2. Textbook Query Optimization

- Algebra Revisited
- Canonical Query Translation
- Logical Query Optimization
- Physical Query Optimization

Algebra Revisited

The algebra needs some more thought:

- correctness is critical for query optimization
- can only be guaranteed by a formal model
- the algebra description in the introduction was too cursory

What we ultimately want to do with an algebraic model:

- decide if two algebraic expressions are equivalent (produce the same result)

This is too difficult in practice (not computable in general), so we at least want to:

- guarantee that two algebraic expressions are equivalent (for some classes of expressions)

This still requires a strong formal model. We accept false negatives, but not false positives.

Tuples

Tuple:

- a (unordered) mapping from attribute names to values of a domain
- sample: [name: "Sokrates", age: 69]

Schema:

- a set of attributes with domain, written $\mathcal{A}(t)$
- sample: {(name,string),(age, number)}

Note:

- simplified notation on the slides, but has to be kept in mind
- domain usually omitted when not relevant
- attribute names omitted when schema known

Tuple Concatenation

- notation: $t_1 \circ t_2$
- sample: [name: "Sokrates", age: 69]$\circ$[country: "Greece"]
  = [name: "Sokrates", age: 69, country: "Greece"]
- note: $t_1 \circ t_2 = t_2 \circ t_1$, tuples are unordered

Requirements/Effects:

- $\mathcal{A}(t_1) \cap \mathcal{A}(t_2) = \emptyset$
- $\mathcal{A}(t_1 \circ t_2) = \mathcal{A}(t_1) \cup \mathcal{A}(t_2)$

Tuple Projection

Consider t = [name: "Sokrates", age: 69, country: "Greece"]

Single Attribute:

- notation $t.a$
- sample: $t.name =$ "Sokrates"

Multiple Attributes:

- notation $t_{|A}$
- sample: $t_{|\{name,age\}} =$ [name: "Sokrates", age: 69]

Requirements/Effects:

- $a \in \mathcal{A}(t)$, $A \subseteq \mathcal{A}(t)$
- $\mathcal{A}(t_{|A}) = A$
- notice: $t.a$ produces a value, $t_{|A}$ produces a tuple

Relations

Relation:

- a set of tuples with the same schema
- sample: {[name: "Sokrates", age: 69], [name: "Platon", age: 45]}

Schema:

- schema of the contained tuples, written $\mathcal{A}(R)$
- sample: {(name,string),(age, number)}

Sets vs. Bags

- relations are sets of tuples
- real data is usually a multi set (bag)

Example:

```
select age
from student
```

age
23
24
24
...

- we concentrate on sets first for simplicity
- many (but not all) set equivalences valid for bags

The optimizer must consider three different semantics:

- logical algebra operates on bags
- physical algebra operates on streams (order matters)
- explicit duplicate elimination ⇒ sets

Set Operations

Set operations are part of the algebra:

- union ($L \cup R$), intersection ($L \cap R$), difference ($L \setminus R$)
- normal set semantic
- but: schema constraints
- for bags defined via frequencies (union $\rightarrow +$, intersection $\rightarrow$ min, difference $\rightarrow -$)

Requirements/Effects:

- $\mathcal{A}(L) = \mathcal{A}(R)$
- $\mathcal{A}(L \cup R) = \mathcal{A}(L) = \mathcal{A}(R)$, $\mathcal{A}(L \cap R) = \mathcal{A}(L) = \mathcal{A}(R)$, $\mathcal{A}(L \setminus R) = \mathcal{A}(L) = \mathcal{A}(R)$

Free Variables

Consider the predicate *age* $= 62$

- can only be evaluated when *age* has a meaning
- *age* behaves a free variable
- must be bound before the predicate can be evaluated
- notation: $\mathcal{F}(e)$ are the free variables of e

Note:

- free variables are essential for predicates
- free variables are also important for algebra expressions
- dependent join etc.

Selection

Selection:

- notation: $\sigma_p(R)$
- sample: $\sigma_{a \geq 2}(\{[a:1],[a:2],[a:3]\}) = \{[a:2],[a:3]\}$
- predicates can be arbitrarily complex
- optimizer especially interested in predicates of the form $attrib = attrib$ or $attrib = const$

Requirements/Effects:

- $\mathcal{F}(p) \subseteq \mathcal{A}(R)$
- $\mathcal{A}(\sigma_p(R)) = \mathcal{A}(R)$

Projection

Projection:

- notation: $\Pi_A(R)$
- sample: $\Pi_{\{a\}}(\{[a:1,b:1],[a:2,b:1]\}) = \{[a:1],[a:2]\}$
- eliminates duplicates for set semantic, keeps them for bag semantic
- note: usually written as $\Pi_{a,b}$ instead of the correct $\Pi_{\{a,b\}}$

Requirements/Effects:

- $A \subseteq \mathcal{A}(R)$
- $\mathcal{A}(\Pi_A(R)) = A$

Rename

Rename:

- notation: $\rho_{a \to b}(R)$
- sample:
  $\rho_{a \to c}(\{[a:1, b:1], [a:2, b:1]\}) = \{[c:1, b:1], [c:2, b:2]\}$?
- often a pure logical operator, no code generation
- important for the data flow

Requirements/Effects:

- $a \in \mathcal{A}(R), b \notin \mathcal{A}(R)$
- $\mathcal{A}(\rho_{a \to b}(R)) = \mathcal{A}(R) \setminus \{a\} \cup \{b\}$

Join

Consider $L = \{[a:1],[a:2]\}, R = \{[b:1],[b:3]\}$

Cross Product:

- notation: $L \times R$
- sample: $L \times R = \{[a:1,b:1],[a:1,b:3],[a:2,b:1],[a:2,b:3]\}$

Join:

- notation: $L \bowtie_p R$
- sample: $L \bowtie_{a=b} R = \{[a:1,b:1]\}$
- defined as $\sigma_p(L \times R)$

Requirements/Effects:

- $\mathcal{A}(L) \cap \mathcal{A}(R) = \emptyset, \mathcal{F}(p) \in (\mathcal{A}(L) \cup \mathcal{A}(R))$
- $\mathcal{A}(L \times R) = \mathcal{A}(L) \cup \mathcal{R}$

Equivalences

Equivalences for selection and projection:

$$
\begin{aligned}
\sigma_{p_1 \wedge p_2}(e) &\equiv \sigma_{p_1}(\sigma_{p_2}(e)) && (1)\\
\sigma_{p_1}(\sigma_{p_2}(e)) &\equiv \sigma_{p_2}(\sigma_{p_1}(e)) && (2)\\
\Pi_{A_1}(\Pi_{A_2}(e)) &\equiv \Pi_{A_1}(e) && (3)\\
& \quad \text{if } A_1 \subseteq A_2 &&\\
\sigma_p(\Pi_A(e)) &\equiv \Pi_A(\sigma_p(e)) && (4)\\
& \quad \text{if } \mathcal{F}(p) \subseteq A &&\\
\sigma_p(e_1 \cup e_2) &\equiv \sigma_p(e_1) \cup \sigma_p(e_2) && (5)\\
\sigma_p(e_1 \cap e_2) &\equiv \sigma_p(e_1) \cap \sigma_p(e_2) && (6)\\
\sigma_p(e_1 \setminus e_2) &\equiv \sigma_p(e_1) \setminus \sigma_p(e_2) && (7)\\
\Pi_A(e_1 \cup e_2) &\equiv \Pi_A(e_1) \cup \Pi_A(e_2) && (8)
\end{aligned}
$$

Equivalences

Equivalences for joins:

$$e_1 \times e_2 \equiv e_2 \times e_1 \quad (9)$$

$$e_1 \bowtie_p e_2 \equiv e_2 \bowtie_p e_1 \quad (10)$$

$$(e_1 \times e_2) \times e_3 \equiv e_1 \times (e_2 \times e_3) \quad (11)$$

$$(e_1 \bowtie_{p_1} e_2) \bowtie_{p_2} e_3 \equiv e_1 \bowtie_{p_1} (e_2 \bowtie_{p_2} e_3) \quad (12)$$

$$\sigma_p(e_1 \times e_2) \equiv e_1 \bowtie_p e_2 \quad (13)$$

$$\sigma_p(e_1 \times e_2) \equiv \sigma_p(e_1) \times e_2 \quad (14)$$

$$\text{if } \mathcal{F}(p) \subseteq \mathcal{A}(e_1)$$

$$\sigma_{p_1}(e_1 \bowtie_{p_2} e_2) \equiv \sigma_{p_1}(e_1) \bowtie_{p_2} e_2 \quad (15)$$

$$\text{if } \mathcal{F}(p_1) \subseteq \mathcal{A}(e_1)$$

$$\Pi_A(e_1 \times e_2) \equiv \Pi_{A_1}(e_1) \times \Pi_{A_2}(e_2) \quad (16)$$

$$\text{if } A = A_1 \cup A_2, A_1 \subseteq \mathcal{A}(e_1), A_2 \subseteq \mathcal{A}(e_2)$$