MICHIGAN
ROSS SCHOOL OF BUSINESS

A Gentle Introduction to SQL

Big Data Camp

June 20, 2019

Teddy DeWitt

(slides inspired by Mike Cafarella)

Learning Overview

- What is a database
- Why is SQL cool?
- Intro to schema and tables
- Running queries
- SQL and Python

What is a database?

- A database is an organized collection of data

- Relational Databases (SQL)
  - ~ SQLite, MySQL, SQL Server, PostgreSQL

- Relational Databases (NoSQL)
  - ~ CouchDB, Cassandra, MongoDB, Redis

- Blockchain Databases
  - ~ Bitcoin, Ethereum, etc.

Fine. What is a *relational* database?

- A relational database is a set of “relations” with two parts
  - ~ Instances - a data table, with rows (records), and columns (fields, attributes)
  - ~ Schema – relation name, columns names, and data format
- Excel comparison
  - ~ Instances are like tabs
  - ~ Schema is tab name, column headers and cell format cells (e.g., number, date, text)

Relational Databases - Cool!! but Tricky?

GREAT!!

- Millions of Rows!!
- Efficient
- Data Safe
- Slicing and Dicing
- Think VLOOKUP & Pivot Tables

Tricky?

- Special Software
- Structured Query Language - SQL

The software is often
free and SQL is basically English!

Relational Databases (1)

- The software is called a Relational Database Management System (RDBMS) – e.g., SQLite
- Your dataset is "a database", managed by an RDBMS
- An RDBMS does lots of things, but mainly:
  - ~ Keeps data safe
  - ~ Gives you a powerful query language

AID	Name	Country	Sport
1	Simone Biles	USA	Gymnastics
2	Usain Bolt	Jamaica	Track
3	Michael Phelps	USA	Swimming

Instance of Athlete Relation

AID	Name	Country	Sport
1	Simone Biles	USA	Gymnastics
2	Usain Bolt	Jamaica	Track
3	Michael Phelps	USA	Swimming

What is the schema?

(aid: integer, name: string, country: string, sport:string)

Let's make this table - Athlete

AID	Name	Country	Sport
1	Simone Biles	USA	Gymnastics
2	Usain Bolt	Jamaica	Track
3	Michael Phelps	USA	Swimming

Creating Relations in SQL

- Create the Athlete relation (table)

```
CREATE TABLE Athlete
(aid INTEGER,
 name CHAR(30),
 country CHAR(20),
 sport CHAR(20))
```

AID	Name	Country	Sport

Adding & Deleting Rows in SQL

```
INSERT INTO Athlete (aid, name, country, sport)
VALUES (1, 'Simone Biles', 'USA', 'Gymnastics')
```

```
INSERT INTO Athlete (aid, name, country, sport)
VALUES (2, 'Usain Bolt', 'Jamaica', 'Track')
```

```
INSERT INTO Athlete (aid, name, country, sport)
VALUES (3, 'Michael Phelps', 'USA', 'Swimming')
```

- And we are going to add another row!

```
INSERT INTO Athlete (aid, name, country, sport)
VALUES (4, 'Havard Lorentzen', 'Norway',
'Speedskating')
```

Table. Athlete. Boom!

AID	Name	Country	Sport
1	Simone Biles	USA	Gymnastics
2	Usain Bolt	Jamaica	Track
3	Michael Phelps	USA	Swimming
4	Harvard Lorentzen	Norway	Speedskating

Getting Data in SQL (1)

- SELECT all of the rows and columns:

```
SELECT *
FROM Athlete
```

AID	Name	Country	Sport
1	Simone Biles	USA	Gymnastics
2	Usain Bolt	Jamaica	Track
3	Michael Phelps	USA	Swimming
4	Harvard Lorentzen	Norway	Speedskating

- Only names and sports:

```
SELECT name, sport
FROM Athlete
```

```
SELECT A.name, A.sport
FROM Athlete A
```

Name	Sport
Simone Biles	Gymnastics
Usain Bolt	Track
Michael Phelps	Swimming
Harvard Lorentzen	Speedskating

Getting Data in SQL (2)

AID	Name	Country	Sport
1	Simone Biles	USA	Gymnastics
2	Usain Bolt	Jamaica	Track
3	Michael Phelps	USA	Swimming
4	Harvard Lorentzen	Norway	Speedskating

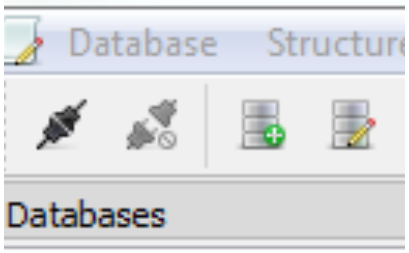
- SELECT names and sports WHERE country is USA:

```
SELECT A.name, A.sport
FROM Athlete A
WHERE A.country = 'USA'
```

Name	Sport
Simone Biles	Gymnastics
Michael Phelps	Swimming

Setup SQLite Studio

- Download SQL_BDC_2019 from the Github Site
- Under Database menu choose “Add a Database” and navigate to wherever you have saved SQL_BDC_2019
- In the Database Menu highlight SQL_BDC_2019 and hit Connect Looks like two plugs connecting
- Click icon that looks like a notepad with a pencil



