<td>Lubber</td>
</tr>
<tr>
<td>31</td>
<td>101</td>
<td>10/11/96</td>
<td>Lubber</td>
</tr>
<tr>
<td>31</td>
<td>101</td>
<td>10/10/96</td>
<td>dustin</td>
</tr>
</tbody>
</table>

---

**Foreign Key**

**I/O cost vs CPU cost**

- We assume $M$ pages in Reserves table, with $PR$ tuples per page, and $N$ pages in Sailors table, with $PS$ tuples per page.
- $M = 1000$
- $PR$ and $PS$ both = 100
- $N = 500$. 
**NESTED LOOPS JOIN**

**I/O cost:** $1000 + 1000 \times 100 \times 500 = 1000 + (5 \times 10^7)$

**Time Taken:** 140 hours
### Table R

<table>
<thead>
<tr>
<th>Page1</th>
<th>Page2</th>
<th>Page3</th>
<th>...</th>
<th>Page1000</th>
</tr>
</thead>
</table>

### Table S

<table>
<thead>
<tr>
<th>Page1</th>
<th>Page2</th>
<th>Page3</th>
<th>...</th>
<th>Page500</th>
</tr>
</thead>
</table>

**I/O cost:**
- **(Best)** $500 + 1000 \times 500 = 500,500$
- **(Worse)** $1000 + 1000 \times 500 = 501,000$

**Min(M,N) + M*N**

**Time Taken:** 1.4 hours
**Blocked Nested Loops Join**

**Table R**
- Block 1
  - Page 1
  - Page 2
  - Page 3
  - ...
  - Page 1000

**Table S**
- Page 1
- Page 2
- Page 3
- ...
- Page 500

**I/O cost:** \[ M + N \cdot M / (B - 2) = 1000 + 500 \cdot 1000 / 100 = 6000 \]

**Time Taken:** Around One minute
If buffersize=100 pages  
\[ \text{i/o cost} = 1000 + \frac{1000}{100} \times 500 = 6000 \]

If buffersize=90 pages  
\[ \text{i/o cost} = 1000 + \frac{1000}{90} \times 500 = 6555.56 \]
INDEX NESTED LOOPS JOIN

**Table R**
- Tuple1(sid1)
- Tuple2(sid2)
- Page2
- Page3
- ...
- Page1000

**Table S’ index (1 page)**
- Sid1
- Sid 2
- ...
- Sid 100

**Table S**
- Page1
- Page2
- Page3
- ...
- Page500

---

**Create Index Table**

**Finding Stage:** $1000 \times 100 \times 1.2 = 120000$ I/O cost

**Retrieving stage:** $1000 \times 100 \times 1 = 100000$ I/O cost
After Sorting…

### Table Reserves

<table>
<thead>
<tr>
<th>sid</th>
<th>bid</th>
<th>day</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>103</td>
<td>12/04/96</td>
<td>guppy</td>
</tr>
<tr>
<td>28</td>
<td>103</td>
<td>11/03/96</td>
<td>ruppy</td>
</tr>
<tr>
<td>31</td>
<td>101</td>
<td>10/10/96</td>
<td>dustin</td>
</tr>
<tr>
<td>31</td>
<td>102</td>
<td>10/12/96</td>
<td>Lubber</td>
</tr>
<tr>
<td>31</td>
<td>101</td>
<td>10/11/96</td>
<td>Lubber</td>
</tr>
</tbody>
</table>

### Table Sailors

<table>
<thead>
<tr>
<th>sid</th>
<th>sname</th>
<th>rating</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>Dustin</td>
<td>7</td>
<td>45</td>
</tr>
<tr>
<td>28</td>
<td>Yuppy</td>
<td>9</td>
<td>35</td>
</tr>
<tr>
<td>31</td>
<td>Lubber</td>
<td>8</td>
<td>55</td>
</tr>
<tr>
<td>36</td>
<td>Guppy</td>
<td>6</td>
<td>36</td>
</tr>
<tr>
<td>44</td>
<td>rusty</td>
<td>5</td>
<td>35</td>
</tr>
</tbody>
</table>
### Sort-Merge Join

- **M = 1000**
- **PR and PS both = 100**
- **N = 500.**

#### I/O cost

<table>
<thead>
<tr>
<th>I/O cost</th>
<th>Time taken</th>
<th>Main memory at each time</th>
</tr>
</thead>
</table>
| Sort Merge Join | Sorting stage: 4349  
Merging stage: 1500  
Total cost: 4649 I/Os | 46.49 seconds | 4 pages  
(1 for loading table R, 1 for loading table S, 1 for merging and 1 for output) |
1. The Reserves table is very large, and a given partition contains many pages of tuples
   • The first page of such a partition may no longer be in the buffer pool when it is requested a second time.
   • This means that when merging, we will have to scan all the R pages in the partition with every tuple in the corresponding Sailors partition, instead of just scanning the Sailors partition once.

