Introduction to SQL

With notes and examples from W3schools apps, Microsoft's AdventureWorksLT database, & SQLZoo

What is SQL?

- SQL stands for ***Structured Query Language***
- SQL lets you access and manipulate databases
- SQL is an ANSI (American National Standards Institute) standard

What Can SQL do?

SQL can execute queries against a database

SQL can retrieve data from a database

SQL can insert records in a database

SQL can update records in a database

SQL can delete records from a database

SQL can create new databases

SQL can create new tables in a database

SQL can create stored procedures in a database

SQL can set permissions on tables, procedures, and views

What are Relational Database Systems (RDBMS)

SQL is a special-purpose programming language designed for managing data held in …..

- **A relational database management system (RDBMS)**
  - The data in RDBMS is stored in database objects called tables.
  - A table is a collection of related data entries and it consists of columns and rows.
  - A single database can house several tables

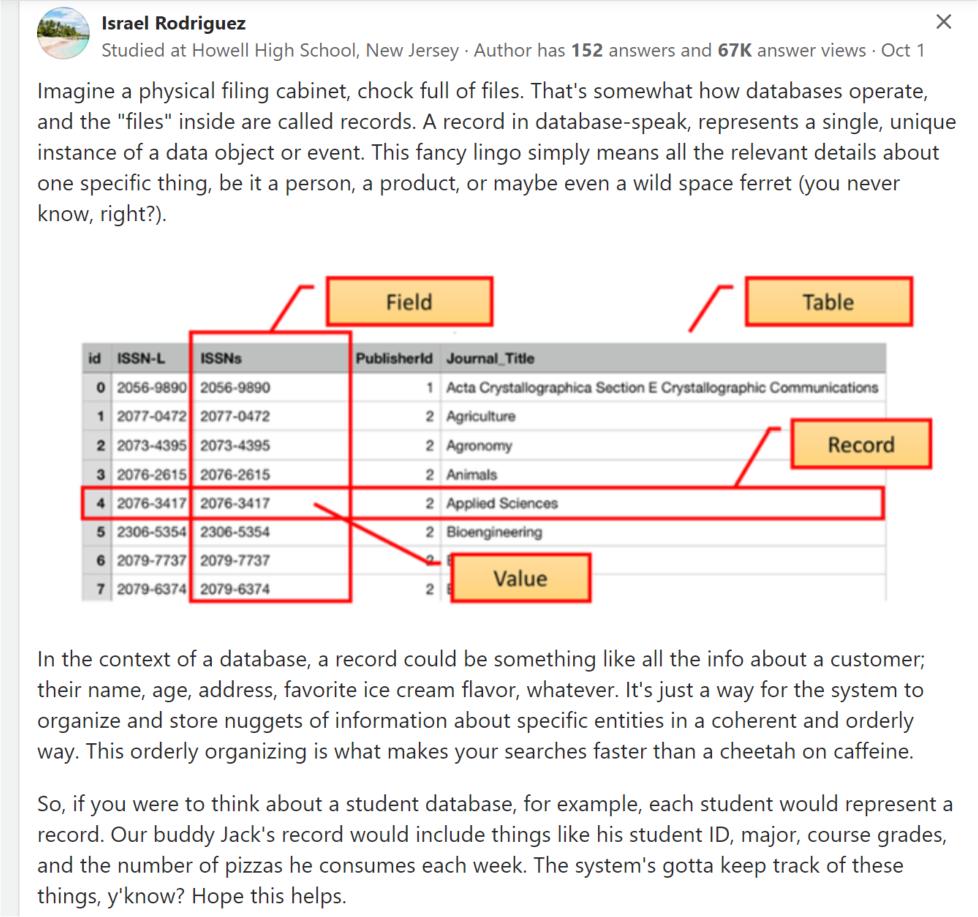
SQL is a Standard - BUT....

Although SQL is an ANSI (American National Standards Institute) standard, **there are different versions of the SQL language.**

However, to be compliant with the ANSI standard, they **all support at least the major commands** (such as SELECT, UPDATE, DELETE, INSERT, WHERE) in a similar manner.

BUT, most of the SQL database programs **also have their own proprietary extensions in addition to the SQL standard**!

Example of the table structure for relational databases



Israel Rodriguez

Studied at Howell High School, New Jersey · Author has **152** answers and **67K** answer views · Oct 1

Imagine a physical filing cabinet, chock full of files. That's somewhat how databases operate, and the "files" inside are called records. A record in database-speak, represents a single, unique instance of a data object or event. This fancy lingo simply means all the relevant details about one specific thing, be it a person, a product, or maybe even a wild space ferret (you never know, right?).

id	ISSN-L	ISSNs	PublisherId	Journal_Title
0	2056-9890	2056-9890	1	Acta Crystallographica Section E Crystallographic Communications
1	2077-0472	2077-0472	2	Agriculture
2	2073-4395	2073-4395	2	Agronomy
3	2076-2615	2076-2615	2	Animals
4	2076-3417	2076-3417	2	Applied Sciences
5	2306-5354	2306-5354	2	Bioengineering
6	2079-7737	2079-7737	2	[illegible]
7	2079-6374	2079-6374	2	[illegible]

In the context of a database, a record could be something like all the info about a customer; their name, age, address, favorite ice cream flavor, whatever. It's just a way for the system to organize and store nuggets of information about specific entities in a coherent and orderly way. This orderly organizing is what makes your searches faster than a cheetah on caffeine.

