Homework 1 Distributed Database Systems

Christoffer Brevik

November 5, 2024

Database used in assignment

Due to the table and variable names used in the assignments. I assume we still use the database described as *Figure 5.3* in the book, for the assignments. Below I show this figure, with the exception of removing the `PAY` table as it is not mentioned in any of the problems:

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E2	M. Smith	Sys. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Sys. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Sys. Anal.

ASG

ENO	PNO	RESP	DUR
E1	P1	Manager	12
E2	P1	Analysist	24
E2	P2	Analysist	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E8	P3	Manager	40

PROJ

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150,000	Montreal
P2	Database Develop.	135,000	New York
P3	CAD/CAM	250,000	New York
P4	Maintenance	310,000	Paris

Problem 8.3

Original Query

We have the following query:

```
SELECT  ENAME, PNAME
FROM    EMP, ASG, PROJ
WHERE   (DUR > 12 OR RESP = "Analyst")
AND     EMP.ENO = ASG.ENO
AND     (TITLE = "Elect. Eng." OR ASG.PNO < "P3")
AND     (DUR > 12 OR RESP NOT= "Analyst")
AND     ASG.PNO = PROJ.PNO
```

DUR and RESP

We see that we have:

```
WHERE   (DUR > 12 OR RESP = "Analyst")
AND     (DUR > 12 OR RESP NOT= "Analyst")
```

Here the `RESP = "Analyst"` and `RESP NOT = "Analyst"` will cancel each other out, and therefore this query only passes contracts with a duration over 12 months. We can therefore only check for this, simplifying the query to:

```
SELECT  ENAME, PNAME
FROM    EMP, ASG, PROJ
WHERE   DUR > 12
AND     EMP.ENO = ASG.ENO
AND     ASG.PNO = PROJ.PNO
AND     (TITLE = "Elect. Eng." OR ASG.PNO < "P3")
```

JOIN *(Not strictly required)*

The query implements a combination of EMP, ASG and PROJ. Therefore we can specify that we actually join the tables by using the `JOIN` query.

```
SELECT ENAME, PNAME
FROM EMP
JOIN ASG ON EMP.ENO = ASG.ENO
JOIN PROJ ON ASG.PNO = PROJ.PNO
WHERE DUR > 12
AND (TITLE = "Elect. Eng." OR ASG.PNO < "P3")
```

Transformed into optimized operator tree

We will not make the query into an optimazed operator tree, also called query tree. Below is a figure made using the tikzpicture library. I've structured the tree as to handle the least amount of data throughout the quering:

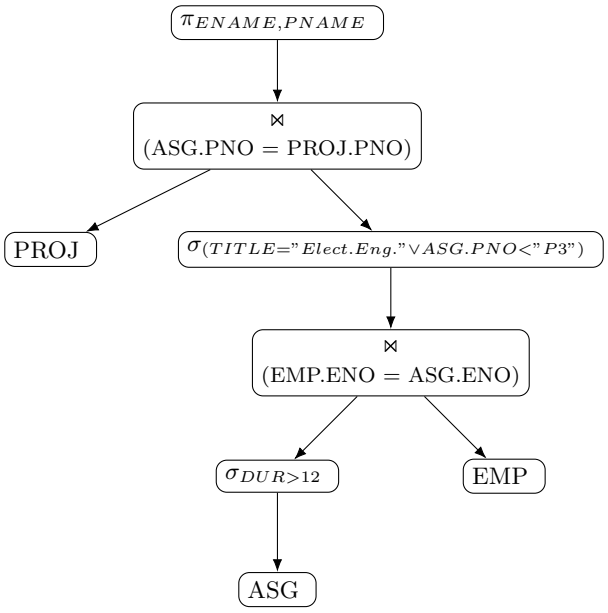

Figure 1: Optimized Query Tree for Problem 8.3

Problem 8.8

From the assignment we have the following query: We also have the following query:

```
SELECT  ENAME
FROM    EMP,ASG,PROJ
WHERE   PROJ.PNO = ASG.PNO
AND     PNAME = "Instrumentation"
AND     EMP.ENO = ASG.ENO
```

