CS3DB3/SE4DB3/SE6M03 TUTORIAL

Mei Jiang

Feb 13/15, 2013

Outline

- Relational Algebra
- SQL and Relational Algebra
- Examples

Relational Algebra

- Basic Operators
  - Select: **σ**
    - σ_{C} (R) where C is a list of conditions
  - Project: **Π**
    - Π_{L} (R) where L is a list of attributes of R
  - Rename: **ρ**
    - $\rho_{R1(A1,\ldots,An)}(R2)$
  - Cartesian product: **x**
  - Union: **∪**
  - Intersection: **∩**
  - Set difference: **−**
- The operators take one or two relations as inputs and produce a new relation as a result.

Relational Algebra (cont.)

- More Operators
  - Natural Join: ⋈
  - Theta Join : $\bowtie_C$
  - Outer Join: ⟕, ⟖, ⟗
  - Eliminate duplicates: **δ**
    - δ (R)
  - Sort tuples: **τ**
    - τ_L (R) where L is list of attributes of R
  - Grouping and Aggregation: **γ**
    - γ_L (R) where L is list of attributes of R that either
      - Grouping attributes
      - AGG(A), where AGG is one of the aggregation operators such as **SUM, AVG, COUNT, MIN, MAX** and A is an attribute.

SQL and Relational Algebra

- **SELECT** A1, A2, …, An
  FROM R1, R2, …, Rm
  WHERE P

  is equivalent to the multiset relational algebra expression

$$\prod_{A1,\, A2, \cdots,\, An} (\sigma_P (R1 \times R2 \times \cdots \times Rm))$$

SQL and Relational Algebra (cont.)

- **SELECT** A1, A2, **AGG**(A3) as AGG3
  FROM R1, R2, …, Rm
  WHERE P
  GROUP BY A1, A2
  - **AGG**() is an aggregation operator , **MIN**(), **MAX**(), etc.
  - is equivalent to the multiset relational algebra expression

$$\gamma_{A1,\ A2,\ AGG(A3) \rightarrow AGG3}(\sigma_P(R1 \times R2 \times \cdots \times Rm))$$

- If only display attribute A1 and AGG3, then

$$\prod_{A1,\ AGG3}(\gamma_{A1,\ A2,\ AGG(A3) \rightarrow AGG3}(\sigma_P(R1 \times R2 \times \cdots \times Rm)))$$

Example -1

- **Course** (course_id, title, dept_name, credits)
- Find the titles of courses in the Comp. Sci. department that have 3 credits.
- SQL

```
SELECT  title
FROM Course
WHERE dept_name='Comp. Sci.' AND credits=3;
```

- Relational Algebra

$$\Pi_{title}(\sigma_{dept\_name="Comp.Sci." \ AND \ credits=3}(course))$$

Example -2

- **Takes** (id, course_id, semester, year, grade)
- **Teaches**(name, course_id, semester, year)
- Find the IDs of all students who were taught by an instructor named Jones.
- SQL

```
SELECT  id
FROM Takes NATURAL JOIN Teaches
WHERE name = 'Jones';
```

- Relational Algebra

$$\Pi_{id} \ (\sigma_{name="Jones"} \ (takes \ \bowtie \ teaches))$$

Example -3

- **Instructor**(id, name, dept_name, salary)
- Find the highest salary of any instructor.
- SQL

```
SELECT max(salary)
FROM Instructor;
```

- Relational Algebra

$$\gamma_{\max(salary)}(Instructor)$$

Example -4

- **Instructor**(id, name, dept_name, salary)
- Find the names of all instructors earning the highest salary.
- SQL

```
SELECT name
FROM Instructor
WHERE salary = ( SELECT max(salary)
                 FROM Instructor);
```

- Relational Algebra

$$\prod_{name} (Instructor \bowtie (\gamma_{\max(salary) \rightarrow salary}(Instructor)))$$

Example -5

- **Takes** (id, course_id, semester, year, grade)
- Find the enrollment of each course that was offered in Fall 2009.
- SQL

```
SELECT  course_id, count(*) as enrollment
FROM Takes
WHERE year=2009 AND semester='Fall'
GROUP BY course_id;
```

- Relational Algebra

$$\gamma_{course\_id,\, count(*) \rightarrow enrollment}(\sigma_{year=2009\, AND\, semester="Fall"}(takes))$$

Example -6

- **Takes** (id, course_id, semester, year, grade)
- Find the maximum enrollment in Fall 2009
- SQL

```
SELECT MAX(enrollment)
FROM  (SELECT  course_id, count(*) as enrollment
         FROM Takes
         WHERE year=2009 AND semester='Fall'
         GROUP BY course_id);
```

- Relational Algebra

$$R := \gamma_{course\_id,\, count(*) \rightarrow enrollment}(\sigma_{year=2009\, AND\, semester="Fall"}(takes))$$

$$\text{Result} = \gamma_{\max(enrollment)}(R)$$

Example -7

- □ **Takes** (id, course_id, semester, year, grade)
- □ Find the course id that had the maximum enrollment in Fall 2009.
- □ SQL

```
SELECT  course_id
FROM Takes
WHERE year=2009 AND semester='Fall'
GROUP BY course_id
HAVING count(*) =
          ( SELECT MAX(enrollment)
            FROM (SELECT  course_id, count(*) as enrollment
                  FROM Takes
                  WHERE year=2009 AND semester='Fall'
                  GROUP BY course_id));
```

Note: We cannot directly SELECT course_id in here, because it's neither a grouping attribute, nor an aggregation function.

Example -7 (cont.)

- **Takes** (id, course_id, semester, year, grade)
- Find the course id that had the maximum enrollment in Fall 2009.
- Relational Algebra

$$R1 := \gamma_{course\_id,\, count(*) \rightarrow enrollment}(\sigma_{year=2009\ AND\ semester="Fall"}(takes))$$

$$R2 := \gamma_{\max(enrollment) \rightarrow enrollment}(R1)$$

$$\text{Result} = \prod_{course\_id} (R1 \bowtie R2)$$

Example -8

- **Works** (pname, cname, salary)
- Find the names of all employees who earn more than every employee of “First Bank”.
- SQL

```
SELECT pname
FROM Works
WHERE salary >ALL (SELECT salary
                   FROM Works
                   WHERE cname= 'First Bank');
```

Example -8 (cont.)

- □ **Works** (pname, cname, salary)
- □ Find the names of all employees who earn more than every employee of “First Bank”.
- □ Relational Algebra

$$R1 := \prod_{w1.pname}(\; \rho_{w1}(works) \bowtie_{(w1.salary \leq w2.salary \, AND \, w2.cname = "First\ Bank")} \rho_{w2}(works))$$

Result = $\prod_{pname}(Works) - R1$

References

- Database System Concepts (6th edition) by A. Silberschatz, H. Korth, S. Sudarshan