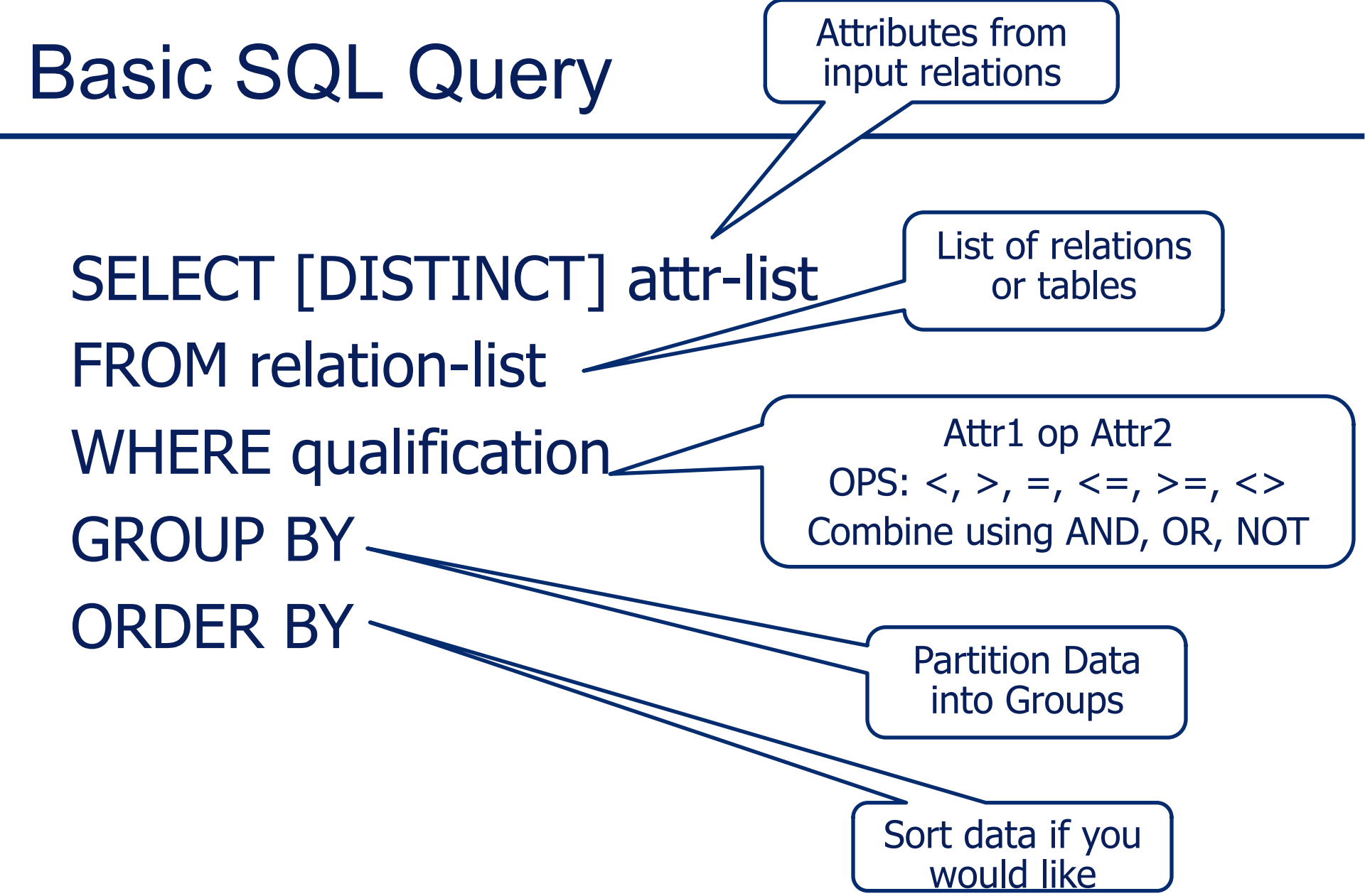
Hands-On #0

- Get your bearings:
  - ~ SELECT * FROM Allstars WHERE ROWID<11
  - ~ SELECT * FROM Twams WHERE ROWID<11

Hands-On #1

- Write queries to find:
  - ~ Names of all the players in the database
  - ~ All info for all players from Detroit
  - ~ Names and teams of the first basemen (Position ID: 3)

Basic SQL Query



Example of Basic Query(1)

- Schema:
  - ~ Sailors (sid, sname, rating, age)
  - ~ Boats (bid, bname, color)
  - ~ Reserves (sid, bid, day)

Example of Basic Query(2)

Boats

bid	bname	color
101	jeff	red
103	boaty	black

Sailors

sid	sname	rating	age
22	dustin	7	45
58	rusty	10	35
31	lubber	8	55

Reserves

sid	bid	day
22	101	Oct-10
58	103	Nov-12
58	103	Dec-13

Example of Basic Query(3)

- Schema:
  - ~ Sailors (sid, sname, rating, age)
  - ~ Boats (bid, bname, color)
  - ~ Reserves (sid, bid, day)
- Find the names of sailors who have reserved boat #103
- Are the names of the sailors and the numbers of the boats reserved in the same place?
- Must JOIN the tables

Example of Basic Query(4)

Reserves x Sailors

sid	bid	day	sid	sname	rating	age
22	101	Oct-10	22	dustin	7	45
22	101	Oct-10	58	rusty	10	35
22	101	Oct-10	31	lubber	8	55
58	103	Nov-12	22	dustin	7	45
58	103	Nov-12	58	rusty	10	35
58	103	Nov-12	31	lubber	8	55
58	103	Dec-13	22	dustin	7	45
58	103	Dec-13	58	rusty	10	35
58	103	Dec-13	31	lubber	8	55

