



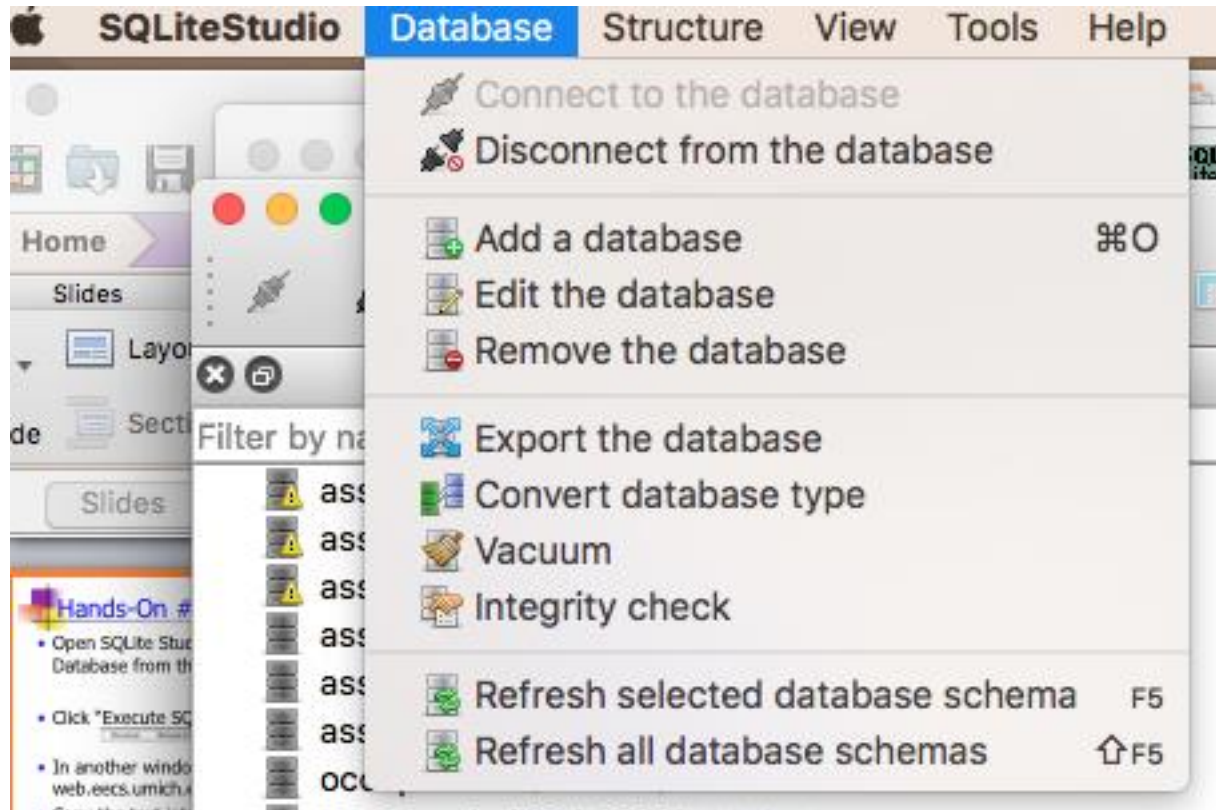
# SQL Challenge

---

- Sit with your teams and bask in your collective knowledge
- Go to the Github page and download
  - SQL Challenge ppt or pdf
  - director\_db\_schema.txt
  - Direct99\_sqlite.csv
  - AT\_Retail\_DB.db
- Read SQL Challenge and Complete the exercises

