Optimization of Logical Queries

Task:

Consider the following relational schema:

- Emp(eid, did, sal, hobby)
- Dept(did, dname, floor, phone)
- Finance(did, budget, sales, expenses)

For the following SQL statement:

1. Translate the query into the relational algebra.
2. Remove redundant joins from the select-project-join subexpressions in the obtained logical query plan.
3. By means of the algebraic laws, further optimize the obtained expression.
Optimization of Logical Queries

Task (continued)

```
SELECT D.floor
FROM Dept D, Emp E
WHERE
    (D.floor = 1
     OR D.floor IN
       (SELECT D2.floor FROM Dept D2, Finance F1
        WHERE F1.budget > 150 AND D2.did = F1.did)
    )
AND E.did = D.did
AND E.did IN (SELECT F2.did FROM Finance F2, Emp E2
              WHERE F2.did = E.did AND E2.did = D.did
              AND E2.eid = E.eid AND F2.expenses = 300)
```
Optimization of Logical Queries

Solution: translation into the relational algebra

First, we normalize the query to a form with only EXISTS and NOT EXISTS subqueries:

```
SELECT D.floor
FROM Dept D, Emp E
WHERE
  (D.floor = 1 OR EXIST
   ( SELECT D2.floor FROM Dept D2, Finance F1
     WHERE F1.budget > 150 AND D2.did = F1.did
     AND D2.floor = D.floor )
  )
AND E.did = D.did
AND EXISTS (SELECT F2.did FROM Finance F2, Emp E2
             WHERE F2.did = E.did AND E2.did = D.did
             AND E2.eid = E.eid AND F2.expenses = 300
             AND E.did = F2.did)
```
Optimization of Logical Queries

Conjunctive Normal Form

```sql
SELECT D.floor
FROM Dept D, Emp E
WHERE ( D.floor = 1
  AND E.did = D.did
  AND EXISTS (SELECT F2.did FROM Finance F2, Emp E2
    WHERE F2.did = E.did AND E2.did = D.did
    AND E2.eid = E.eid AND F2.expenses = 300 AND E.did = F2.did)
  ) OR (  
    EXIST ( SELECT D2.floor FROM Dept D2, Finance F1
      WHERE F1.budget > 150 AND D2.did = F1.did
      AND D2.floor = D.floor)
    AND E.did = D.did
    AND EXISTS (SELECT F2.did FROM Finance F2, Emp E2
      WHERE F2.did = E.did AND E2.did = D.did
      AND E2.eid = E.eid AND F2.expenses = 300 AND E.did = F2.did) )
```
Optimization of Logical Queries

Normalize to UNION

Q1 = SELECT D.floor
    FROM Dept D, Emp E
    WHERE D.floor = 1
    AND E.did = D.did
    AND EXISTS (SELECT F2.did FROM Finance F2, Emp E2
                WHERE F2.did = E.did AND E2.did = D.did
                AND E2.eid = E.eid AND F2.expenses = 300
                AND E.did = F2.did)
Optimization of Logical Queries

Normalize to UNION

Q2 = SELECT D.floor
FROM Dept D, Emp E
WHERE
  EXIST ( SELECT D2.floor FROM Dept D2, Finance F1
    WHERE F1.budget > 150 AND D2.did = F1.did
    AND D2.floor = D.floor)
  AND E.did = D.did
AND EXISTS (SELECT F2.did FROM Finance F2, Emp E2
    WHERE F2.did = E.did AND E2.did = D.did
    AND E2.eid = E.eid AND F2.expenses = 300
    AND E.did = F2.did)

The new query is Q1 UNION Q2.
Optimization of Logical Queries

Translation of the innermost subqueries

\[
\text{SELECT F2.did FROM Finance F2, Emp E2}
\text{WHERE F2.did = E.did AND E2.did = D.did}
\text{AND E2.eid = E.eid AND F2.expenses = 300}
\text{AND E.did = F2.did}
\]

