Bases de datos

¿Qué es una Base de Datos?

"Conjunto exhaustivo **no redundante** de datos **estructurados** organizados independientemente de su utilización e implementación en máquina, accesibles en tiempo real y compartibles por **usuarios concurrentes** que tienen necesidad de información diferente y no predecible en el tiempo"

Ciclo del dato

1

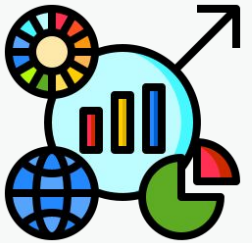
Colectan datos de ventas

2

Consultan los analistas

3

Se agrega la información



4

Se utiliza par la toma de decisiones

5

Decisiones que afectan al cliente final

6

Que repercute en las ventas de nuevo

Tipos de bases de datos

Relacionales (SQL)

- Colección de elementos de datos organizados en un conjunto de tablas formalmente descritas
- Se basan en la organización de la información en partes pequeñas que se integran mediante identificadores o claves(keys).
- Tienen mayor capacidad de almacenamiento
- Son menos vulnerables ante fallas.

No relacionales (NoSQL)

- Diseñados para modelos de datos específicos
- Tienen esquemas flexibles
- No tienen un identificador que sirva para relacionar dos o más conjuntos de datos
- Son fáciles de desarrollar, tanto en funcionalidad como en rendimiento a escala.

Tipos de bases de datos

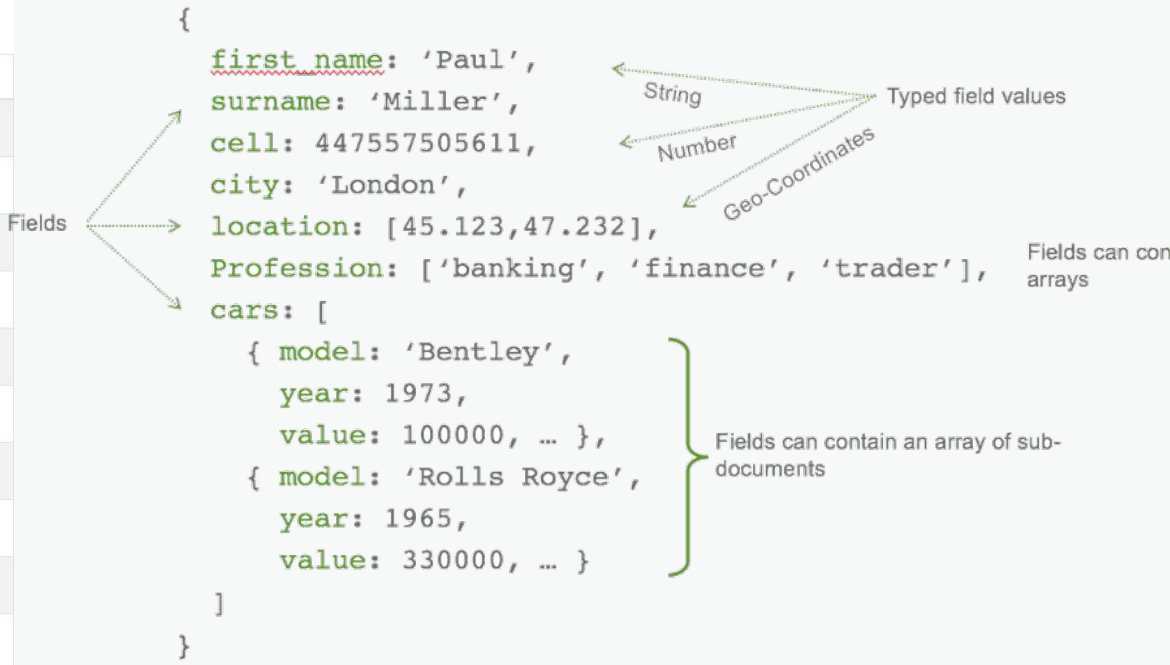
Relacionales (SQL)

