MySQL

**DB** is any collection of related data like a shopping list, your friends…etc. And can be stored in different ways [ on paper, in your mind, on your computer “**the most common use case**”]

Computers are great at keeping track of large amount of information

زي Amazon.com بيحتاج متابعة البيانات بشكل مستمر والتغييرات اللي بتحصل وتسجيل العملاء والأرباح والطلبات وملايين البيانات اللي بتضاف كل فترة والداتا اللي بتخزنها امازون valuable, critical secured and

**So** we use DBMS (Database management system): a special SW program that create and maintain a database making it easy to manage amounts of information

**CRUD** → the four main operations in DB

*There are two types of DB.*

* Rational DB (SQL)

Organize data into one or more tables each one has columns and rows, and a unique key identifies each row

* Non-rational (NoSQL/ not just SQL)

Organize data in anything but a traditional table [graphs, documents, key value hash,..]

Structure Query Language (SQL)*: standardized language for interacting with RDBMS, to perform CRUD****,*** *define tables and structures.*

DB Queries

Requests made to the data management system for specific information.

كلما كانت DB معقدة وكبيرة كلما زادت صعوبة الوصول لجزء معين من المعلومات اللي احنا عاوزينها

A google search is a query.

### A query is a set of instruction given to the RDBMS that tell what info you want it to retrieve for you.

**QUERY** is a block of SQLdesigned to ask DBMS for particular piece of information.

Text

Description automatically generated with medium confidence

*Tables and keys*

زي ما قولنا فوق الـ table عبارة عن صفوف وأعمدة كل عمود بيعبر عن single attribute وكل صف بيعبر عن entry or a value of attribute

**Primary key**: uniquely identify the specific row and must contain UNIQUE values {num, string, any type}, and cannot contain NULL values.

فيه نوعين من الـ primary key

* Surrogate key also called a synthetic primary key, is generated when a new record is inserted into a table automatically by a database والنوع دا ملوش وجود في العالم الحقيقي وملوش دلالة على عكس النوع الثاني زي مثلا ترتيب ا و ترقيم
* Natural key is a type of unique key in a database formed of attributes that exist and are used in the external world outside the database. ex. SSN, ID

**Foreign key**: an attribute that links another database table and refers to the primary key of another table.

دا بيعرفنا ايه الـ relationships اللي بين الجداول

**Composite key**:is a combination of two or more columns in a table that can be used to uniquely identify each row in the table.

 زي first\_name and last\_name واستخدام تاني لما نضيف اتنين foreign keys مع بعض في جدول معين بيكونوا primary key للجدول دا

*SQL basics*

SQL is a hybrid language it's basically 4 types of languages  in one

1. **Data query language (DQL)**

Used to query the DB for info and get the info that already stored there

1. **Data definition language(DDL)**

Used for defining the DB schema (the overall layout of DB)

1. **Data control language (DCL)**

Used for controlling access to the data and user & permissions management

1. **Data manipulation language (DML)**

Used for inserting ,updating,deleting from the DB

*Data types*

**INT(6)** →whole numbers in 6 digits

**DECIMAL** (m,n)→ decimal numbers like 3.2

**VARCHAR (2)**→string of text in 2 characters

**BLOB** → Binary large objects ,stores large data

**DATE** → any date YYYY-MM-DD

**TIMESTAMP** →just like DATE with time in hours and minutes used for recording

*Creating tables*

To create table we write **CREATE TABLE + NAME\_TABLE**

To write their attributes inside () we write **name\_attr + datatype + if primary key +null /not null/unique**

To show this table we write **DESCRIBE + NAME\_TABLE ;**

To delete the table we write **DROP  TABLE + NAME\_TABLE;**

To modify table we use **ALTER TABLE + NAME\_TABLE + OPERATION (ADD)+ name\_attr ;**

**Inserting Data**

* To add values inside columns we use **INSERT INTO + Name\_TABLE  + VALUES ( , , ) ;** Inside parentheses we write the value in the order of attributes
* To grab all information in the table we use **SELECT \* FROM NAME\_TABLE ;**

**Update & delete Data**

* To change a value, we write

**UPDATE NAME\_TABLE**

**SET  name-attr = new\_value**

**WHERE name\_attr = old\_value ;**

* To delete a value we write

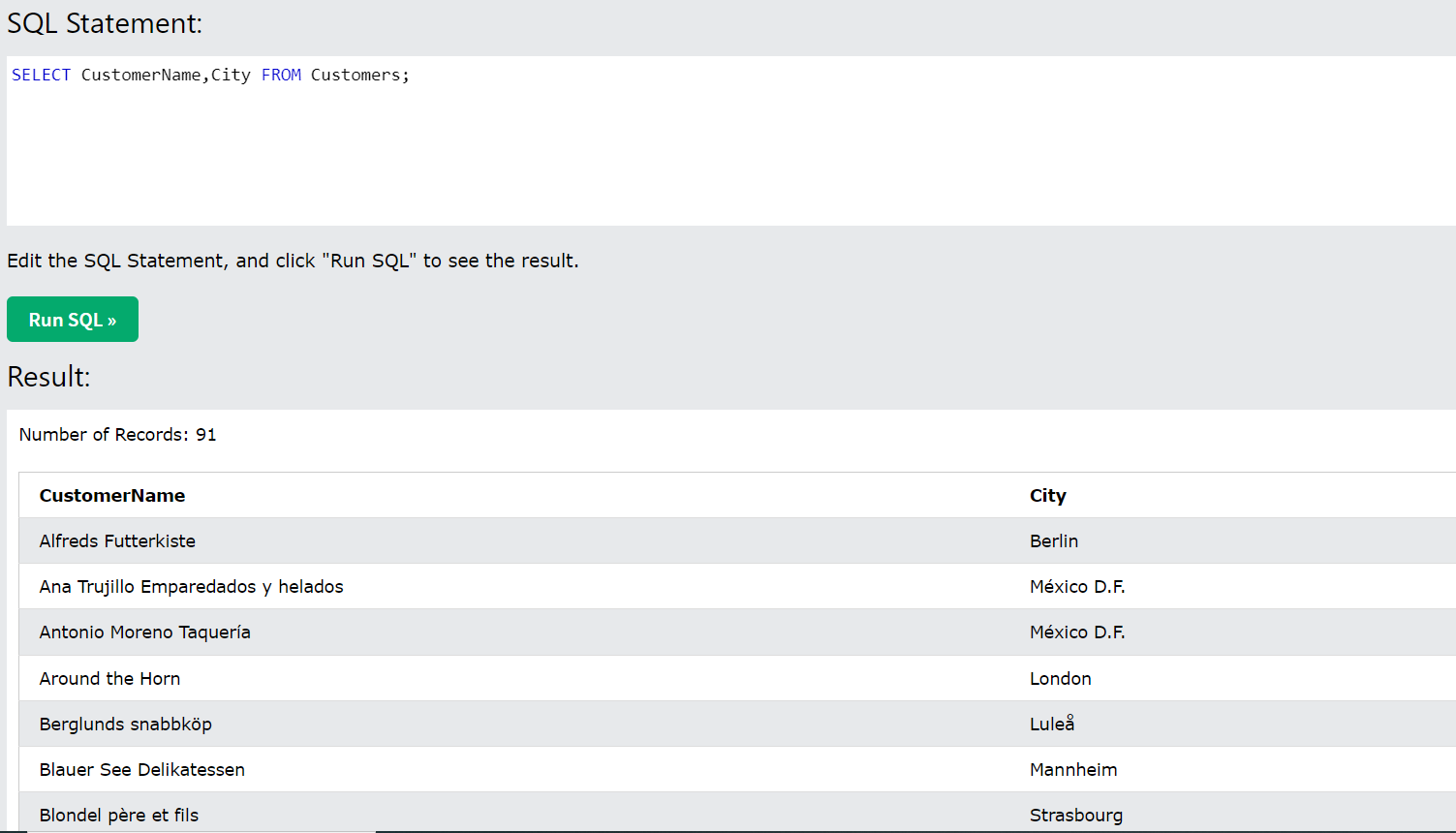
**DELETE FROM NAME\_TABLE WHERE name\_attr = value ;**

