Query Decomposition and Data Localization

Query Decomposition and Data Localization consists of two steps:

- Mapping of calculus query (SQL) to algebra operations (select, project, join, rename) [Query Decomposition]
- Applying data distribution information to the algebra operations [Data Localization]

Query Decomposition

- Normalization
  - Manipulate query quantifiers and qualification
- Analysis
  - Detect and reject “incorrect” queries
- Simplification
  - Eliminate redundant predicates
- Restructuring
  - Use transformation rules to optimize query

Lexical and syntactic analysis
- Check validity (similar to compilers)
- Check for attributes and relations
- Type checking on the qualification

Put into normal form
- Conjunctive normal form
  \[
  (p_{11} \lor p_{12} \lor \cdots \lor p_{1n}) \land \cdots \land (p_{m1} \lor p_{m2} \lor \cdots \lor p_{mn})
  \]
- Disjunctive normal form
  \[
  (p_{11} \land p_{12} \land \cdots \land p_{1n}) \lor \cdots \lor (p_{m1} \land p_{m2} \land \cdots \land p_{mn})
  \]
- OR’s mapped into union
- AND’s mapped into join or selection
Consider a query:

SELECT ENAME, RESP
FROM EMP, ASG, PROJ
WHERE EMP.ENO = ASG.ENO
AND ASG.PNO = PROJ.PNO
AND PNAME = "CAD/CAM"
AND DUR ≥ 36
AND TITLE = "Programmer"

If the graph is not connected the query is wrong.
Consider the following query:

```sql
SELECT TITLE
FROM EMP
WHERE EMP.ENAME = "J. Doe"
OR (NOT(EMP.TITLE = "Programmer")
    AND EMP.TITLE = "Elect. Eng.")
AND NOT(EMP.TITLE = "Programmer")
```

Simplified query:

```sql
SELECT TITLE
FROM EMP
WHERE EMP.ENAME = "J. Doe"
```

---

Commutativity of binary operations
- \( R \times S = S \times R \)
- \( R \otimes S = S \otimes R \)
- \( R \cup S = S \cup R \)

Associativity of binary operations
- \( (R \times S) \times T = R \times (S \times T) \)
- \( (R \otimes S) \otimes T = R \otimes (S \otimes T) \)

Idempotence of unary operations
- \( \Pi_A(\Pi_A(R)) = \Pi_A(R) \)
- \( \sigma_{p1(A1)}(\sigma_{p2(A2)}(R)) = \sigma_{p1(A1) \land p2(A2)}(R) \)

Commuting selection with binary operations
- \( \sigma_{p(A)}(R \times S) \iff (\sigma_{p(A1)}(R)) \times S \)
- \( \sigma_{p(A1)}(R \otimes (A2, B2)) \iff (\sigma_{p(A1)}(R) \otimes (A2, B2)) \)
- \( \sigma_{p(A)}(R \cup T) \iff \sigma_{p(A1)}(R) \cup \sigma_{p(A1)}(T) \)
- \( A \) belongs to \( R \) and \( T \)

Commuting projection with binary operations
- \( \Pi_C(R \times S) \iff \Pi_{A1}(R) \times \Pi_{B1}(S) \)
- \( \Pi_C(R \otimes (A,B)) \iff \Pi_{A1}(R) \otimes (A,B) \Pi_{B1}(S) \)
- \( \Pi_C(R \cup S) \iff \Pi_{A1}(R) \cup \Pi_{A1}(S) \)
- \( C = A' \cup B', A' \subseteq A, B' \subseteq B \)
Data Localization

**Input:** algebraic query on global conceptual schema

**Purpose:** determine which fragments are involved
- Substitute global query with data manipulation queries on fragments
- Optimize the global query

**Example:** Assume EMP is fragmented into EMP1, EMP2, EMP3 as follows:
- EMP1 = ENO ≤ "E3"(EMP)
- EMP2 = "E3" < ENO ≤ "E6"(EMP)
- EMP3 = ENO > "E6"(EMP)

