SQL

A gentle introduction

What is SQL?

Structured **Q**uery **L**anguage is a standard database language used to create, manage and query data from relational databases.

What is SQL?

Structured **Q**uery **L**anguage is a standard **database** language used to create, manage and **query** data from **relational** databases.

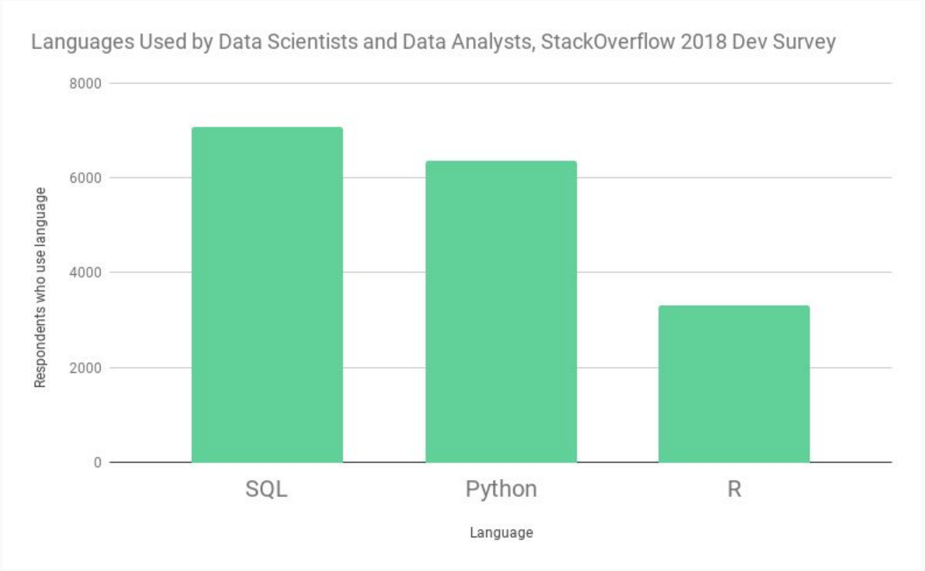
What is SQL?

Structured **Q**uery **L**anguage is a standard **database** language used to create, manage and **query** data from **relational** databases.

(Don´t worry. We´ll get there.)

Why learning SQL?

- Manipulate **large** amounts of **data** efficiently.
- SQL is a **highly demanded** skill in the job market!



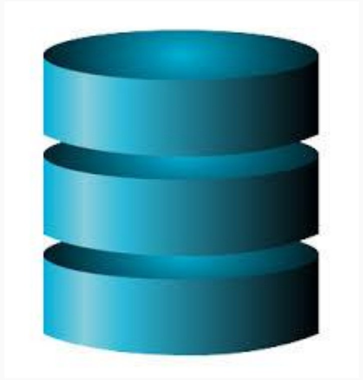
Some useful terminology

Database

- A database is an **organized collection of data** from which you can access and/or modify information.
- They are designed to ensure **speed, efficiency, integrity,** and **consistency**.
- There are two main types:
    - **Relational**
    - **Non-relational**

Database

(Tip: You can think of a **database** as a fancy electronic files cabinet)

