We have become more and more connected to each other and organizations with the burgeoning of the internet of things. The more we use online services the more data we use and create. Data is stored in databases. Data warehouses and data lakes are bigger and more sophisticated databases to preserve large volumes of data that are often used by analytics teams to unlock insights about business strategies and exploitations.

Big Data is about harvesting, categorizing, and studying large amounts of data generated by countless sources.

Blockchain technology is an evolved type of database that enhances distributed computing in peer-to-peer environments with a secure decentralized approach to managing transactional data known as distributed ledgers.

Interacting with a database means that we use queries, or questions in technical jargon, to interrogate the database. The database management solution will then display the result of our query through a table organized in attributes that we mentioned in our query.

When the same data is collected and stored by different departments using different database management solutions and labeling attributes differently causes data silos within the same organization. This can lead to flawed or contradictory information which misleads an organization into studying inaccurate patterns make misinformed decisions.

An integrated database is a collection of data that can be used and easily accessed across departments. It is big yet it is concise, meaning there aren’t repeating attributes and the entities are well-defined to avoid misunderstandings.

Data about data is called metadata.

An enterprise’s database isn’t owned by anyone, but it is a shared resource managed by an administrator (DBA), who is responsible for creating and maintaining the database to serve its users.

The Database Management System is a software package that grants access to an organization’s database. It works with programs that put in place storage structures, load data, keep it secure, and allow for queries from programs and users. A DBMS can also feature programs that display data in formats readable to either programs or users, rendering the data editable, or hiding sensitive information from unrecognized or unauthenticated users.

End users are all who will use the database to do their job. They may/may not be experienced programmers or data scientists. Thus, it is important that any application featured in the database management solution works as intended and does not pose an obstacle for them to access the data they are authorized to view and utilize.

Sophisticated/casual users use queries to retrieve, insert, delete, or update data only if they have been authorized by the administrator.

Data scientists are granted access to vast amounts of data to perform analysis

Naive users are users who do not necessarily know any query language who invoke programs or select options listed in a menu to perform select operations within the database.

Hierarchical DBMS vs Network DBMS vs Relational DBMS

In a hierarchical data model, data is organized in a tree-like structure: it contains nodes that are connected by branches, and the primary node is called the root node. Each node has exactly one parent, but one parent can have many children.

In a network DBMS, data is presented using directed graphs. Thus, data nodes can be interrelated, and relationships can vary from 1:1, 1:many, or M:M

The hierarchical and network models were powerful and efficient, but they were complex and required users to understand data structures and access paths to data. In other words, they were designed to work with programs rather than being interactive for end users. Hence, the industry decided to adopt a user-friendlier solution, such as the rational data model.

The relational data model uses simple tables to organize data in an industry-wide standardized format, which could cause obstacles for specific organizations that might require more flexibility.

Thus, the entity-relationship (ER) data model was developed. It attempts to capture the meaning of the data it represents. It is most often used in the design phase for databases. It addresses the need to store and manipulate complex data that is not easy to model using the simple tables of the rational model, as well as the development of programming languages using the object-oriented paradigm, like Java.

Data warehouses are a method of capturing data consolidated from many databases. A data warehouse usually stores historical data about an organization to perform analytics.

Data lakes are repositories that store data in their natural format. The raw data is transformed when needed for use in data analytics.

**Domain of the attribute**– set of allowable values for attribute. Ex: credit would be between 0 – 150.

Attribute may have **null values**.

**Multivalued Attribute** – some attributes may have multiple values.

Ex: students may have more than one email address.

**Composite attribute** – some attributes can be decomposed into smaller elements. Ex: Address can be broken down into street, city, state, zip, country.

**Derived attribute** – value that can be calculated from other sources. Ex, age can be calculated from date\_of\_birth

**Superkey** is an attribute or set of attributes that uniquely identifies a row in a table.

Ex an attribute stuid of student table because it can uniquely identify each student.

Ex a set/combination of attributes could be {stuid, credits}

**Candidate key** is a superkey with no redundant data.

Ex: studid or social security number is but credits is not.

**Composite key** is a key of two or more attributes that uniquely identifies the row. Ex: {lastname, firstname, address}

**Primary key** is the candidate key chosen identifying entities and accessing records. Ex: stuid

**Alternate key** is the candidate key which are not selected as the primary key.

Ex: if social security number is stored on table.

**Secondary key**: attribute or set of attributes used for accessing records, but not necessarily unique.

Ex: LastName

**Foreign key** is a key that references the primary key of another table – The foreign key connects the two tables together and is used to establish the parent/child relationship.

Ex: enroll table (StuID) to student table (StuID)