Relational Algebra
Heuristic Optimization

SWEN304/SWEN435

Lecturer: Dr Hui Ma

Engineering and Computer Science

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

Example Relational Algebra Query

- Which students have enrolled **in two or more** papers?

ENROLMENT

ID	paper
101	CO102
101	CO103
102	CO102
103	MA101

- we first rename paper to paper' and join the resulting relation with Enrolment, hence

$$X := \text{ENROLMENT} * \rho_{paper \rightarrow paper'}(\text{ENROLMENT})$$

ID	paper	paper'
101	CO102	CO102
101	CO102	CO103
101	CO103	CO102
101	CO103	CO103
102	CO102	CO102
103	MA101	MA101

Database Design Quality

- This gives us for each student every pair of papers he or she has enrolled in
- We are not interested in pairs which have same entries but the ones with different entries for the pair

$$Y := \sigma_{paper \neq paper'}(X)$$

IDk	paper	Paper’
101	CO102	CO103
101	CO103	CO102

Relational Algebra Example

- We get the student that have enrolled in two or more courses

$\pi_{ID}(Y)$

ID
101

- Represented in one query

$$\pi_{ID}\ (\sigma_{paper \neq paper'}(ENROLMENT * \rho_{paper \rightarrow paper'}(ENROLMENT)))$$

A Sample Relational Database

Student

LName	FName	StudId	Major
Smith	Susan	131313	Comp
Bond	James	007007	Math
Smith	Susan	555555	Comp
Cecil	John	010101	Math

Course

PName	CourId	Points	Dept
DB Sys	C302	15	Comp
SofEng	C301	15	Comp
DisMat	M214	22	Math
Pr&Sys	C201	22	Comp

Grades

StudId	CourId	Grade
007007	C302	A+
555555	C302	ω
007007	C301	A
007007	M214	A+
131313	C201	B-
555555	C201	C
131313	C302	ω
007007	C201	A
010101	C201	ω

Heuristic Query Optimization

- Process for heuristics optimization
  1. The parser of a high-level query generates an initial internal representation
  2. Apply heuristics rules to optimize the internal representation
  3. A query execution plan is generated to execute groups of operations based on the access paths available on the files involved in the query

- The main heuristic is to apply first the operations that reduce the size of intermediate results
  - E.g., Apply SELECT and PROJECT operations before applying the JOIN or other binary operations

Using Heuristics in Query Optimization (1)

- General Transformation Rules for Relational Algebra Operations:

1. Cascade of σ: A conjunctive selection condition can be broken up into a cascade (sequence) of individual σ operations:
   - $\sigma_{c1 \wedge c2 \wedge \ldots \wedge cn}(N) = \sigma_{c1}(\sigma_{c2}(\ldots(\sigma_{cn}(N))\ldots))$

   $\sigma_{grade=\text{'A+'} \wedge courId=\text{'C302'}}(Grades)$
   $=\sigma_{grade=\text{'A+'}}(\sigma_{courId=\text{'C302'}}(Grades))$

2. Commutativity of σ: The σ operation is commutative:
   - $\sigma_{c1}(\sigma_{c2}(N)) = \sigma_{c2}(\sigma_{c1}(N))$

   $\sigma_{grade=\text{'A+'}}(\sigma_{courId=\text{'C302'}}(Grades))$
   $=\sigma_{courId=\text{'C302'}}(\sigma_{grade=\text{'A+'}}(Grades))$

Using Heuristics in Query Optimization (1)

- General Transformation Rules for Relational Algebra Operations:

3. Cascade of π: In a cascade (sequence) of π operations, all but the last one can be ignored:
   - $\pi_{List1}\ (\pi_{List2}\ (\ldots(\pi_{Listn}(N))\ldots)\ ) = \pi_{List1}(N)$

     $\pi_{StudId}\ (\pi_{StudId,\ LName}(Student))$

     $= \pi_{StudId}(Student)$

4. Commuting σ with π: If the selection condition c involves **only** the attributes A1, ..., An in the projection list, the two operations can be commuted:
   - $\pi_{A1,\ A2,\ \ldots,\ An}\ (\sigma_c\ (N)) = \sigma_c\ (\pi_{A1,\ A2,\ \ldots,\ An}\ (N))$