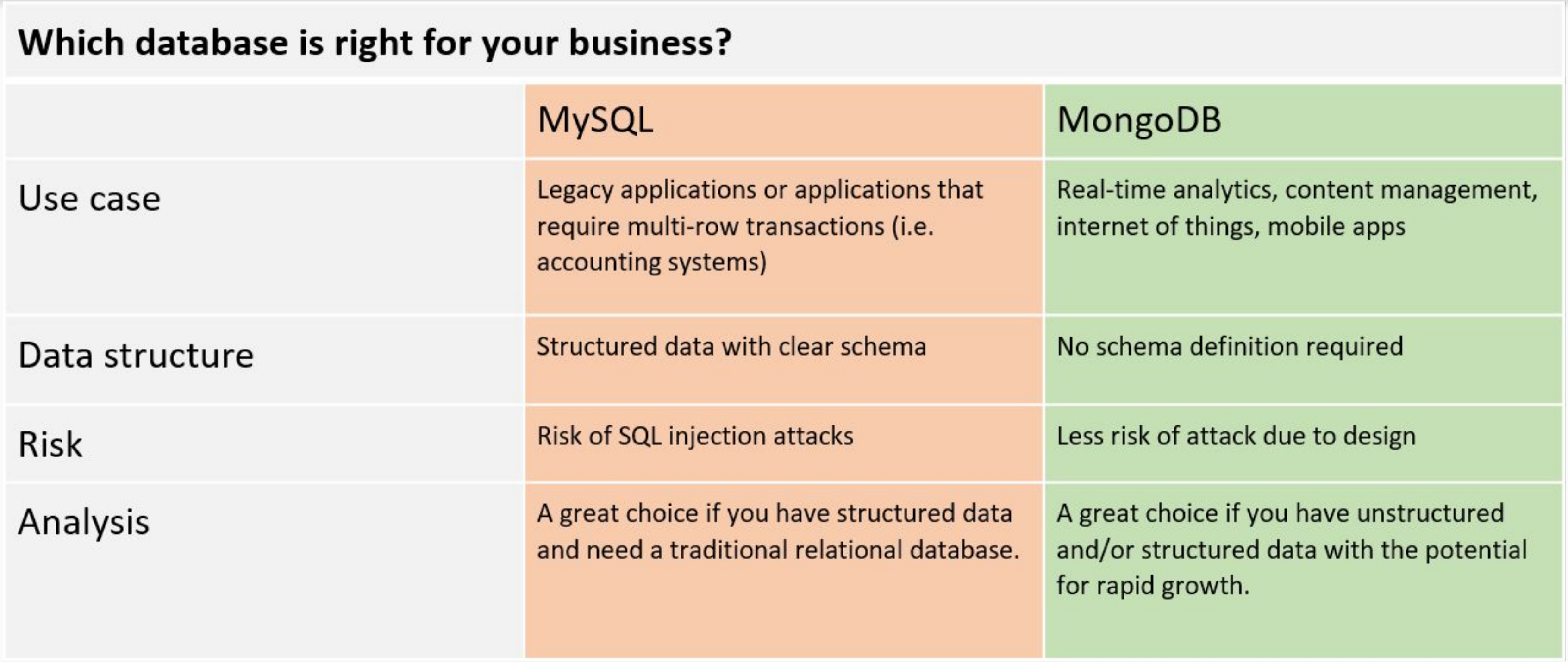


Database **M**anagement **S**ystems (**DBMS**)

To **interact with DB** we use specific pieces of software called **DBMS**. In this bootcamp you will learn two of the most popular ones:

- **R**elational **D**atabase **M**anagement **S**ystem (**RDBMS**):
  - **MySQL**
- **N**on-**R**elational **D**atabase **M**anagement **S**ystem (**NRDBMS**)
  - **MongoDB**

Database Management Systems (DBMS)



Which database is right for your business?

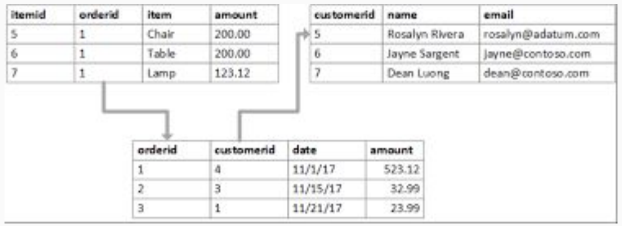
	MySQL	MongoDB
Use case	Legacy applications or applications that require multi-row transactions (i.e. accounting systems)	Real-time analytics, content management, internet of things, mobile apps
Data structure	Structured data with clear schema	No schema definition required
Risk	Risk of SQL injection attacks	Less risk of attack due to design
Analysis	A great choice if you have structured data and need a traditional relational database.	A great choice if you have unstructured and/or structured data with the potential for rapid growth.

What is SQL?

Structured **Q**uery **L**anguage is a standard **database** language used to create, manage and **query** data from **relational** databases.