v-Datatable example

	First Name	Last Name	Hometown	Data of Birght	Created	Updated	
☐	Lucca	Lin	Melbourne	13/02/1975	23/04/2019	23/04/2019	Edit
☐	Zahid	Werner	Sydney	18/09/1979	23/04/2019	23/04/2019	Edit
☐	Gabriel	Griffiths	Chicago	25/11/1984	23/04/2019	23/04/2019	Edit
☐	Talha	Tucker	Berlim	27/01/1999	23/04/2019	23/04/2019	Edit
☐	Aariz	Piper	Auckland	11/07/1964	23/04/2019	23/04/2019	Edit
☐	Macsen	Schultz	Rio de Janeiro	01/10/1987	23/04/2019	23/04/2019	Edit
☐	Sebastian	Cervantes	Brisbane	13/11/1994	23/04/2019	23/04/2019	Edit
☐	Tayyab	Lister	Perth	14/12/1997	23/04/2019	23/04/2019	Edit
☐	Anum	Warren	Manaus	17/02/1951	23/04/2019	23/04/2019	Edit
☐	Areeba	Stein	Rome	18/03/1954	23/04/2019	23/04/2019	Edit

Items per page 10 << < 1 2 > >>

No relacionales (NoSQL)



Tipos de bases de datos

Relacionales (SQL)

Oracle SQL Developer

Microsoft SQL Server

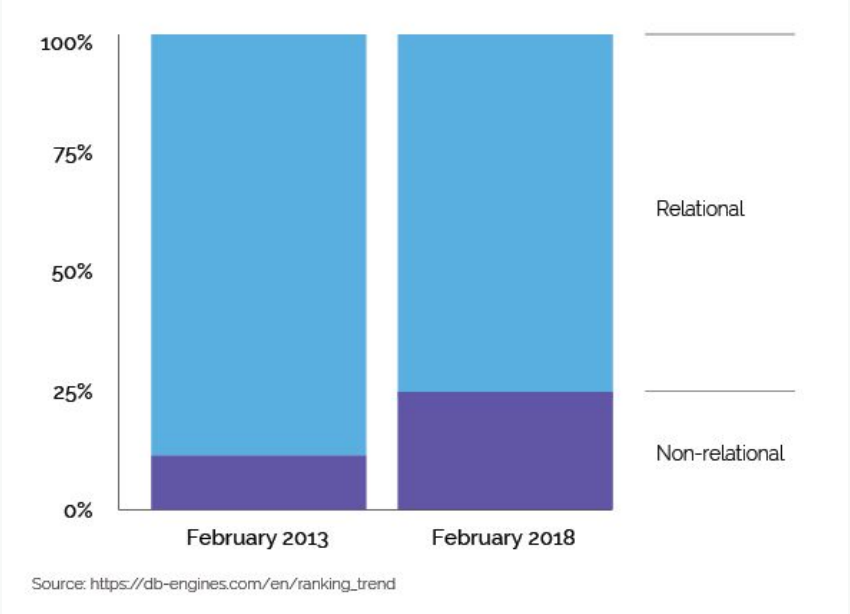
PostgreSQL

No relacionales (NoSQL)

mongoDB

cassandra

redis

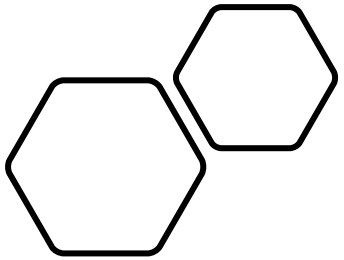


¿Qué es SQL?

"SQL (Structured Query Language) es un lenguaje de dominio específico utilizado en programación, diseñado para **administrar, y recuperar información** de sistemas de gestión de bases de datos relacionales. Una de sus principales características es el cálculo relacional para efectuar consultas con el fin de recuperar, de forma sencilla, información de bases de datos, así como realizar cambios en ellas."

