# Diagram Description automatically generatedLecture 1: intro.

Database:

A collection of related data (electronic file cabinet).

* Represent some aspects of the real world.
  + Called mini world || universe of discourse (UoD).
* A logical collection of data with some meaning.
* Designed, built, and populated with data for a specific purpose.

Database Management System (DBMS):

A computerized system that helps the user create & maintain databases.

* DBMS is a general-purpose software.
  + It helps Define, Construct, Manipulate, and Share databases.
* An app accesses the database by sending queries/requests to the DBMS.
  + Can read & write data to the database.

Properties of the database approach:

* Self-describing.
  + Achieved using meta data & system catalog.
* A separation layer between programs & data & data abstraction.
  + Program data independence: The structure of data is stored separately from the access programs.
  + Data model: a type of data abstraction used to provide theoretical representation.

* Supports multiple views of the data.
* Allows sharing of data.
* Can process Multiuser transactions.

Actors on the scene.

1. Database administrator (DBA).

Responsible for:

* + Authorizing access.
  + Coordinating & monitoring.
  + Acquiring software & hardware resources.

1. Database designers.

Responsible for:

* + Identifying the data that should be stored.
  + Choosing the appropriate structure to represent & store the data.

1. End users.

The people whose jobs require access to the database.

1. System analysts.

Responsible for:

* + Determine the requirements of the end users.
  + Develop specifications for standard canned transactions.

1. Application programmers.

Responsible for:

* + Implementing specifications set by analyst.
  + Testing.
  + Debugging.
  + Documenting.
  + Maintaining transactions.

Analysts & programmers are referred to as developers || software engineers.

# Lecture 2: intro (Cont’d).

## Advantages of using the DBMS approach:

* Controls redundancy.
* The views of different user groups are integrated during the design.
* Restricts unauthorized access.
  + Has a security & authorization subsystem.
  + Is a privileged software.
* Provides persistent storage for program objects.
* Provides storage structures & search techniques for efficient Query processing.
* Provides backup & recovery.
* Provides multiple user interfaces.
* Represents complex relationships among the data.
* Enforces integrity constraints.

## History of database applications.

* Mid 1960s – 1980s.
  + Early database application using hierarchical & network systems.
  + Databases were implemented on large & expensive mainframes.
* Early 1980s.
  + Start of data abstraction & application flexibility using relational databases.
  + Didn’t use physical storage pointers || record placement to access related data records (slow).
* End of 1980s.
  + The emergence of object-oriented programming languages.
  + Object oriented databases (OODBS) were developed afterwards.
    - Because of the need to store & share complex structured objects.
* Early 1990s.
  + E-commerce emerged as a major application on the web.

## Database new applications:

* Scientific applications.
  + Storing large amounts of data generated by scientific experiments.
* Images Storage & retrieval.
  + scanned news, personal photos, satellite images, medical images.
* Videos Storage & retrieval.
  + Movies, video clips.
* Data mining applications.
  + Analyze large amounts of data.
* Spatial applications.
  + Store & analyze spatial locations of data.
* Time series applications.
  + Store info like economic data linked to certain times.

NOSQL: storing some data using SQL and some without SQL.

When not to use DBMS:

* High initial investment in hardware, software, and training.
* Overhead for providing security, concurrency control, backup & recovery.

# Lecture 3: System Concepts & Architecture

* Modern DBMS packages are modular in design & have a client/server architecture.
* Cloud computing consists of large servers that manage “big data”.

Data abstraction: the suppression of details of data organization & storage and highlighting important features.

OR hiding implementation from the user and only showing what is important.

Data Model: collection of concepts used to describe database structures.

* Specifies dynamic aspect || behavior of the database application.

## Data model categories.

* High level (conceptual) model.
  + Provide concepts close to how users view data.
  + Entity: representation of a real-world object.
  + Attribute: representation of some property describing an entity.
  + Relationship: an association (link) between two entities.
* Low level (physical) model.
  + concepts describing the details of how data is stored on computer’s storage media.
* Representational (implementation) model.
  + Concepts are easily understood by end users.
  + Hides details about storage.
  + Can be implemented directly.
  + Include:
    - Relational data model (legacy model).
    - Network & hierarchical models (used in the past).
    - Record-based models (uses record structures).
    - Object data model.
    - Self-describing data models (combines description & values).

## Table Description automatically generatedDB Schema.

* Description of a database.
* Specified during database design.
* Schema diagram: a schema represented as a diagram.
* Database state (snapshot): data in database at a particular moment.

## Important DB characteristics:

* Use of a catalog to store DB description (schema).
  + Makes it self-describing.
* Insulation of programs & data (program data & program operation).
* Support of multiple user views.

## Data independence.

* Logical independence:
  + The ability to change conceptual schema without changing external schema || application programs.
* Physical independence:
  + The ability to change internal schema without changing conceptual schema.

## DB languages:

* Data Definition Language (DDL):
  + Identifies descriptions of schema constructs & stores them into DBMS catalog.
* Storage Definition Language (SDL):
  + Used to specify internal schema.
  + Modern DBMS don’t have a language that performs this role.
* View Definition Language (VDL):
  + Specify user views & their mappings to the conceptual schema.
* Data Manipulation Language (DML):
  + Retrieval, insertion, deletion, data modification.

## DB Interfaces:

* Menu-based interfaces for web clients || browsing.
* Apps for mobile devices.
  + Provide mobile users with access to their data.
* Form-based interfaces.
  + Displays a form for each user.
* Natural language interfaces.
  + Accepts requests in natural language & tries to understand them.
* Keyword-based database search.
  + Similar to web search engines.
  + Accepts strings of natural language words & matches them with documents that include those words.
* Interface for parametric users.
  + Their users have a set of operations that are performed repeatedly.

## DB system utilities.

Tools that help DataBase Admins (DBA) manage the DB system.

* Loading.
  + Used to load existing data files into the database.
* Backup.
  + Creates a backup copy of the database.
* DB storage reorganization.
  + Used to reorganize a set of DB files.
* Performance monitoring.
  + Provides statistics to the DBA.

## DBMS Classification.

### Model based classification.

* Relational data model.
  + Systems based on this model are known as SQL-systems.
* Big data systems.
  + Called key-value storage systems || NOSQL systems.
* Tree-structured data model.
  + Called native XML DBMSs.
  + Extended Markup Language (XML).
* Various data models.
  + Document-based data model.
  + Graph-based data model.
  + Key-value data model.

### Number of users classification.

* Single-user systems: support only one user at a time.
* Multi-user systems: include the majority of DBMSs & supports multiple users.

### Number of sites of distribution classification.

* Centralized.
  + Data is stored at a single site (location).
  + Can support multiple users.
* Distributed.
  + Distributed DBMS (DDBMS) can have multiple locations connected by a network.
  + Data is replicated in case of a failure.
  + Homogeneous DDBMS: uses same DBMS at all locations.
  + Heterogeneous DDBMS: uses different DBMS software at each location.
  + Federated DBMS: DBMSs are loosely coupled with a degree of autonomy.

# Lecture 4: