iterator interface

\text{get-next (opt)} \text{ to produce input to op2}

\begin{itemize}
\item \text{DISTINCT}
\item \text{Project B, C}
\item \text{Select A = 5 from R}
\end{itemize}
Sequential scan $\sigma_c(R)$

- If $t.c$ is true, put into output buffer

- $n = 1$

- Cost = $\text{PAGES}(R)$
Duplicate Removal (SR)

"DISTINCT"

Cost: If m-1 blocks hold all unique values → PAGES (R)
GROUP BY \( \delta_{A,B}, \sum(C), \) 
\( \text{avg}(D) \)

COST = If results fit in M-1 block
\( \text{PAGES}(R) \)
\text{\texttt{R \text{UNION ALL} \texttt{S}}}

\text{\textbf{cost} = \texttt{PAGES}(\texttt{R}) + \texttt{PAGES}(\texttt{S})}

\textbf{OUTPUT}
\text{R \ Union \ S}

\text{Cost = IF } M-1 \text{ is sufficient memory }
\text{PAGES}(R) + \text{PAGES}(S)
\[ R \setminus S \]

- \( M \) blocks allocated
- Is tuple in \( R \)?
  - Yes: delete from memory
  - No: disregard
- Store unique \( R \) tuples

Databases

\[ \text{cost} = \frac{M-1}{\text{block}} \]

- Can hold unique values of \( R \)
- Then \( \text{pages}(R) + \text{pages}(S) \)
**Index Scan**

```sql
SELECT A, B
FROM T
WHERE C = 10;
```

Index `I1` on `R(C)`

Diagram:
- `I1` as the root node.
- `T` as the tree structure.
- `M = 1`:
  - `R` as the internal node.
  - `C`:
    - `M = 1`:
      - Page of `R` found tuple.
- Output `A, B` of tuple.

Additional notes:
- `ROOT + INTERNAL' (1 for each)`
- `LEAF NODES CONTAINING C=10`
- `(PAGES CONTAINING TUPLES IF NEEDED)`
Block nested loop join

\[ R \bowtie S \]

inner

outer

\[ M = 2 \]

output

How many times do we read \( S \)?
pages \( (e) \) times

How many times do I read \( R \)? A time

Cost = \( \text{PAGES}(e) + \text{PAGES}(S) \times \text{PAGES}(e) \)
Block nested loop join \( M > 2 \)

\[ \text{How many times do we read } S? \]

\[ \text{PAGES}(S) + \left\lceil \frac{\text{PAGES}(R)}{M-1} \right\rceil \times \text{PAGES}(S) \]
<table>
<thead>
<tr>
<th>PAGES(E)</th>
<th>PAGES(S)</th>
<th>M</th>
<th>COST</th>
<th>RMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>500</td>
<td>2</td>
<td>100 + 100 * 500</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>500</td>
<td>101</td>
<td>100 + 500</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>500</td>
<td>21</td>
<td>100 + 5 * 500</td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>100</td>
<td>2</td>
<td>500 + 500 * 100</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>101</td>
<td>500 + 5 * 100</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>21</td>
<td>500 + 25 * 100</td>
<td></td>
</tr>
</tbody>
</table>
EXTERNAL SORT

M blocks allocated

$M \geq \text{PAGES}(E)$

$\xrightarrow{\text{sort}}$ output

$\text{COST} = \text{PAGES}(E)$
EXTERNAL SORT \( M \subseteq PAGES(\mathbb{R}) \)

2 phases \( \Rightarrow \) assume \( M \) is fixed throughout

Phase 1: Read \( \rightarrow \) sort \( \rightarrow \) write to temporary \( \rightarrow \) write to disk

\[ \text{cost} = 2 \times \text{PAGES}(\mathbb{R}) \]
EXTERNAL SORT

PHASE 2 → MERGE

If \( M \geq \# \text{sorted groups} \)

find smallest tuple

\( M \uparrow \)

SORTED GROUPS
EXTERNAL SORT

PHASE 2 → MERGE

IF $M \leq \#\text{sorted groups}$

find smallest tuple

write to disk

SORTED GROUPS

$M$
<table>
<thead>
<tr>
<th>External Sort</th>
<th>Pages(s)</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>200</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>10</td>
</tr>
<tr>
<td>Phase 1</td>
<td>2000</td>
<td>20</td>
</tr>
<tr>
<td>Sorted Phase 2</td>
<td>100</td>
<td>10</td>
</tr>
</tbody>
</table>

- Cost = \frac{S}{2^k} - \frac{M}{4^k} + \frac{R}{2^k} - \frac{\text{output}}{2^k}
- \text{pages} = \frac{M}{2^k} - \frac{\text{output}}{2^k}
Pages(R) = 2000
M = 25

Phase 1 → Cost = 24,000 (Read + Write)

\[
\frac{2000}{25} = 80
\]

Phase 2 → Cost = 41,000 (Read + Write)

[\sqrt{\frac{80}{25}}] = 4 \quad \text{to 4 sorted groups}

Phase 2 → Cost = 2,000 (Read + Output)

Total = 10,000 Pages
EXTERNAL SORT EXAMPLE

+ PASS E

M = 4

PAGES(C) = 8

1, 2, 3, 4,
5, 6, 17, 8

MEMORY

DISK

18 14 71 18 29 15 45 6
31 16 61 12 18 19 21 22 23 28
12 4 5 7 8 9 10 13 14 15 18
3 6 11 12 16 18 19 20 21 22 23 28
ORDER BY y SORT
SORT MERGE JOIN

R \rightarrow S
R.A \rightarrow S.B

SORT y ON R.A \rightarrow R

SORT S.B \rightarrow S

WRITE
Group by A
count (B)
sum (C)

Group by
current A
current count of
B
sum of
C

sort on BA
Hashing

MT1 blocks

Each bucket roughly has

\[
\frac{\text{PAGES}(R)}{M}
\]

... pages

Buckets

1 ... M
Hash \rightarrow \text{Remove duplicates} \rightarrow \text{Group by} \rightarrow \text{Union / Difference / Intersection} \rightarrow \{ \text{Apply to each bucket separately} \}

Hash Join \rightarrow Hash(R) \rightarrow Hash(S)
QUERY PLAN

cost ESTIMATION

\[
\text{on-the-fly} = \text{cost} = 0
\]

\[
\text{100 + 2500} = 2600
\]

\[\text{Total cost} = 2600\]
Query Plan Cost Estimation

Total cost = (cost of index scan) + 500 (read once)

\[ \prod_{m=1}^{T} \text{on-the-fly} \]

19 blocks are allocated to

Index scan on \( \Sigma \)

\( m=2 \)

\( \sigma_{rc=400} \)

\( R \)

\( 400 \)

Suppose is \( \sigma_{rc=400} \) to 10 pages