Example of Basic Query (5)

- Find the names of sailors who have reserved boat #103

```
SELECT S.sname
FROM Sailors S, Reserves R
WHERE S.sid = R.sid AND R.bid = 103
```

sname
rusty
rusty

Example of Basic Query(6)

- Find the names of sailors who have reserved boat #103

```
SELECT S.sname
FROM Sailors S INNER JOIN Reserves R
ON S.sid = R.sid
WHERE R.bid = 103
```

This is a JOIN – new school. Use the new school

sname
rusty
rusty

Using DISTINCT

3. Project columns in attr-list
   (eliminate duplicates only if DISTINCT)

```
SELECT DISTINCT S.sname
FROM Sailors S INNER JOIN Reserves R
ON S.sid = R.sid
WHERE R.bid = 103
```

What's the effect of adding DISTINCT?

sname
rusty

Another Example

- Find the colors of boats reserved by a sailor named rusty

```
SELECT B.color
FROM Sailors S INNER JOIN Reserves R
INNER JOIN Boats B
ON S.sid = R.sid AND R.bid = B.bid
WHERE S.sname = 'rusty'
```

Hands-On #2

- Write queries to find:
  - ~ Team names for all teams with attendance more than 2,000,000
  - ~ Player ID and home stadium for all Allstars
  - ~ TeamID, attendance for teams that had an all-star player

ORDER BY clause

- Most of the time, results are unordered
- You can sort them with the ORDER BY clause

Attribute(s) in ORDER BY clause must be in SELECT list.

Find the names and ages of all sailors, in increasing order of age

```
SELECT S.sname, S.age
FROM Sailors S
ORDER BY S.age[ASC
```

Find the names and ages of all sailors, in decreasing order of age

```
SELECT S.sname, S.age
FROM Sailors S
ORDER BY S.age DESC
```

ORDER BY clause

```
SELECT S.sname, S.age, S.rating
FROM Sailors S
WHERE S.age > 40
ORDER BY S.age ASC, S.rating DESC
```

What does this query compute?

Find the names, ages, & ratings of sailors over the age of 40.

Sort the result in increasing order of age.

If there is a tie, sort those results in decreasing order of rating.

Hands-On #3

- Use the database loaded last time
- A twist:
  - ~ Find TeamID and attendance values for teams that had an all-star player ORDERED BY ATTENDANCE

Aggregate Operators

```
SELECT  COUNT (*) FROM  Sailors S
```

```
SELECT COUNT (DISTINCT S.name)
FROM Sailors S
```

COUNT (*)
COUNT ([DISTINCT] A)
SUM ([DISTINCT] A)
AVG ([DISTINCT] A)
MAX (A) *Can use Distinct*
MIN (A) *Can use Distinct*

```
SELECT  AVG (S.age)
FROM  Sailors S
WHERE  S.rating=10
```

```
SELECT  AVG ( DISTINCT S.age)
FROM  Sailors S
WHERE  S.rating=10
```

Hands-On #4

- Use our previous Allstar and Teams DB
- Find:
  - ~ Average attendance for all teams
  - ~ Average attendance among teams that had an all-star player

GROUP BY

- Conceptual evaluation
  - ~ Partition data into groups according to some criterion
  - ~ Evaluate the aggregate for each group

Example: *For each rating level, find the age of the youngest sailor*

```
SELECT  MIN (S.age), S.rating
FROM  Sailors S
GROUP BY  S.rating
```

Excel Equivalent: *Think about the results you would want from a pivot table….*

Hands-On #5

- With our same old database, first try a simple one:
  - ~ Show all teamIds that had an all-star, along with number of all-star players

Hands-On #5

- Harder:
  - ~ Show all team names that had an all-star, along with number of all-star players

NULL Values in SQL

- NULL represents ‘unknown’ or ‘inapplicable’
- WHERE clause eliminates rows that don’t evaluate to true

What does this query return?

```
SELECT sname
FROM sailors
WHERE age > 45
      OR age <= 45
```

sailors

sid	sname	rating	age
22	dustin	7	45
58	rusty	10	NULL
31	lubber	8	55

Yes, it returns just dustin and lubber!

NULL Values in Aggregates

- NULL values generally ignored when computing aggregates

```
SELECT AVG(age)
FROM sailors
```

Returns 50!

