DATABASE DESIGN

Database design is the process of structuring and organizing data to allow efficient access, proper management, and information security. Good database design is essential to ensure consistency, reliability, and speed in data insertion, updating, deletion, and search operations.

What is a database?

A database is a set of data organized so that it can be easily consulted, managed, and updated. The data is stored in a permanent memory medium and is designed so that multiple users can access it at the same time. Databases are created and managed by database management systems (dbms). A dbms is a set of software tools that can manage large amounts of structured, shared, and permanent data that can be updated and queried. A dbms also ensures reliability and confidentiality. A database is, therefore, a collection of data managed by a dbms
the dbms is therefore a software (or even a collection of software) that allows you to create databases, insert data, modify it and perform searches on it.
• DATABASE MANAGEMENT: a dbms allows you to create a database, insert, modify, delete and search for data. you can also interface with a dbms using programming languages or applications that adopt standard protocols and languages.
• DATA PERSISTENCE AND CONSISTENCY: a dbms stores data in a permanent memory and allows its recovery in the event of malfunctions through some backup and restore functions. it also ensures that the data is consistent with each other. a modification operation is successful if all the data to which it refers are modified, in which case the operation is said to be atomic (i.e., indivisible).
• PRIVACY AND SECURITY: only authorized users (usually through username and password) can access data and only those for which they have authorization.
• DATA INTEGRITY: in a dbms there is a special user, the dba (database administrator), who is authorized to make changes to the structures containing the data and can establish constraints on them; for example, imposing that, for the age, an integer greater than or equal to 0 must be entered. the dbms then guarantees that these constraints are always respected, so that it is not possible to enter invalid values. as we will see later, integrity must be seen in relation to what has been defined through the constraints.
• SCALABILITY: unlike what happens when using text files or spreadsheets, performance remains almost unchanged regardless of the quantity of data, whether it is little or very much.
• TRANSACTION SUPPORT: a transaction is a sequence of operations correlated with each other that are executed atomically; that is, if even just one of these operations fails, the entire sequence is canceled and the database returns to the state before its beginning. only

if all the operations of the sequence are successful the database is modified and this preserves the integrity and consistency of the data.
-• METADATA MANAGEMENT: metadata is the information that describes the data, for example the names of the tables and fields, which are also saved in the database in special tables accessible to the administrator.
There are different types of databases, each with specific characteristics for different types of applications. Here are the main categories:
RELATIONAL DATABASES (RDBMS)
A relational database is a type of database that organizes data in tables (called relations) composed of rows (tuples) and columns (attributes). It is based on the relational model proposed by Edgar F. Codd in 1970 and uses the SQL (Structured Query Language) language for data management and querying.

Main features:

Table structure: Data is organized in tables with rows and columns, where each row represents an instance and each column an attribute.
Primary keys and foreign keys: Each table has a primary key (Primary Key, PK) that uniquely identifies each row. Foreign keys (Foreign Key, FK) connect tables to each other, establishing relationships between data.
Referential integrity: Ensures data consistency between tables linked by foreign keys, preventing operations that would violate relationship constraints.
Logical and physical independence of data: The logical structure of the database is separated from its physical implementation, allowing changes without impacting the application.
SQL language: The relational database is managed and queried using SQL, which allows data insertion, update, deletion and selection operations.
Atomicity, Consistency, Isolation, Durability (ACID): Relational database systems ensure the correctness of transactions through these four properties:
• Atomicity: Each transaction is either fully executed or canceled.
• Consistency: The database remains in a valid state before and after a transaction.

• Isolation: Concurrent transactions do not interfere with each other.
• Durability: A completed transaction remains saved in the database even in the event of failures. Data normalization: Process that eliminates redundancies and inconsistencies in data by splitting it into multiple linked tables.
Security and access control: Management of users with differentiated permissions to ensure data confidentiality and integrity.
Examples of popular relational DBMS:
• MySQL
• PostgreSQL
• Oracle Database
• Microsoft SQL Server
• SQLite

HIERARCHICAL DATABASES

A hierarchical database is a type of database that organizes data in a tree structure, where each parent record can have multiple child records, but each child has only one parent. This structure is similar to a folder with subfolders in an operating system.