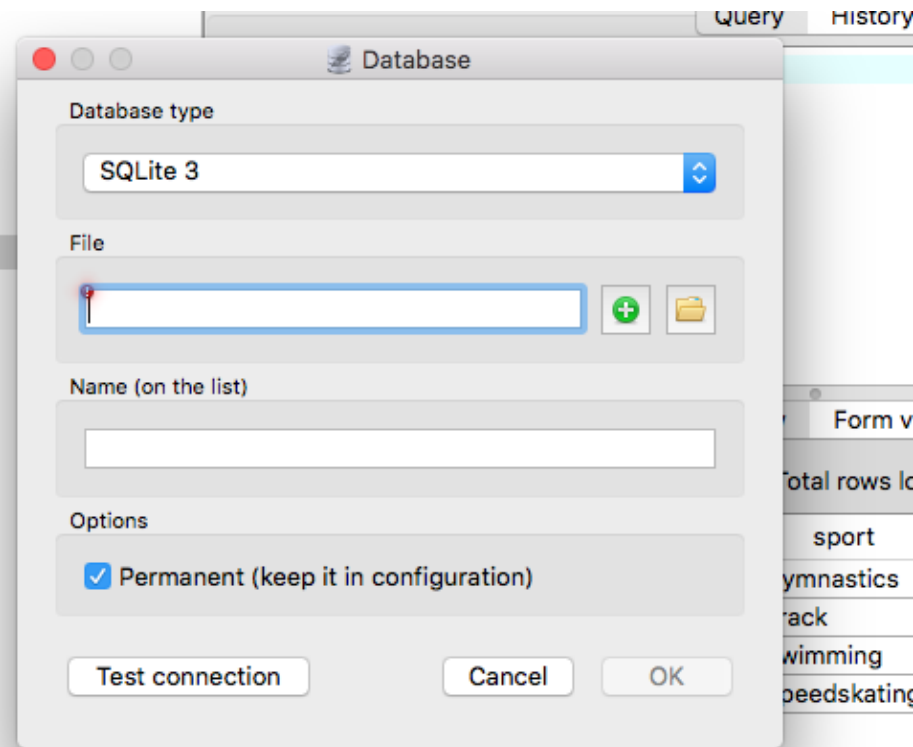
# Directors Revisited ()

- Open SQLite Studio and Select “Add a Database from the Database Menu”



# Directors Revisited ()

- Click the green plus button to add a new file





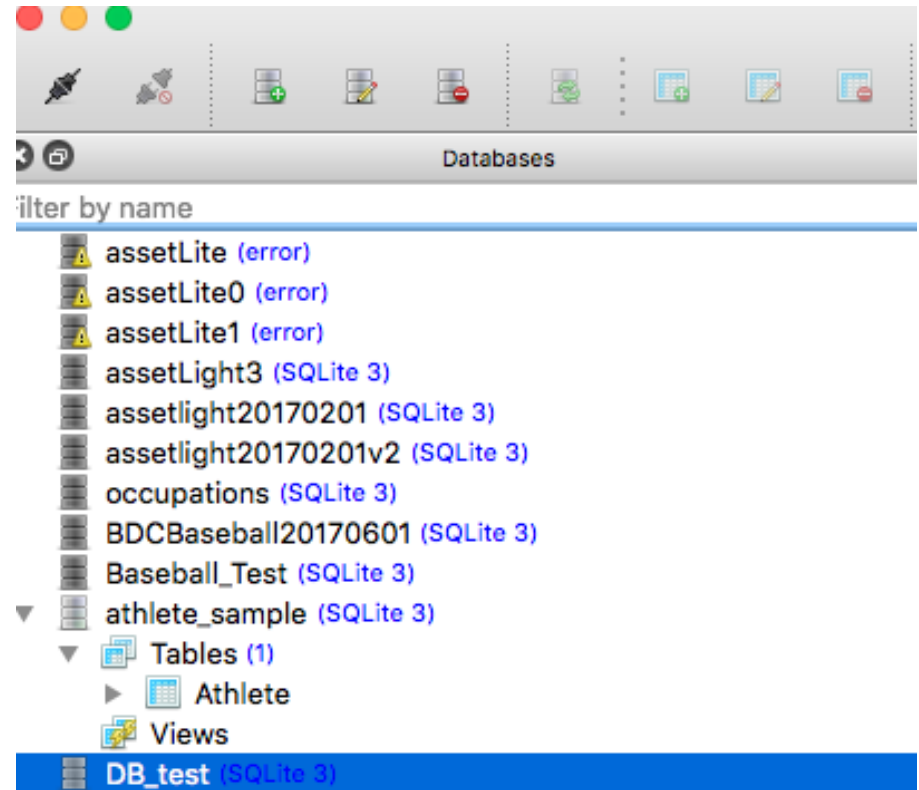
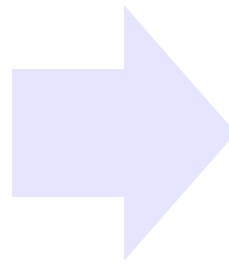
# Directors Revisited ()

---

- Name it something appropriate like Director\_DB and save it to a folder where you can find it.