Wikipedia

Características de una BD relacional



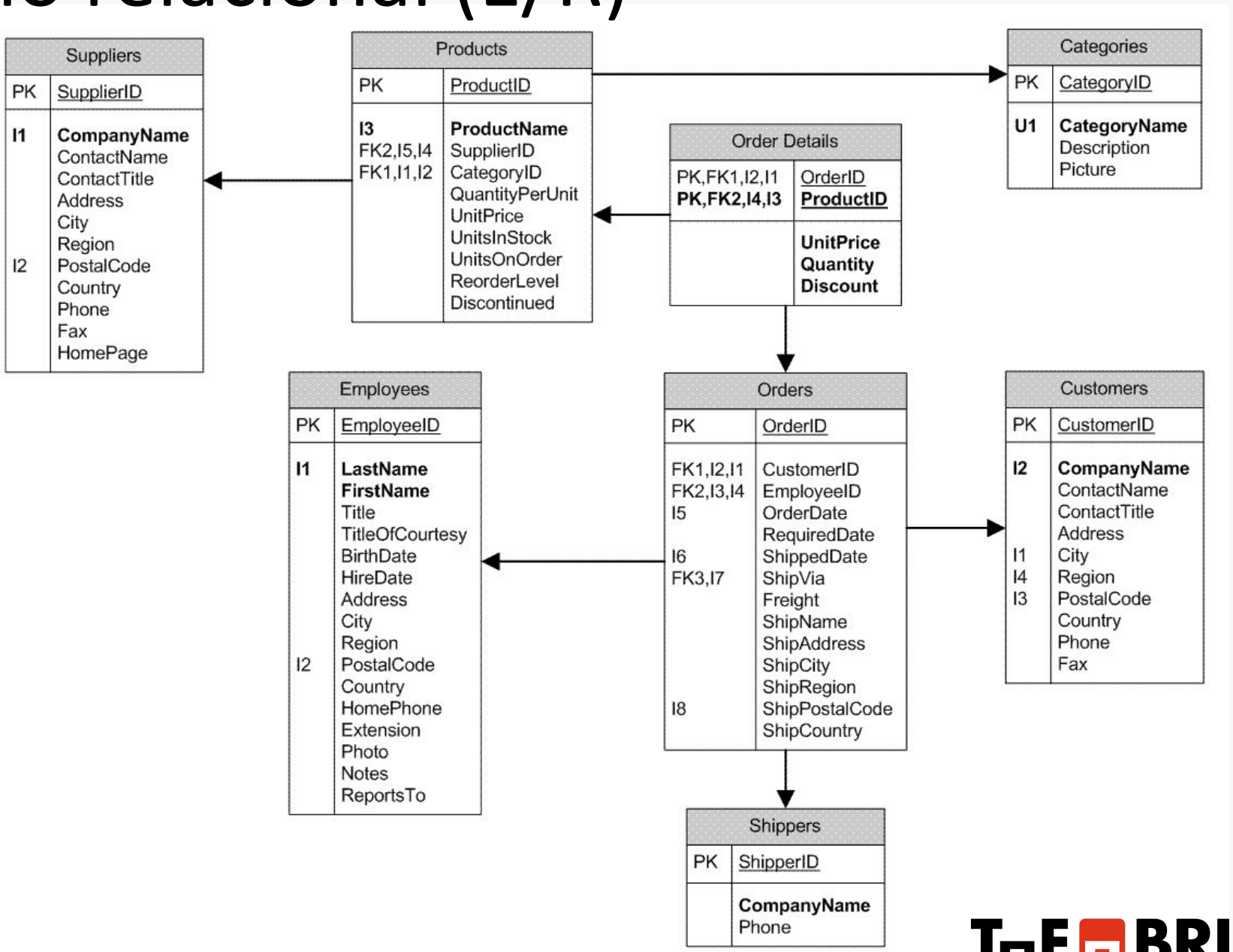
- Independencia lógica y física de los datos.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios.
- Integridad de los datos.
- Consultas complejas optimizadas.
- Seguridad de acceso y auditoría.
- Respaldo y recuperación.
- Acceso a través de lenguajes de programación estándar.

Pedido	Id Cliente	Nombre Cliente	Edad
0001	AGR5	María	29
0002	AGR5	María	29
0003	UXR5	Celedonio	87
0004	UXR5	Celedonio	87
0005	AGR5	María	29
0006	PLR5	Paco	55
0007	PLR5	Paco	55
0008	PLR5	Paco	55
0009	PLR5	Paco	55
0010	UXR5	Celedonio	87

