

# CSE 344: Section 2

## A SeQueL to SQL

January 11<sup>th</sup>, 2018

# Administrivia

WQ1 due **Wednesday, January 17<sup>th</sup>** at 11:00 PM

HW1 due **Wednesday, January 17<sup>th</sup>** at 11:30 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 style for renaming attribute operations to more intuitive labels
- Essential for self joins (ex: FROM [table] AS T1, [table] AS T2)
- You can alias without “AS” in the FROM clause (i.e. “AS” keyword can be omitted)

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

# Aggregates

- Aggregates will make the query return a single tuple.

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

**SUM(attribute)**

**MIN/MAX(attribute)**

**AVG(attribute)**

...

# Filters

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

[ex] `SELECT * FROM table LIMIT 1;`

**DISTINCT** - only returns different values (gets rid of duplicates)

[ex] `SELECT DISTINCT column_name FROM table;`



# Grouping and Ordering

**GROUP BY** [attribute], ..., [attribute\_n]

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

**ORDER BY**

# SQL Query Evaluation Order

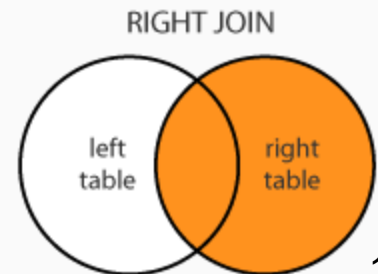
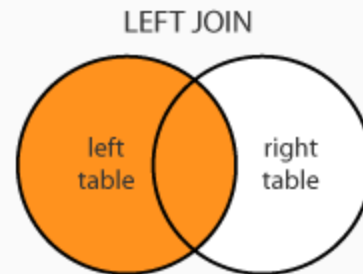
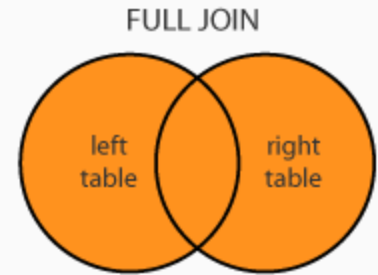
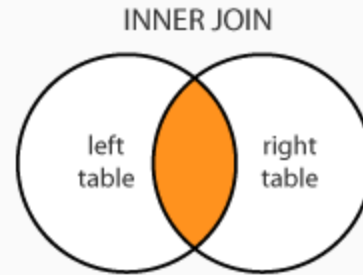
# FWGHOS

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

# Joining

Inner vs. Outer

Self Joins



For more information and different types of joins see:

<https://blogs.msdn.microsoft.com/craigfr/2006/08/16/summary-of-join-properties/>

# 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

# 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  $\langle \text{cond} \rangle(x_1, \dots, x_n)$ :

output( $x_1.a_1, \dots, x_n.a_n$ )