# Directors Revisited ()

- Select/highlight your DB File in the right hand window and click Connect to database





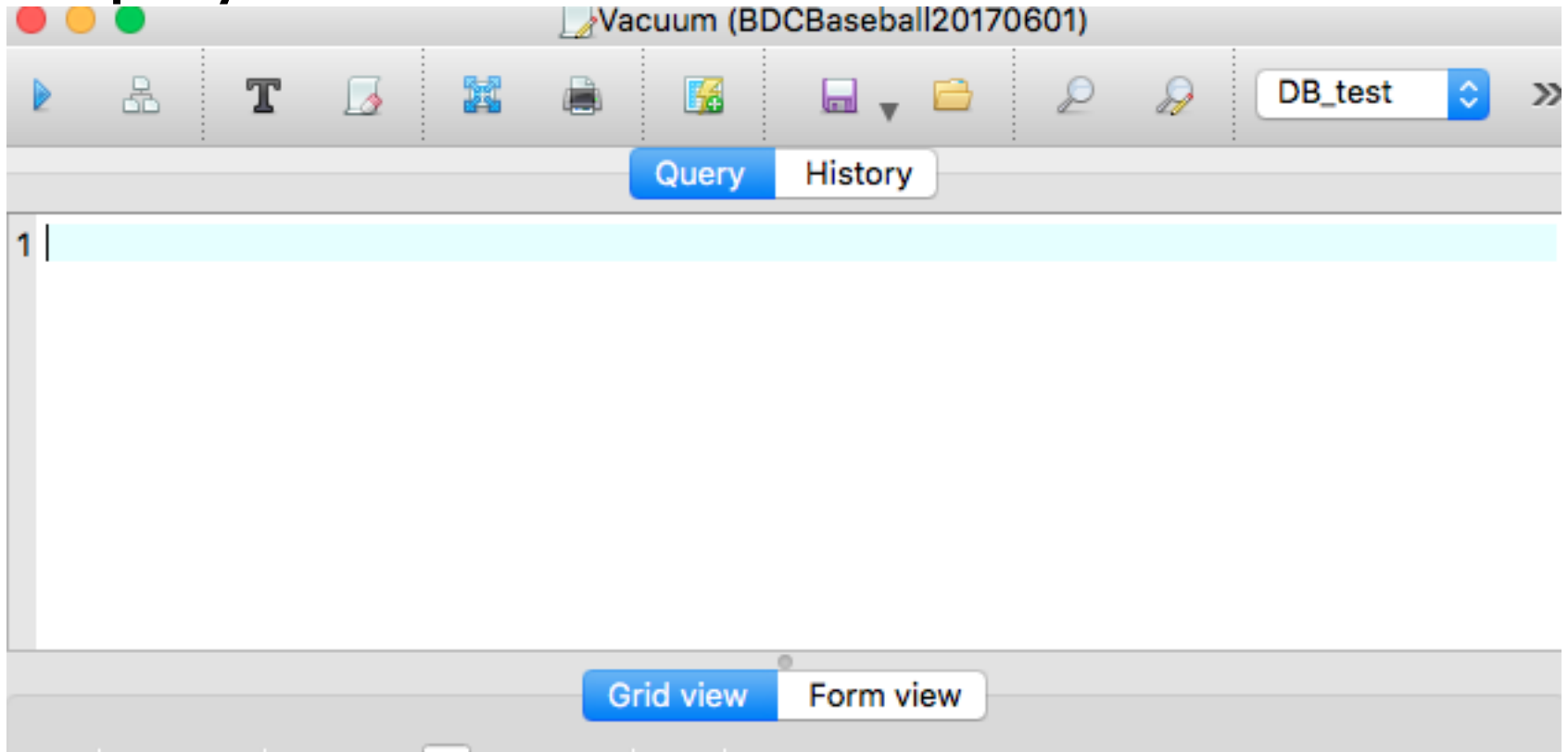
# Directors Revisited ()

---

- Open the file `director_db_schema.txt`
- Highlight all of the text you see and copy it to your clipboard.

# Directors Revisited ()

- Paste all of the text in the query window, highlight all of the text and then click the “play” button.