**Basic Queries**

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

The data returned is stored in a result table, called the result-set.

**SELECT CustomerName,City FROM Customers;**

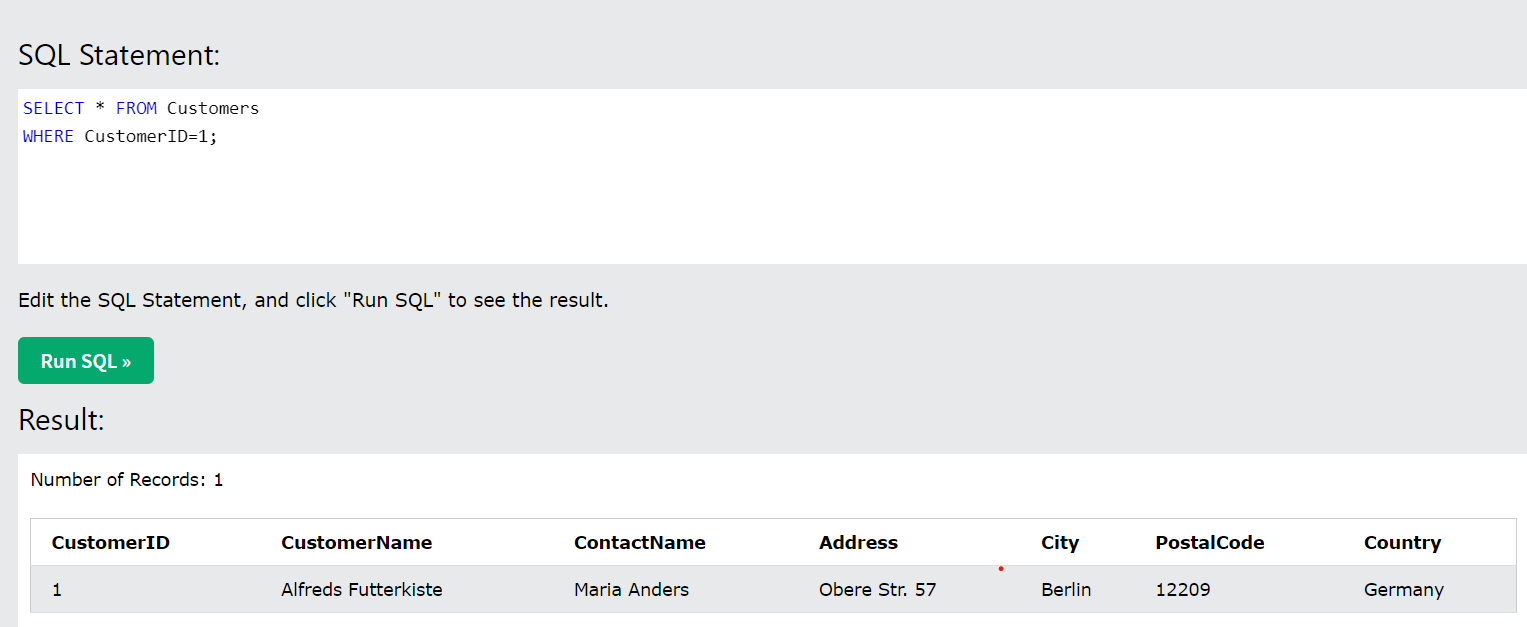


The **WHERE** clause is used to filter records.

It is used to extract only those records that fulfill a specified condition.

