**Class 15: Joins I**

### Summary

- 
  - #pages \( M \)  
  - #rows per page \( P_r \)  
  - \( S(\text{sid}, \text{sname}, \text{rating}, \text{age}) \)  
  - \( N = 500 \)  
  - \( P_s < 80 \)  
  - \( R(\text{sid}, \text{bid}, \text{day}, \text{sname}) \)  
  - \( M = 1000 \)  
  - \( P_r = 100 \)

### Selection

(a)  
- Cheapest access path  
- Retrieve tuples  
- Apply remaining selection conditions

(b)  
- Get rid of from all matching indexes  
- Intersection rid  
- Retrieve tuples & apply remaining sel. cond.

- **unsorted**: \( M \) I/Os  
- **sorted**: \( \log_2 M + f \cdot M \)  
- **clustered**: \( \log f \cdot M + f \cdot M \)  
- **unclustered**: \( \log f \cdot M + f \cdot M \cdot P_r \)

### Projection

- Sort + discard unwanted fields & duplicates  
- Hash + cc

### Cost

\( M + 2 \cdot T \) pages after removing unwanted fields

---

**Joins**:  
- Nested-Loop Joins  
- Sort-Merge Joins  
- Hash Joins  
- Remaining op (joins + agg)

Any interesting query containing a join
\begin{align*}
\text{SELECT} & \quad \text{FROM} \quad R, S \quad \text{WHERE} \quad R.\text{sid} = S.\text{sid} \\
\text{R} \bowtie \text{S} & \quad \text{discuss as \# I/Os discard output} \\
\text{\overbrace{(R \times S)}} & \quad \text{Simple Nested-Loop Join} \quad R \bowtie S \quad i=j \\
\quad \quad \forall r \in R \quad \text{-- outer} \\
\quad \quad \forall s \in S \quad \text{-- inner} \\
\quad \quad \text{if } r_i = s_j \text{ then add } \langle r, s \rangle \text{ to the result} \\
\text{Cost} & \\
(M \cdot Pr) \cdot N + M = & \quad (1000 \cdot 100) \cdot 500 + 1000 = \underline{50,001,000} \quad \# \text{I/Os} \\
\# \text{rows of } R & \quad 1 \quad \text{I/O} \rightarrow 2 \text{ms} \\
\text{R} \quad M=1000 \rightarrow 4 \text{MB} \\
\text{S} \quad N=500 \rightarrow 2 \text{MB} \\
\text{Swap } R \text{ with } S & \\
(N \cdot Ps) \cdot M + N = & \quad \underline{40,000,500} \quad \text{I/Os} \\
\text{Page-oriented Nested-Loop Join} & \\
\quad \forall \text{ page br in } R \\
\quad \forall \text{ page bs in } S \\
\quad \forall \text{ tuple r in } \text{br} \\
\quad \forall \text{ tuple s in } \text{bs} \\
\quad \text{if } r_i = s_j \text{ then add } \langle r, s \rangle \text{ to the result}
\end{align*}
Cost

\[ M \cdot N + M = 1000 \cdot 500 + 1000 = 501,000 \rightarrow 17 \text{ min} \]

Smaller outer?

\[ N \cdot M + N = 500 \cdot 1000 + 500 = 500,500 \]

- Index Nested Loop Join

\[ \text{for } \text{tuple } r \text{ in } R \]
\[ \text{probe index to fetch } s \text{ such that } s_i = r_j \]
\[ \text{add } (r, s) \text{ to result} \]

Cost

\[ M + M \cdot \text{Pr} = \text{cost of finding matching tuples through the index} \]
\[ \rightarrow \text{Hash index} 1.2 \text{ I/Os} \]
\[ \rightarrow \text{B*-Tree} 2-4 \text{ I/Os} \]

clustered \( \rightarrow \) 1 I/O per page of matching tuples

unclustered \( \rightarrow \) 1 I/O per matching tuple

Example 1: hash idx on aid of S

Scan R: (M)

\[ \text{for each tuple in } R \]
\[ \text{fetch data entry (1,2)} \]
\[ \text{go to file (1)} \]

\[ M + M \cdot \text{Pr} \cdot (2 + 1) \rightarrow 1000 \cdot 100 (2.2) = 221,000 \rightarrow 7 \text{ min} \]
Example 2: hash idx on sid of R