Key Features

1. Tree Structure
• Data is organized in a hierarchy with a root node and multiple levels of sub-nodes.
• Each parent can have multiple children, but each child has only one parent (one-to-many, 1:N relationship).
2. One-to-Many (1:N) Relationships
• A single parent node can have multiple child nodes.
• Does not directly support many-to-many (N:N) relationships unless the data is duplicated.
3. Rigid Access Path
• To access a child node, you have to traverse the hierarchy from top to bottom.
• This makes retrieval fast if the structure is well-defined, but inefficient for traversal searches.
4. Efficiency in Predefined Searches
• Very efficient if queries follow the natural hierarchy.
• Not suitable if data needs to be retrieved with cross-sectional searches (e.g. find all employees regardless of department).
5. Difficulty in Editing and Updating
• Changing the structure is complicated: If a node needs to be moved, it may be necessary to reorganize the entire database. Duplication of data to represent complex relationships.
6. Greater Security and Control
• Being based on a fixed hierarchical structure, it is easier to control access to the various levels.
Today, hierarchical databases have been almost completely replaced by relational databases (SQL) that offer greater flexibility.

NETWORK DATABASES

A network database is a type of database that organizes data in a graph structure, where each record can have multiple links to other records, allowing many-to-many (N:N) relationships between data. It was developed in the 1960s as an evolution of the hierarchical model to overcome its limitations and was standardized by CODASYL (Conference on Data Systems Languages)
Main Features
1. Graph Structure
• Data is organized in a network of nodes and edges, similar to a graph.
• Each record can have multiple connections to other records, allowing complex relationships.
• Links between records are represented by direct pointers, making navigation more efficient than in hierarchical databases.
2. Many-to-Many (N:N) Relationships
• Unlike the hierarchical model (which only supports one-to-many, 1:N), a network database can represent many-to-many (N:N) without the need to duplicate data.

• This makes it suitable for situations where elements are highly interconnected, such as in a university management system where a student may be enrolled in multiple courses and a course may have multiple students.
3. Direct Data Access
• It uses physical pointers to connect records, making data search and access very fast.
• Unlike the relational model, it does not require complex SQL queries with joins, but allows direct access to related

records.

4. Complexity in Management

• The use of physical pointers makes network databases more complex to manage than relational databases.
• Any change to the network structure may require a redesign of the edges and nodes.

5. Greater Flexibility than Hierarchical Databases

• Unlike the hierarchical model, where each node has only one parent, here a node can be connected to multiple parent and child nodes.
• This makes the database more flexible and reduces data duplication.

OBJECT DATABASE

An object database is a type of database that combines the concepts of object-oriented programming (OOP) with data management.
• The stored elements are not only data structured in tables (as in relational databases), but also objects, which include attributes and methods.
• Each object belongs to a class and can have relationships with other objects, supporting concepts such as inheritance, encapsulation, and polymorphism.

Main Features

1. Object-Based Structure

• Data is stored in the form of objects instead of tables.
• Each object is an instance of a class, which defines its attributes and methods.
• Classes can inherit characteristics from other classes (inheritance).

2. Support for Object Persistence

• An object stored in the database retains the state and behaviors defined in its class.
• It can be retrieved and used without having to convert it to a different format.

3. Object Relationships

• Objects can have direct associations with each other.
• No complex joins like in relational databases are needed, since objects are already linked via references.

4. Encapsulation and Methods in Objects

• Each object can have methods that perform operations on the data, reducing the need to write separate SQL code.
• Users can interact with objects without worrying about their internal structure

5. Support for Inheritance and Polymorphism

• Classes can inherit attributes and methods from other classes, avoiding data redundancy.
• An object can be treated as an instance of its parent class (polymorphism).

6. More Complex than Relational Databases

• Managing relationships between objects can become complex for very large structures.
• Fewer standardized tools than SQL databases.

XML Databases

An XML database is a type of database designed to store, manage, and query structured data in XML (eXtensible Markup Language) format.
• XML is a hierarchical and self-describing format, ideal for representing data with flexible structures.
• XML databases are used in applications that require data exchange between different systems, such as web services, electronic documents and software configurations.

Main Features

1. XML-Based Structure

• Data is stored in XML documents instead of relational tables.
• Each document has a hierarchical structure with nested nodes.
• XML elements can have attributes and textual values.

Here the data is organized in a hierarchical structure, where university is the root node, and each student is a child node.

2. Hierarchy and Nesting

• The tree structure allows nesting data, easily representing parent-child relationships.
• Separate tables are not needed as in relational databases.

3. Flexibility in Data Structure

• XML does not require a fixed structure like SQL tables.
• New fields can be added without changing the entire schema.

• Ideal for data with variable or non-homogeneous structures.

4. Support for XML Queries (XPath, XQuery)

• XML databases use specific languages to query data, such as:

o XPath → to navigate between XML nodes.

o XQuery → to query and transform XML data.

5. Native or Relational Storage

XML databases can be:

1. Native XML → store data directly in XML format, optimizing searches.
2. XML on Relational Databases → store XML data in XML type columns within a SQL database.

6. Platform Independence and Interoperability

• XML is universal and readable by any system.

• Facilitates the exchange of data between different applications (e.g. between a database and a web app).

• Used in web services (SOAP, REST) to transmit information between servers.