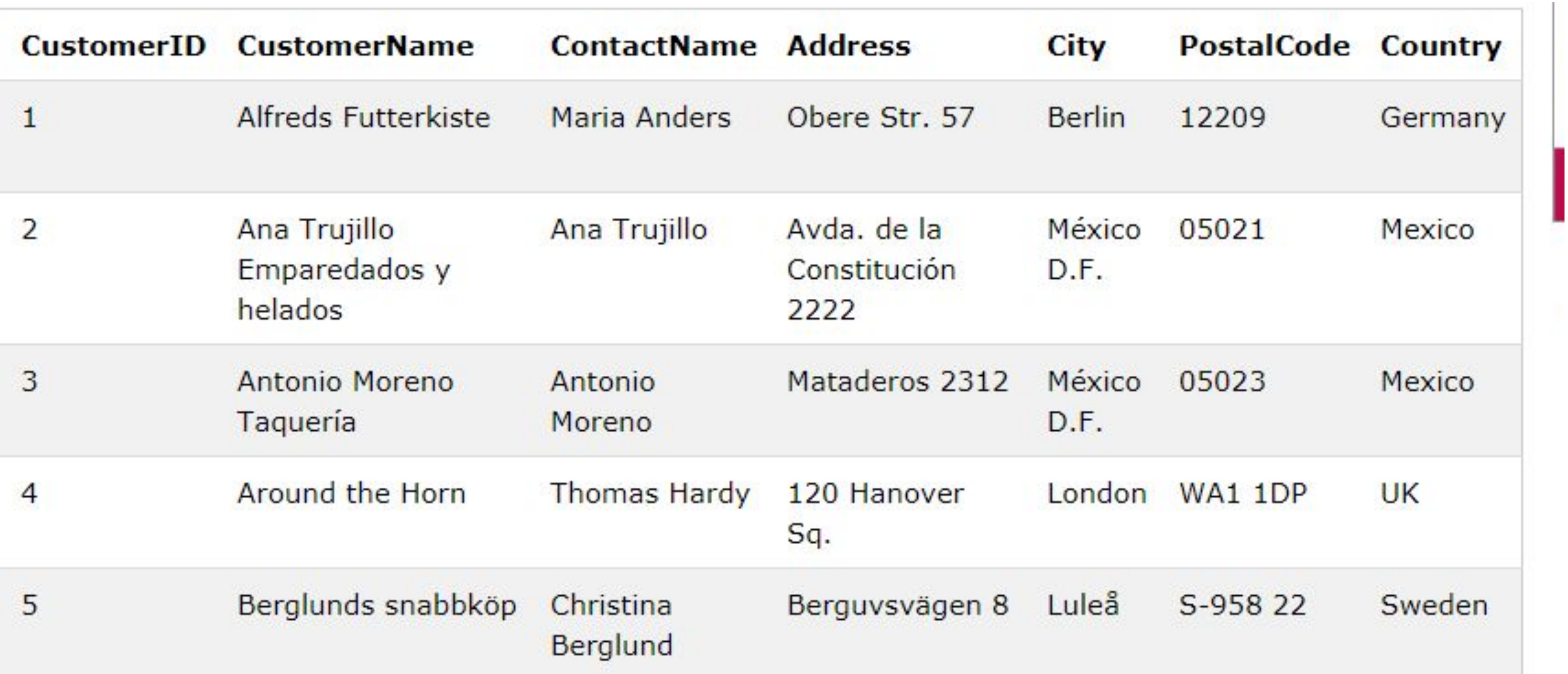
So, if you were to think about a student database, for example, each student would represent a record. Our buddy Jack's record would include things like his student ID, major, course grades, and the number of pizzas he consumes each week. The system's gotta keep track of these things, y'know? Hope this helps.

Examples of Database Programs:

RDBMS is the basis for SQL, and …

- For all modern database systems such as **MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access**.
- A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data.

Sample Table (table name = “Customers”):



CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden

SQL Statements

Most of the actions you need to perform on a database are done with SQL statements.

EXAMPLE: Following SQL statement selects all the records in the "Customers" table:

```
SELECT * FROM Customers;
```

SQL Statements

Keep in Mind That...

- SQL keywords are NOT case sensitive: select is the same as SELECT

  BUT it’s helpful to write SQL keywords in upper-case.

Semicolon after SQL Statements?

A semicolon is the **standard way to separate each SQL statement in database systems that allow more than one SQL statement to be executed in the same call to the server.**

We will use semicolon at the end of each SQL statement.

