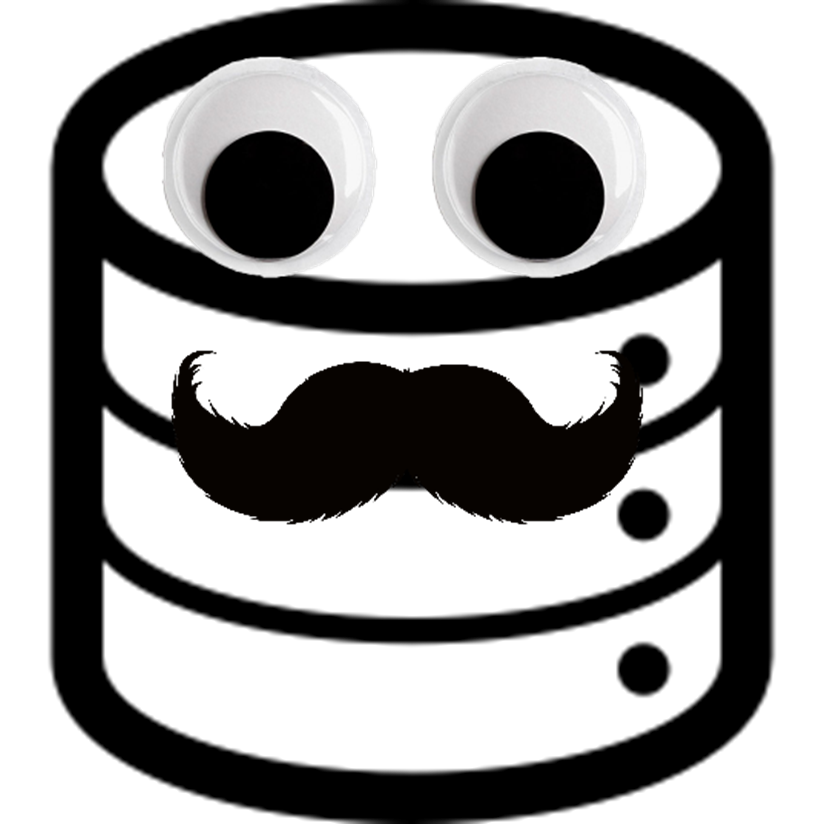
*WELCOME TO CLASS*

[*Please Fill Out Survey at End*](https://www.surveymonkey.com/r/NHRDMQK)



*SQL for Smart Folks\* Bootcamp*

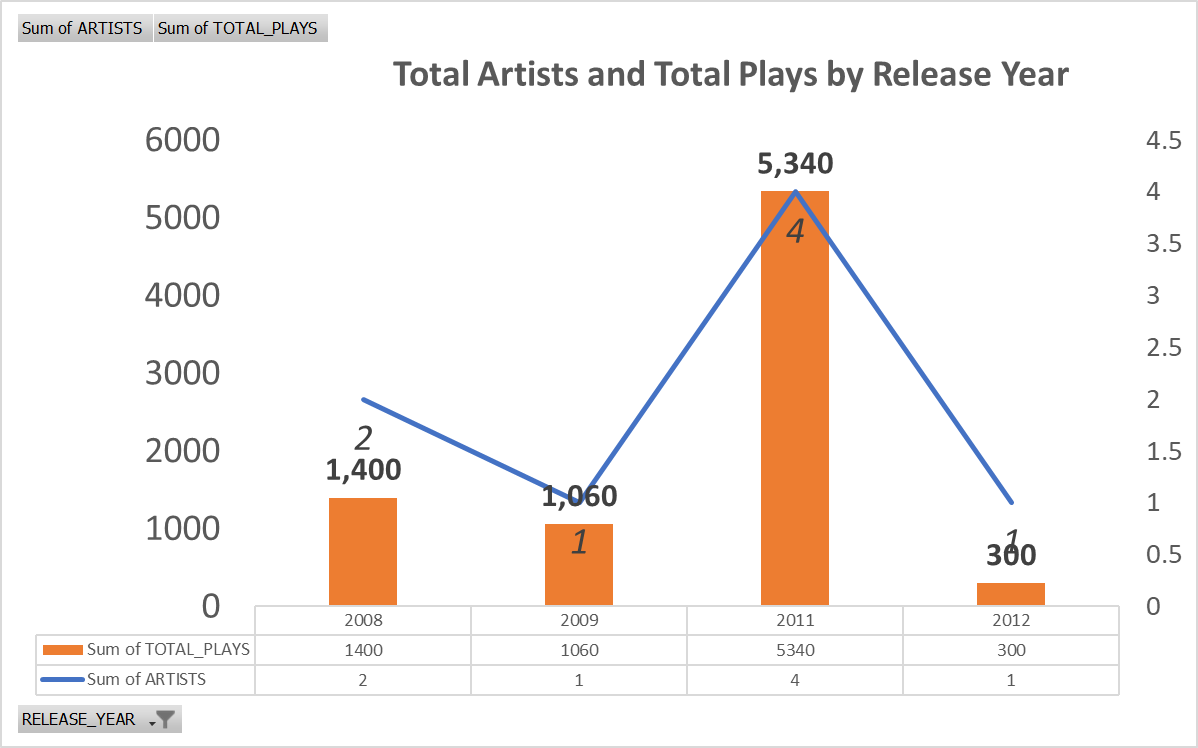
Instructor: [Michael Valeri](https://www.linkedin.com/in/mvaleri/)