**SELECT \* FROM Customers**

**WHERE CustomerID=1;**

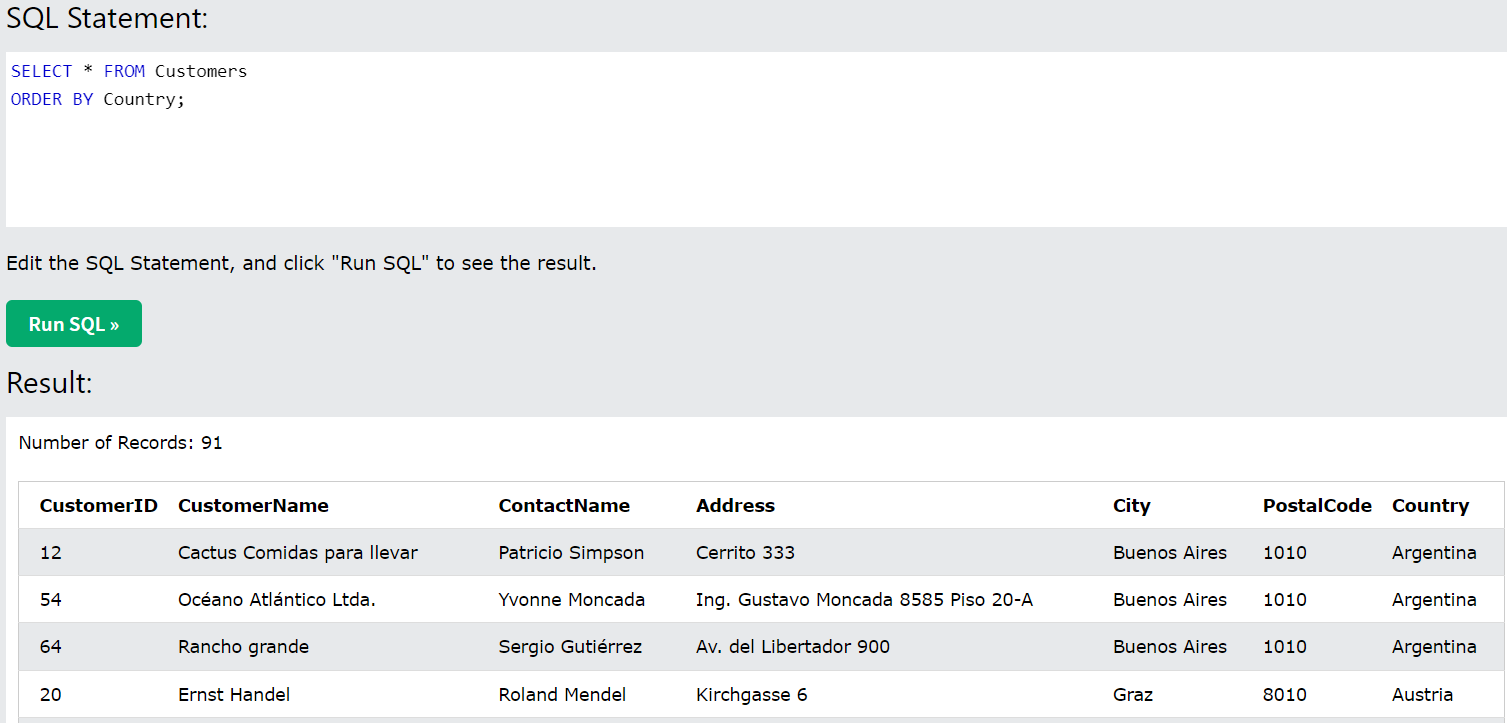


The **ORDER BY** keyword is used to sort the result-set in ascending or descending order.

The **ORDER BY** keyword sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.

**SELECT \* FROM Customers**

**ORDER BY Country;**



The **IN** operator allows you to specify multiple values in a WHERE clause.

The **IN** operator is a shorthand for multiple **OR** conditions.

**SELECT \* FROM Customers**

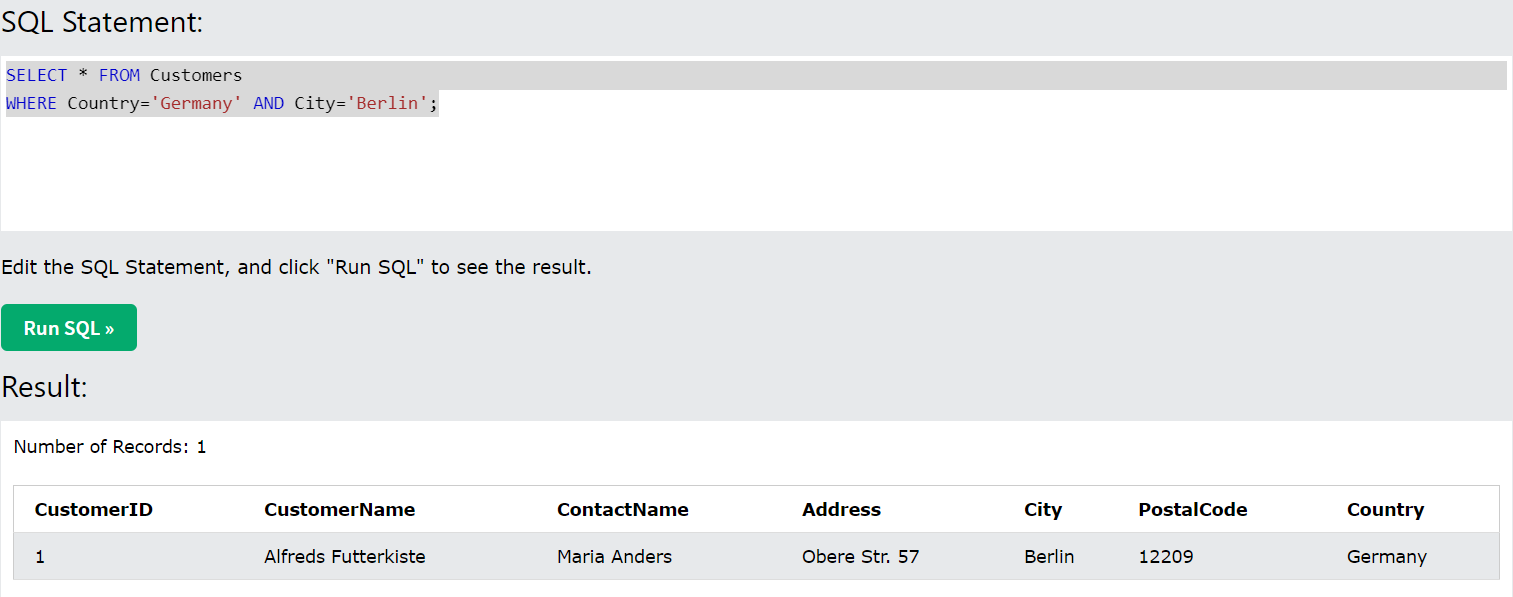
**WHERE Country IN ('Germany', 'France', 'UK');**

