CSE 344: Section 3
Relational Algebra

July 5th, 2018
Deadlines

WQ3: July 6, 11 pm
Homework 3: July 11, 11:30 pm
### RA Operators

<table>
<thead>
<tr>
<th>Standard:</th>
<th>Joins:</th>
<th>Extended:</th>
</tr>
</thead>
<tbody>
<tr>
<td>⫋ Union</td>
<td>⨝ - Nat. Join</td>
<td>δ - Duplicate Elim.</td>
</tr>
<tr>
<td>– - Diff.</td>
<td>⨝ - L.O. Join</td>
<td>γ - Group/Agg.</td>
</tr>
<tr>
<td>σ - Select</td>
<td>⨝ - R.O. Join</td>
<td>τ - Sorting</td>
</tr>
<tr>
<td>π - Project</td>
<td>⨝ - F.O. Join</td>
<td></td>
</tr>
<tr>
<td>ρ - Rename</td>
<td>x - Cross Product</td>
<td></td>
</tr>
</tbody>
</table>
A Few More SQL Keywords

(() INTERSECT ()
(() UNION ()
(() EXCEPT ()
Grouping & Aggregation (γ) Notation

Grouping:

\[ \gamma_{\text{attr}_1, \ldots, \text{attr}_k} \]

Aggregation:

\[ \gamma_{\text{count/sum/max/min(attr)} \rightarrow \text{alias}} \]

Grouping & aggregation:

\[ \gamma_{\text{attr}_1, \ldots, \text{attr}_k, \text{count/sum/max/min(attr)} \rightarrow \text{alias}} \]
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…)
Query Plans (Example SQL -> RA)

Select-Join-Project structure

Make this SQL query into RA (remember FWGHOS):

```sql
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
```
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 (-)
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 $\leq 50$

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