**Example:** ASG fragmented into ASG1 and ASG2 as follows:
- ASG1 = ENO ≤ "E3"(ASG)
- ASG2 = ENO > "E3"(ASG)

**Example:** Replace EMP by (EMP1 ∪ EMP2 ∪ EMP3) and ASG by (ASG1 ∪ ASG2) in all queries
Data Localizations Issues

- Reduction of horizontal fragmentation (HF)
  - Reduction with selection
  - Reduction with join
- Reduction of vertical fragmentation (VF)
  - Find empty relations

Consider relation \( R \) with horizontal fragmentation \( F = \{ R_1, R_2, \ldots, R_k \} \), where \( R_i = \sigma_{p_i}(R) \)

Rule can be formalized by the following:

\[
\sigma_{p_j}(R_i) = \emptyset \iff p_i(x) \land p_j(x) = false, x \in R_i
\]

Beneficial when fragmentation predicate is inconsistent with the query selection predicate (returns empty relations)

Example:

```sql
SELECT * FROM EMP WHERE ENO = 'E5'
```
**Data Localizations Issues (Reduction of HF with Join 2/4)**

- **Possible if fragmentation is done on the join attribute**
- **Distribute join over union**
  \[(R_1 \cup R_2) \times S \iff (R_1 \times S) \cup (R_2 \times S)\]
- **Rule can be formalized by the following. Given** \(R_i = \sigma_{p_i}(R)\) and \(R_j = \sigma_{p_j}(R)\), \(j = 1, 2, \ldots, k\)
  \[R_i \times R_j = \emptyset \iff p_i(x) \land p_j(x) = false, x \in R_j\]

**Example**

- EMP1 = ENO \(\leq\) E3(EMP)
- EMP2 = E3 \(\leq\) ENO \(\leq\) E6(EMP)
- EMP3 = ENO \(\geq\) E6(EMP)
- ASG1 = ENO \(\leq\) E3(ASG)
- ASG2 = ENO \(\geq\) E3(ASG)

**Query**

```
SELECT * FROM EMP, ASG WHERE EMP.ENO = ASG.ENO
```

---

**Data Localizations Issues (Reduction of HF with Join 3/4)**

- **Fragmentation:**
  - EMP1 = ENO \(\leq\) E3(EMP)
  - EMP2 = E3 \(\leq\) ENO \(\leq\) E6(EMP)
  - EMP3 = ENO \(\geq\) E6(EMP)
  - ASG1 = ENO \(\leq\) E3(ASG)
  - ASG2 = ENO \(\geq\) E3(ASG)

**Query**

```
SELECT * FROM EMP, ASG WHERE EMP.ENO = ASG.ENO
```

---

**Data Localizations Issues (Derived Horizontal Fragmentation 4/4)**

- **Assume a good fragmentation of EMP is**
  - EMP1 = \(\sigma_{\text{TITLE} = \text{"Programmer"}}(\text{EMP})\)
  - EMP2 = \(\sigma_{\text{TITLE} \neq \text{"Programmer"}}(\text{EMP})\)
- **To achieve efficient joins one can also fragment ASG:**
  - ASG1 = ASG \(\leq\) ENO EMP1
  - ASG2 = ASG \(\geq\) ENO EMP2

---

**Data Localizations Issues (Reduction of VF)**

- **Find empty relations**
- **Let** \(R\) **be fragmented into** \(R_1, R_2, \ldots, R_k\), \(R_i = \Pi_A(R), A_i \subseteq A\). **Then** \(\Pi_D(R_i)\) **is empty iff** \(D\) **is not in** \(A_i\).
- **For example**
  - EMP1 = \(\Pi_{\text{ENO}, \text{ENAME}}(\text{EMP})\)
  - EMP2 = \(\Pi_{\text{ENO}, \text{TITLE}}(\text{EMP})\)
- **Query**

```
SELECT \text{ENAME} FROM EMP
```

---
Query decomposition and data localization maps calculus query into algebra operations and applies data distribution information to the algebra operations.

Query decomposition consists of normalization, analysis, simplification, and restructuring.

Data localization reduces horizontal fragmentation with join and selection, and vertical fragmentation with joins, and aims to find empty relations.