2. In a worst case scenario where all tuples in both relations contain the same value in the join attribute, the merging phase will require us to read the complete second relation for every tuple in the first relation. This makes the I/O cost \( O(MN) \).
• Refinement

• Combines the merging phase of sorting with merging phase of join
• Total cost is $3(M+N)$ only
• Requires more buffer pages in main memory
• Replacement Sort
//Partition R into k partitions
foreach tuple do
read r and add it to buffer page h(ri);

//Partition S into k partitions
foreach tuple do
  read s and add it to buffer page h(sj);

//Probing phase
for do {
  //Build in-memory hash table for Rl, using h2
  foreach tuple   Rl do
    read r and insert into hash table using h2(ri);

  //Scan Sl and probe for matching Rl tuples
  foreach tuple   Sl do {
    read s and probe table using h2(sj);
    for matching R tuples r, output <r,s>
  }

  Clear hash table to prepare for next partition
}
Hash Join

Table R

Tuple 1
Tuple 2
Tuple 3
...
Tuple N

Hash Table for Partition R₁

Tuple 1
Tuple 2
...
Tuple 10

Hash Table for Partition S₁

Tuple 1
Tuple 2
Tuple 3
...
Tuple 10

Tuple N

Partition R₁

Partition S₁

Hash h
Hash Join

**Table R**
- Tuple 1
- Tuple 2
- Tuple 3
- ...
- Tuple N

**Hash Table for Partition R1**
- Resorted Tuple 1
- Resorted Tuple 2
- ...
- Resorted Tuple 10

**Partition** $R_1$, $R_2$, $R_n$
## Hybrid Hash Join

<table>
<thead>
<tr>
<th>Scenario</th>
<th>I/O cost</th>
<th>Time taken</th>
<th>Main memory at each time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid Hash Join</td>
<td>• Partition by partition</td>
<td>• $750 + 1500 + 750 = 3000$</td>
<td>• R partition in in-memory hash table</td>
</tr>
<tr>
<td></td>
<td>• Entire R in in-memory</td>
<td>• $500 + 1000 = 1500$</td>
<td>• All of R in the hash table</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• $3000$ I/Os * 10ms per I/Os</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>= $30,000$ ms = 30s</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• $1500 * 10$ ms = 15s</td>
<td></td>
</tr>
<tr>
<td>Hash Join</td>
<td>Partition by partition</td>
<td>$3 * (500 + 1,000) = 4,500$ I/Os;</td>
<td>Partition r of R and Partition s of S that corresponds with each other.</td>
</tr>
</tbody>
</table>
### Analysis and Comparisons

- \( M = 1000 \)
- \( PR \) and \( PS \) both = 100
- \( N = 500 \)

<table>
<thead>
<tr>
<th>Scenario</th>
<th>I/O cost</th>
<th>Time taken</th>
<th>Data in main memory at each time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nested Loops Join</td>
<td>( 1000 + 1000 \times 100 \times 500 ) ( = 1000 + (5 \times 10^7) )</td>
<td>140 hours</td>
<td>Total: 4 pages.</td>
</tr>
<tr>
<td>Page-at-a-time</td>
<td>Best: ( 500 + 1000 \times 500 ) ( = 500,500 )</td>
<td>About 1.4 hours</td>
<td>Total: 4 pages.</td>
</tr>
<tr>
<td>Block Nested Loops Join</td>
<td>Each block can contain 100 pages</td>
<td>Best: ( \min(m,n) + N \times M / (B - 2) ) ( = 500 + 500 \times 1000 / 100 = 5500 )</td>
<td>Around one minute</td>
</tr>
<tr>
<td>Index Nested Loops Join</td>
<td>Outer table is the Reserve table ( (r.sid ) is the foreign key of ( S ) table)</td>
<td>Total cost: 220,000 I/O s</td>
<td>36 minutes</td>
</tr>
<tr>
<td>Sort-Merge Join</td>
<td>Total cost: 4649 I/O s</td>
<td>46.49 seconds</td>
<td>About four pages</td>
</tr>
<tr>
<td>Hybrid Hash Join</td>
<td>Partition by partition ( R ) in-memory</td>
<td>750 + 1500 = 3000 I/Os; 500 + 1000 = 1500 I/Os</td>
<td>30 seconds</td>
</tr>
<tr>
<td>Hash Join</td>
<td>Partition by partition ( R ) in-memory</td>
<td>3*(500 + 1,000) = 4,500 I/Os;</td>
<td>45 seconds</td>
</tr>
</tbody>
</table>