This subquery is translated as follows:

\[
e_1 = \pi_{F2.did,E.*,D.*} F2.did=E.did \land E2.did=D.did \land E2.eid=E.eid
\sigma_{F2.expenses=300 \land E.did=F2.did}(\rho_D(\text{Dept}) \times \rho_E(\text{Emp}) \times \rho_{F2}(\text{Finance}) \times \rho_{E2}(\text{Emp}))
\]
Optimization of Logical Queries

Translation of the innermost subquery

```
SELECT D2.floor FROM Dept D2, Finance F1
WHERE F1.budget > 150 AND D2.did = F1.did
AND D2.floor = D.floor
```

This subquery is translated as follows:

\[ e_2 = \pi_{D2.floor, D.*} \sigma_{F1.budget > 150 \land D2.did = F1.did} \]
\[ \sigma_{D2.floor = D.floor} (\rho_D(\text{Dept}) \times \rho_{D2}(\text{Dept}) \times \rho_{F1}(\text{Finance})) \]
Optimization of Logical Queries

Translation of the Middle Queries

Q1 = SELECT D.floor FROM Dept D, Emp E
WHERE D.floor = 1 AND E.did = D.did
AND EXISTS (SELECT F2.did FROM Finance F2, Emp E2
WHERE F2.did = E.did AND E2.did = D.did
AND E2.eid = E.eid AND F2.expenses = 300
AND E.did = F2.did)

The translation of the from part gives

\[ e_3 = (\rho_D(\text{Dept}) \times \rho_E(\text{Emp})) \]

To de-correlate we compute:

\[ f = \hat{e}_3 \Join \pi_{D.*,E.*}(e_1) \]

Note that \( \hat{e}_3 \) is empty and hence

\[ f = \pi_{D.*,E.*}(e_1) \]

To this expression we add the WHERE and SELECT clause:

\[ e_4 = \pi_{D.floor}(\sigma_{D.floor=1 \land E.did=D.did}(\pi_{D.*,E.*}(e_1))) \]
Optimization of Logical Queries

Translation of the Middle Queries

\[
Q2 = \text{SELECT } D.\text{floor} \\
\text{FROM Dept D, Emp E} \\
\text{WHERE} \\
\quad \text{EXIST ( SELECT D2.floor FROM Dept D2, Finance F1} \\
\quad \quad \text{WHERE } F1.\text{budget} > 150 \text{ AND D2.}\text{did} = F1.\text{did} \\
\quad \quad \quad \text{AND D2.floor} = D.\text{floor}) \\
\quad \text{AND E.}\text{did} = D.\text{did} \\
\quad \text{AND EXISTS (SELECT F2.}\text{did} \text{FROM Finance F2, Emp E2} \\
\quad \quad \text{WHERE F2.}\text{did} = E.\text{did} \text{ AND E2.}\text{did} = D.\text{did} \\
\quad \quad \quad \text{AND E2.}\text{eid} = E.\text{eid} \text{ AND F2.}\text{expenses} = 300 \\
\quad \quad \quad \text{AND E.}\text{did} = F2.\text{did})}
\]

The translation of the from part gives

\[
e_5 = (\rho_D(\text{Dept}) \times \rho_E(\text{Emp}))
\]

To de-correlate we compute:

\[
f' = e_5 \Join (\pi_{D,*}(e_1) \Join \pi_D(e_2)) = (\pi_{D,*}(e_1) \Join \pi_D(e_2))
\]

Solution of the exercises
To this expression we add the WHERE and SELECT clause:

\[ e_6 = \pi_{D.\text{floor}} \sigma_{E.did = D.did}(\pi_{D.*}E.\pi_{\ast}(e_1) \Join \pi_{D.*}(e_2)) \]
Optimization of Logical Queries