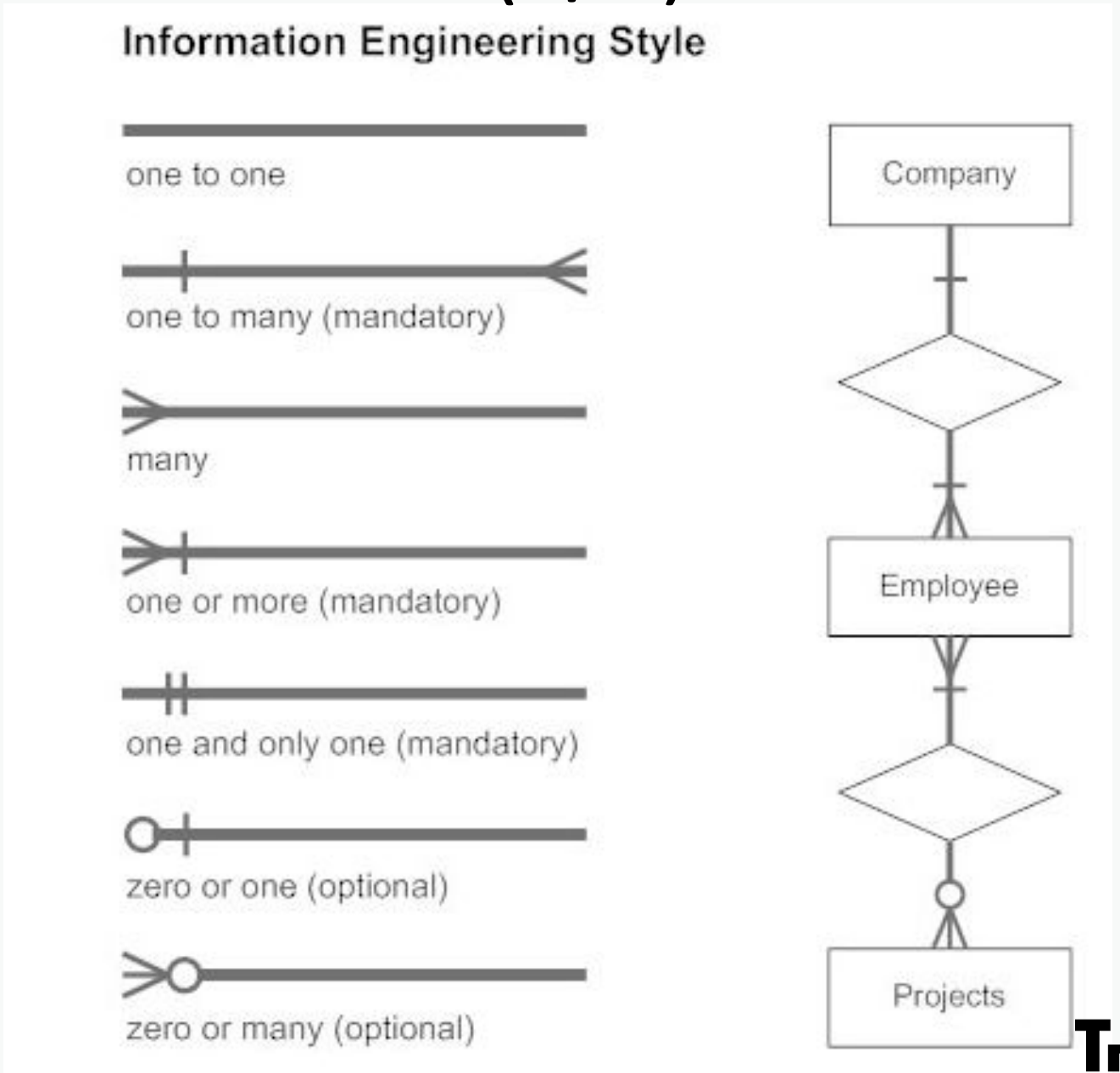
Pedido	Id Cliente
0001	AGR5
0002	AGR5
0003	UXR5
0004	UXR5
0005	AGR5
0006	PLR5
0007	PLR5
0008	PLR5
0009	PLR5
0010	UXR5