Some of The Most Important SQL Commands

SELECT - extracts data from a database

UPDATE - updates data in a database

DELETE - deletes data from a database

INSERT INTO - inserts new data into a database

CREATE DATABASE - creates a new database

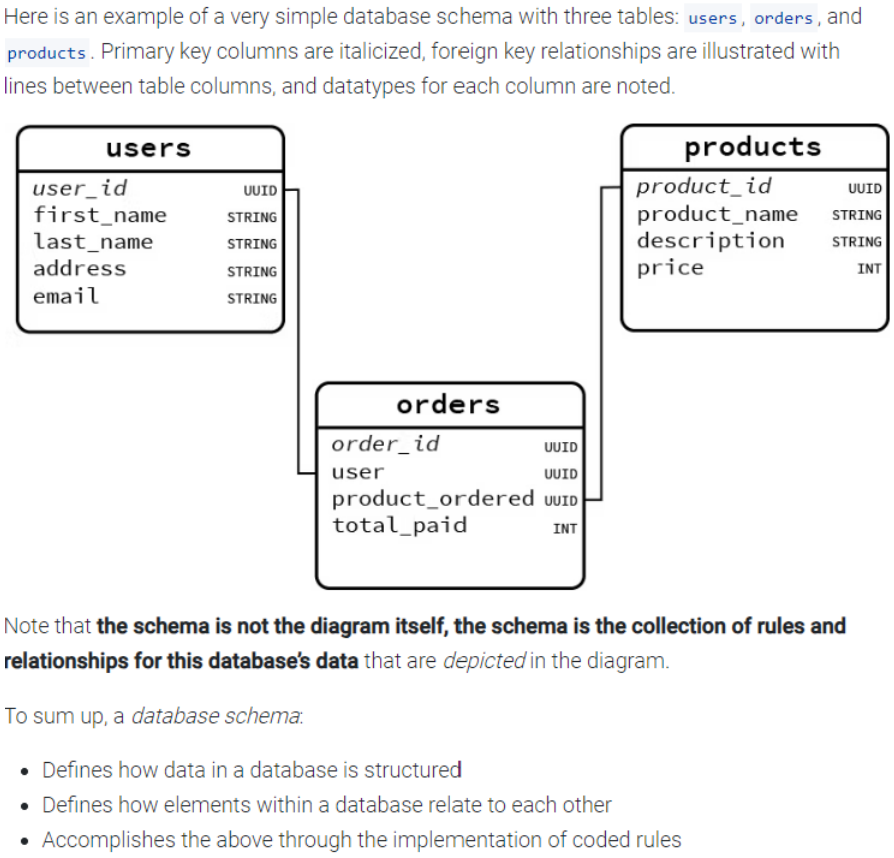
ALTER DATABASE - modifies a database

CREATE TABLE - creates a new table

ALTER TABLE - modifies a table

DROP TABLE - deletes a table

SQL: Defining a Schema



Here is an example of a very simple database schema with three tables: `users`, `orders`, and `products`. Primary key columns are italicized, foreign key relationships are illustrated with lines between table columns, and datatypes for each column are noted.

users	
user_id	UUID
first_name	STRING
last_name	STRING
address	STRING
email	STRING

products	
product_id	UUID
product_name	STRING
description	STRING
price	INT

orders	
order_id	UUID
user	UUID
product_ordered	UUID
total_paid	INT

Note that **the schema is not the diagram itself, the schema is the collection of rules and relationships for this database's data** that are *depicted* in the diagram.

To sum up, a *database schema*:

- Defines how data in a database is structured
- Defines how elements within a database relate to each other
- Accomplishes the above through the implementation of coded rules

Source

The SQL CREATE TABLE Statement

- Used to create a new table in a database.

Syntax:

```
CREATE TABLE table_name (
 column1 datatype,
 column2 datatype,
 column3 datatype
);
```

The SQL CREATE TABLE Statement

Example

```
CREATE TABLE Persons (
  PersonID int,
  LastName varchar(255),
  FirstName varchar(255),
  Address varchar(255),
  City varchar(255)
);
```

Result = New table with columns, but no data

SQL Data Types

Each column in a database table is required to have a name and a data type.

Data types might have different names in different types of databases. Always check the documentation!

MySQL Data Types...

In MySQL there are three main data types: text, number, and date.

Main numeric data types:

INT: Integer (note: INTEGER in sqllite and sql server)

INT(size): Integer where the maximum number of digits may be specified in parenthesis

DOUBLE(size,d): A large number with a floating decimal point. Size is maximum number of digits and d is maximum number of digits to the right of the decimal

Fun fact: Why do some programming languages like R and SQL refer to numerical data types as doubles?

Both data types represent numbers with decimals,
but a float is 32 bits in size while a double is 64 bits.
A double is twice the size of a float — thus the term double.

Create a new table via W3 SQL App:

Navigate here and create “Persons” table:

https://www.w3schools.com/sql/trysql.asp?filename=trysql_create_table

Then Delete code, create a new table called “Classes”.

Should list your classes with variables: Class_name, day, and department.

The INSERT INTO statement is used to insert new records in a table.

- Possible to write the INSERT INTO statement in two ways...

The INSERT INTO statement is used to insert new records in a table.

One way...

```
INSERT INTO table_name (column1, column2,
column3)

VALUES (value1, value2, value3);
```

The INSERT INTO statement is used to insert new records in a table.

If you are adding values for all the columns of the table, you do not need to specify the column names in the SQL query.

However, **make sure the order of the values is in the same order as the columns in the table**

The INSERT INTO statement is used to insert new records in a table.

```
Another way...
```

```
INSERT INTO table_name VALUES (value1,
value2, value3);
```

The INSERT INTO statement is used to insert new records in a table.