Relational vs **Non-Relational** Databases

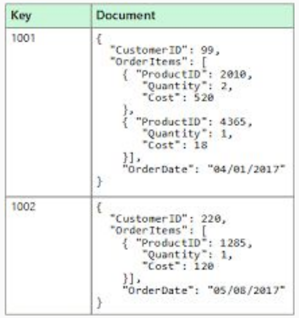
- **Relational** databases:
  - Are based on relational algebra.
  - Store information in **tabular** form
  - Use **SQL**
- **Non-Relational** Databases
  - Store data in **non-tabular** form ("documents")
  - **NoSQL**



itemid	orderid	item	amount
5	1	Chair	200.00
6	1	Table	200.00
7	1	Lamp	123.12

customerid	name	email
5	Rosalyn Rivera	rosalyn@adatum.com
6	Jayne Sargent	jayne@contoso.com
7	Dean Luong	dean@contoso.com

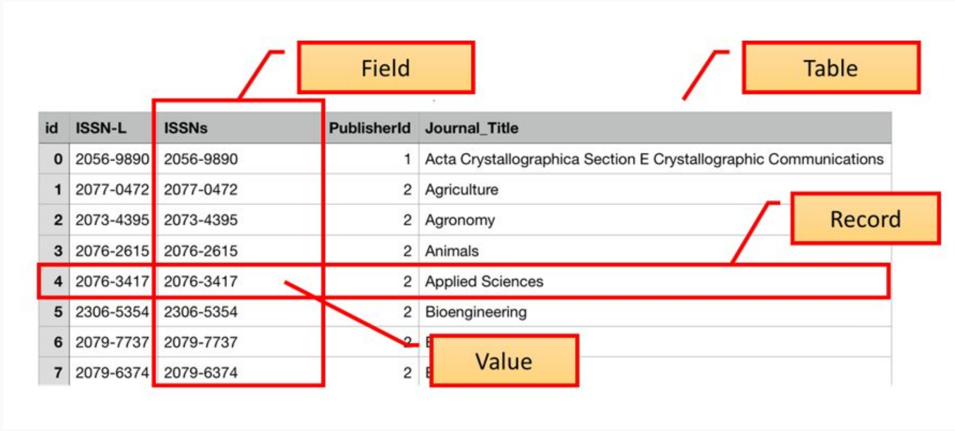
orderid	customerid	date	amount
1	4	11/1/17	523.12
2	3	11/15/17	32.99
3	1	11/21/17	23.99



Key	Document
1001	`{ "CustomerID": 99, "OrderItems": [ { "ProductID": 2010, "Quantity": 2, "Cost": 520 }, { "ProductID": 4365, "Quantity": 1, "Cost": 18 }], "OrderDate": "04/01/2017" }`
1002	`{ "CustomerID": 220, "OrderItems": [ { "ProductID": 1285, "Quantity": 1, "Cost": 120 }], "OrderDate": "05/08/2017" }`

Relational vs **Non-Relational** Databases

- According to the relational model, data is stored in “relations”, which are perceived by the users as **tables**.
- Each “relation” is composed of tuples (or records or **rows**) and attributes (or fields or **columns**)


Field
Table
Record
Value

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]

Relational vs **Non-Relational** Databases

This link seems incredible… but it is true!

What is SQL?

Structured **Q**uery **L**anguage is a standard **database** language used to create, manage and **query** data from **relational** databases.