sailors

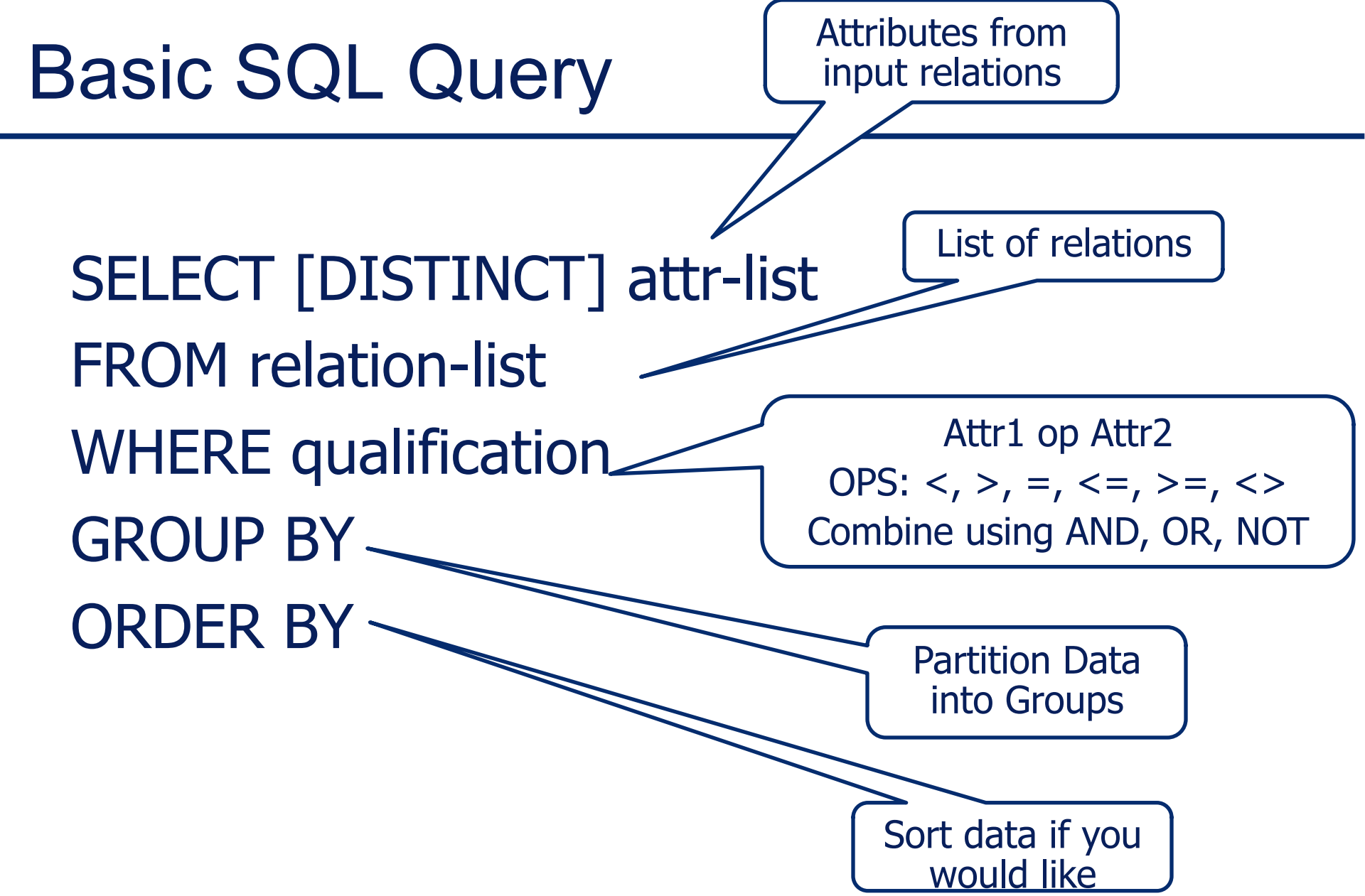
sid	sname	rating	age
22	dustin	7	45
58	rusty	10	NULL
31	lubber	8	55

Questions?

APPENDIX

Using Joins To Make Network Ties

Basic SQL Query



The Power of Joins (1)

```
SELECT name, COUNT(A.playerID) AS
playerCount
FROM Allstars A
INNER JOIN Teams T
ON A.teamID = T.teamID
GROUP BY name
ORDER BY playerCount DESC
```

The Power of Joins (2)

- *There needs to be a common identifier between tables for the join to be useful*

- *Could you join a table with itself……*

Board of Directors

- *What is a board of directors?*
- *What is a board interlock?*
- *What is a social network?*
- *What do I need to create a social network map of board interlocks?*

The Query

```
SELECT A.companyname AS TiedTo,
B.companyname AS TiedFrom FROM
Director A INNER JOIN Director B
ON a.directorname=b.directorname
WHERE a.companyname <>
b.companyname
```

Overview of Join Types

(borrowed from Jose Portilla)