- Scan S \(N\)
- \(H\) probe hash idx (1.2)
- find matching tuples 2.5

\[ N = N \cdot Ps (1.2 + 2.5) = 500 + 500 \cdot 80 \cdot (2.2) = 148,500 \implies 5 \text{ min} \]

Block Nested Loop Joins

- 1 page for streaming inner S
- 1 page for output
- \(k\) pages for holding blocks (of \(N\)) of outer R

- 1 block of \(k\) pages of \(R\)
- \(H\) page bs in \(S\)
- \(H\) tuple \(r\) in \(k\) pages of \(R\)
- \(H\) tuple \(s\) in \(bs\)
  - \(r_i = s_j\) add \(r, s\) to the result

Cost:

- Scan outer \(R\): \(M\) I/Os
- Scan inner for each block of \(R\)

\[ M + \frac{M}{N} \cdot N \implies 1000 + \frac{500 \cdot 1000}{K} \implies 100 \text{ pages} \]

\[ N + \frac{N}{K} \cdot M \implies 500 + \frac{500 \cdot 1000}{K} \implies 3500 \implies 11 \text{ sec} \]
• Sort-Merge Join

- both sorted on the join attribute
  useful: 1) both or one relations sorted on join attr
          2) output should be sorted on join attr.

- many duplicates may lead to backtracking

Cost: Sort R + Sort S + M + N

worst case? M N if all is equal

Cost: \((M+N) \cdot 2 \cdot \#\text{passes} + M + N\)

2 passes?

\[
\left\lfloor \frac{N}{B} \right\rfloor = B - 1 \approx \frac{N}{B} = B - 1 \Rightarrow B^2 - B - N = 0
\]

B \approx \sqrt{N^2 + 1} = 33

Cost = (M+N) \cdot S = 1500 \cdot S = \boxed{7500} \ \text{I/Os} \Rightarrow 15 \text{sec}

BNLJ w/ 33 buffers

\[
M = M - \frac{N}{B} = 1000 + \frac{500 \cdot 1000}{33} \approx \boxed{1000 + 15151}
\]

\[
N + \frac{MN}{K} = 500 + \frac{500 \cdot 1000}{33} \approx \boxed{500 + 15151}
\]

if \(k = 100\) SMJ cannot do better than \(\boxed{7500}\)

BNLJ will do as low as \(\boxed{5500}\)
Refined Sort-Merge Join

Assume \( B > \sqrt{M} \) and \( B > \sqrt{N} \)

After pass 0

\[
R \rightarrow \frac{M}{B} \text{ runs} \quad B > \sqrt{M} \Rightarrow \frac{1}{B} < \frac{1}{\sqrt{M}} \Rightarrow \frac{M}{B} < \sqrt{M} < B
\]

\[
S \rightarrow \frac{N}{B} \text{ runs} \quad B > \sqrt{N} \Rightarrow \frac{N}{B} < \sqrt{N} < B
\]

After pass 0 either \( R, S \) \# runs \( \leq B \)

Consider using replacement sort

It results to runs with size \( \leq 2B \)

\# sorted runs after pass 0 using replacement sort

\[
R \rightarrow \frac{M}{2B} < \frac{B}{2} \quad S \rightarrow \frac{N}{2B} < \frac{B}{2}
\]

We allocate a buffer per sorted run per file

\[
\text{cost} = (M+N)^3
\]

Read \( R \) \rightarrow writing \( < \frac{B}{2} \) \# runs of \( R \) \( 2M \)

Read \( S \) \rightarrow writing \( < \frac{B}{2} \) \# runs of \( S \) \( 2N \)

Read \( R \) and \( S \) and merge on the fly: \( M+N \)

\[
(M+N)^3 = \boxed{4500} \text{ I/Os} \rightarrow \boxed{7S}
\]