# Directors Revisited ()

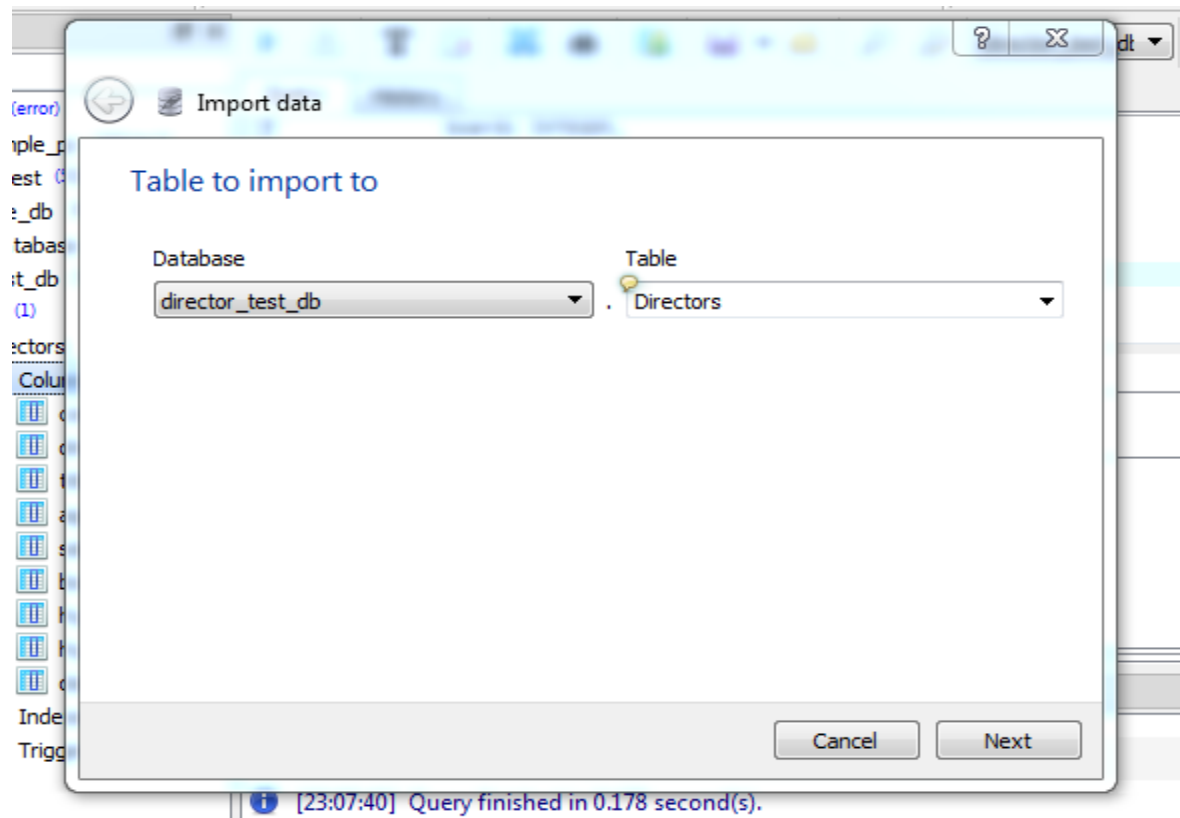
---

- If all has gone according to plan you should have a database in the left hand pane with one table “Directors” with nine columns