*\*YOU – Common People*

**0. Pre-Work – Download PG Admin and Getting Curriculum**

|  |
| --- |
| ***\*GETTING STARTED GUIDE\****  *Step 1:* [*Download PG Admin 4*](https://www.pgadmin.org/download/)*. Select OS from list. If trouble opening on Mac after download, see troubleshoot guide* [*here*](https://www.pgadmin.org/faq/)*.*    *<Windows/MAC 🡪 Select latest version from list 🡪 2nd Option .EXE/.DMG*    *Step 2: Find SQL Bootcamp Curriculum* [*here*](https://github.com/mvaleri12/valeri-analytics-sql-bootcamp)  *<Open Link above 🡪 Download Zip>*  *<Find Zip File in Downloads Folder called valeri-analytics-sql-bootcamp-master>*  *<Open Zip File 🡪 Drag folder onto Desktop so you can find later>*    *Step 3: Connect to AWS server on PG Admin 4; credentials on Page 6*  *Step 4: Start brainstorming questions on introduction example, Page 3* |

1. **Introduction – Business Review**

****

*What questions can you expect to be generated from the above chart?*

|  |
| --- |
|  |

1. **The Analytics Framework – Where does SQL fit in?**

|  |
| --- |
|  |

1. **Relational Database Fundamentals – Entity Relationship Diagrams (ERDs)**

|  |
| --- |
|  |

1. **Datatype Fundamentals – Tables, Schemas & Granularity**

*ALBUMS (Table Schema)*

|  |  |
| --- | --- |
| **Field/Column** | **Data Type** |
| row |  |
| artist |  |
| album |  |
| release\_date |  |
| genre |  |
| plays |  |
| rating |  |
| org\_price |  |
| market\_value |  |
| burned |  |
| playable |  |

*ALBUMS (Table Granularity)*

Each row/record in this table represents a\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

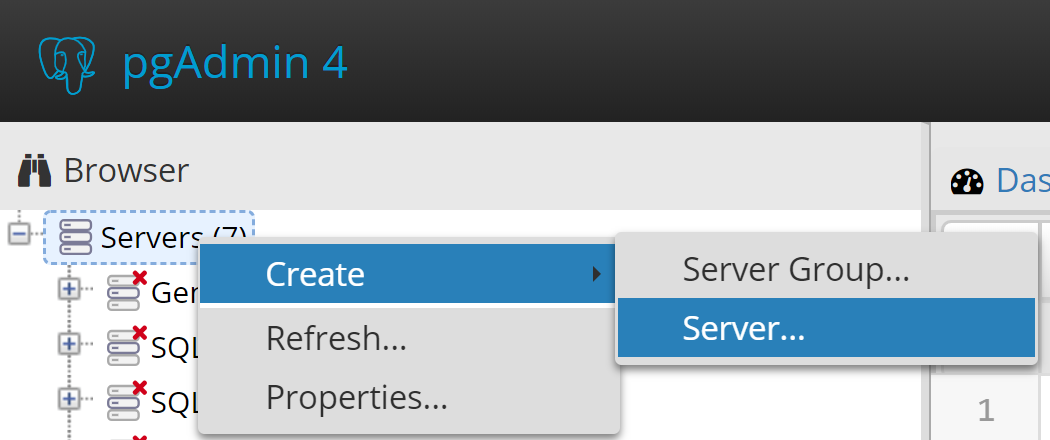
Takeaway: You should NEVER guess \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