Example for “Customers” table:

```
INSERT INTO Customers (CustomerName,
ContactName, Address, City, PostalCode,
Country)
VALUES ('Cardinal', 'Tom B. Erichsen',
'Skagen 21', 'Stavanger', '4006', 'Norway');
```

Create a new table via W3 SQL App:

Navigate here and add new values to “Customers” table:

https://www.w3schools.com/sql/trysql.asp?filename=trysql_insert_colname

Then Delete code and add a new row to “Classes” table.

Add row of data to your variables: Class_name, day, and department

The SELECT statement is used to select data from a database.

Syntax:

```
SELECT column1, column2

FROM table_name;
```

column1 and *column2* are the field names of the table you want to select data from

The SELECT statement is used to select data from a database.

Select all columns using *

```
SELECT * FROM table_name;
```

Example:

```
SELECT CustomerName,City FROM Customers;
```

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_columns

The SELECT DISTINCT statement is used to return only distinct (different) values.

Syntax:

```
SELECT DISTINCT column1, column2, ...

FROM table_name;
```

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_distinct

SQL WHERE Clause

The WHERE clause is used to filter records.

:used to extract only those records that **fulfill a specified condition.**

```
SELECT column1, column2, ...

FROM table_name

WHERE condition;
```

SQL WHERE Clause

The WHERE clause is used to filter records.

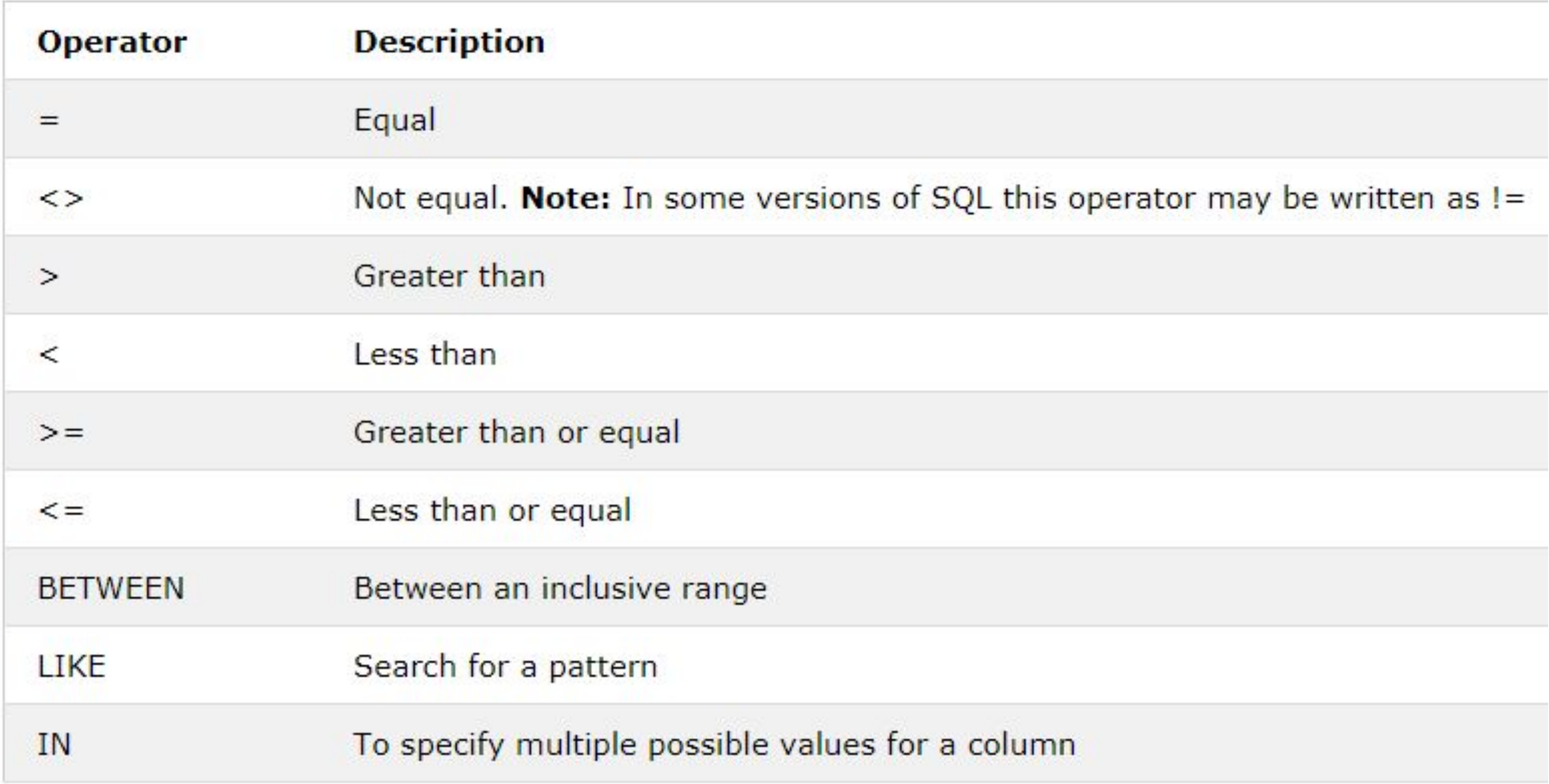
:used to extract only those records that **fulfill a specified condition.**

```
SELECT * FROM Customers

WHERE Country='Mexico';
```

Notice the “=” operator, there are more...