### 

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

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

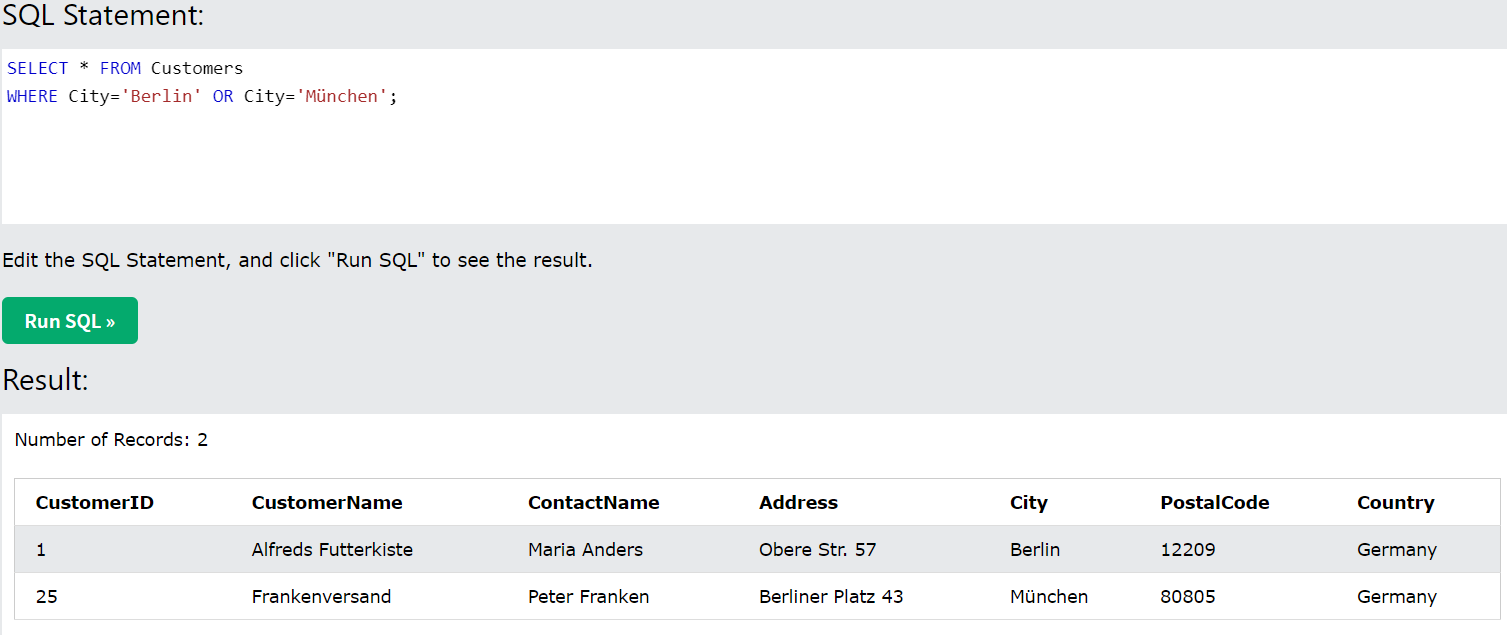
**SELECT \* FROM Customers**

**WHERE Country='Germany' AND City='Berlin';**

* The **OR** operator displays a record if any of the conditions separated by **OR** is TRUE.

**SELECT \* FROM Customers**

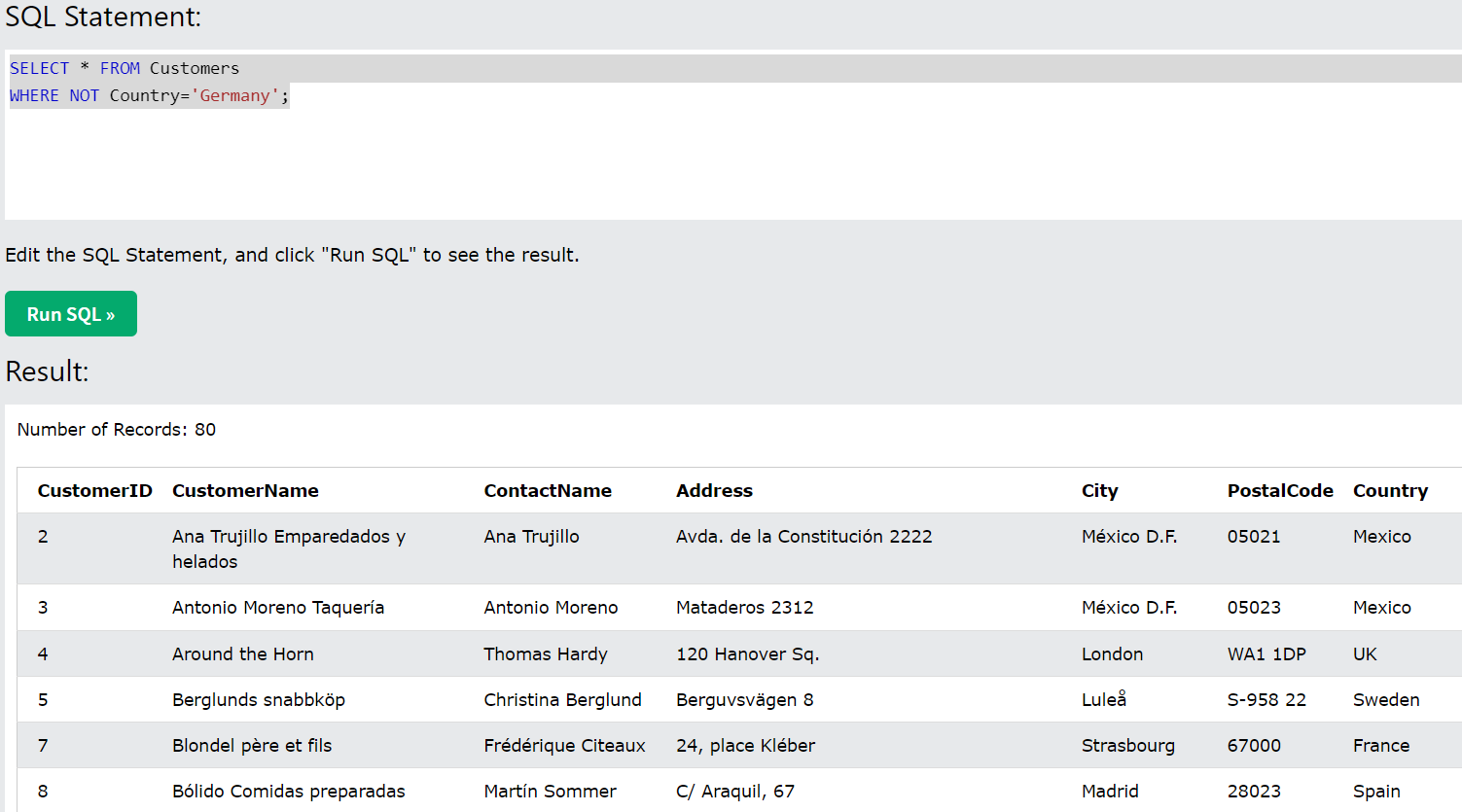
**WHERE City='Berlin' OR City='München';**

****

The **NOT** operator displays a record if the condition(s) is NOT TRUE.

**SELECT \* FROM Customers**

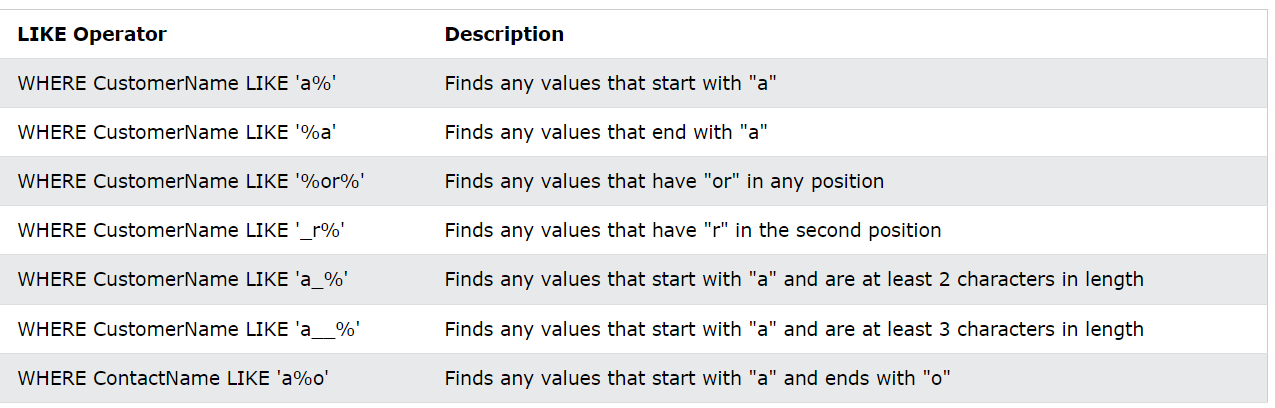
**WHERE NOT Country='Germany';**



The **LIKE** operator is used in a WHERE clause to search for a specified pattern in a column.