1. **Connecting to an RDB (Red) – PostGreSQL Server Details**

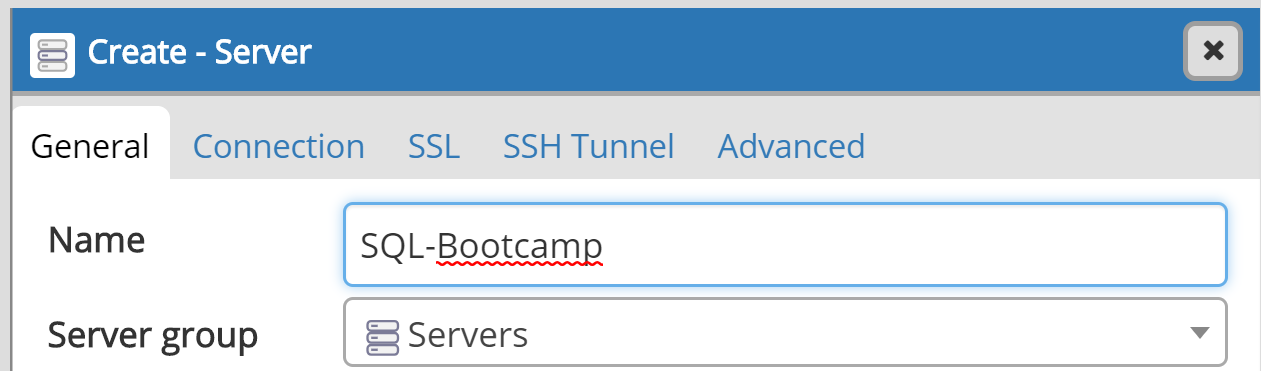
\*Make sure to DISCONNECT from VPN or Corporate Network\*

1. Open PG-Admin 4

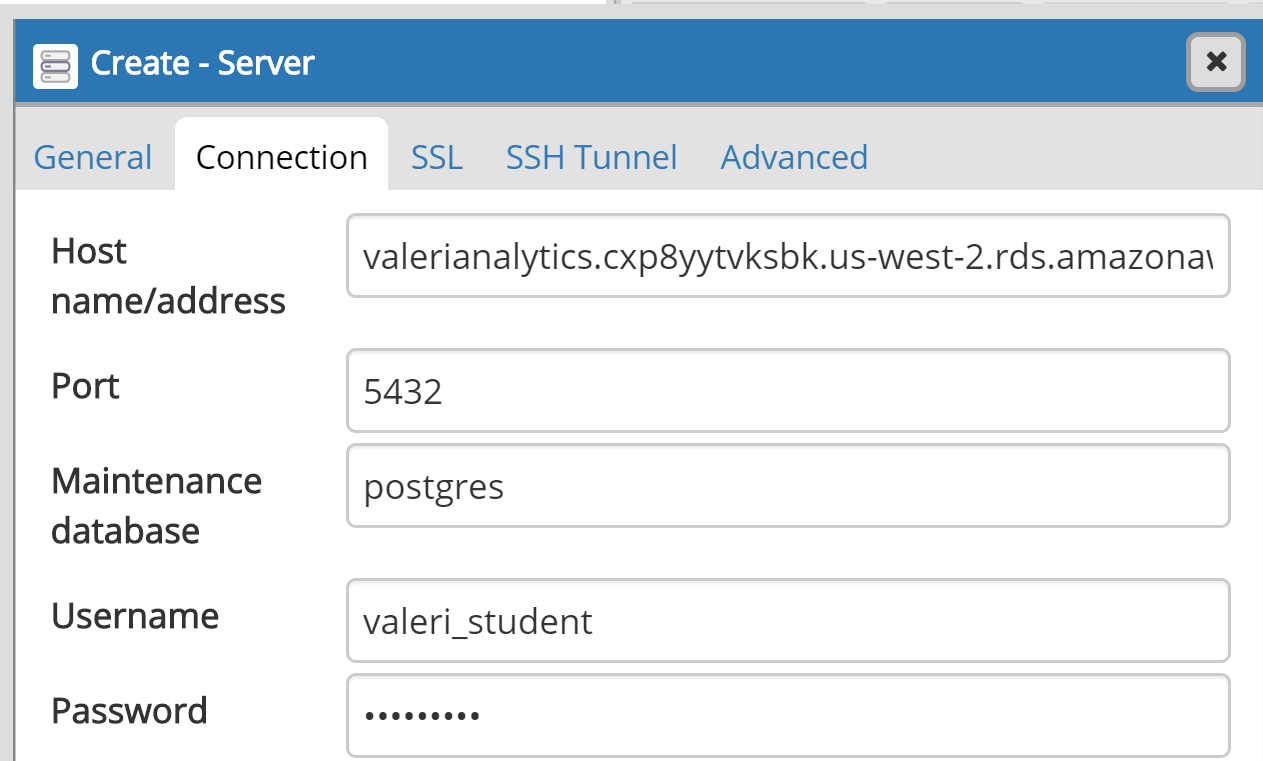
2. Right click “Servers” 🡪 “Create” 🡪 “Server”