SQL WHERE Clause Operators



Operator	Description
=	Equal
<>	Not equal. **Note:** In some versions of SQL this operator may be written as !=
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern
IN	To specify multiple possible values for a column

Practice with W3 SQL App:

Navigate here and run SQL:

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_where

Then try to return values not equal to “Mexico”

On “OrderDetails” data, find Quantities greater than 10

SQL AND and OR Operators

WHERE clause can be combined with AND and OR operators.

AND and OR operators are used to filter records based on more than one condition:

- *AND operator displays a record if all the conditions separated by AND are TRUE.*
- *The OR operator displays a record if any of the conditions separated by OR are TRUE.*

```
SELECT column1, column2, ...

FROM table_name

WHERE condition1 AND condition2 AND
condition3 ...;
```

```sql
SELECT column1, column2, ...
FROM table_name
WHERE condition1 OR condition2 OR condition3
...;
```

SQL NOT Operator

WHERE clause can ALSO be combined with NOT operator.

```
SELECT column1, column2, ...

FROM table_name

WHERE NOT condition;
```

Practice And, Or, and NOT

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_where_and

Change “And” to “Or” and pay attention to the number of rows in result of the query.

Add a “Not” to the query and pay attention to rows.

Between and Like

BETWEEN syntax:

```
SELECT column_name(s)

FROM table_name

WHERE column_name BETWEEN value1 AND value2;
```

Between and Like

Example:

```
SELECT * FROM Products

WHERE Price BETWEEN 10 AND 20;
```

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_between

Between and Like

LIKE:used in a WHERE clause to search for a specified pattern in a column.

There are two wildcards used in conjunction with the LIKE operator:

% - The percent sign represents zero, one, or multiple characters

_ - The underscore represents a single character

Between and Like

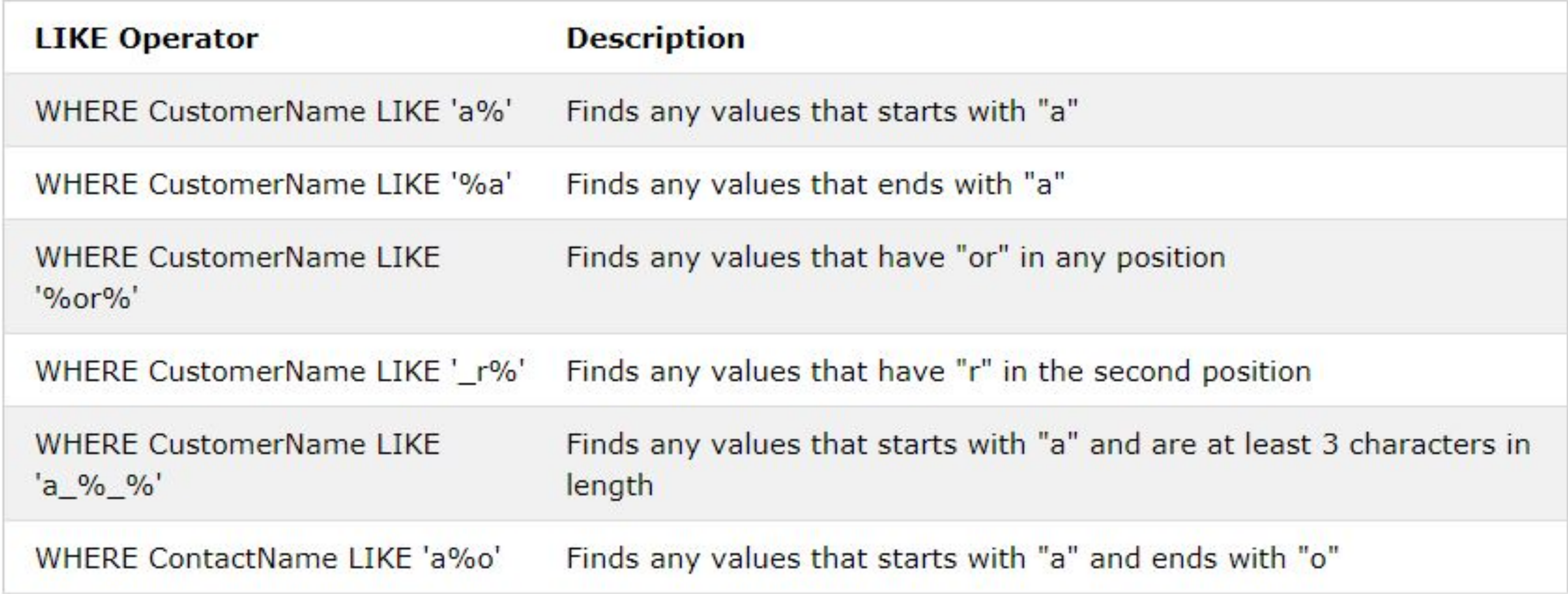
LIKE Syntax:

```
  SELECT column1, column2, ...

  FROM table_name

  WHERE columnN LIKE pattern;
```