     $\pi_{StudId,\ major}\ (\sigma_{major='Comp'}(Student))$

     $= \sigma_{major='Comp'}(\pi_{StudId,\ major}(Student))$

     $\pi_{StudId,\ Fname}(\sigma_{major='Comp'}(Student))$

     $\neq \sigma_{major='Comp'}(\pi_{StudId,\ Fname}(Student))$

Using Heuristics in Query Optimization (2)

- General Transformation Rules for Relational Algebra Operations (contd.) ($\bowtie$ represent both natural join * and equi-join $\bowtie_C$)

5. Commutativity of $\bowtie$ (and x): The $\bowtie$ operation is commutative as is the x operation:
   - $N_1 \bowtie_C N_2 = N_2 \bowtie_C N_1$; $N_1 \times N_2 = N_2 \times N_1$

Student $\bowtie_{StudId-StudId}$ Grades = Grades $\bowtie_{StudId-StudId}$ Student

Student x Grades = Grades x Student

Using Heuristics in Query Optimization (2)

6. Commuting σ with join $\bowtie$ (or x): If all the attributes in the selection condition c involve only the attributes of one of the relations being joined—say, N_1—the two operations can be commuted as follows:
   - $\sigma_c (N_1 \bowtie N_2) = \sigma_c (N_1) \bowtie N_2$

     $\sigma_{major='Math'}$ (Student * Grades)

     = $\sigma_{major='Math'}$ (Student) * Grades

- Alternatively, if the selection condition c can be written as (c1 and c2), where condition c1 involves only the attributes of N_1 and condition c2 involves only the attributes of N_2, the operations commute as follows:
   - $\sigma_c(N_1 \bowtie N_2) = \sigma_{c1}(N_1) \bowtie \sigma_{c2} (N_2)$

   $\sigma_{major='Math' \wedge grade='A+'}$ (Student * Grades)

   = $\sigma_{major='Math'}$ (Student) * $\sigma_{grade='A+'}$ (Grades)

Using Heuristics in Query Optimization (3)

- General Transformation Rules for Relational Algebra Operations (contd.):

7. Commuting π with $\bowtie$ (or x): Suppose that the projection list is AL = $\{A_1, ..., A_n, B_1, ..., B_m\}$, where $A_1, ..., A_n$ are attributes of N_1 and $B_1, ..., B_m$ are attributes of N_2.

   a) If the join condition c involves only attributes in AL, the two operations can be commuted as follows:

   - $\pi_{AL} (N_1 \bowtie_C N_2) = (\pi_{A1, ..., An} (N_1)) \bowtie_C (\pi_{B1, ..., Bm} (N_2))$

$\pi_{StudId,Major,Grade}$ (Student $\bowtie_{StudId=StudId}$ Grades)

= ($\pi_{StudId,Major}$ (Student)) $\bowtie_{StudId=StudId}$ ($\pi_{StudId,Grade}$ (Grades))

Using Heuristics in Query Optimization (3)

- b) If the join condition C contains additional attributes not in AL, these must be added to the projection list, and a final π operation is needed.
  - $\pi_{AL}(N_1 \bowtie_C N_2) = \pi_{AL}(\pi_{AL1}(N_1) \bowtie_C \pi_{AL2}(N_2))$,

  where AL_i contains the common attributes in Ni and AL and the common attributes of N1 and N2

$\pi_{LName,\ Grade}$ (Student $\bowtie_{StudId=StudId}$ Grades)

$= \pi_{LName,\ Grade}$ ($\pi_{StudId,Lname}$ (Student) $\bowtie_{StudId=StudId}$ $\pi_{StudId,Grade}$ (Grades))

Using Heuristics in Query Optimization (4)

- General Transformation Rules for Relational Algebra Operations (contd.):

8. Commutativity of set operations: The set operations υ and ∩ are commutative but "–" is not.

π_{StudId}(Student) ∩ π_{StudId}(Grades) = π_{StudId}(Grades) ∩ $_{StudId}$(Student)