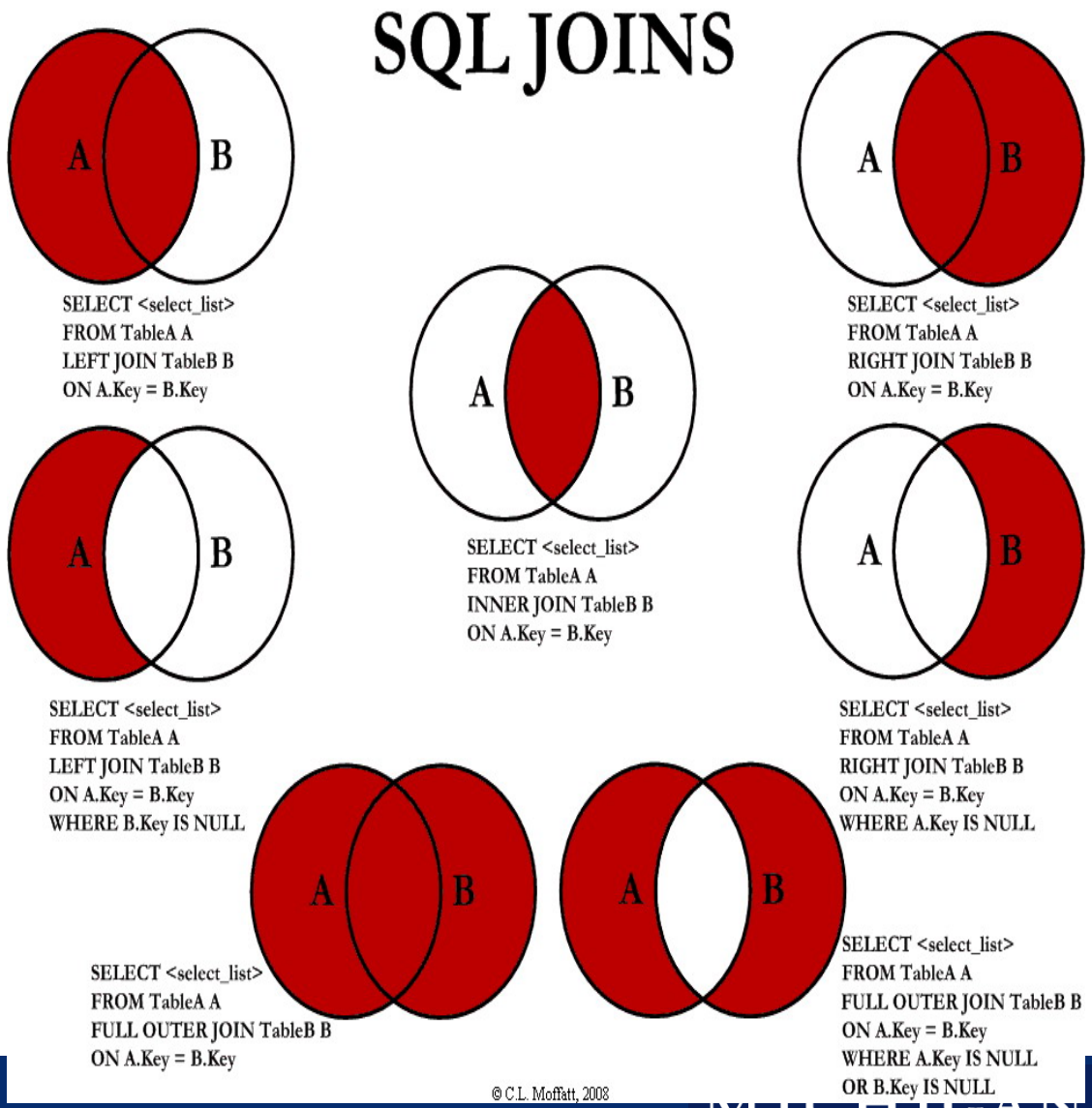
Overview

- This will begin to build our understanding of the various JOIN types.
- This presentation is a resource for you in this lecture!

JOINS Overview

- As we learn about more types of JOINS it will become very useful to reference a JOINS Venn Diagram figure.
- These are very easy to find via a Google Image Search!

JOINS Overview



JOINS Overview

- The example table for our discussion:

A

id	name
1	Pirate
2	Monkey
3	Ninja
4	Spaghetti

B

id	name
1	Rutabaga
2	Pirate
3	Darth Vader
4	Ninja

- Items in red are present in *both* tables.

Original Tables

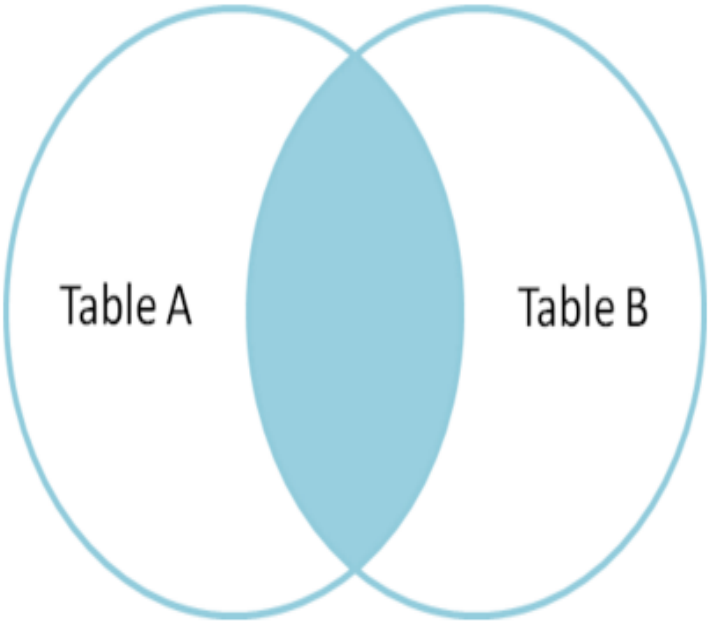
```
id name       id  name
-- ----       --  ----
1  Pirate     1   Rutabaga
2  Monkey     2   Pirate
3  Ninja      3   Darth Vader
4  Spaghetti  4   Ninja
```

INNER JOIN

Inner join produces only the set of records that match in both Table A and Table B.

```
SELECT * FROM TableA
INNER JOIN TableB
ON TableA.name = TableB.name
```

```
id  name       id   name
--  ----       --   ----
1   Pirate     2    Pirate
3   Ninja      4    Ninja
```