Like Operator examples



LIKE Operator	Description
WHERE CustomerName LIKE 'a%'	Finds any values that starts with "a"
WHERE CustomerName LIKE '%a'	Finds any values that ends with "a"
WHERE CustomerName LIKE '%or%'	Finds any values that have "or" in any position
WHERE CustomerName LIKE '_r%'	Finds any values that have "r" in the second position
WHERE CustomerName LIKE 'a_%_%'	Finds any values that starts with "a" and are at least 3 characters in length
WHERE ContactName LIKE 'a%o'	Finds any values that starts with "a" and ends with "o"

Like Operator examples

Try it:

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_like

The ORDER BY keyword

Sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.

```
SELECT column1, column2, ...

FROM table_name

ORDER BY column1, ... DESC;
```

The ORDER BY keyword

Note: you can order by mulitiple columns

```
SELECT * FROM Customers
ORDER BY Country ASC, CustomerName DESC;
```

Note:”ASC” means ascending

Practice:https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_orderby3

SQL Aggregate Functions: COUNT(), AVG() and SUM()

The COUNT() function returns the number of rows that matches a specified criteria.

The AVG() function returns the average value of a numeric column.

The SUM() function returns the total sum of a numeric column.

SQL COUNT(), AVG() and SUM() Functions

```
SELECT COUNT(column_name)

FROM table_name

WHERE condition;
```

SQL COUNT(), AVG() and SUM() Functions

```
SELECT AVG(column_name)

FROM table_name

WHERE condition;
```

SQL COUNT(), AVG() and SUM() Functions

```
SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

SQL Aggregate Functions: COUNT(), AVG() and SUM()

Practice:

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_count

Take a count of ProductIDs, Then calculate the average price, then use the OrderDetails data to sum up all orders.

Also try MAX() and MIN()

GROUP BY Statement

Often used with aggregate functions (COUNT, MAX, MIN, SUM, AVG) to ***group the result-set by one or more columns.***

```
SELECT column_name(s)

FROM table_name

WHERE condition

GROUP BY column_name(s)

ORDER BY column_name(s);
```

GROUP BY Statement

Example:

```
SELECT COUNT(CustomerID), Country

FROM Customers

GROUP BY Country;
```

GROUP BY Statement

Example using “AS” to create new column name:

```
SELECT COUNT(CustomerID) AS ID_Count,
Country

FROM Customers

GROUP BY Country;
```

Practice Group By

Navigate to:

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_groupby

Can you run this query and then rerun it to create a new variable name for the count of customer ids?

SQL HAVING Clause

Added to SQL because the WHERE keyword could not be used with grouped results.

```
SELECT column_name(s)

FROM table_name

GROUP BY column_name(s)

HAVING condition
```

SQL HAVING Clause

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country
HAVING COUNT(CustomerID) > 5;
```

Practice:https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_having

Updating and Deleting Table Data

Be very careful to backup data when you do this!

UPDATE Syntax

```
UPDATE table_name

SET column1 = value1, column2 = value2

WHERE condition;
```

Note: Where condition subsets data to data you want to change

Updating and Deleting Table Data

Example

```
UPDATE Customers
SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
WHERE CustomerID = 1;
```

Try it: https://www.w3schools.com/sql/trysql.asp?filename=trysql_update_2

Updating and Deleting Table Data

The DELETE statement is used to delete existing records in a table.

```sql
DELETE FROM table_name

WHERE condition;
```

ALTER TABLE Statement

used to add, delete, or modify columns in an existing table.

Add column:

```
ALTER TABLE table_name

ADD column_name datatype;
```

ALTER TABLE Statement

used to add, delete, or modify columns in an existing table.

Delete column:

```
ALTER TABLE table_name

DROP COLUMN column_name;
```

ALTER TABLE Statement

used to add, delete, or modify columns in an existing table.

Delete column:

```
ALTER TABLE table_name

DROP COLUMN column_name;
```

ALTER TABLE Statement

used to add, delete, or modify columns in an existing table.

Change column:

```
ALTER TABLE table_name

ALTER COLUMN column_name datatype;
```

Note: These can change depending on database management system (e.g. sometimes “MODIFY”)

Selecting data in one database using values from another: IN statements

Customers is table 1 and suppliers is table 2:

```
SELECT * FROM Customers

WHERE Country IN (SELECT Country FROM
Suppliers);
```

Notes: Subsetting customers by supplier countries. MUST put select statements for data #2 in parentheses.

