CSE 344: Section 3 Relational Algebra

July 5th, 2018

Deadlines

WQ3: July 6, 11 pm

Homework 3: July 11, 11:30 pm

RA Operators

Standard:

- ∪ Union
- – - Diff.
- σ - Select
- π - Project
- ρ - Rename

Joins:

- ⋈ - Nat. Join
- ⟕ - L.O. Join
- ⟖ - R.O. Join
- ⟗ - F.O. Join
- × - Cross Product

Extended:

- δ - Duplicate Elim.
- ɣ - Group/Agg.
- τ -Sorting

A Few More SQL Keywords

```
(<sub>) INTERSECT (<sub>)

(<sub>) UNION (<sub>)

(<sub>) EXCEPT (<sub>)
```

Grouping & Aggregation (γ) Notation

Grouping:

$$\gamma_{attr\_1, \ldots, attr\_k}$$

Aggregation:

$$\gamma_{count/sum/max/min(attr) \rightarrow alias}$$

Grouping & aggregation:

$$\gamma_{attr\_1, \ldots, attr\_k, count/sum/max/min(attr) \rightarrow alias}$$

SQL to RA for basic queries

1. Put the table names at the bottom of the tree
2. Select for tuples (WHERE clause) on individual tables (Student.age > 10)
3. Join the tables by their join predicates (Student.cid = Class.id)
4. Select for tuples (WHERE clause) on combined tables
5. Group by and aggregate (GROUP BY clause)
6. Filter the appropriate groups (HAVING clause)
7. Project relevant attributes (SELECT a, b, c…)

R(b)
T(a, c)

Query Plans (Example SQL -> RA)

Select-Join-Project structure

Make this SQL query into RA (remember FWGHOS):

```
SELECT R.b, T.c, max(T.a) AS T_max
  FROM Table_R AS R, Table_T AS T
 WHERE R.b = T.b
 GROUP BY R.b, T.c
HAVING max(T.a) > 99
```

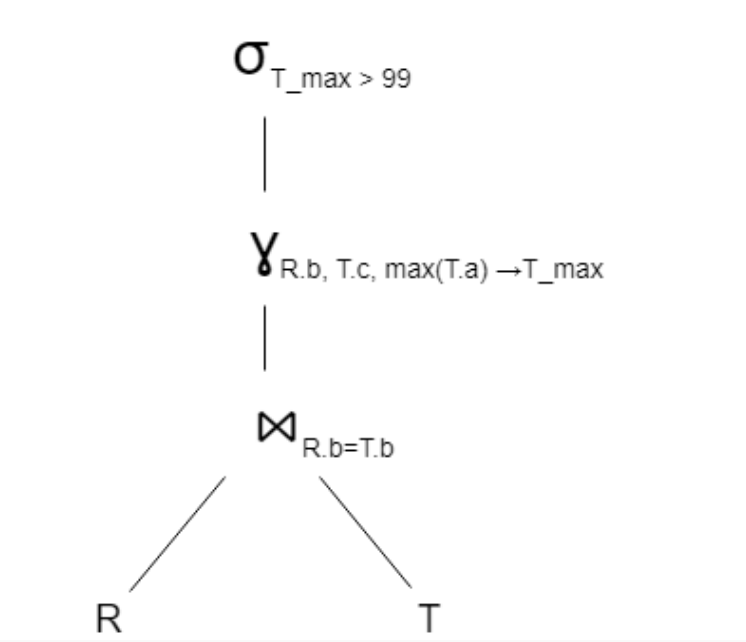
R(b)
T(a, c)

Query Plans (Example SQL -> RA)

Select-Join-Project structure

Make this SQL query into RA (remember FWGHOS):

```
SELECT R.b, T.c, max(T.a) AS T_max
  FROM Table_R AS R, Table_T AS T
 WHERE R.b = T.b
 GROUP BY R.b, T.c
HAVING max(T.a) > 99
```



SQL to RA for universal (nested) queries

- **Monotonic query**: If we add tuples to the input, no tuples can be lost from the output
- Universal queries are NOT monotonic
  - existence of a single contradicting tuple can make an output value invalid
- In SQL, universal queries must use nested subqueries
- In RA, use set difference operator (-)

Product(pname, cname, price)

SQL to RA for universal (nested) queries

Example: Find the companies that only sell products that cost more than $50

Step 1: Find all companies

Step 2: Find all companies that sell **any** product for <= 50

Step 3: Find the set difference between steps 1 and 2

Product(pname, cname, price)

SQL to RA for universal (nested) queries

Example: Find the companies that only sell products that cost more than $50

Step 1: Find all companies

Step 2: Find all companies that sell **any** product for <= 50

Step 3: Find the set difference between steps 1 and 2