π_{StudId}(Student) - π_{StudId}(Grades) ≠ π_{StudId}(Grades) - $_{StudId}$(Student)

9. Associativity of ⋈ , x, υ, and ∩ : These four operations are individually associative; that is, if θ stands for any one of these four operations (throughout the expression), we have
   - $(N_1\ \theta\ N_2)\ \theta\ N_3 = N_1\ \theta\ (N_2\ \theta\ N_3)$

(Student * Grades) * Course = Student * (Grades * Course)

Using Heuristics in Query Optimization (4)

10. Commuting σ with set operations: The σ operation commutes with $\cup$, $\cap$, and $-$. If θ stands for any one of these three operations, we have
    - $\sigma_c (N_1 \ \theta \ N_2) = \sigma_c (N_1) \ \theta \ \sigma_c (N_2)$

$\sigma_{StudID=`007007'} (\pi_{StudId,}(Student) - \pi_{StudId}(Grades))$

$= \sigma_{StudID=`007007'} (\pi_{StudId,}(Student)) - \sigma_{StudID=`007007'} (\pi_{StudId}(Grades))$

Using Heuristics in Query Optimization (5)

- General Transformation Rules for Relational Algebra Operations (contd.):

11. The π operation commutes with $\cup$.

$$\pi_{AL} (N_1 \cup N_2) = (\pi_{AL} (N_1)) \cup (\pi_{AL} (N_2))$$

12. Converting a (σ, x) sequence into $\bowtie$: If the condition c of a σ that follows a x Corresponds to a join condition, convert the (σ, x) sequence into a $\bowtie$ as follows:

$$\sigma_C (N_1 \times N_2) = (N_1 \bowtie_C N_1)$$

$\sigma_{StudId=StudID}$ (Student x Grades)

= Student $\bowtie_{StudId=StudId}$ Grades)

Using Heuristics in Query Optimization (6)

- Summary of Heuristics for Algebraic Optimization:
  1. The main heuristic is to apply first the operations that reduce the size of intermediate results.
  2. Perform select operations as early as possible to reduce the number of tuples and perform project operations as early as possible to reduce the number of attributes. (This is done by moving select and project operations as far down the tree as possible, e.g. rule 6, 7, 10, 11)
  3. The select and join operations that are most restrictive should be executed before other similar operations. (This is done by reordering the leaf nodes of the tree among themselves and adjusting the rest of the tree appropriately.)

Heuristic Query Optimization Exercise

- For a given relation schema:

MOVIE(title, genre, director)

ACTOR(mtitle, name, DoB)

MOVIE_AWARD(title, award_name, year_of_award)

$IC = \{ACTOR[mtitle] \subseteq MOVIE[title],$
$MOVIE\_AWARD[title] \subseteq MOVIE[title]\}$

- Draw query trees for the following queries

$$\pi_{title}\ (\sigma_{genre= \text{'war'} \wedge name= \text{'Tom Cruise'}}((MOVIE) \bowtie_{title\ =\ mtitle} (ACTOR)))$$

$$\pi_{director}\ (\sigma_{award\_name\ =\ \text{'Oscar'}}(MOVIE\ \ *\ \ MOVIE\_AWARD))$$

- Optimize the query trees using heuristic rules

Heuristic Query Optimization Exercise

MOVIE(title, genre, director)

ACTOR(mtitle, name, year_born)

MOVIE_AWARD(title, production_year, award_name year_of_award)

$$\pi_{title}\left(\sigma_{genre='war' \wedge name='Tom\ Cruise'}\left((MOVIE) \bowtie_{title=mtitle} (ACTOR)\right)\right)$$

Heuristic Query Optimization Exercise

MOVIE(title, genre, director)

ACTOR(mtitle, name, year_born)

MOVIE_AWARD(title, production_year, award_name year_of_award)

$$\pi_{director}\ (MOVIE\ *\ \sigma_{award\_name\ =\ 'Oscar'}\ (MOVIE\_AWARD))$$