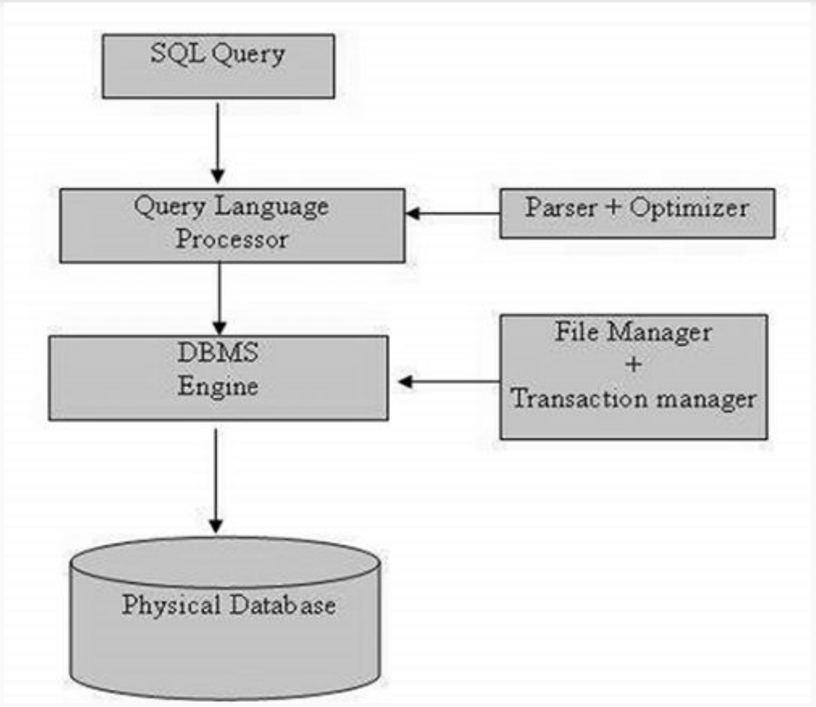
Query

- “**query**” (“consulta”) is a piece of code to ask the computer for some data.
- Since SQL is a ***declarative*** programming language, you must only focus on **what** you want, rather than the details on **how** to do it. For example: *How many products are there in my DB?:*

```
SELECT COUNT(product_id) FROM products;
```

Query

This is a simple diagram to explain how "the magic" happens when you type a query:



What is SQL?

Structured **Q**uery **L**anguage is a standard **database** language used to create, manage and **query** data from **relational** databases.

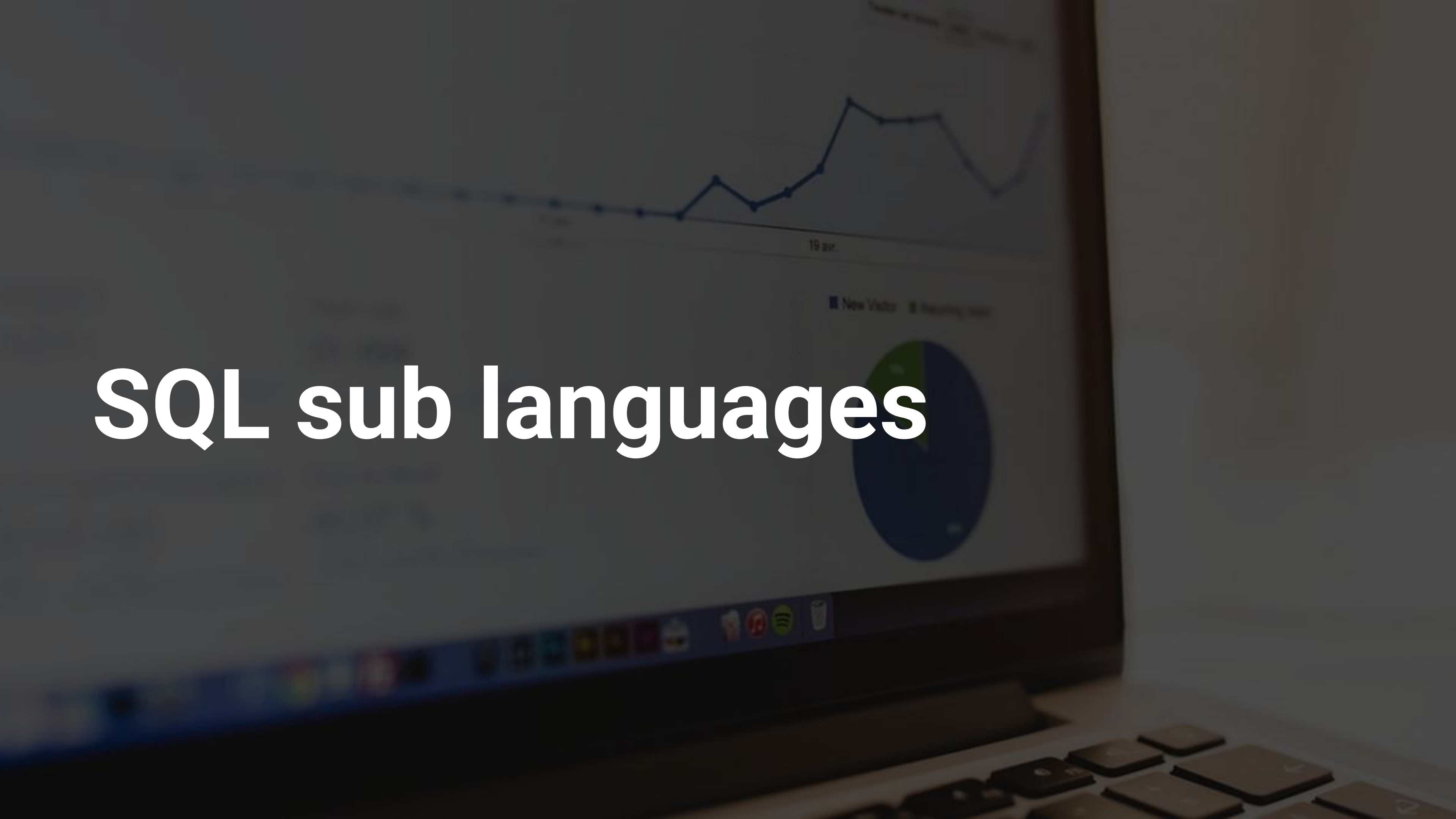
Any questions?