Translation of the Whole Query

Q1 UNION Q2

Since the schemas of $e_4$ and $e_6$ are the same, the union is straightforward:

$$e = e_4 \cup e_6$$

Written in full:

$$e = \pi D.\text{floor}\sigma D.\text{floor}=1\land E.\text{did}=D.\text{did}$$

$$\pi D.\ast, E.\ast \sigma F_2.\text{did}=E.\text{did}\land E_2.\text{did}=D.\text{did}\land E_2.\text{eid}=E.\text{eid}\land F_2.\text{expenses}=300\land E.\text{did}=F_2.\text{did}$$

$$(\rho_D(\text{Dept}) \times \rho_E(\text{Emp}) \times \rho_{F2}(\text{Finance}) \times \rho_{E2}(\text{Emp}))$$

$$\cup$$

$$\pi D.\text{floor}\sigma E.\text{did}=D.\text{did}$$

$$[\pi D.\ast, E.\ast \sigma F_2.\text{did}=E.\text{did}\land E_2.\text{did}=D.\text{did}\land E_2.\text{eid}=E.\text{eid}\land F_2.\text{expenses}=300\land E.\text{did}=F_2.\text{did}$$

$$(\rho_D(\text{Dept}) \times \rho_E(\text{Emp}) \times \rho_{F2}(\text{Finance}) \times \rho_{E2}(\text{Emp}))$$

$$\Join [\pi D.\ast \sigma F_1.\text{budget}>150\land D_2.\text{did}=F_1.\text{did}\land D_2.\text{floor}=D.\text{floor}$$

$$(\rho_D(\text{Dept}) \times \rho_{D2}(\text{Dept}) \times \rho_{F1}(\text{Finance}))]]$$
Optimization of Logical Queries

Redundant Joins Removal

The query comprises the following maximal select-project-join subexpressions:

1. $\pi_{D.\text{floor}} \sigma_{D.\text{floor}=1} \land E.\text{did}=D.\text{did} \pi_{D.*} \sigma_{...}(\rho_D(\text{Dept}) \times \rho_E(\text{Emp}) \times \rho_{F2}(\text{Finance}) \times \rho_{E2}(\text{Emp}))$
2. $[\pi_{D.*} \sigma_{...}(\rho_D(\text{Dept}) \times \rho_E(\text{Emp}) \times \rho_{F2}(\text{Finance}) \times \rho_{E2}(\text{Emp})]$
3. $(\rho_D(\text{Dept}) \times \rho_{D2}(\text{Dept}) \times \rho_{F1}(\text{Finance}))$

Note that "$F_1.\text{budget} > 150$" cannot be included in a select-project-join expression. Also note that the third expression does not contain redundant joins (Why?).
Optimization of Logical Queries

Redundant Joins Removal

The first expression corresponds to:

\[ Q_1(“1”) \leftarrow \text{Dept}(a_1, a_2, “1”, a_4), \text{Emp}(b_1, a_1, b_3, b_4), \text{Finance}(a_1, c_2, c_3, “300”), \text{Emp}(b_1, a_1, d_3, d_4) \]

The first and third atoms cannot be removed (Why?)

We check whether we can remove the second atom:

\[ Q_2(“1”) \leftarrow \text{Dept}(a_1, a_2, “1”, a_4), \text{Finance}(a_1, c_2, c_3, “300”), \text{Emp}(b_1, a_1, d_3, d_4) \]

The corresponding canonical database: \( D_2(“1”) = \{\text{Dept}(a_1, a_2, “1”, a_4), \text{Finance}(a_1, c_2, c_3, “300”), \text{Emp}(b_1, a_1, d_3, d_4)\} \)

Clearly ("1") \( \in Q_1(D_2) \) because of the matching

\[
\begin{align*}
a_1 &\rightarrow a_1 & a_2 &\rightarrow a_2 & a_4 &\rightarrow a_4 \\
b_1 &\rightarrow b_1 & b_3 &\rightarrow d_3 & b_4 &\rightarrow d_4 \\
c_2 &\rightarrow c_2 & c_3 &\rightarrow c_3 & d_3 &\rightarrow d_3 & d_4 &\rightarrow d_4
\end{align*}
\]

hence \( Q_2 \subseteq Q_1 \). The other direction always holds. Hence \( Q_1 \equiv Q_2 \)
Optimization of Logical Queries

Redundant Joins Removal

No other atom can be removed (Why?).