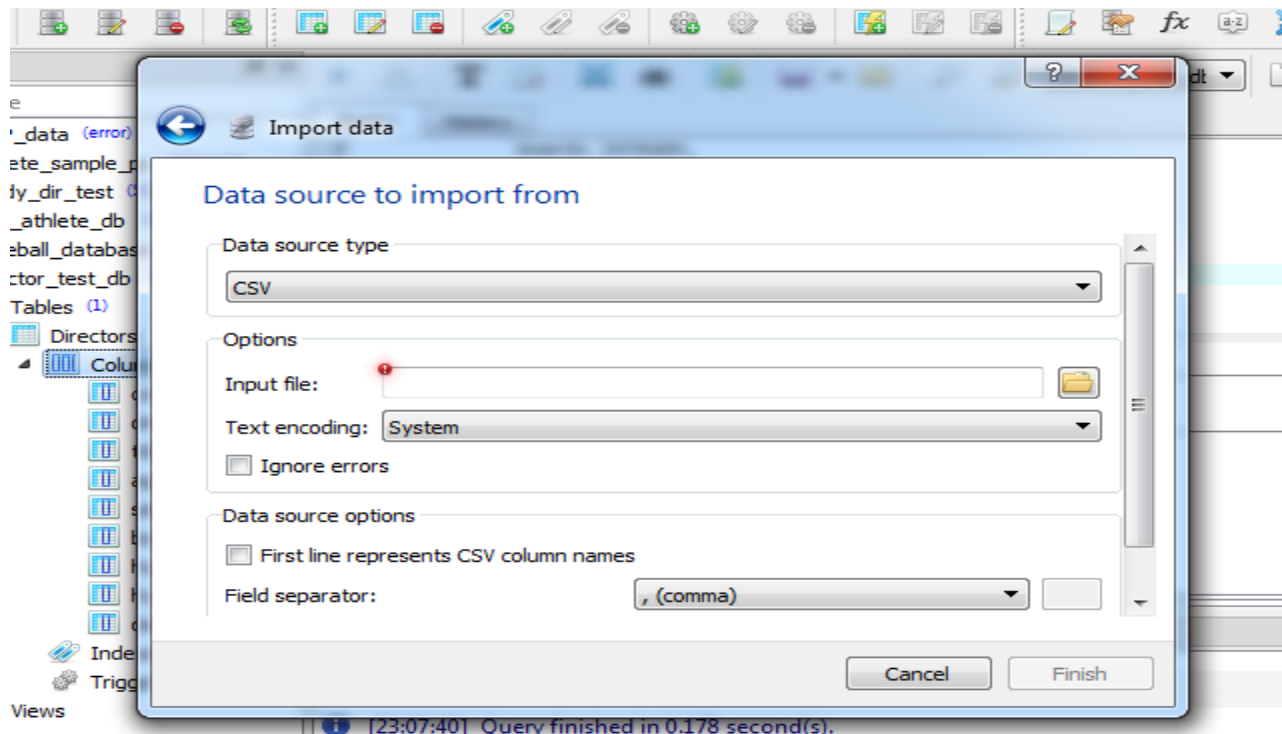
# Directors Revisited ()

- Right click the table name Directors and select “Import into the table” You should get the image below.



# Directors Revisited ()

- Click Next which will bring you to the screen below. Follow the instructions to locate and import the file "direct99\_sqlite.csv"





# Directors Revisited ()

---

- It may take a little while to load the data in to the table (especially on a PC) but I promise you that it is working! 😊 When the data is loaded.
- After the data is loaded you should try three things
  - Write queries that gives you the names of all of the directors for Apple, JP Morgan and Caterpillar
  - Reproduce the query from yesterday that gives you all of the networks pairs for the corporate interlocks.



# Directors Revisited ()

---

- HINTS

- You probably don't know how the company is named. But you can do a wild card search with the % symbol and the LIKE commands. For example if I was looking for Sarah Lee I might say `"SELECT* from Directors WHERE company_name LIKE '%sarah%'"`
- Did we join the table with itself yesterday?

# Ann Taylor and SQL – “SELECT”



***“Good Morning! Rumor has it that you have learned some SQL. That’s great. I would like to give you access to our DB. Go to the Github page, download AT\_Retail\_DB.db and add it to SQLite Studio with the “Add A Database” Functionality***

# Ann Taylor and SQL – “SELECT”



***“Get Familiar with the DB! Try running `SELECT * FROM Employees LIMIT 10` to get a sense of what’s in that table. Repeat for the other tables! Then I will have some questions for you!”***

# Ann Taylor and SQL – “SELECT”



***“My name is Jenna!  
How might you find out  
my employee  
information?”***

# Ann Taylor and SQL – “SELECT”



***“You know..I think this Crystal Pendant is radiant! Can you write a query to tell me which employees have sold this item?”***



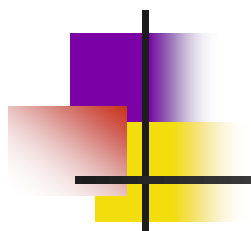


# Ann Taylor and SQL – “SELECT”



***“So I am really competitive! Can you write a query that gives me the employee ID, first name, last name and total sales of each employee? I want to know if I am at the top! Remember, I’m JENNA!!”***

***(Hint-You may have to join three tables!)***



# APPENDIX

## ✿ SQL SELECT STATEMENTS

### **SELECT \* FROM tbl**

Select all rows and columns from table tbl

### **SELECT c1,c2 FROM tbl**

Select column c1, c2 and all rows from table tbl

### **SELECT c1,c2 FROM tbl WHERE conditions ORDER BY c1 ASC, c2 DESC**

Select columns c1, c2 with where conditions and from table tbl order result by column c1 in ascending order and c2 in descending order

### **SELECT DISTINCT c1, c2 FROM tbl**

Select distinct rows by columns c1 and c2 from table tbl.