Id Cliente	Nombre Cliente	Edad
AGR5	María	29
UXR5	Celedonio	87
PLR5	Paco	55

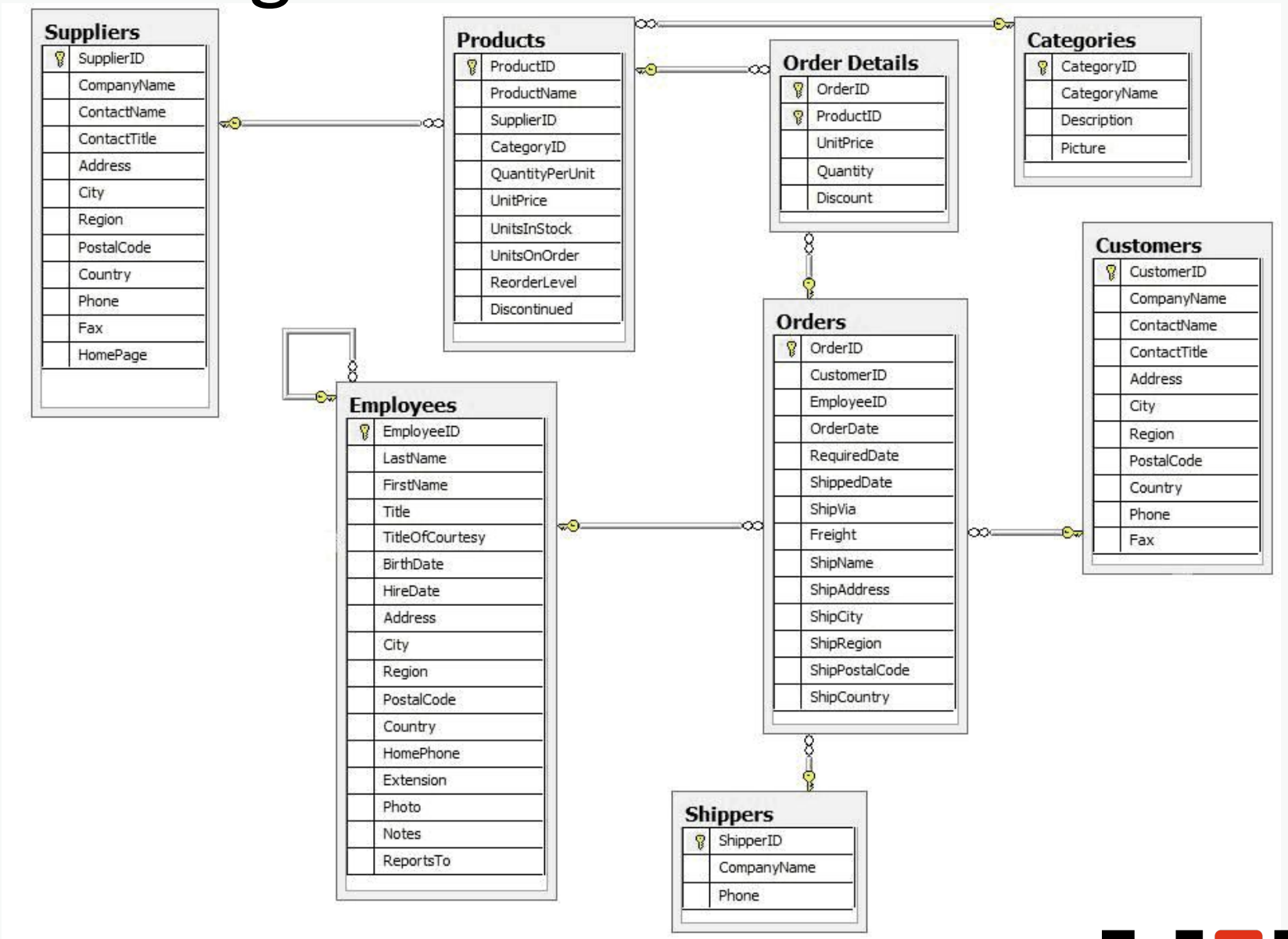
Modelo relacional (E/R)



Modelo relacional (E/R)



Modelo lógico



SQL con Python

In [1]:
```
import pandas as pd
import sqlite3
```

In [2]:
```
# connect withe the chinook database
connection = sqlite3.connect("chinook.db")

# cursor object
crsr = connection.cursor()
```

In [21]:
```
#Base Queries
query = '''
SELECT *
FROM tracks
'''

sql_query(query)
```

Out[21]:

	TrackId	Name	AlbumId	MediaTypeId	GenreId	Composer	Milliseconds	Bytes	UnitPrice
0	1	For Those About To Rock (We Salute You)	1	1	1	Angus Young, Malcolm Young, Brian Johnson	343719	11170334	0.99
1	2	Balls to the Wall	2	2	1	None	342562	5510424	0.99
2	3	Fast As a Shark	3	2	1	F. Baltes, S. Kaufman, U. Dirkscneider & W. Ho...	230619	3990994	0.99

Sentencias SQL

¿Qué es una query?
Se trata de una consulta a base de datos.

SQL cheat sheet

REBELLABS by ZEROTURNAROUND

Basic Queries

```
-- filter your columns
   SELECT col1, col2, col3, ... FROM table1
-- filter the rows
   WHERE col4 = 1 AND col5 = 2
-- aggregate the data
   GROUP by ...
-- limit aggregated data
   HAVING count(*) > 1
-- order of the results
   ORDER BY col2
```

Useful keywords for **SELECTS**:

DISTINCT - return unique results
BETWEEN a **AND** b - limit the range, the values can be numbers, text, or dates
LIKE - pattern search within the column text
IN (a, b, c) - check if the value is contained among given.