3. “General” 🡪 Write “SQL-Bootcamp” in Name



4. “Connection” 🡪 Write below credentials in fields, then click “Save”



|  |  |
| --- | --- |
| *PRIMARY SERVER DETAILS*  Hostname/address: **valerianalytics.cxp8yytvksbk.us-west-2.rds.amazonaws.com**  Port: **5432**  Maintenance\_DB: **postgres**  Username: **valeri\_student**  Password: **shawshank** | *SECONDARY SERVER*  Hostname/address: **analyticsga.cuwj8wuu6wbh.us-west-2.rds.amazonaws.com**  Port: **5432**  Maintenance\_DB: **postgres**  Username: **analytics\_student**  Password: **analyticsga** |

1. **How to Extract Data out of an RDB – The Basic Syntax of SQL**

|  |  |  |
| --- | --- | --- |
| **SEQ.** | **KEYWORD** | **WHAT IT DOES** |
| 1 | SELECT |  |
| 2 | FROM |  |
| 3 | WHERE |  |
| 4 | GROUP BY |  |
| 5 | HAVING |  |
| 5 | ORDER BY |  |
| 6 | LIMIT |  |

Takeaways: SQL keywords are case \_\_\_\_\_\_\_\_\_\_\_; but the \_\_\_\_\_\_\_\_ matters.

1. **Asking Questions in SQL – SELECT, FROM, ORDER BY and LIMIT Keywords**

*Instructor Guided*

|  |  |
| --- | --- |
| **Question (English)** | **Notes** |
| I want to see ALL records in the *ALBUMS* table… |  |
| I want to see a SAMPLE of 10 ALBUMS and ARTISTS in the *ALBUMS* table… |  |
| I want to see a list of ARTISTs in the ALBUMS table… |  |
| I want to see a DISTINCT list of ARTISTS in the ALBUMS table… |  |
| I want to see a list of ALBUMs in the ALBUMS table, sorted from A-Z… |  |
| I want to see a list of ALBUMS their GENRE, sorted by highest to lowest price |  |
| I want a list of ALBUMS and the profit realized from selling each ALBUM |  |
| I want to order the list of ALBUMS by lowest profit album to highest profit ALBUM |  |

1. **Filtering data in SQL – WHERE keyword**

*Instructor Guided*

|  |  |
| --- | --- |
| **Question (English)** | **Notes** |
| 1 Condition, 1 Column: Text Example: I want to see a list of Rock Albums… |  |
| 1 Condition, 1 Column: Integer Example: I want to see a list of Albums with over 1000 plays |  |
| 1 Condition, 1 Column: Date Example: I want to see a list of Albums released in 2011 |  |
| Multiple Conditions, 1 Column Example: I want to see a list of Rock and Rap ALBUMs… |  |
| Multiple Conditions, Multiple Columns Example: I want to see a list of Rock and Rap Albums with at least a 4 rating.... |  |
| NULL Conditions: I want to see a list of Albums with no ARTIST name… |  |
| OR Conditions: I want to see a list of Rock Albums with a rating of at least 4 or they album was burned. |  |
| Pattern Matching: I want see a list of ARTISTS that have a ‘the’ in their name… |  |

Takeaways:

The \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ affects the syntax in the \_\_\_\_\_\_\_\_\_\_\_\_ clause.

