DATA 514: Section 2
A SeQuel to SQL

January 15, 2019
Homework

WQ2, HW2 due Monday!

(Jan 21 at 11:00 PM)
SQL 3-Valued Logic

SQL has 3-valued logic

- **FALSE = 0**
  
  
  [ex] price < 25 is FALSE when price = 99

- **UNKNOWN = 0.5**
  
  
  [ex] price < 25 is UNKNOWN when price = NULL

- **TRUE = 1**
  
  
  [ex] price < 25 is TRUE when price = 19
SQL 3-Valued Logic (con’t)

Formal definitions:

- C1 AND C2 means min(C1,C2)
- C1 OR C2 means max(C1,C2)
- NOT C means means 1-C

The rule for SELECT ... FROM ... WHERE C is the following:

- if C = TRUE then include the row in the output
- if C = FALSE or C = unknown then do not include it
Importing Files (HW2)

First, make the table.
Then, import the data.

.mode csv
.import ./population.csv Population
.import ./gdp.csv GDP
.import ./airport.csv Airport

.import /path/to/file NameOfTable
Aliasing

- Good practice to use more intuitive labels
- Required to disambiguate for self joins
  (e.g.) FROM [table] AS T1, [table] AS T2)

- You can alias without “AS” in the FROM clause (“AS” keyword can be omitted)

SELECT [attribute] AS [attribute_name]
FROM [table] AS [table_name]
... [table_name].[attribute_name] ...
Aggregates

- Aggregates return a single tuple.

**COUNT**(attribute) - counts the number of tuples

**SUM**(attribute)

**MIN/MAX**(attribute)

**AVG**(attribute)

...
Filters

**LIMIT number** - limits the number of tuples returned

[e.g.] SELECT * FROM table LIMIT 1;

Note: MS SQL Server uses **TOP** instead of **LIMIT**

**DISTINCT** - only returns different values (eliminates duplicates)

[e.g.] SELECT DISTINCT column_name FROM table;
Grouping and Ordering

**GROUP BY** [attribute], ..., [attribute_n]

**HAVING** [predicate] - operates on groups

**WHERE** filters before grouping; **HAVING** filters after

**ORDER BY**

https://www.w3schools.com/sql/sql_having.asp
SQL Query Evaluation Order

FWGHOS
(From, Where, Group By, Having, Order By, Select)
SQL Query Evaluation Order

FWGHOS

From, Where, Group By, Having, Order By, Select

- Transact-SQL documentation: see link
Joining

Inner vs. Outer

Self Joins
Join Semantics

- For now, we are primarily focusing on “nested loops” semantics

- NOT the most efficient implementation on a large database! (we will talk about other ways to join later in the course)
  - Hash Join
  - Sort-Merge Join

For more information and different types of joins see:
Nested Loop Semantics

```
SELECT x_1.a_1, ..., x_n.a_n
FROM x_1, ..., x_n
WHERE <cond>

for each tuple in x_1:
    ...
    for each tuple in x_n:
        if <cond>(x_1, ..., x_n):
            output(x_1.a_1, ..., x_n.a_n)
```