Adding Primary Key ID variables to data

In MySQL:

```sql
CREATE TABLE Persons (ID int NOT NULL,

LastName varchar(255) NOT NULL,

FirstName varchar(255),

PRIMARY KEY (ID)

);
```

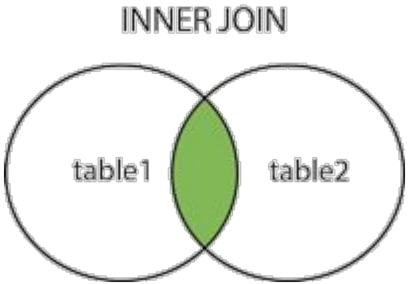
Note: PRIMARY KEY function tells MySQL that ID is unique identifier for data

Table Joins

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

- **(INNER) JOIN**: Returns records that have matching values in both tables
- **LEFT (OUTER) JOIN**: Return all records from the left table, and the matched records from the right table
- **RIGHT (OUTER) JOIN**: Return all records from the right table, and the matched records from the left table
- **FULL (OUTER) JOIN**: Return all records when there is a match in either left or right table

Inner Joins



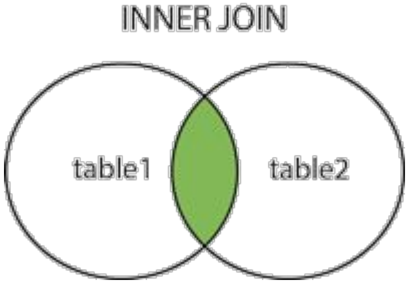
Example

```
SELECT Orders.OrderID,
Customers.CustomerName, Orders.OrderDate
FROM Orders
INNER JOIN Customers ON
Orders.CustomerID=Customers.CustomerID;
```

Note: Data_name . Column_name

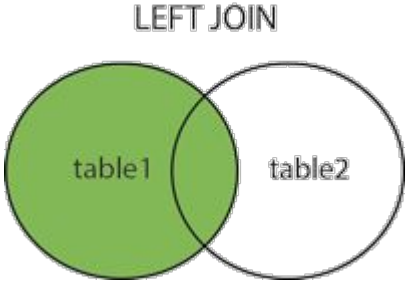
Inner Joins

Example



https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_join

Left Joins



Example

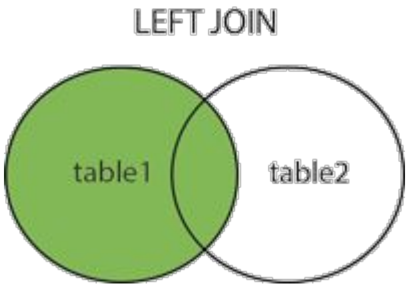
```
SELECT Customers.CustomerName, Orders.OrderID

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID =
Orders.CustomerID

ORDER BY Customers.CustomerName;
```

Left Joins



Example

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_join_left

SQL NULL Values

A field with a NULL value is a field with no value.

Use IS NULL or IS NOT NULL to isolate values

```
SELECT column_names

FROM table_name

WHERE column_name IS NULL;
```

SQL Excercises

1. Take W3Schools.com SQL Quiz to test your knowledge

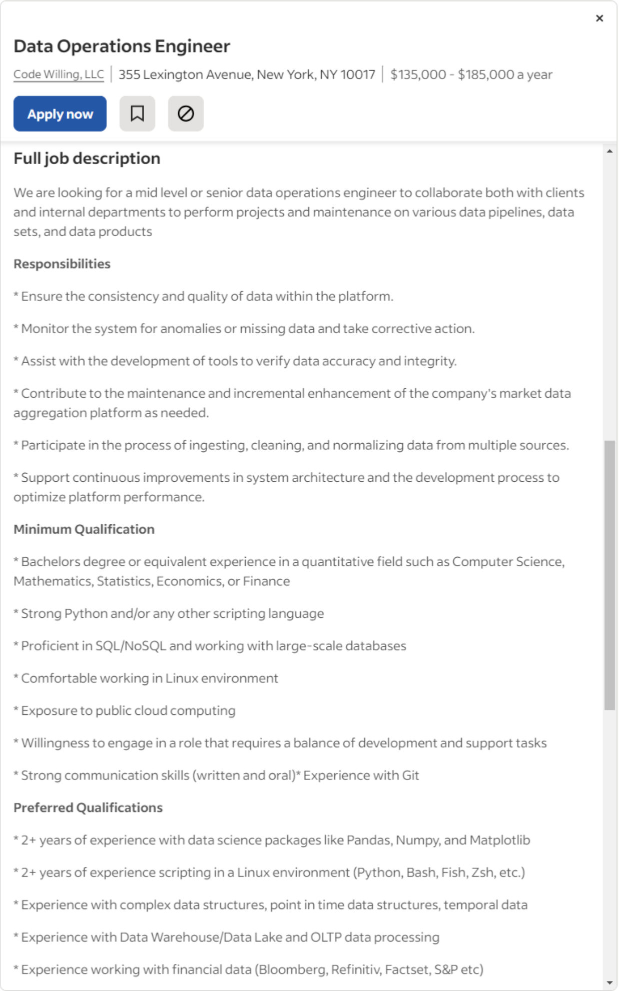
https://www.w3schools.com/quiztest/quiztest.asp?qtest=SQL

2. Answer easy, medium, and hard questions from SQL ZOO related to similar data

https://sqlzoo.net/wiki/AdventureWorks

Note: sqlzoo likes single quotes, not double quotes

Examples of jobs requiring SQL



Data Operations Engineer

Code Willing, LLC | 355 Lexington Avenue, New York, NY 10017 | $135,000 - $185,000 a year

Full job description

We are looking for a mid level or senior data operations engineer to collaborate both with clients and internal departments to perform projects and maintenance on various data pipelines, data sets, and data products

Responsibilities

* Ensure the consistency and quality of data within the platform.