For 1 condition, 1 column I use \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

For multiple conditions, 1 column I use \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

For multiple conditions, multiple columns I use \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

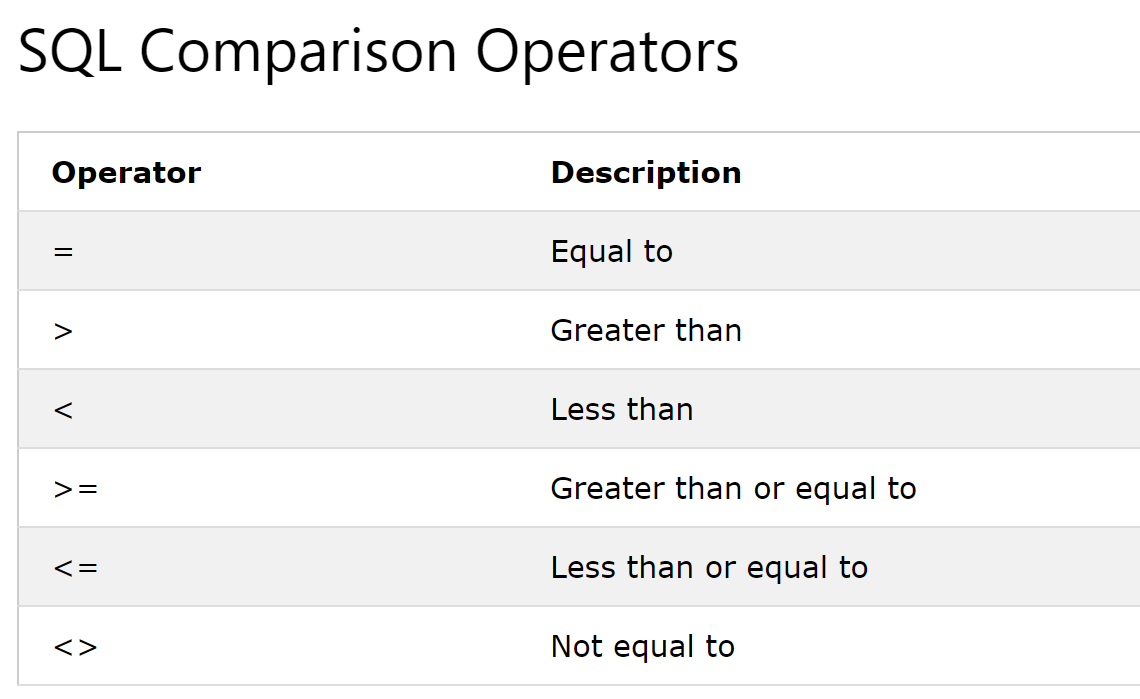
For NULL conditions, I use \_\_\_\_\_\_\_\_\_\_\_\_.

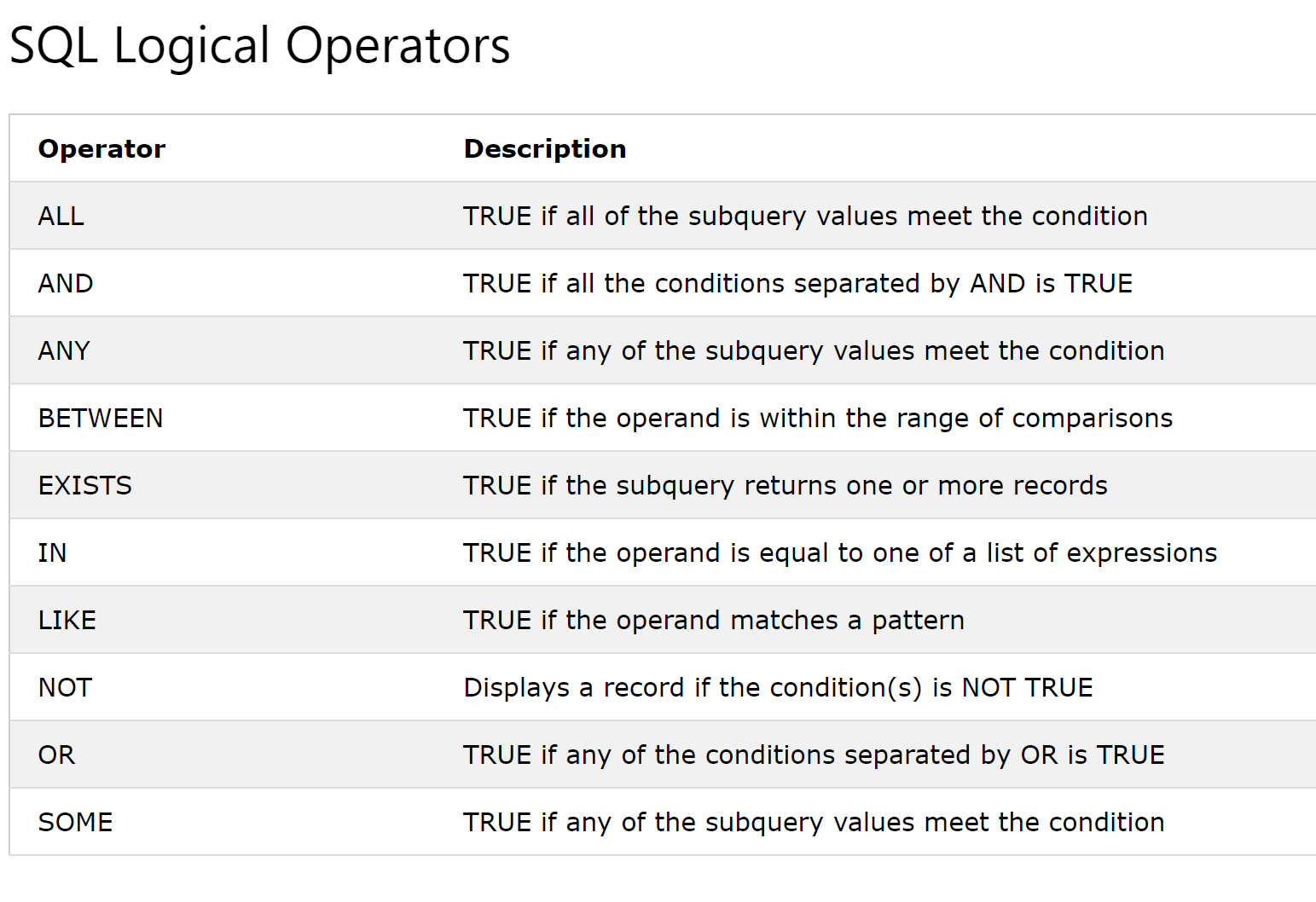
USING \_\_\_\_\_\_\_\_ will \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ all previous conditions. We can control for this by\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

For pattern matching use\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ to normalize the

column field\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ and \_\_\_\_\_\_\_\_\_\_\_\_\_ function to

search.





1. **Aggregating Data in SQL – SUM, COUNT, MIN, MAX, and AVG functions**

|  |  |
| --- | --- |
| **FUNCTION** | **WHAT DOES IT DO?** |
| SUM(COLUMN\_X) | Sums the values in entire column |
| COUNT(COLUMN\_X) | Counts the number of records in column (that aren’t null) |
| COUNT(DISTINCT COLUMN\_X) | Counts the number of distinct records in column (that aren’t null) |
| MIN(COLUMN\_X) / MAX(COLUMN\_X) | Finds min/max value in column |

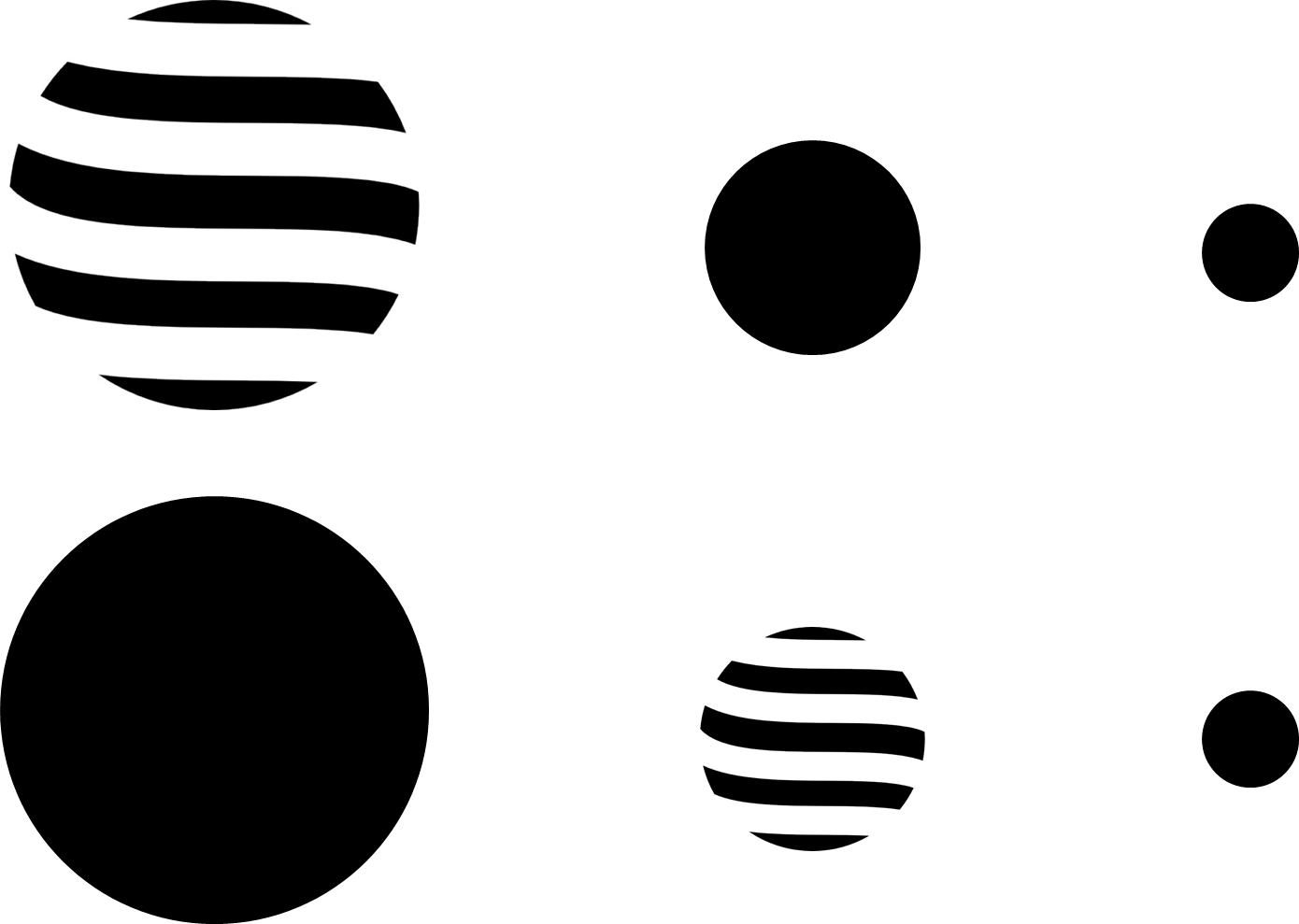
*Instructor Guided*

|  |  |
| --- | --- |
| **Question (English)** | **Notes** |
| What is the SUM of the Total Plays in the table? |  |
| What is the COUNT of Artists in the Table? |  |
| What is the DISTINCT COUNT of Artists in the Table? |  |

Takeaways: Using an aggregate without a \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ is not helpful.

1. **Slicing Aggregates by Dimensions – Using GROUP BY in SQL**

How many rocks are in the box? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

****

*Rock Count by Size*

|  |  |
| --- | --- |
| **Rock Size** | **Count** |
| Big |  |
| Medium |  |
| Small |  |

*Rock Count by Stripes (T / F)*

|  |  |
| --- | --- |
| **IS\_STRIPED** | **Count** |
| TRUE |  |
| FALSE |  |

*Rock Count by Stripes and Size*

|  |  |  |
| --- | --- | --- |
| **Rock Size** | **IS\_STRIPED** | **Count** |
| Big | Y |  |
| Big | N |  |
| Medium | Y |  |
| Medium | N |  |
| Small | N |  |

*ROCKS\_IN\_THE\_BOX (Table Schema)*

|  |  |
| --- | --- |
| **Field/Column** | **Data Type** |
| rock\_id | text |
| is\_striped | boolean |
| rock\_size | text |
| weight | int |