### **SELECT c1, aggregate(expr) FROM tbl GROUP BY c1**

Select column c1 and use aggregate function on expression expr, group columns by column c1.

### **SELECT c1, aggregate(expr) AS c2 FROM tbl GROUP BY c1 HAVING c2 > v**

Select column c1 and c2 as column alias of the result of aggregate function on expr. Filter group of records with c2 greater than value v

## ✿ SQL UPDATE TABLE

### **INSERT INTO tbl(c1,c2,...) VALUES(v1,v2...)**

Insert data into table tbl

### **INSERT INTO tbl(c1,c2,...) SELECT c1,c2.. FROM tbl2 WHERE conditions**

Insert data from tbl2 into tbl

### **UPDATE t SET c1 = v1, c2 = v2...**

Update data in table tbl  
**WHERE conditions**

### **DELETE FROM tbl WHERE conditions**

Delete records from table tbl based on WHERE conditions.

### **TRUNCATE TABLE tbl**

Drop table tbl and re-create it, all data is lost

## ✿ SQL TABLE STATEMENTS

### **CREATE TABLE tbl( c1 datatype(length) c2 datatype(length) ... PRIMARY KEY(c1)**

)  
Create table tbl with primary key is c1

### **DROP TABLE tbl**

Remove table tbl from database.

### **ALTER TABLE tbl ADD COLUMN c1 datatype(length)**

Add column c1 to table tbl

### **ALTER TABLE tbl DROP COLUMN c1**

Drop column c1 from table tbl

## ✿ SQL JOIN STATEMENTS

### **SELECT \* FROM tbl1 INNER JOIN tbl2 ON join-conditions**

Inner join table tbl1 with tbl2 based on join-conditions.

### **SELECT \* FROM tbl1 LEFT JOIN tbl2 ON join-conditions**

Left join table tbl1 with tbl2 based on join-conditions.

### **SELECT \* FROM tbl1 RIGHT JOIN tbl2 ON join-conditions**

Right join table tbl1 with tbl2 based on join-conditions.

### **SELECT \* FROM tbl1 RIGHT JOIN tbl2 ON join-conditions**

Full outer join table tbl1 with tbl2 based on join-conditions.



**TABLE 7.2** SQL Data Manipulation Commands

COMMAND OR OPTION	DESCRIPTION
INSERT	Inserts row(s) into a table
SELECT	Selects attributes from rows in one or more tables or views
WHERE	Restricts the selection of rows based on a conditional expression
GROUP BY	Groups the selected rows based on one or more attributes
HAVING	Restricts the selection of grouped rows based on a condition
ORDER BY	Orders the selected rows based on one or more attributes
UPDATE	Modifies an attribute's values in one or more table's rows
DELETE	Deletes one or more rows from a table
COMMIT	Permanently saves data changes
ROLLBACK	Restores data to their original values

**TABLE  
7.2**

**SQL Data Manipulation Commands (continued)**

COMMAND OR OPTION	DESCRIPTION
<b>COMPARISON OPERATORS</b>	
=, <, >, <=, >=, <>	Used in conditional expressions
<b>LOGICAL OPERATORS</b>	
AND/OR/NOT	Used in conditional expressions
<b>SPECIAL OPERATORS</b>	Used in conditional expressions
BETWEEN	Checks whether an attribute value is within a range
IS NULL	Checks whether an attribute value is null
LIKE	Checks whether an attribute value matches a given string pattern
IN	Checks whether an attribute value matches any value within a value list
EXISTS	Checks whether a subquery returns any rows
DISTINCT	Limits values to unique values
<b>AGGREGATE FUNCTIONS</b>	Used with SELECT to return mathematical summaries on columns
COUNT	Returns the number of rows with non-null values for a given column
MIN	Returns the minimum attribute value found in a given column
MAX	Returns the maximum attribute value found in a given column
SUM	Returns the sum of all values for a given column
AVG	Returns the average of all values for a given column



# Useful Resources

---

- URLs

- <http://www.w3schools.com/sql/>
- <http://www.tutorialspoint.com/sqlite/>
- [http://www.tutorialspoint.com/sqlite/sqlite\\_python.htm](http://www.tutorialspoint.com/sqlite/sqlite_python.htm)

- Books

- Learning SQL – Alan Beaulieu

- Online Courses

- Udemy – The Complete SQL Bootcamp (\$)