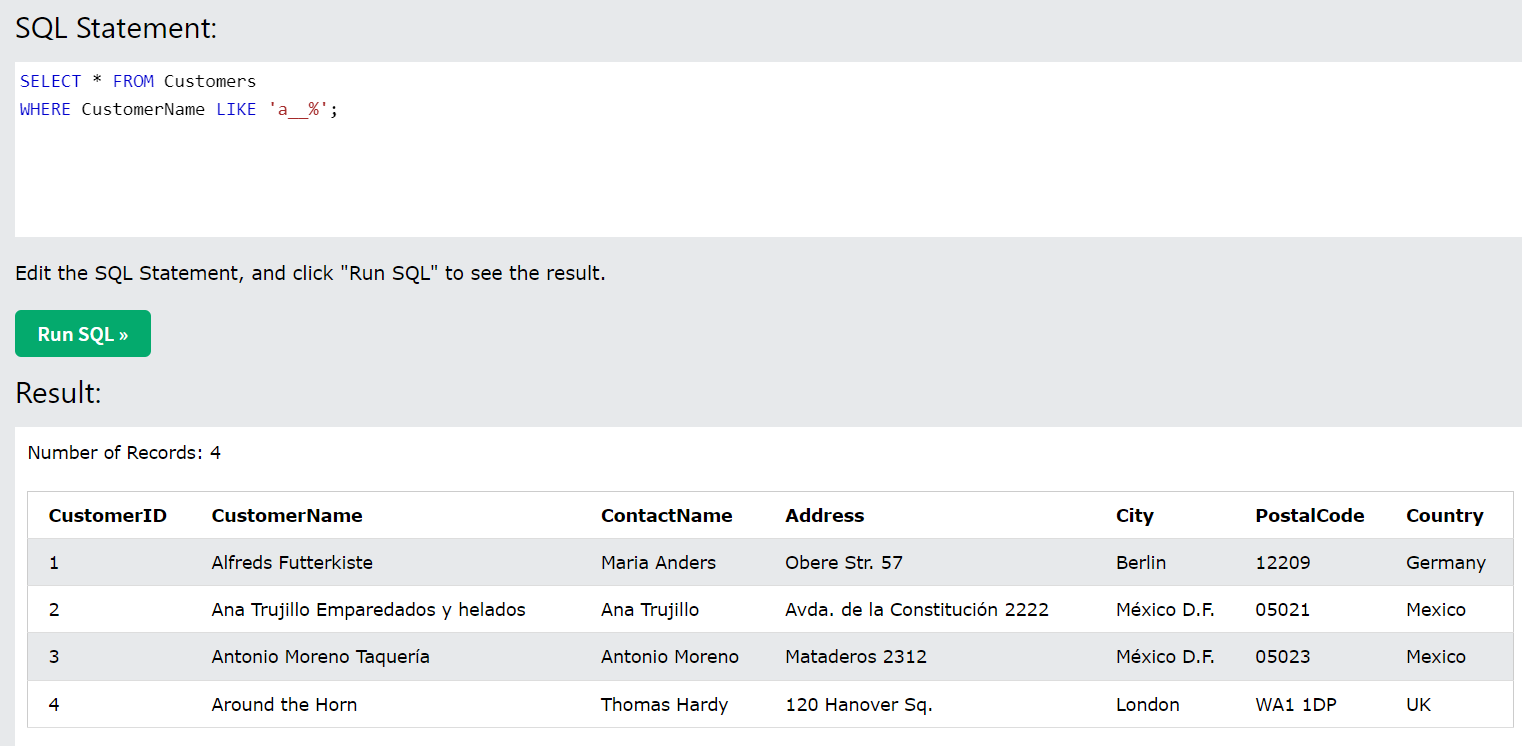
There are two wildcards often used in conjunction with the **LIKE** operator:

* The percent sign (%) represents zero, one, or multiple characters
* The underscore sign (\_) represents one, single character



**SELECT \* FROM Customers**

**WHERE CustomerName LIKE 'a\_\_%';**

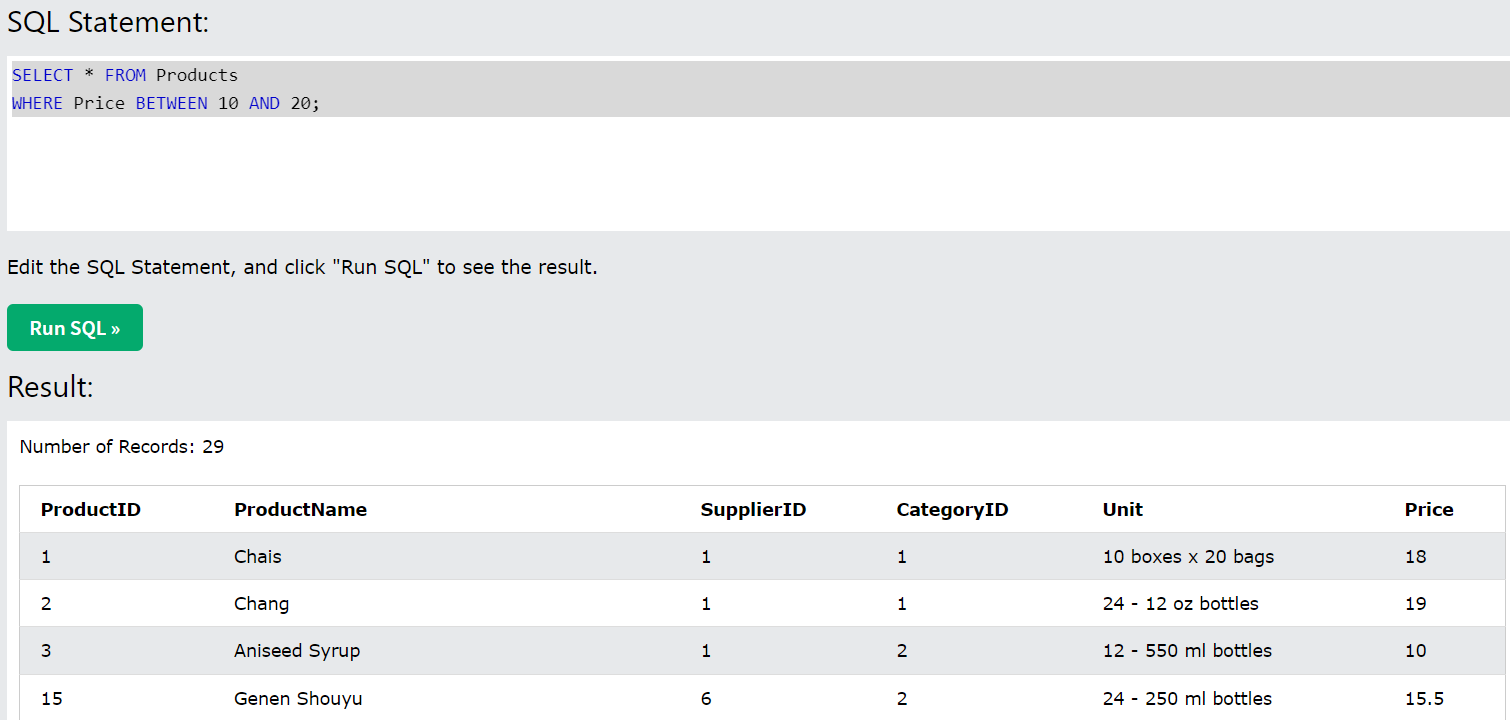


The **BETWEEN** operator selects values within a given range. The values can be numbers, text, or dates.