Data Modification

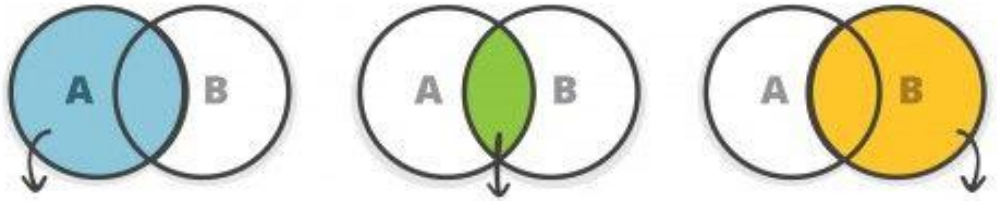
```
-- update specific data with the WHERE clause
   UPDATE table1 SET col1 = 1 WHERE col2 = 2
-- insert values manually
   INSERT INTO table1 (ID, FIRST_NAME, LAST_NAME)
      VALUES (1, 'Rebel', 'Labs');
-- or by using the results of a query
   INSERT INTO table1 (ID, FIRST_NAME, LAST_NAME)
      SELECT id, last_name, first_name FROM table2
```

Views

A **VIEW** is a virtual table, which is a result of a query.
They can be used to create virtual tables of complex queries.

```
CREATE VIEW view1 AS
SELECT col1, col2
FROM table1
WHERE ...
```

The Joy of JOINs



LEFT OUTER JOIN - *all rows from table A, even if they do not exist in table B*

INNER JOIN - *fetch the results that exist in both tables*

RIGHT OUTER JOIN - *all rows from table B, even if they do not exist in table A*

Updates on JOINed Queries

You can use **JOIN**s in your **UPDATE**s

```
UPDATE t1 SET a = 1
FROM table1 t1 JOIN table2 t2 ON t1.id = t2.t1_id
WHERE t1.col1 = 0 AND t2.col2 IS NULL;
```

NB! Use database specific syntax, it might be faster!

Semi JOINs

You can use subqueries instead of **JOIN**s:

```
SELECT col1, col2 FROM table1 WHERE id IN
   (SELECT t1_id FROM table2 WHERE date >
      CURRENT_TIMESTAMP)
```

Indexes

If you query by a column, index it!

```
CREATE INDEX index1 ON table1 (col1)
```

Don't forget:
Avoid overlapping indexes
Avoid indexing on too many columns
Indexes can speed up **DELETE** and **UPDATE** operations

Useful Utility Functions

```
-- convert strings to dates:
   TO_DATE (Oracle, PostgreSQL), STR_TO_DATE (MySQL)
-- return the first non-NULL argument:
   COALESCE (col1, col2, "default value")
-- return current time:
   CURRENT_TIMESTAMP
-- compute set operations on two result sets
   SELECT col1, col2 FROM table1
   UNION / EXCEPT / INTERSECT
   SELECT col3, col4 FROM table2;
```

Union - returns data from both queries
Except - rows from the first query that are not present in the second query
Intersect - rows that are returned from both queries

Reporting

Use aggregation functions

COUNT - return the number of rows
SUM - cumulate the values
AVG - return the average for the group
MIN / MAX - smallest / largest value