Time to get your hands dirty!

- Type `mysql -u root -p` (mac) or `winpty mysql -u root -p` (Windows) in the terminal to test your MySQL installation .
- Open Sequel Pro (or MySQL Workbench) and check that you are connected to your localhost MySQL server.
- Prework review: Code along!!
- The police needs you!

SQL

A gentle introduction (day 2)



SQL sub languages

SQL sub languages

The syntax of SQL is usually divided in **different categories** or sub languages depending on the **type of operations/statements** they perform. The main components of SQL´s syntax are:

- Data Definition Language (**DDL**)
- Data Manipulation Language (**DML**)
- Data Control Language (**DCL**)
- Transaction Control Language (**TCL**)

Data Definition Language (**DDL**)

Data Definition Language is used to **define or modify the database objects** (schemas, tables, etc.) The most common operations are:

- `CREATE`: it is used to create entire databases and database objects such as tables
- `DROP`: it deletes objects from database (beware!)
- `RENAME`: it renames objects
- `TRUNCATE`: it deletes the data within a table (but not the table)

Data Manipulation Language (DML)

Its statements allow us to **manipulate the data** in the tables of a database.

- `SELECT… FROM…`: retrieve data from the database
- `INSERT INTO… VALUES…`: used to insert data into tables.
- `UPDATE… SET… WHERE`: renew existing data of your tables.
- `DELETE… FROM… WHERE`: like `TRUNCATE`, but we can specify what we want to remove (rather than all the records in a table).

Data Control Language (DML)

With only two statements **it manages the rights** users have in a database

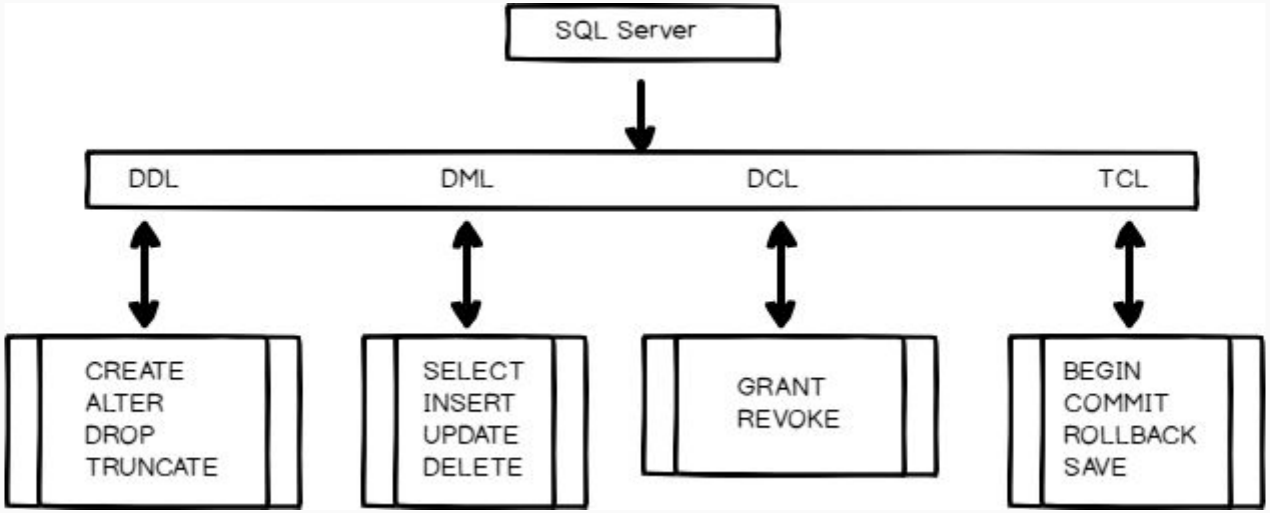
- `GRANT`: give permissions.
- `REVOKE`: revoke permissions.

Transaction Control Language (TCL)

Not every change you make to a database is **saved** automatically. MySQL comes with an option `autocommit` enabled by default, so we don't have to worry much about this.

- `COMMIT:` saves the transaction in the database. Changes can not be undone
- `ROLLBACK:` Reverts to the last committed state.

SQL sub languages (summary)

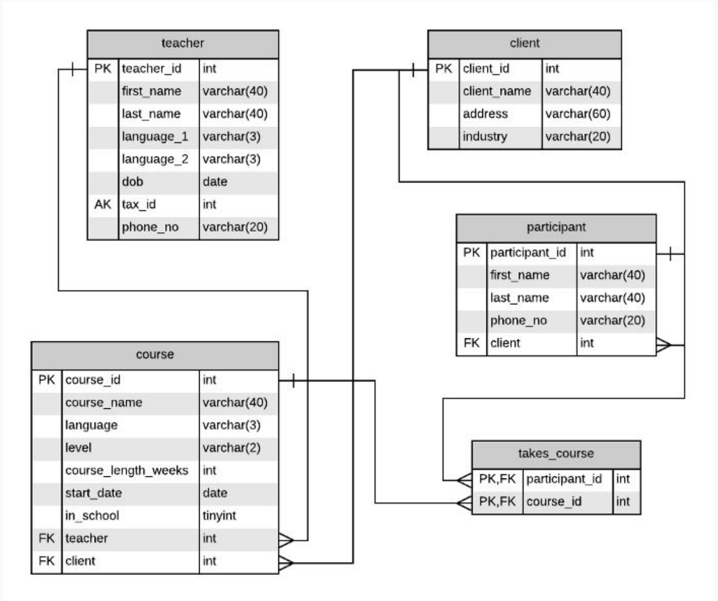


Relational schemas

Relational schemas

We use a relational schema to describe how our database looks like. They key components to any relational schema are:

- Tables
- Primary Keys
- Foreign Keys
- Relationships



	teacher	
PK	teacher_id	int
	first_name	varchar(40)
	last_name	varchar(40)
	language_1	varchar(3)
	language_2	varchar(3)
	dob	date
AK	tax_id	int
	phone_no	varchar(20)

	client	
PK	client_id	int
	client_name	varchar(40)
	address	varchar(60)
	industry	varchar(20)

	participant	
PK	participant_id	int
	first_name	varchar(40)
	last_name	varchar(40)
	phone_no	varchar(20)
FK	client	int

	course	
PK	course_id	int
	course_name	varchar(40)
	language	varchar(3)
	level	varchar(2)
	course_length_weeks	int
	start_date	date
	in_school	tinyint
FK	teacher	int
FK	client	int

	takes_course	
PK,FK	participant_id	int
PK,FK	course_id	int

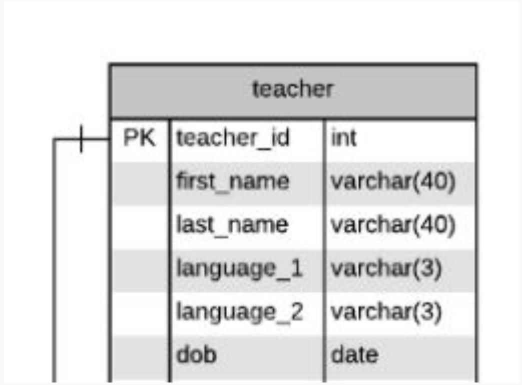
Note: here you have a super nice series of posts in towardsdatascience where they discuss these concepts. Our live coding session will follow them closely, so refer to them for reviewing or deep diving!

Relational schemas: primary keys (pk)

A column (or a set of columns) whose value exists (can not be null) and is unique for every record in a table.

Each table can have one and only one primary key.

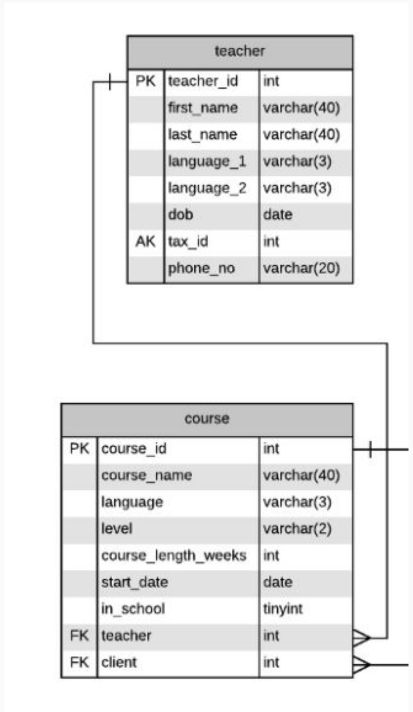
They are they **unique identifiers of a table**.



teacher		
PK	teacher_id	int
	first_name	varchar(40)
	last_name	varchar(40)
	language_1	varchar(3)
	language_2	varchar(3)
	dob	date

Relational schemas: foreign keys (fk)

A column (or a set of columns) that **identifies the relationship** between one table (child) and another (parent) --via the primary key of the latter.



teacher		
PK	teacher_id	int
	first_name	varchar(40)
	last_name	varchar(40)
	language_1	varchar(3)
	language_2	varchar(3)
	dob	date
AK	tax_id	int
	phone_no	varchar(20)

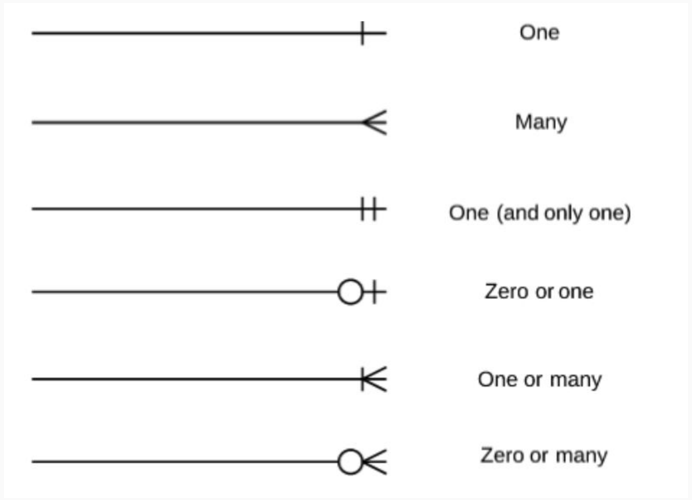
course		
PK	course_id	int
	course_name	varchar(40)
	language	varchar(3)
	level	varchar(2)
	course_length_weeks	int
	start_date	date
	in_school	tinyint
FK	teacher	int
FK	client	int

Relational schemas: Relationships

Relationships tell you how much of the data from a foreign key can be seen in the primary key column(s) of the table the data is related to, and vice versa. There are three main types:

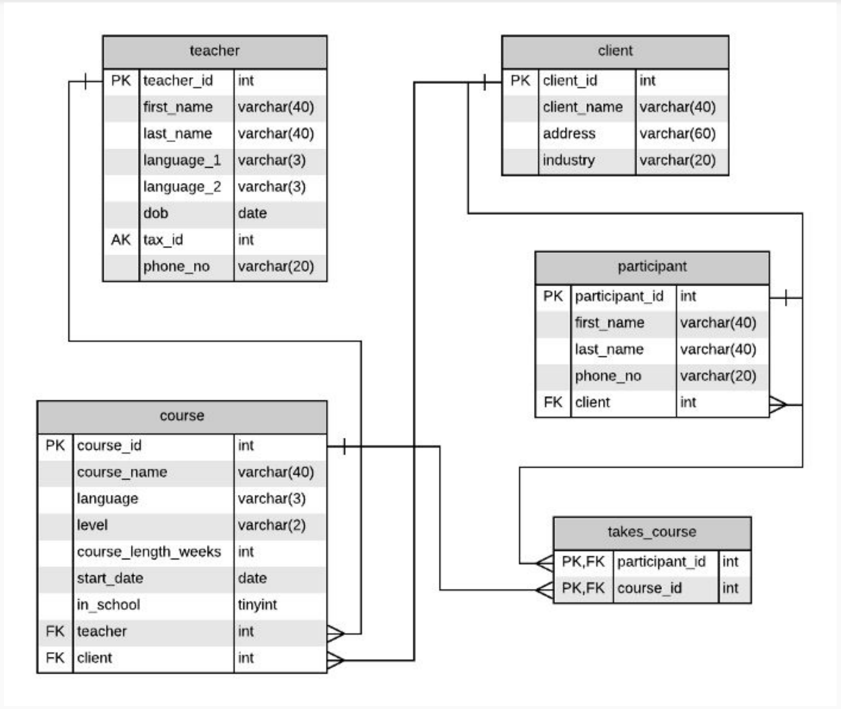
- `one-to-one`
- `one-to-many`
- `many-to-many`

Note: The process of defining these relationships is often referred to as *normalization.*



Relational schemas: Relationships

What is the type of relationship between… ?



teacher		
PK	teacher_id	int
	first_name	varchar(40)
	last_name	varchar(40)
	language_1	varchar(3)
	language_2	varchar(3)
	dob	date
AK	tax_id	int
	phone_no	varchar(20)

client		
PK	client_id	int
	client_name	varchar(40)
	address	varchar(60)
	industry	varchar(20)

participant		
PK	participant_id	int
	first_name	varchar(40)
	last_name	varchar(40)
	phone_no	varchar(20)
FK	client	int

course		
PK	course_id	int
	course_name	varchar(40)
	language	varchar(3)
	level	varchar(2)
	course_length_weeks	int
	start_date	date
	in_school	tinyint
FK	teacher	int
FK	client	int

takes_course		
PK,FK	participant_id	int
PK,FK	course_id	int

Let´s code!!

[link to the code](#) used in the classin

Further materials

Nice introductory, short video to relational schemas.

SQL Cheat Sheet

What is SQL?

SQL is a database language used to query and manipulate the data in the database.

MySQL/Language/Definitions

- **Data Definition Language(DDL)**
- **Data Manipulation Language(DML)**
- **Data Control Language(DCL)**
- **Data Query Language(DQL)**
- **Data Transfer Language(DTL)**

Querying from a Table

- SELECT **a, b** FROM **T**; (Querying Data in Columns a, b from Table T)
- SELECT ***** FROM **T**; (Querying all rows and columns from a table)
- SELECT **a, b** FROM **T** WHERE **Condition;** (Query data and filter rows with a condition)
- SELECT DISTINCT **a** FROM **T** WHERE **condition**; (Query distinct rows from a table)
- SELECT **a, b** FROM **T** ORDER BY ASC/DESC; (Sort the result set in ascending or descending order)
- SELECT **a, b** FROM **T** ORDER BY **a** LIMIT **n** OFFSET **Offset**; (Skip Offset of rows and return the next n rows)
- SELECT **a,** aggregate**(b)** FROM **T** GROUP BY **A**; (Group rows using an aggregate function)
- SELECT **a,** aggregate**(b)** FROM **T** GROUP BY **A** HAVING **condition**; (Filter groups using HAVING Clause)