The **BETWEEN** operator is inclusive: begin and end values are included.

**SELECT \* FROM Products**

**WHERE Price BETWEEN 10 AND 20;**



**ON DELETE CASCADE**

Use the ON DELETE CASCADE option to specify whether you want rows deleted in a child table when corresponding rows are deleted in the parent table. If you do not specify cascading deletes, the default behavior of the database server prevents you from deleting data in a table if other tables reference it.

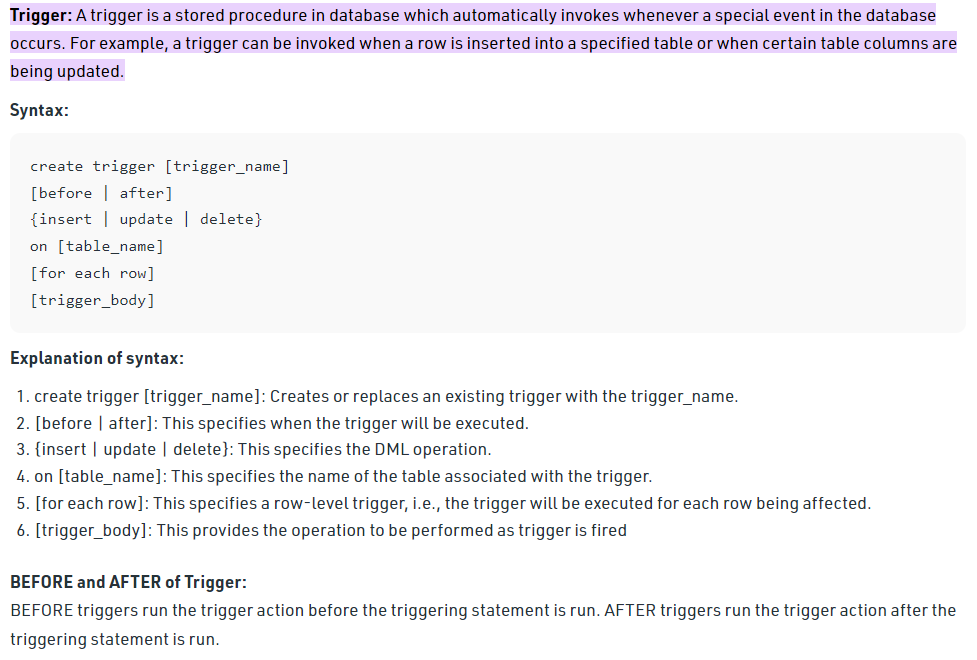
If you specify this option, later when you delete a row in the parent table, the database server also deletes any rows associated with that row (foreign keys) in a child table. The principal advantage to the cascading-deletes feature is that it allows you to reduce the quantity of SQL statements you need to perform delete actions.

**On Delete Set Null**

On Delete Set Null clause sets all the records of the column which is defined as a foreign key in the child table to Null if the corresponding record in the parent table is deleted.

**Triggers**

Trigger is a block of SQL code which we can write and ‘ll define a certain action that should happen when certain operation gets performed on DB .it’s extremely powerful and useful



Because a trigger resides in the database and anyone who has the required privilege can use it, a trigger lets you write a set of SQL statements that multiple applications can use. It lets you avoid redundant code when multiple programs need to perform the same database operation.

**Database schema**

All the different tables and the different attributes on those tables and have a different relationship.

**Entity هو اي شيء ف الواقع مستقل ذاته اقدر اعبر عنه Attributes سواء كان شيء ملموس او غير ملموس و يربط entities دي بحاجة اسمها relations**

**ERD (Entity Relationship Diagram)**

بستخدمه عشان أقدر احول الداتا لديزين وشكل مفهوم أقدر اطلع منه **relations** مع بعض

