# Data Management for Big Data *SQL*

Andrea Brunello andrea.brunello@uniud.it



#### Introduction

SQL (Structured Query Language) is the **de facto standard** for interacting with relational databases. It allows one to:

- define and modify the database logical schema
- populate the database, modify and retrieve information
- perform administrative tasks, for instance, manage security and access rights
- customize concurrency control strategies



#### History

- IBM Sequel language developed as a part of System R project in the early 70s
- ANSI and ISO standard:
  - first formalization, SQL-86
  - latest standard, SQL:2019
- Commercial systems offer most, if not all, SQL-92 features, plus varying proprietary features
- Actually, SQL implementations offered by different DBMS vendors are slightly different from one another: https://www.postgresql.org/docs/current/sql.html
- Here, we are going to refer to PostgreSQL



#### DDL and DML

- SQL is a declarative language composed of two parts:
  - DDL (Data Definition Language): that allows to specify the schema of the database (i.e., tables, constraints, ...)
  - DML (Data Manipulation Language): that allows to interact with the content of the database (i.e., the database instance) by means of *queries*
- SQL features a very intuitive syntax
- It is standardized within ISO/IEC, so to allow for a better interoperability between different relational DBMSs (at least for what concerns its basic features)



#### SQL, Data Manipulation Language

The fundamental block of an SQL query is composed of **three** clauses

SELECT: list of attributes, aggregation functions, expressions

FROM : data sources, tables

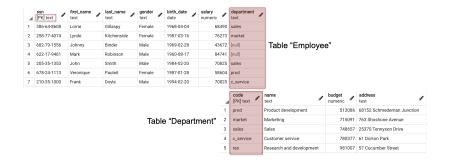
WHERE: filtering conditions for the data that has to be extracted

Other clauses: ORDER BY, HAVING, GROUP BY



# SQL, Data Manipulation Language Exemplary setting

#### Let us consider the following two tables





Extract the name, surname, and monthly salary all employees of gender female and with a total salary greater than 60000

SELECT first\_name, last\_name, salary/12
FROM employee
WHERE gender = 'Female' AND salary > 60000;

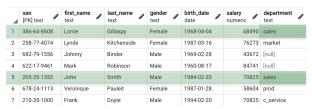
4	ssn [PK] text	first_name text	last_name /	gender text	birth_date date	salary numeric	department text
1	386-64-8608	Lorrie	Gillaspy	Female	1968-04-04	68490	sales
2	258-77-4074	Lynde	Kitchenside	Female	1987-03-16	76273	market
3	682-79-1556	Johnny	Binder	Male	1969-02-28	43672	[null]
4	622-17-9461	Mark	Robinson	Male	1960-08-17	84741	[null]
5	205-35-1353	John	Smith	Male	1984-02-20	70825	sales
6	678-24-1113	Veronique	Pauleit	Female	1987-01-28	58604	prod
7	210-35-1000	Frank	Doyle	Male	1994-02-20	70825	c_service

4	first_name text	last_name text	round numeric
1	Lorrie	Gillaspy	5707.50
2	Lynde	Kitchenside	6356.08



Extract the average salary of employees working in the sales department

SELECT AVG(salary)
FROM employee
WHERE department = 'sales';







Extract the name and surname of employees working at a department with a budget greater than 720000

**SELECT** employee.first\_name, employee.last\_name **FROM** employee

**WHERE** department.budget > 720000;

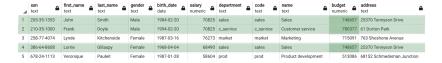


Each row of employee is combined with every other row of department; then, only the resulting rows that satisfy the JOIN condition are kept





At this point, the WHERE condition is applied over the remaining rows



We then obtain the final result, keeping just the attributes specified in the SELECT clause





#### References

A. Silberschatz, H.F. Korth, S. Sudarshan *Database system concepts*, 7th Edition, 2020.

PostgreSQL's website: https://www.postgresql.org/