* Monitor the system for anomalies or missing data and take corrective action.

* Assist with the development of tools to verify data accuracy and integrity.

* Contribute to the maintenance and incremental enhancement of the company's market data aggregation platform as needed.

* Participate in the process of ingesting, cleaning, and normalizing data from multiple sources.

* Support continuous improvements in system architecture and the development process to optimize platform performance.

Minimum Qualification

* Bachelors degree or equivalent experience in a quantitative field such as Computer Science, Mathematics, Statistics, Economics, or Finance

* Strong Python and/or any other scripting language

* Proficient in SQL/NoSQL and working with large-scale databases

* Comfortable working in Linux environment

* Exposure to public cloud computing

* Willingness to engage in a role that requires a balance of development and support tasks

* Strong communication skills (written and oral)* Experience with Git

Preferred Qualifications

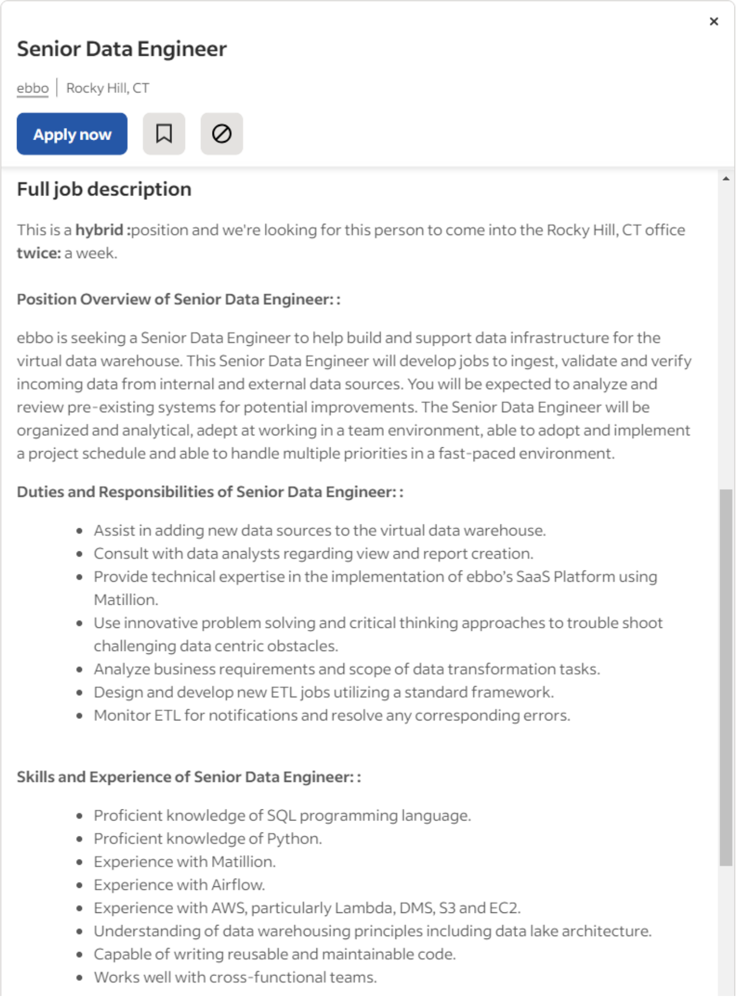
* 2+ years of experience with data science packages like Pandas, Numpy, and Matplotlib

* 2+ years of experience scripting in a Linux environment (Python, Bash, Fish, Zsh, etc.)

* Experience with complex data structures, point in time data structures, temporal data

* Experience with Data Warehouse/Data Lake and OLTP data processing

* Experience working with financial data (Bloomberg, Refinitiv, Factset, S&P etc)



Senior Data Engineer

ebbo | Rocky Hill, CT

Full job description

This is a **hybrid :**position and we're looking for this person to come into the Rocky Hill, CT office **twice:** a week.

Position Overview of Senior Data Engineer: :

ebbo is seeking a Senior Data Engineer to help build and support data infrastructure for the virtual data warehouse. This Senior Data Engineer will develop jobs to ingest, validate and verify incoming data from internal and external data sources. You will be expected to analyze and review pre-existing systems for potential improvements. The Senior Data Engineer will be organized and analytical, adept at working in a team environment, able to adopt and implement a project schedule and able to handle multiple priorities in a fast-paced environment.

Duties and Responsibilities of Senior Data Engineer: :

- Assist in adding new data sources to the virtual data warehouse.
- Consult with data analysts regarding view and report creation.
- Provide technical expertise in the implementation of ebbo's SaaS Platform using Matillion.
- Use innovative problem solving and critical thinking approaches to trouble shoot challenging data centric obstacles.
- Analyze business requirements and scope of data transformation tasks.
- Design and develop new ETL jobs utilizing a standard framework.
- Monitor ETL for notifications and resolve any corresponding errors.

Skills and Experience of Senior Data Engineer: :

- Proficient knowledge of SQL programming language.
- Proficient knowledge of Python.
- Experience with Matillion.
- Experience with Airflow.
- Experience with AWS, particularly Lambda, DMS, S3 and EC2.
- Understanding of data warehousing principles including data lake architecture.
- Capable of writing reusable and maintainable code.
- Works well with cross-functional teams.