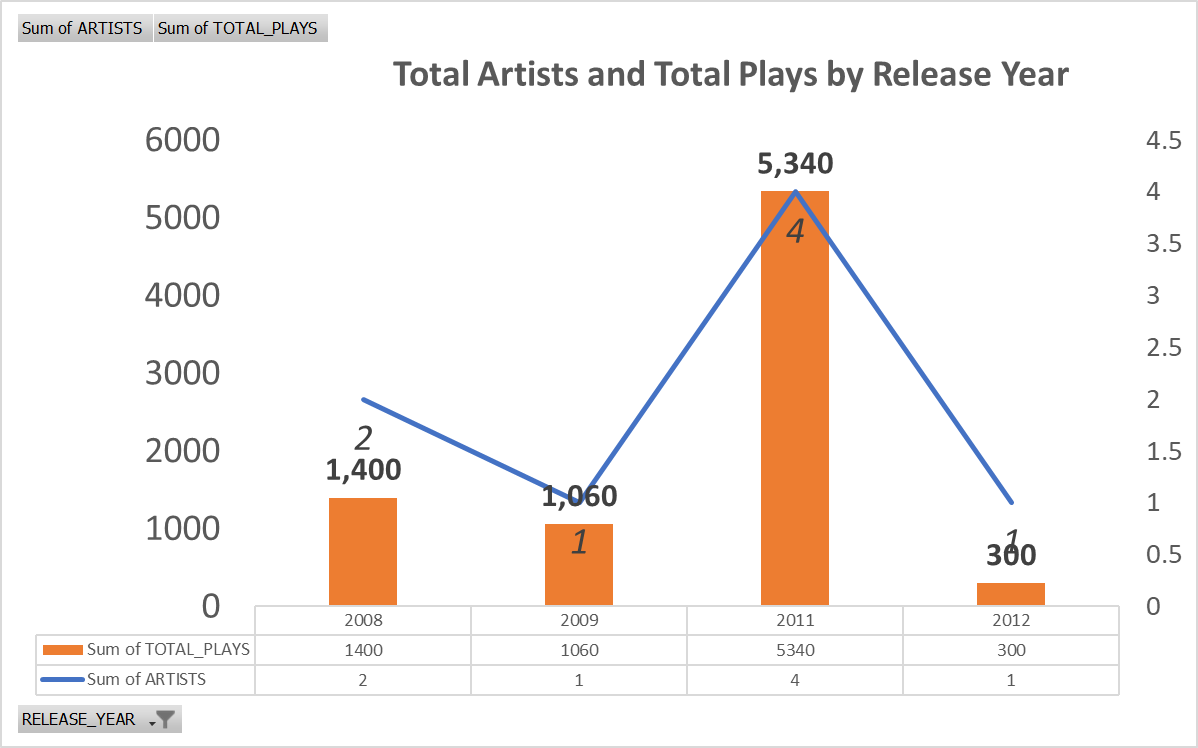
*Instructor Guided: Write an SQL statement that counts the number of rocks by Size and Stripe*

**SELECT**

**FROM** *ROCKS\_IN\_THE\_BOX*

**GROUP BY**

Takeaways: You use a GROUP BY when you want to understand an AGGREGATE by \_\_\_\_\_\_\_\_\_\_\_\_\_\_. The GROUP BY and SELECT statement are identical; except you don’t include \_\_\_\_\_\_\_\_\_\_\_\_ or \_\_\_\_\_\_\_\_\_\_\_\_\_\_ in the GROUP BY.

****

*Independent Practice: Write a SQL statement that counts the number artists and sum total plays by release year.*

*How could we further break this down? Modify the above code.*

1. **Relating Multiple Tables – Using JOINs in SQL**

*“I want to see Total Sales by Customer State”*



**Table: CUSTOMERS**

**Table: SALES**

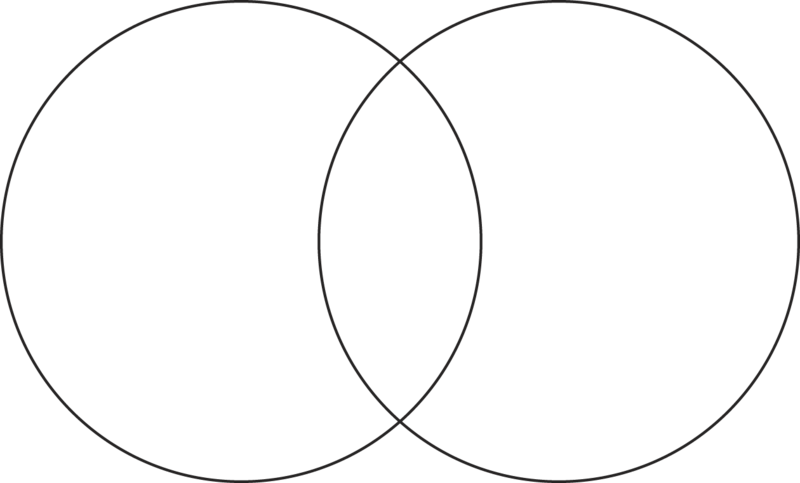
1. Why do we use a JOIN?

2. What is required to JOIN two tables?

3. What is a JOIN similar too?

4. How do we visualize a join

|  |  |
| --- | --- |
| **TABLE A:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **TABLE B:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
| Primary Table (most important) | Secondary Table (nice to have) |



*Write a query that JOINs the SALES* table to the *CUSTOMERS* table.

Draw the difference between a LEFT JOIN and a JOIN

1. **Circles on Circles - Different Types of JOINs**

