CSE460/560 DATA MODELS AND QUERY LANGUAGES

Relational Algebra – Algebraic Laws of RA

Cheng-En Chuang

(Slides Adopted from Jan Chomicki and Ning Deng)

University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

Outline

1. Query Rewriting
2. Algebraic Laws

Outline

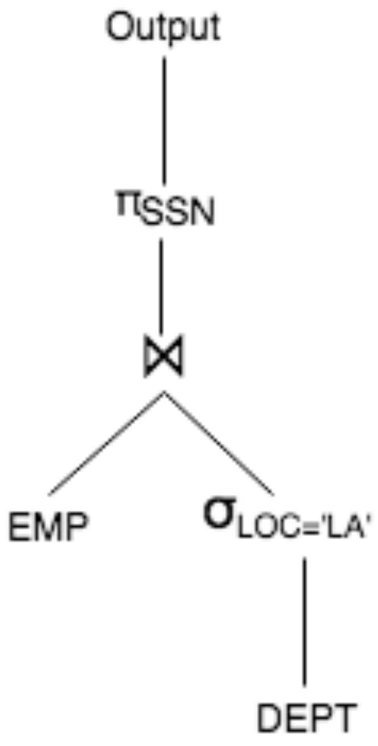
1. **Query Rewriting**
2. Algebraic Laws

Query Rewriting

- EMP(SSN, SAL, DNAME)
- DEPT(DNAME, LOC)
- Query
  - Find all the SSNs of the employees who work in a department located in LA
  - $\pi_{SSN}\left(\sigma_{EMP.DNAME=DEPT.DNAME \land DEPT.LOC='LA'}(EMP \times DEPT)\right)$
  - $\pi_{SSN}\left(\sigma_{EMP.DNAME=DEPT.DNAME}\left(\sigma_{DEPT.LOC='LA'}(EMP \times DEPT)\right)\right)$
  - $\pi_{SSN}\left(\sigma_{EMP.DNAME=DEPT.DNAME}\left(EMP \times \sigma_{DEPT.LOC='LA'}(DEPT)\right)\right)$
  - $\pi_{SSN}(EMP \bowtie \sigma_{DEPT.LOC='LA'}(DEPT))$
- 🍭What is the difference between these queries?

Query Rewriting (RA Tree)

- $\pi_{SSN}(EMP \bowtie \sigma_{DEPT.LOC='LA'}(DEPT))$



Outline

1. Query Rewriting
2. **Algebraic Laws**

Improving Query Plans

- Algebraic Laws (equivalence rules)
  - Commutative and Associative
  - Pushing down SELECTION
  - Pushing down PROJECT
  - …
- Cost based Optimization (We will not talk about this)

Commutative and Associative Operators

- $R \times S = S \times R; (R \times S) \times T = R \times (S \times T)$
- $R \bowtie S = S \bowtie R; (R \bowtie S) \bowtie T = R \bowtie (S \bowtie T)$
- $R \cup S = S \cup R; (R \cup S) \cup T = R \cup (S \cup T)$
- $R \cap S = S \cap R; (R \cap S) \cap T = R \cap (S \cap T)$
- How about θ-join?
  - R(a,b) , S(b,c), T(c,d)
  - $R \bowtie_{\theta} S = S \bowtie_{\theta} R$
  - $\left(R \bowtie_{\theta_1} S\right) \bowtie_{\theta_2 \wedge \theta_3} T = R \bowtie_{\theta_1 \wedge \theta_3} (S \bowtie_{\theta_2} T)$
    - θ_2 involves attributes only from S and T

Pushing Down SELECTION

- SELECTION σ tend to reduce the size relation
- Move the selections down the tree as far as they will go
  - Without changing what the expression does
- Splitting Laws
  - $\sigma_{C_1 \wedge C_2}(R) = \sigma_{C_1}(\sigma_{C_2}(R))$
  - $\sigma_{C_1 \vee C_2}(R) = \sigma_{C_1}(R) \cup \sigma_{C_2}(R)$
- Union and Difference
  - $\sigma_C(R \cup S) = \sigma_C(R) \cup \sigma_C(S)$
  - $\sigma_C(R - S) = \sigma_C(R) - \sigma_C(S)$

Pushing Down SELECTION

- Assumption: Relation *R* has all attributes mentioned in *C*
  - $\sigma_C(R \times S) = \sigma_C(R) \times S$
  - $\sigma_C(R \bowtie S) = \sigma_C(R) \bowtie S$
  - $\sigma_{C_1}\left(R \bowtie_{C_2} S\right) = \sigma_{C_1}(R) \bowtie_{C_2} S$
  - $\sigma_C(R \cap S) = \sigma_C(R) \cap \sigma_c(S)$

Pushing Down PROJECTION

- PROJECT π is generally less useful than pushing down SELECTION
  - Why?
- Principle
  - Pushing down the PROJECTION as long as it eliminates only attributes
    - That are neither used by any operator above on the RA tree
    - Nor are in the result of the entire expression

Pushing Down PROJECTION

- Cascading
  - $\pi_{A_1,\ldots,A_n}\left(\pi_{A_1,\ldots,A_{n+k}}(R)\right) = \pi_{A_1,\ldots,A_n}(R)$
- With SELECTION
  - $\pi_{A_1,\ldots,A_n}\left(\sigma_C(R)\right) = \sigma_c(\pi_{A_1,\ldots,A_n}(R))$
  - If attr(c) $\subseteq \{A_1, \ldots, A_n)$

Pushing Down PROJECTION

- With Cartesian Product
  - $\pi_{A_1,\dots,A_{n+k}}(S \times R) = \pi_{A_1,\dots,A_n}(S) \times \pi_{A_{n+1},\dots,A_{n+k}}(R)$
  - Where
    - $A_1, \dots, A_n$ come from *S*
    - $A_{n+1}, \dots, A_{n+k}$ come from *R*
- With Union
  - $\pi_{A_1,\dots,A_n}(R \cup S) = \pi_{A_1,\dots,A_n}(R) \cup \pi_{A_1,\dots,A_n}(S)$

Proving Laws

- $\forall R_1, R_2 : \sigma_C(R_1 - R_2) = \sigma_C(R_1) - \sigma_C(R_2)$
  - A tuple *t* in LHS iff
    - $t \in \pi_C(R_1 - R_2)$ iff
    - $t \in (R_1 - R_2) \wedge C(t)$ iff
    - $t \in R_1 \wedge t \notin R_2 \wedge C(t)$
  - A tuple *t* in RHS iff
    - $t \in \sigma_C(R_1) - \sigma(R_2)$ iff
    - $t \in \sigma_C(R_1) \wedge t \notin \sigma_C(R_2)$ iff
    - $t \in R_1 \wedge C(t) \wedge t \notin R_2 \vee \neg C(t)$ iff
    - $t \in R_1 \wedge t \notin R_2 \wedge C(t)$

Disproving Laws

- $\forall R_1, R_2 : \pi_A(R_1 - R_2) = \pi_A(R_1) - \pi_A(R_2)$
  - $R_1 = \{(1,2), (1,3)\}$
  - $R_2 = \{(1,2)\}$

Recommended Reading

Database Systems: The Complete Book
Chapter 16.1 – 16.2