BROUGHT TO YOU BY XRebel

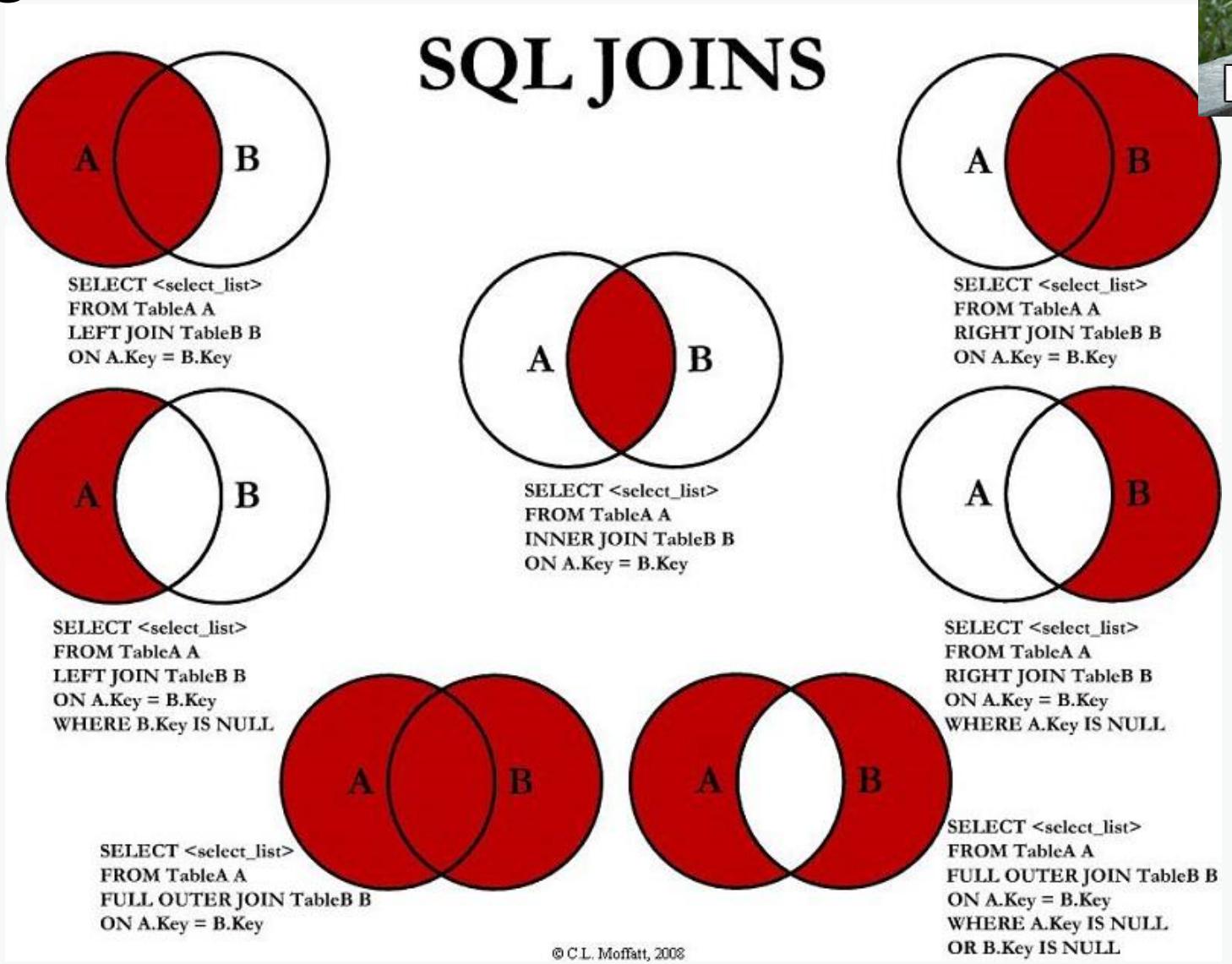
Sentencias

- **SELECT**: se utiliza para seleccionar datos de una base de datos.
- **SELECT DISTINCT**: se usa para devolver solo valores distintos (únicos).
- **FROM**: se utiliza para especificar de qué tabla seleccionar o eliminar datos.
- **WHERE**: se utiliza para filtrar registros.
- Operadores **AND, OR, NOT y BETWEEN**: se suelen combinar con WHERE y se utilizan para filtrar registros basados en más de una condición.
- **ORDER BY**: se utiliza para ordenar el conjunto de resultados en orden ascendente o descendente. Por defecto es ascendente.
- **GROUP BY**: agrupa las filas que tienen los mismos valores en filas resumen, como "encontrar el número de clientes en cada país".