The optimal query is hence

\[ Q_2(“1”) \leftarrow \text{Dept}(a_1, a_2, “1”, a_4), \text{Finance}(a_1, c_2, c_3, “300”), \text{Emp}(b_1, a_1, d_3, d_4) \]

Translating this query back to the relational algebra, we obtain:

\[
\pi_{D.\text{floor}}(\left[ \sigma_{D.\text{floor}=1 \land E_2.\text{did}=D.\text{did} \land F_2.\text{did}=E_2.\text{did} \land E_2.\text{did}=D.\text{did} \land F_2.\text{expenses}=300} \\
(\rho_D(\text{Dept}) \times \rho_{F_2}(\text{Finance}) \times \rho_{E_2}(\text{Emp})) \right])
\]

Solution of the exercises
Optimization of Logical Queries

RedundantJoinsRemoval

The second expression is:

\[
\left[ \pi_{D.*} \sigma_{F_2.did=E.did \land E_2.did=D.did \land E_2.eid=E.eid \land F_2.expenses=300 \land E.did=F_2.did} (\rho_D(Dept) \times \rho_E(Emp) \times \rho_{F_2}(Finance) \times \rho_{E_2}(Emp)) \right]
\]

Translated:

\[Q_3(a_1, a_2, a_3, a_4, b_1, b_3, b_4) \leftarrow \text{Dept}(a_1, a_2, a_3, a_4), \text{Emp}(b_1, a_1, b_3, b_4), \text{Finance}(a_1, c_2, c_3, "300"), \text{Emp}(b_1, a_1, d_3, d_4)\]

We cannot remove the second atom, this time (why?)
Optimization of Logical Queries

Redundant Joins Removal

However, with a similar mapping as for the first expression, the fourth atom can be removed, and we obtain:

\[ Q_4(a_1, a_2, a_3, a_4, b_1, b_3, b_4) \leftarrow \text{Dept}(a_1, a_2, a_3, a_4), \text{Emp}(b_1, a_1, b_3, b_4), \text{Finance}(a_1, c_2, c_3, "300") \]

We have thus \( Q_4 \subseteq Q_3 \). The other direction always holds. Hence \( Q_3 \equiv Q_4 \).

Translating this query back to the relational algebra, we obtain:

\[
[\pi_{D.\ast,E.\ast}\sigma_{F_2.did=E.did \land E.did=D.did \land F_2.expenses=300 \land E.did=F_2.did} (\rho_D(\text{Dept}) \times \rho_E(\text{Emp}) \times \rho_{F_2}(\text{Finance}))]
\]
Optimization of Logical Queries

Redundant Joins Removal

The third expression is:

$$(\rho_D(\text{Dept}) \times \rho_{D2}(\text{Dept}) \times \rho_{F1}(\text{Finance}))$$

Translated:

$$Q_5(a_1, \ldots, a_4, b_1, \ldots, b_4, c_1, \ldots, c_4) \leftarrow \text{Dept}(a_1, a_2, a_3, a_4), \text{Dept}(b_1, b_2, b_3, b_4), \text{Finance}(c_1, c_2, c_3, c_4)$$

