Database

What is a data?

In computer science, a data is the representation of an information.

For example, your name, your age, your height, your weight, etc. are data that represent / characterize you.

What is a database?

A database, or BDD, is therefore a collection of data.
It is a set of data modeling the objects of a part of the real world and serving as a support for a computer application.
Simply put, it allows you to store and manipulate data.

Database Management System

What is a DBMS?

A DBMS is a set of software that allows users to access a database, manipulate data, insert, edit, and search for specific data.

Searches for a given data can be executed from the value of a data designated by a name in a set of objects (for example, we can search for all people who are more than 1m70), but also from of relationships between objects (for example, we can search all the friends of a person).

The DBMS makes sharing transparent, ie it gives the illusion to each user that he or she alone is working with the data.

Do not confuse BDD and SBGD

A database is a collection of data that is stored and available on demand. The DBMS is the software by which one accesses the database, it manages the machine aspect: the performances, the storage. It allows you to insert data, edit, search, etc.

Of course, to be able to manipulate this data from the DBMS, we need a language (SQL for example). We will see this part later in this course.

DBMS architecture

What is DBMS architecture?

DBMS architecture facilitates the design, development, implementation and maintenance of a database. It's important to understand that a database stores critical information for a company. Selecting the appropriate database architecture allows for fast and secure access to this data.

1-tier

architecture The simplest architecture is the 1-tier or 'single-tier' architecture. With this architecture, the client, the server, and the database all reside on the same machine.
Much used in the development / testing phase, such an architecture is rarely used in production.

2-tier

architecture A 2-tier architecture is an architecture where:
> The physical / graphical layer runs on a client (PC, mobile, tablet, etc.).
> The data is stored on a server.

The 2-tier architecture provides additional security for the DBMS because it is not exposed directly to the end user.

3-tier

architecture The 3-tier architecture is an architecture where:

> The graphics layer runs on a client (PC, mobile, tablet, etc.).
> The application layer is on a server.
> The data is stored on another server.

This DBMS architecture contains an application layer between the user and the DBMS, which is responsible for communicating the user's request to the DBMS system and sending the DBMS response to the user.

The application layer also processes functional logic, constraints, and rules before transmitting data to the user or to the DBMS.

The three-tier architecture is the most common DBMS architecture in production.

The relational data model

What is the relational model?

The relational model presents the database as a collection of relationships. A relationship is a table of values.
In the relational model, the data is stored in the form of tables (logical layer! Not physical). Each row of the table represents a collection of linked data values. The data is represented as a set of relationships.

When we talk about relational model, we also talk about "relational database management system" (RBMS).

Relational Model Structurerelational

Themodel is characterized by:

> Tables: Relationships are stored as a table. A table has columns and rows.
Each line represents a record.
Each column represents an attribute.

> Attributes: Attributes are the properties that define a relationship. For example: name, first name, age.

> Tuple: A tuple is a record (a row) of a table.

> Degree: The degree is the number of attributes in a table.

> Key: A key is an attribute or set of attributes that identifies each row of a given table differently. For example, a unique number (like prisoners).

Integrity

Constraints Relational Integrity Constraints are conditions, rules that must be followed to obtain a valid table.

There are several types of integrity constraints. Integrity constraints on a RDBMS are mainly divided into three categories:

> Domain
Constraint> Key
Constraint> Referential Integrity

Constraint Integrity

Constraints Relational Integrity Constraints are conditions, rules that must be respected for get a valid table.

There are several types of integrity constraints. Integrity constraints on a RDBMS are mainly divided into three categories:

> Domain
ConstraintKey
>Constraint> Referential Integrity

Constraint a) Domain Constraints Domain

constraints can be violated if an attribute value is not not of the appropriate data type.

Domain constraints specify this in each tuple and the value of each attribute must be unique. Standard data types are: integers, real numbers, characters, Booleans, strings, and so on.

Example:
We have a table with an attribute (column) 'Country'.
We define a domain 'DOMAIN1' as all the countries of the earth.
The possible values for the attribute (column) are thus limited to the values defined in 'DOMAIN1'.

b) Key Constraints

An attribute that can uniquely identify a tuple in a table is called the key. The value of the attribute for different tuples in the relationship must be unique. In addition, duplicates are prohibited.

c) Referential integrityReferential integrity

constraintsconstraints are based on the concept of foreign keys.
A foreign key identifies a column of a table as referencing a column from another table.
However, this key element must exist in the table.

Operations

In a relational data model, there are 4 basic operations:

> INSERT that is used to add data to a table
> DELETE which allows to delete a tuple
> MODIFY which makes it possible to change the value of an attribute
> SELECT which allows to choose / select a dataset

Best Practices

- The cells in the table must contain a single value
- Each column must have a unique name
- Two rows can not be identical
- The values of an attribute must come from the same domain

The keys

What is a key? What's the point ?

As we saw earlier, a key is an attribute or set of attributes that helps you identify a line (tuple) in a relation (table). It allows to find the relation between two tables. A key can be a combination of one or more columns of a table.

Keys help identify a row of data in a table. In an application, a table can contain thousands of records. In addition, the recordings could be duplicated. The keys thus make it possible to uniquely identify a table record despite these difficulties.

There are several types of Keys, each with different properties / roles.

The Candidate Keycandidate

Akey is a concept at the modeling level. These are any attributes of a relation that can retrieve a single element of the relation (tuple) by their value (s).

For a car, a candidate key could be its registration number, or its serial number, which is supposed to be unique.
For one person, his security number (in France) is a candidate key. His email address may be another.

Some rules for candidate keys:
> Must guarantee unique values
> Candidate keys can be composed of several attributes
> Must not be null

The Primary Key

The primary key is a concept at the DBMS level.
A column or group of columns in a table that uniquely identifies each row in that table is called the primary key.

To take the example of the car, if I choose the registration number as my primary key it is he who will allow me to uniquely identify a car.

Some rules for primary keys:
> Two lines can not have the same primary key
> For each line, it is mandatory to have a primary key
> The primary key can not be null.
> The value of a primary key can not be changed if a foreign key refers to this primary key.

The Alternative Key

All keys that are not primary keys are called "alternative keys".
This is a candidate key that is not a primary key.
(Miss France: all miss are candidates, miss France elected primary key, the other alternatives)

The Foreign Key

A foreign key is a column that is added to create a relationship with another table. Foreign keys help us maintain data integrity and also allow navigation between two different instances of an entity. Each relationship between the tables must be supported by a foreign key.

The Compound Key

The composite key has several fields that allow you to uniquely recognize a specific record. It is possible that each column is not unique in itself in the database. However, when combined with the other column (s), the combination of composite keys becomes unique.