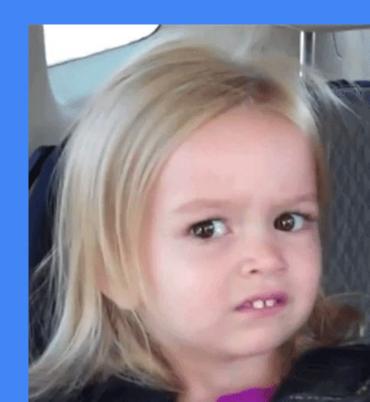
CSE 414: Section 4 Relational Algebra

January 28, 2021

Prati Tal



Announcements



Monotonicity

Definition: A query Q is **monotone** if whenever we add tuples to one or more input tables, the answer to the query will not lose **any** output tuples

Theorem: If Q is a SELECT-FROM-WHERE query that does not have subqueries, and no aggregates, then it is monotone

(Non-)monotonic Queries

- "Can we take back outputs by looking at more data?"
- Is this a monotonic query?

```
SELECT count(*)
FROM T1
GROUP BY T1.attr
```

(Non-)monotonic Queries

- "Can we take back outputs by looking at more data?"
- Is this a monotonic query?

```
SELECT count(*)
FROM T1
GROUP BY T1.attr
```

No! This query does not satisfy **set containment**.

Ex:

Current output: {(6), (23), (10)} After more data: {(6), (23), (11)}

 $\{(6), (23), (10)\} \nsubseteq \{(6), (23), (11)\}$



RA - Motivation

- SQL is a declarative language
 - Computers don't understand it well
- Relational Algebra is that middle ground
- While executing, multiple RA tree are formed and the best one is chosen

RA Operators

 \cap - Intersect R1 \cap R2 = R1-(R1-R2)

 $R1 \cap R2 = R1 \bowtie R2$

Standard:

∪ - Union

- Diff.

σ - Select

π - Project

ρ - Rename

Joins:

≥ - L.O. Join

R.O. Join

⋈ - F.O. Join

×- Cross Product

Extended:

δ - Duplicate Elim.

γ - Group/Agg.

τ - Sorting

A Few More SQL Keywords

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

Y Notation

Grouping and aggregation on group:

Yattr_1, ..., attr_k, count/sum/max/min(attr) -> alias

Aggregation on the entire table:

Ycount/sum/max/min(attr) -> alias

Some pro tips:

Where and Having have the same operator:

You can use an operator multiple times

Project is the SQL Select:) $\pi = \text{SELECT from SQL and NOT } \sigma$

Use FWGHOS while making a tree!

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
```

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

TR.b, T.c, T_max(T_max) > 99

TR.b, T.c, T_max(T_max) = T.b

T)))
```

Now for some practice!