## قبل ما اطلع الـ entities والـ relations للموديل اللي عندي لازم أفكر الاول ف كام نقطة

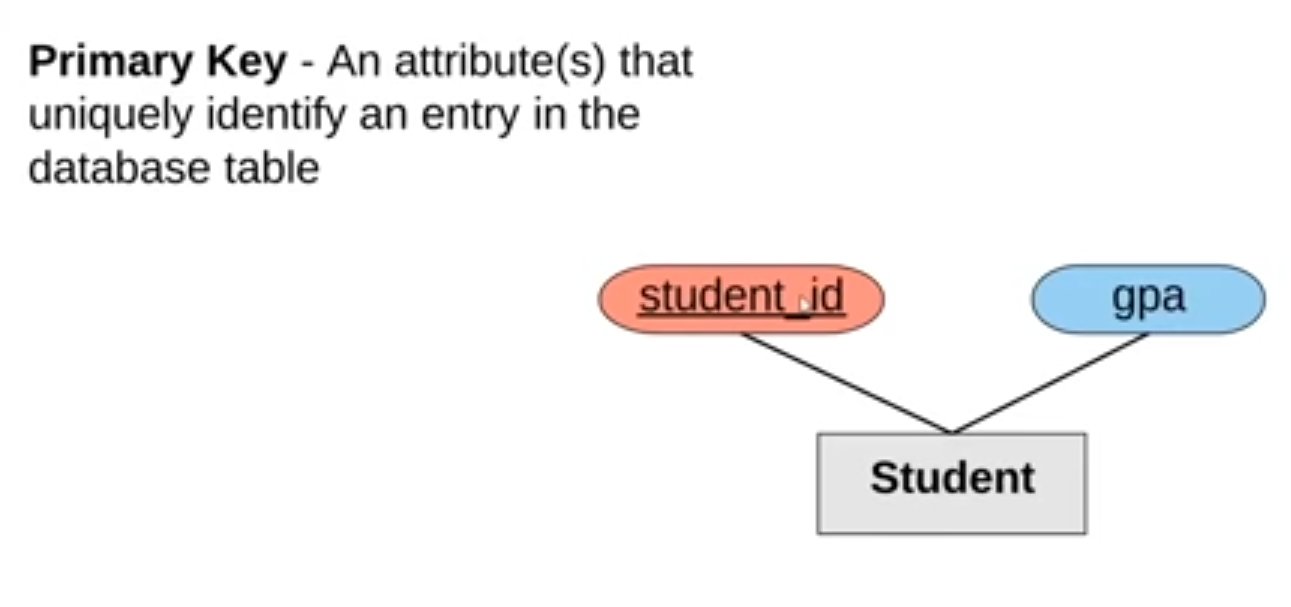
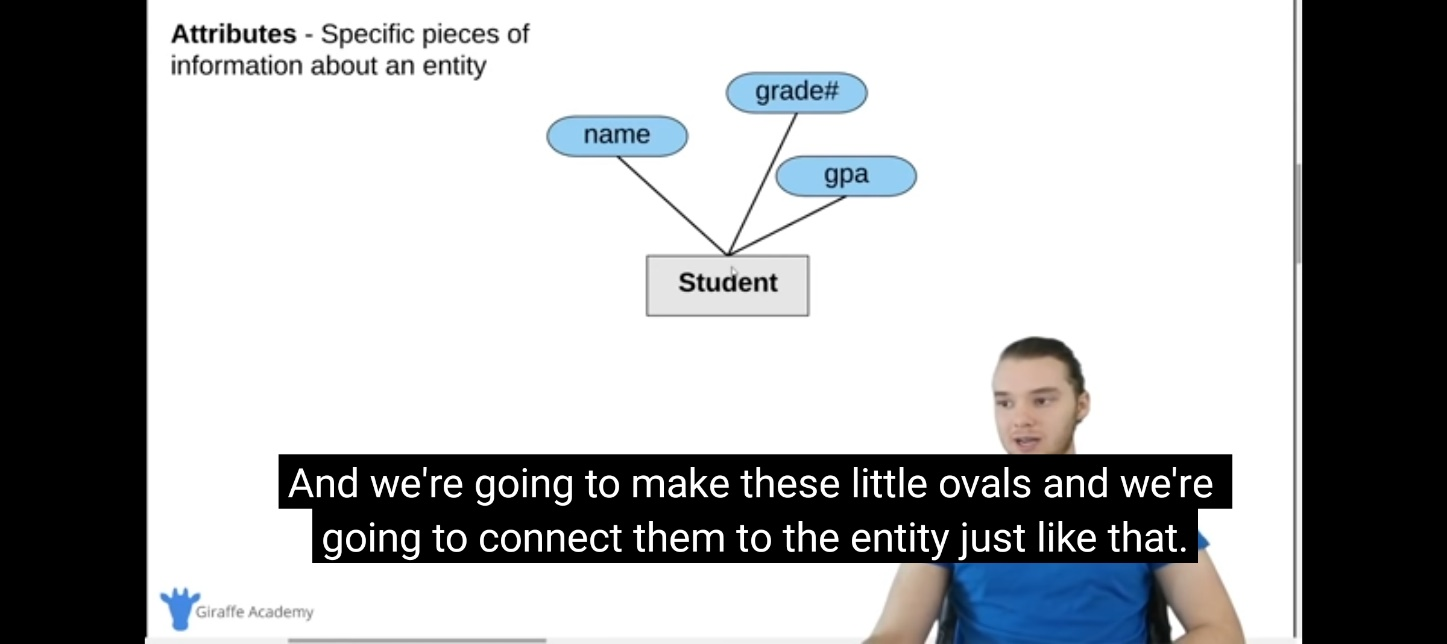
## ايه هي ال entities اللي بتوصف الموديل دا من خلال الـ characteristics اللي عطيهالي

## ايه الـ attributesاللي بتوصف الانتيتي دي

## ايه الـ attributes

## هل فيه attributes مميزة مثلا مبتتكررش اميز بيها

بعدين اشوف العلاقات ما بين ال entities وبعضها

  
  