Original Tables

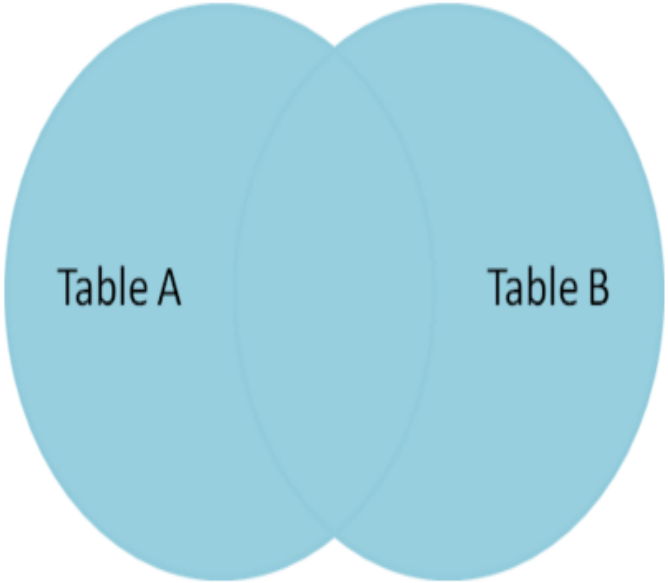
```
id name       id  name
-- ----       --  ----
1  Pirate     1   Rutabaga
2  Monkey     2   Pirate
3  Ninja      3   Darth Vader
4  Spaghetti  4   Ninja
```

FULL OUTER JOIN

Full outer join produces the set of all records in Table A and Table B, with matching records from both sides where available. If there is no match, the missing side will contain null.

```
SELECT * FROM TableA
FULL OUTER JOIN TableB
ON TableA.name = TableB.name
```

```
id    name       id    name
--    ----       --    ----
1     Pirate     2     Pirate
2     Monkey     null  null
3     Ninja      4     Ninja
4     Spaghetti  null  null
null  null       1     Rutabaga
null  null       3     Darth Vader
```



Original Tables

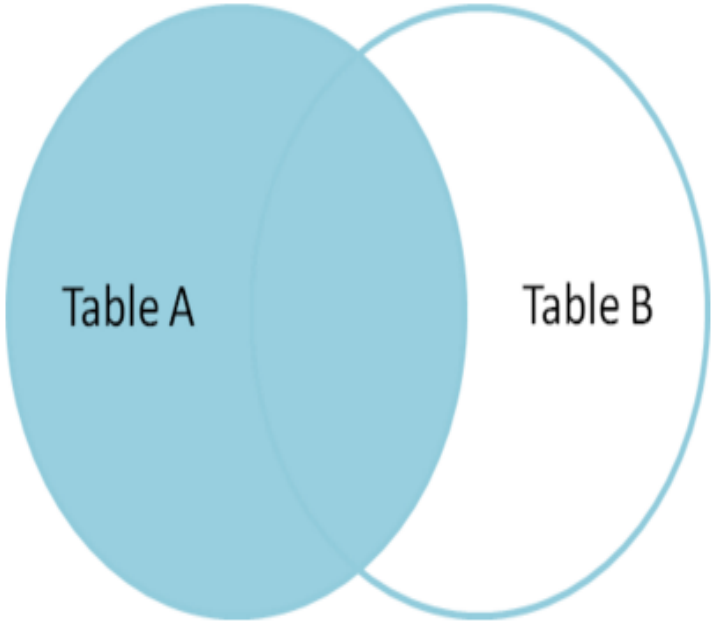
```
id name       id  name
-- ----       --  ----
1  Pirate     1   Rutabaga
2  Monkey     2   Pirate
3  Ninja      3   Darth Vader
4  Spaghetti  4   Ninja
```

LEFT OUTER JOIN

Left outer join produces a complete set of records from Table A, with the matching records (where available) in Table B. If there is no match, the right side will contain null.

```
SELECT * FROM TableA
LEFT OUTER JOIN TableB
ON TableA.name = TableB.name
```

```
id  name       id    name
--  ----       --    ----
1   Pirate     2     Pirate
2   Monkey     null  null
3   Ninja      4     Ninja
4   Spaghetti  null  null
```



Original Tables

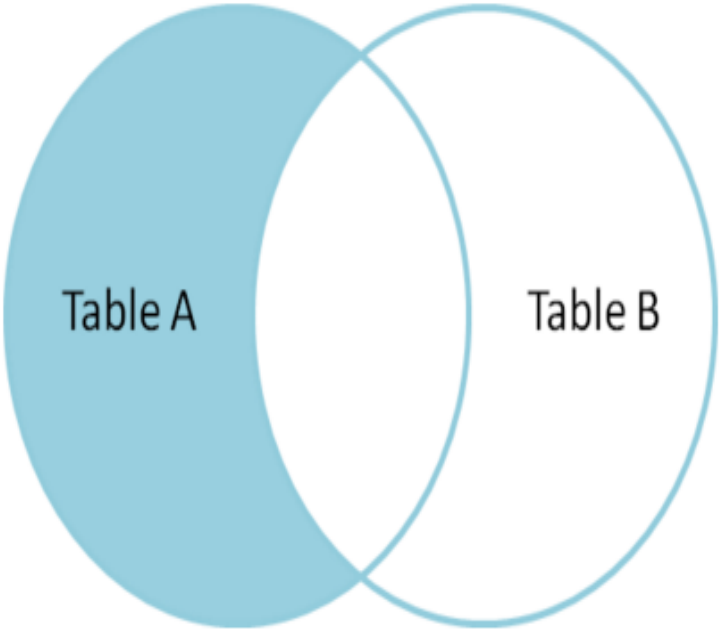
```
id name       id  name
-- ----       --  ----
1  Pirate     1   Rutabaga
2  Monkey     2   Pirate
3  Ninja      3   Darth Vader
4  Spaghetti  4   Ninja
```