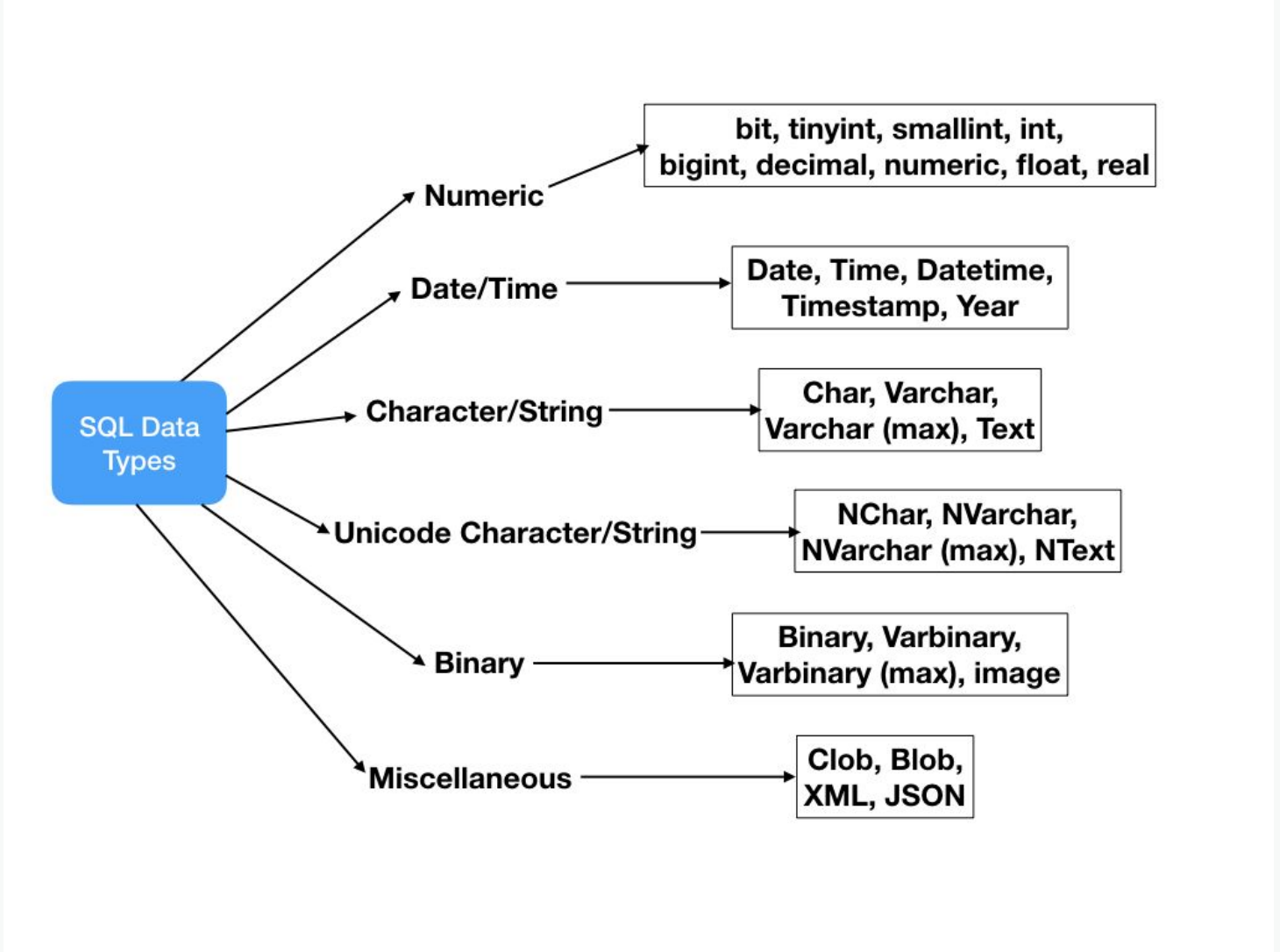
JOINS

- JOIN es una instrucción para combinar datos de diversas tablas.
- Los principales son:
  - **(INNER) JOIN:** Solo obtiene los campos comunes de varias tablas, en función a una columna dada. Se muestra sólo la **intersección.**
  - **LEFT (OUTER) JOIN**: Devuelve todos los registros de la primera tabla (izquierda) y los coincidentes de la segunda (derecha).
  - **RIGHT (OUTER) JOIN**: Devuelve todos los registros de la segunda tabla (derecha) y los coincidentes de la primera (izquierda).
  - **FULL (OUTER) JOIN:** Muestra todas las filas de ambas tablas, sin importar que no existan coincidencias.
  - Se usará NULL como un valor por defecto para los casos en los no haya coincidencias.
  - La tabla asociada al FROM será la tabla LEFT y la tabla de después del JOIN será la tabla RIGHT.

JOINS



DATA TYPES



Recursos

- https://www.w3schools.com/sql/
- https://sqlzoo.net
- https://play.google.com/store/apps/details?id=com.sololearn.sql&hl=es_419
- https://www.genbeta.com/desarrollo/asi-arqueras-nand-juego-mesa-espanol-que-ayuda-a-aprender-lenguaje-sql
- https://www.smartdraw.com/entity-relationship-diagram/