  
****

Diagram

Description automatically generated

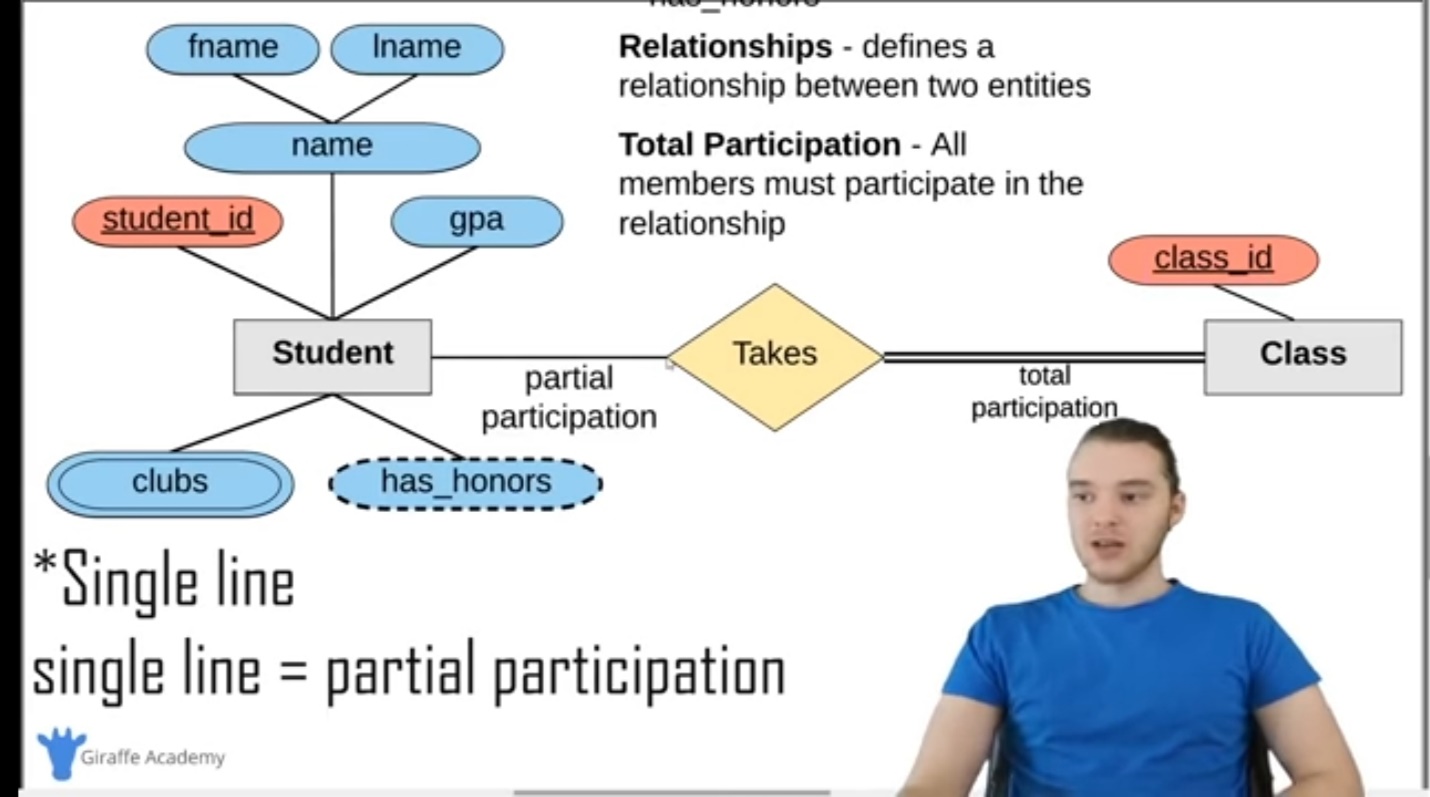
Diagram

Description automatically generated

Diagram

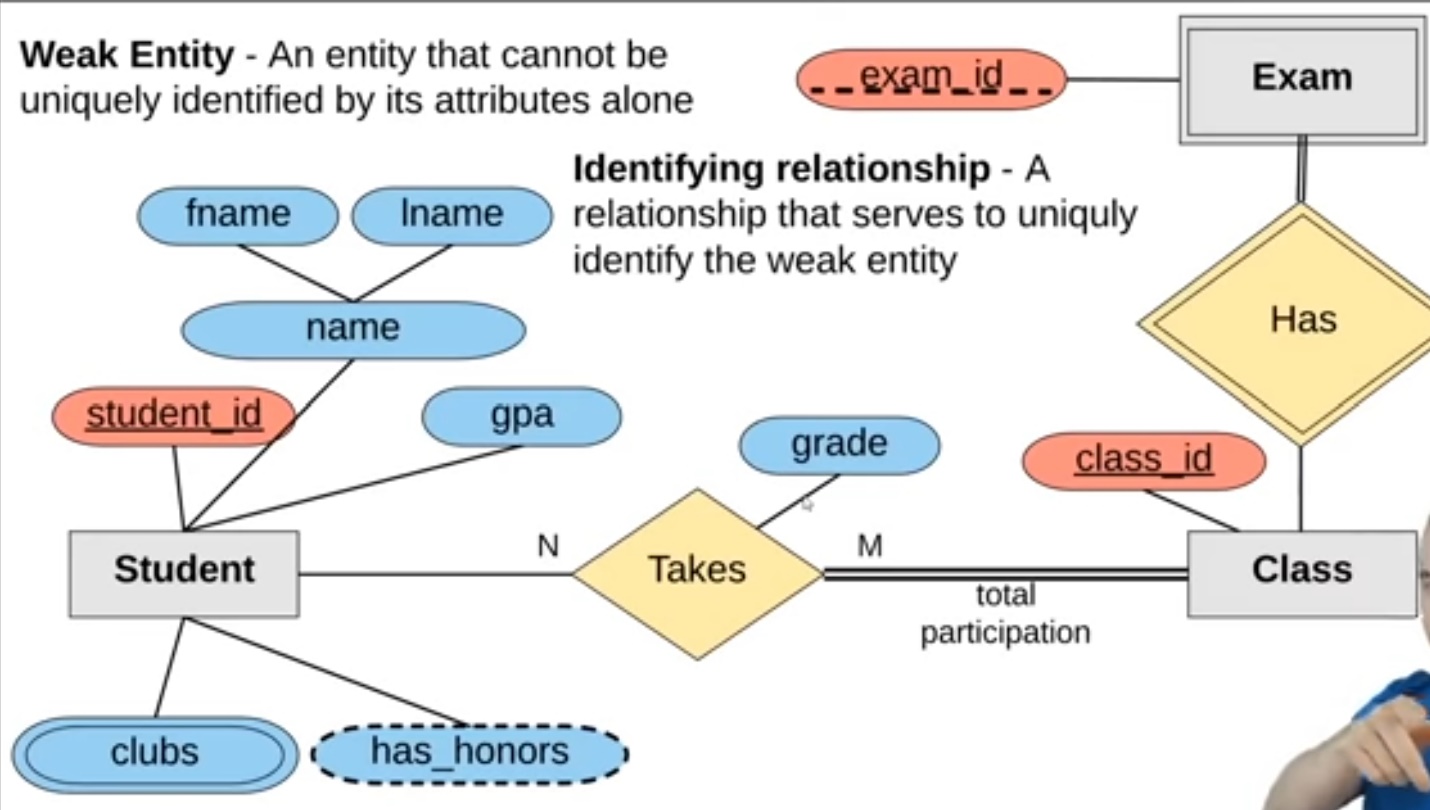
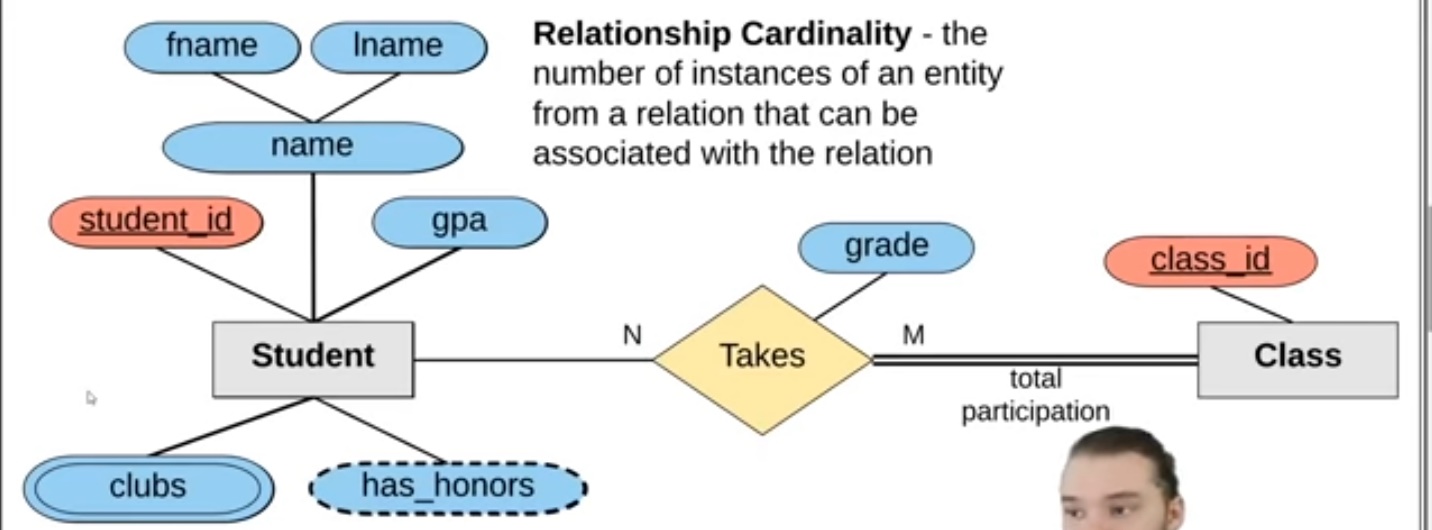
Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated



# REFERENCE

* [SQL Tutorial](https://www.w3schools.com/sql/default.asp)

* [SQL Tutorial - Full Database Course for Beginners](https://youtu.be/HXV3zeQKqGY)