LEFT OUTER JOIN with WHERE

To produce the set of records only in Table A, but not in Table B, we perform the same left outer join, then **exclude the records we don't want from the right side via a where clause**.

```
SELECT * FROM TableA
LEFT OUTER JOIN TableB
ON TableA.name = TableB.name
WHERE TableB.id IS null

id  name       id    name
--  ----       --    ----
2   Monkey     null  null
4   Spaghetti  null  null
```



Original Tables

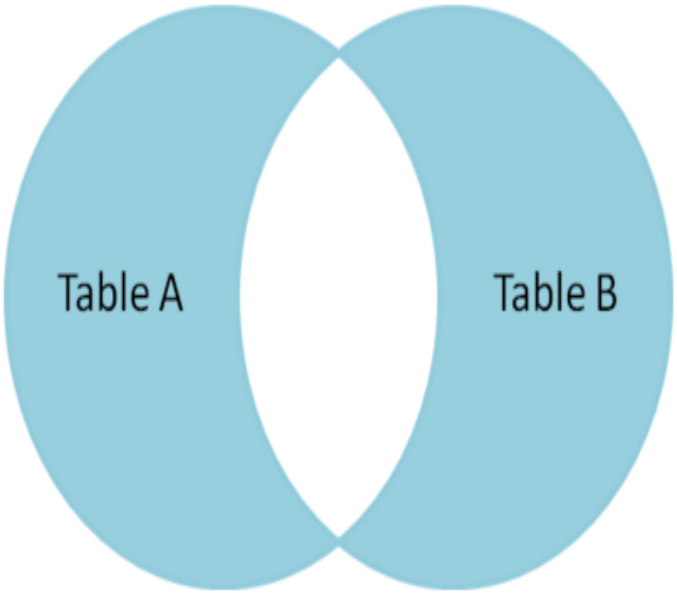
```
id name       id  name
-- ----       --  ----
1  Pirate     1   Rutabaga
2  Monkey     2   Pirate
3  Ninja      3   Darth Vader
4  Spaghetti  4   Ninja
```

FULL OUTER JOIN with WHERE

To produce the set of records unique to Table A and Table B, we perform the same full outer join, then **exclude the records we don't want from both sides via a where clause**.

```
SELECT * FROM TableA
FULL OUTER JOIN TableB
ON TableA.name = TableB.name
WHERE TableA.id IS null
OR TableB.id IS null
```

```
id    name       id    name
--    ----       --    ----
2     Monkey     null  null
4     Spaghetti  null  null
null  null       1     Rutabaga
null  null       3     Darth Vader
```



Data Analyst SQL Practice

Scenario - Eastern University Endowment

- You are a new equity analyst and your manager know about your SQL skills….
- …So he has put you in charge of all data pulls from the database!!

Hands-On #0

- Get your bearings first:
  - ~ See what is in the Financial table
  - ~ SELECT * FROM Financial where ROWID=30477
  - ~ SELECT * FROM Financial where ROWID=1940
  - ~ SELECT * FROM Financial where ticker='AMZN'

Hands-On #1 - Internet Company Revenue

- *Revenue made by Ticker-AMZN in all years*
- *Revenue made by CompanyName - 'ALPHABET INC' in all years*
- *Revenue made by Zillow in all years*
  - ~ *Try company name like '%Zillow%'*

Hands-On #1 - Internet Company Revenue

- *Revenue made by Ticker-AMZN in all years*
  - ~ SELECT * FROM Financial WHERE Ticker= 'AMZN‘
  - ~ SELECT fiscalyear, revenue FROM Financial WHERE Ticker= 'AMZN'

Hands-On #1 - Internet Company Revenue

- *Revenue made by CompanyName - 'ALPHABET INC' in all years*
  - ~ SELECT * FROM Financial WHERE companyname = 'ALPHABET INC'
  - ~ SELECT fiscalyear, revenue FROM Financial WHERE companyname = 'ALPHABET INC'

Hands-On #1 - Internet Company Revenue

- *Revenue made by Zillow*
  - ~ *Try companyname LIKE '%Zillow%'*
  - ~ SELECT * FROM Financial WHERE companyname LIKE '%ZILLOW%'
  - ~ SELECT fiscalyear, revenue FROM Financial WHERE companyname LIKE '%ZILLOW%'

Hands-On #2 Sectors

- Provide a list of company names, tickers and industry sector names for all companies in SIC2=54
- Provide a list of company names, tickers industry sector name, fiscal year and revenue for all companies in SIC2=54
- Provide a list of company names, tickers industry sector name, fiscal year and revenue for all companies in the “Pharmaceutical Preparations” sector (SIC2 or SIC4?)