No atoms can be removed (why?)
Optimization of Logical Queries

Redundant Joins Removal

The optimized expression is therefore:

\[ e = \pi_{D.\text{floor}}\left( [\sigma_{D.\text{floor}=1} \land E_2.\text{did}=D.\text{did} \land F_2.\text{did}=E_2.\text{did} \land E_2.\text{did}=D.\text{did} \land F_2.\text{expenses}=300 \right. \]
\[ \left. (\rho_D(\text{Dept}) \times \rho_{F_2}(\text{Finance}) \times \rho_{E_2}(\text{Emp})) \right] \]
\[ \cup \]
\[ \pi_{D.\text{floor}}\left( \left[ \pi_{D.*,E.*} [\sigma_{F_2.\text{did}=E.\text{did} \land E.\text{did}=D.\text{did} \land F_2.\text{expenses}=300 \land E.\text{did}=F_2.\text{did} \right. \right. \]
\[ \left. \left. (\rho_D(\text{Dept}) \times \rho_E(\text{Emp}) \times \rho_{F_2}(\text{Finance})) \right] \right. \]
\[ \bigtimes \left[ \pi_{D.*} [\sigma_{F_1.\text{budget}>150} \land D_2.\text{did}=F_1.\text{did} \land D_2.\text{floor}=D.\text{floor} \right. \[
\left. \left. (\rho_D(\text{Dept}) \times \rho_{D_2}(\text{Dept}) \times \rho_{F_1}(\text{Finance})) \right] \right. \]

Solution of the exercises
Cost-based plan selection

Task

(refer to the handouts for the full exercise)

Construct a sufficiently optimal physical query plan for:

$$\pi_{E.eid, D.did, P.pid} \sigma_{E.sal = 50000}(E) \bowtie \sigma_{D.budget \geq 20000}(D) \bowtie P$$

Assume that employee salaries are uniformly distributed over the range $[10009, 110008]$ and that project budgets are uniformly distributed over $[10000, 30000]$. There are clustered indexes available on $E.sal$, $D.did$ and $P.pid$. 
Cost-based plan selection

Solution

\[
\begin{align*}
\pi & \quad \Join \quad \sigma \quad E \\
\sigma & \quad \Join \quad \sigma \quad D \\
\Join & \quad \sigma \quad P
\end{align*}
\]
Cost-based plan selection

Solution

\[ \pi \downarrow \sigma \downarrow E \quad \sigma \downarrow D \quad \varnothing \quad P \]
Cost-based plan selection

Solution

Subexpression:

$$\sigma_{E.sal = 50000}(E)$$

First possibility: we use the clustered index on E.sal to get the records such that E.sal = 50000.

The number of tuples that satisfy the salary requirement is estimated to:

$$\left\lceil \frac{1}{110008 - 10009 + 1} \times 20000 \right\rceil = 1 \text{ tuples}$$

Hence, the result can be stored in 1 block:

$$\left\lceil \frac{20 \text{ bytes}}{4000 \text{ bytes/block}} \right\rceil = 1 \text{ block}$$

A table scan would cost:

$$\left\lceil \frac{20000 \text{ tuples}}{4000 \text{ bytes/block}} \div \frac{20 \text{ bytes/tuple}}{20 \text{ bytes/tuple}} \right\rceil = 100 \text{ block I/Os}$$
Cost-based plan selection

Solution

$T = 1$
$B = 1$
Cost-based plan selection

Solution

\[
\pi \Join \sigma E \sigma D P
\]
Cost-based plan selection

Solution

Subexpression:

\[ \sigma_{D.\text{budget} \geq 20000}(D) \]

The number of tuples returned is estimated to 2501:

\[ \left\lceil \frac{30000 - 20000 + 1}{30000 - 10000 + 1} \times 5000 \text{ departments} \right\rceil = 2501 \text{ tuples} \]

This corresponds to 26 Blocks:

\[ \frac{2501}{\frac{4000 \text{ bytes/block}}{40 \text{ bytes/tuple}}} = 26 \text{ blocks} \]

Since no index is available, a table scan is our only possibility:

\[ \frac{5000}{\frac{4000 \text{ bytes/block}}{40 \text{ bytes/tuple}}} = 50 \text{ blocks} \]
Cost-based plan selection

Solution

\[ \pi \bowtie \sigma_E \sigma_D P = 2501 \]
\[ B = 26 \]