We also have the following fragments:

- $PROJ_1$ and $PROJ_2$ indirectly fragmented using the values of `PNO`:
  - $PROJ_1 = \sigma_{\texttt{PNO} \leq \text{"P2"}}(PROJ)$
  - $PROJ_2 = \sigma_{\texttt{PNO} > \text{"P2"}}(PROJ)$
- ASG_1 and ASG_2 based on join with `PROJ`:
  - $ASG_1 = ASG \bowtie_{\texttt{PNO}} PROJ1$
  - $ASG_2 = ASG \bowtie_{\texttt{PNO}} PROJ2$
- EMP_1 and EMP_2 based on a vertical fragmentaton of EMP, where the fragments contain:
  - EMP_1 = ENO, ENAME
  - EMP_2 = ENO, TITLE

Firstly we see that the query only needs the ENAME-value from the query. Therefore we see that we don't need EMP_2. Furthermore we only want projects with the PNAME-value of "Instrumentation". By looking at out figures we see that this PNAME value is dependent on the primary key PNO=1, which is also the basis for the fragmentation of $PROJ$. We see that $PROJ_1$ contains the data for $PNO <= 1$, so we only need this fragment. As ASG is dependent on $PROJ$, we only need ASG_1 for our query too.

Knowing this, we can only use the relevant fragments, concluding with the following query:

```
SELECT ENAME
FROM EMP1, ASG1, PROJ1
WHERE PROJ1.PNO = ASG1.PNO
AND EMP1.ENO = ASG1.ENO
AND PNAME = "Instrumentation"
```

Problem 9.2

Given Information

- Relations and Sites:
  - `EMP` is located at Site 1.
  - `ASG` is located at Site 2.
  - `PROJ` is located at Site 3.
- Sizes of Relations:

$$\begin{aligned} \text{size}(\texttt{EMP}) &= 100 \\ \text{size}(\texttt{ASG}) &= 200 \\ \text{size}(\texttt{PROJ}) &= 300 \end{aligned}$$

- Join Sizes:

$$\begin{aligned} \text{size}(\texttt{EMP} \bowtie \texttt{ASG}) &= 200 \\ \text{size}(\texttt{ASG} \bowtie \texttt{PROJ}) &= 200 \end{aligned}$$

Optimal Join Program

Our objective is to minimize the total transmission time by choosing an optimal order and location for joins. This means minimizing the amount of data we transfer between the sites. The join sizes are equal, so here we can focus on transferring the smallest relations first, making sure we only send the smallest relations and joins (Therefore ending on the site containing the largest relation $PROJ$):

1. **First Join (EMP and ASG):** Since size(`EMP`) = 100 and size(`ASG`) = 200, we join `EMP` and `ASG` first.
   - Perform this join at **Site 2** (where `ASG` is located), we sendt `EMP` from Site 1 to Site 2.
   - Transmission Cost: Transfer of `EMP` (100 units) from Site 1 to Site 2.
2. **Second Join (ASG-PROJ):** The result of $\texttt{EMP} \bowtie \texttt{ASG}$ (size 200) is then joined with `PROJ`.
   - Perform this join at **Site 3** (where `PROJ` is located), transferring the 200 tuples of $\texttt{EMP} \bowtie \texttt{ASG}$ from Site 2 to Site 3.
   - Transmission Cost: Transfer of $\texttt{EMP} \bowtie \texttt{ASG}$ (200 units) from Site 2 to Site 3.

Total Transmission Cost

Transmission of `EMP` from Site 1 to Site 2: 100 units

Transmission of $\texttt{EMP} \bowtie \texttt{ASG}$ result from Site 2 to Site 3: 200 units

Total Transmission Cost = $100 + 200 = 300$ units