Hands-On #2 Sectors

- Provide a list of company names, tickers and industry sector names for all companies in SIC2=54

```
SELECT c.ticker, c.companyname,
s.description
FROM Company C INNER JOIN SIC2
S ON c.sic2=s.codevalue
WHERE s.codevalue=54
```

Hands-On #2 Sectors

- Provide a list of company names, tickers industry sector name, fiscal year and revenue for all companies in SIC2=54

```
SELECT c.ticker, c.companyname,
s.description, f.fiscalyear, f.revenue
FROM Company C INNER JOIN SIC2 S
INNER JOIN Financial F ON
c.sic2=s.codevalue AND c.gvkey=f.gvkey
WHERE s.codevalue=54
```

Hands-On #2 Sectors

- Provide a list of company names, tickers industry sector name, fiscal year and revenue for all companies in the "Pharmaceutical Preparations" sector (SIC2 or SIC4?)

```
SELECT c.ticker, c.companyname, s.description,
f.fiscalyear, f.revenue
FROM Company C INNER JOIN SIC4 S INNER
JOIN Financial F ON c.sic4=s.codevalue AND
c.gvkey=f.gvkey
WHERE s.codevalue=2834
```

Hands-On #3 – Pharma Revenue

- Provide a list of company names, tickers industry sector name, fiscal year and revenue for all companies in the "Pharmaceutical Preparations" sector for Fiscal Year 2015 ORDERED BY REVENUE DESCENDING

Hands-On #3 - Pharma Revenue

- Provide a list of company names, tickers industry sector name, fiscal year and revenue for all companies in the “Pharmaceutical Preparations” sector for Fiscal Year 2015 ORDERED BY REVENUE DESCENDING

```
SELECT c.ticker, c.companyname, s.description,
f.fiscalyear, f.revenue
FROM Company C INNER JOIN SIC4 S INNER JOIN
Financial F ON c.sic4=s.codevalue AND c.gvkey=f.gvkey
WHERE s.codevalue=2834 AND f.fiscalyear=2015
ORDER BY REVENUE DESC
```

Hands-On #4 Food Shops Revenue

- Provide a list of company names, tickers industry sector name, fiscal year and revenue for all companies in SIC2=54. Where 2014 Revenue is greater than 20 BILLION DOLLARS!! (Revenue field is already in millions of dollars.) ORDER BY Revenue ASCENDING

Hands-On #4 Food Shops Revenue

- Provide a list of company names, tickers industry sector name, fiscal year and revenue for all companies in SIC2=54. Where 2014Revenue is greater than 20 BILLION DOLLARS!! (Revenue field is already in millions of dollars.) ORDER BY Revenue ASCENDING

```
SELECT c.ticker, c.companyname, s.description,
f.fiscalyear, f.revenue
FROM Company C INNER JOIN SIC2 S INNER
JOIN Financial F ON c.sic2=s.codevalue AND
c.gvkey=f.gvkey
WHERE s.codevalue=54 and f.fiscalyear=2014 and
f.revenue > 20000
ORDER BY F.Revenue ASC
```

Hands-On #5 Counts and Averages

- Count the number of companies in the Food Shop sector in 2014
- Find the average revenue for companies in the Food Shop sector in 2015
- Count the number of companies in the Broker dealer sector in 2015 (SIC4=6211)
- Find Average Revenue for companies in the Broker dealer sector in 2015 (SIC4=6211)

Hands-On #5 Counts and Averages

- Count the number of companies in the Food shop sector in 2014

```
SELECT COUNT(*)  FROM Company C
INNER JOIN SIC2 S INNER JOIN Financial
F ON c.sic2=s.codevalue AND
c.gvkey=f.gvkey WHERE s.codevalue=54
and f.fiscalyear=2014
```

Hands-On #5 Counts and Averages

- Find the average revenue for companies in the food shops sector in 2015

```
SELECT AVG (f.REVENUE)
FROM Company C INNER JOIN SIC2 S
INNER JOIN Financial F ON
c.sic2=s.codevalue AND c.gvkey=f.gvkey
WHERE s.codevalue=54 and
f.fiscalyear=2015
```

Hands-On #5 Counts and Averages

- Count the number of companies in the Broker dealer sector in 2015 (SIC4=6211)

```
SELECT COUNT(DISTINCT(f.ticker))
FROM Company C INNER JOIN SIC4 S
INNER JOIN Financial F ON
c.sic4=s.codevalue AND c.gvkey=f.gvkey
WHERE s.codevalue=6211 AND
f.fiscalyear=2015
```

Hands-On #5 Counts and Averages

- Find Average Revenue for companies in the Broker dealer sector in 2015 (SIC4=6211)

SELECT AVG(f.revenue) FROM Company C INNER JOIN SIC4 S INNER JOIN Financial F ON c.sic4=s.codevalue AND c.gvkey=f.gvkey

WHERE s.codevalue=6211 AND f.fiscalyear=2015

Hands-On #6 - Group By

- Provide SIC4 code, sector name and count of all companies in
  - ~ Bottled and canned soft drinks
  - ~ Wines, brandy and Brandy spirits
  - ~ Bottled and canned soft drinks
  - ~ Distilled and blended liquors
  - ~ HINT if you need to address multiple criteria in a where clause you can try WHERE Code in (A,B,C,D)

Hands-On #6 - Group By

```
SELECT s.codevalue, s.description,
count(c.ticker) FROM SIC4 S INNER JOIN
Company c ON s.codevalue=c.SIC4
WHERE S.codevalue IN (2082, 2084, 2086,
2085)
GROUP BY S.codevalue
```

Hands-On #7

- Harder:
  - ~ Provide two digit SIC Code, sector name and Average 2015 Revenue for each sector and order by avg revenue descending

Hands-On #7

```
SELECT s.codevalue, s.description,
count(c.ticker) AS Count, avg(f.revenue) AS
AverageRevenue
FROM COMPANY C INNER JOIN Financial
F INNER JOIN SIC2 S ON
s.codevalue=c.SIC2 AND c.gvkey=f.gvkey
WHERE f.fiscalyear=2015
GROUP BY S.codevalue
ORDER BY AverageRevenue DESC
```