Solution of the exercises 27
Cost-based plan selection

Solution

\[
\begin{align*}
\pi \\
\Join \\
\sigma \\
\sigma \\
E \\
D \\
P
\end{align*}
\]
Cost-based plan selection

Solution

Subexpression: \( P \)

A table scan on \( P \) requires 500 block I/O’s. This is also the estimated number of blocks returned:

\[
\frac{1000 \text{ tuples}}{4000 \text{ bytes/block}} \times \frac{4000 \text{ bytes/block}}{2000 \text{ bytes/tuple}} = 500 \text{ blocks}
\]
Cost-based plan selection

Solution

\[
\pi \bigtriangleup \sigma \ E \ \sigma \ D \ P
\]

\[
T = 1000 \\
B = 500
\]
Cost-based plan selection

Solution

Solution of the exercises 31
Cost-based plan selection

Solution

Now, we must determine an ordering for the joins. We consider all pairs of joins and keep the one with the smallest cost.

\[
\sigma_{e.\text{sal}=50000}(E) \quad \text{and} \quad \sigma_{d.\text{budget}\geq20000}(D)
\]

The selection on each side requires one buffer to execute, leaving only 10 buffers for the join.

The output of \(e_1\) contains only 1 tuples, and can therefore be computed in 1 block. Since \(1 = B(e_1) \leq M = 10\), we can apply the one-pass join algorithm. Its cost is

\[
B(e_1) + B(e_2) = 1 + 26 = 27 \text{ I/O's}
\]

An index-join cannot be used on \(e_2\) since it is not a base relation. All other join methods always cost more than one-pass join. Hence the one-pass join is preferred.
Cost-based plan selection

Solution

Solution of the exercises 33
Cost-based plan selection

Solution

The second join pair is:

\[ \sigma_{D{.budget \geq 20000}(D) \quad \text{and} \quad P} \]

We have 11 buffers at our disposal, given that we need 1 buffer to perform the selection in \( e_2 \). It is not possible to use a one-pass join, since \( 26 = B(e_2) \geq M = 11 \) and \( 500 = B(P) \geq M = 11 \).

We have enough memory to perform an optimized sort-merge join:

\[
8 = \left\lfloor \frac{B(e_2)}{M \left\lceil \log_M B(e_2) \right\rceil - 1} \right\rfloor + \left\lfloor \frac{B(P)}{M \left\lceil \log_M B(P) \right\rceil - 1} \right\rfloor \leq M = 11 \text{ available buffers}
\]

This optimized sort-merge join has a cost of:

\[
2B(e_2) \left\lceil \log_M B(e_2) \right\rceil + 2B(P) \left\lceil \log_M B(P) \right\rceil - B(e_2) - B(P)
\]

\[
= 2 \times 26 \times 2 + 2 \times 500 \times 3 - 26 - 500
\]

\[
= 2578 \text{ I/O's}
\]
Cost-based plan selection

Solution

Assuming that the clustered index on P.pid is a $BTree$, it ensues that $P$ is already sorted on this join attribute. Given that we then only need to sort $e_2$, the cost of a non-optimized sort-merge join is:

$$2B(e_2) \lceil \log_M B(e_2) \rceil + B(e_2) + B(P)$$

Furthermore, we can optimize the last merge:

4 necessary buffers = $\left\lceil \frac{B(e_2)}{M} \right\rceil + 1 \leq M = 11$ available buffers

The cost thereof is:

$$2B(e_2)(\lceil \log_M B(e_2) \rceil - 1) + B(e_2) + B(P)$$

= $2 \times 26 \times 1 + 26 + 500$

= 578 I/Os
Cost-based plan selection

Solution

The cost of an hash-join is:

$$2B(e_2) \left\lceil \log_{M-1} B(e_2) - 1 \right\rceil + 2B(P) \left\lceil \log_{M-1} B(e_2) - 1 \right\rceil + B(e_2) + B(P)$$

$$= 2 \times 26 \times 1 + 2 \times 500 \times 2 + 26 + 500$$

$$= 2578 \text{ I/O’s}$$

It is also possible to use an index-join, using the clustered index on \( P.p\ id \). This method has a cost of:

$$B(e_2) + T(e_2) \times \left\lceil \frac{B(P)}{V(P, p\ id)} \right\rceil = 26 + 2501 \times 1 = 2527 \text{ I/O’s}$$

Hence, the optimized sort-merge join and hash join have the same cost. We assume that the index on \( P.p\ id \) is a BTree, and sorting \( P \) is not necessary. In that case the optimized sort-merge join that only sorts \( e_2 \) is preferred.
Cost-based plan selection

Solution

Solution of the exercises 37
Cost-based plan selection

Solution

The third join pair is:

\[ \sigma_{E.\text{sal}=50000}(E) \quad \text{and} \quad P \]

Note that this join is a full cartesian product. A one-pass join is available at the following cost:

\[ B(e_1) + B(P) = 1 + 500 = 501 \text{ I/O’s} \]

No index can help up for this join, and the one-pass join algorithm gives the best cost.
Cost-based plan selection

Solution

\[
\begin{align*}
\pi & \bowtie \sigma \\
E & \sigma \quad D & P
\end{align*}
\]

Cost = 27
Single pass join

\[
\begin{align*}
\pi & \bowtie \sigma \\
E & \sigma \quad D & P
\end{align*}
\]

Cost = 578
Optimized sort-merge join

\[
\begin{align*}
\pi & \bowtie \sigma \\
D & \sigma \quad P
\end{align*}
\]

Cost = 501
Single pass join

Solution of the exercises 39
Cost-based plan selection

Solution

The join-pair with the least cost is therefore:

$$\sigma_{e.\text{sal}=50000}(E) \quad \text{and} \quad \sigma_{D.\text{budget} \geq 20000}(D)$$

Where an one-pass join on $E.\text{did}$ is used. Therefore, only 2 buffers are necessary (why?).

The estimated number of tuples in the output of this join is:

$$\frac{T(e_1) \times T(e_2)}{\max(V(e_1, \text{did}), V(e_2, \text{did}))} = \frac{1 \times 2501}{20} = 126$$

These records are 60 bytes long and can be stored in 2 blocks
Cost-based plan selection

Solution

\[
\pi \sigma \sigma E \sigma D P
\]

\[
T = 126 \\
B = 2
\]
Cost-based plan selection

Solution

\[\pi \bowtie \sigma \sigma \preceq E \bowtie \sigma \preceq D \bowtie P\]
Cost-based plan selection

Solution

We still need to find the best way to join the whole expression

\[ \sigma_{e\cdot \text{sal}=50000}(E) \bowtie_{e_3} \sigma_{D\cdot \text{budget}\geq20000}(D) \] and \( P \)

The output of \( e_3 \) fits in 2 blocks. Given that \( 2 = B(e_3) \leq M = 12 \), a one-pass join is possible. The cost thereof is:

\[ B(e_3) + B(P) = 2 + 500 = 502 \]

This joins can also be performed by means of an index-join, using the clustered index on \( P\cdot \text{pid} \).

\[ B(e_3) + T(e_3) \times \left[ \frac{B(P)}{V(P, \text{pid})} \right] = 2 + 125 \times 1 = 127 \text{ I/O's} \]

Hence, the index-join is preferred.
Cost-based plan selection

Solution

\[ \pi \bigcirc \bigoplus \sigma E \bigcirc \bigoplus \sigma D \bigcirc \bigcirc P \]
Cost-based plan selection

**Solution**

The projection \( \pi_{E.eid,D.did,P.pid} \) can be performed on the fly at the same time as the last join.

Notice that we did not need to materialize any of the intermediate results.
Cost-based plan selection

Solution

Solution of the exercises 46