**Data**

* Data stand for Discrete Analog/Array Type Accumulated. It refers to the set of information and facts that is represented and stored in computer system. It can be in various form such as text, numbers, images, audio and videos etc.

**Database**

* Database is an organized collection of data and database is controlled by DBMS.
* The main purpose of database is to operate large amount of information by storing, retrieving and managing data, and it store in the form of files or set of files on magnetic disk, tape or other secondary devices.

**Database Management System**

* DBMS is a software that is used to manage the database such as MYSQL and ORACLE.
* It provides the user interface to perform various operation like creating the database, storing data in it, updating data and lot more.
* DBMS allow us to Data definition, Data updating, Data retrieving and User Administration.
* **Characteristics** -
  + It uses the digital repository establish on server to store and manage the data.
  + It contains automatic backup and recovery procedures.
  + It contains ACID properties which maintain data in healthy state.
* **Advantages** – Controls Data Redundancy, Data sharing, recovery and backup.
* **Disadvantages** – Cost of Hardware and Software, Size, Higher Impact of failure.

**Types of Databases** –

1. **Centralized** – store data on single CPU and bound to single physical location.
2. **Distributed** – data store in different database system and connected via communication links
   1. Homogenous – database system executes on same os.
   2. Heterogeneous – database systems execute on different os.
3. **Relational** – It is based on relational data model provide by E.F Codd. It stores data in the form of table which contains rows(tuple) and columns(attributes).
4. **No SQL** – Stands for Not only SQL. It is used to store wide range of dataset. It stores data in many forms –
   1. Key-Values – storing as key-values
   2. Document Oriented – It store data in JSON like documents.
   3. Graph Database- It store data in graph like structure.
   4. Wide Column Store- store data in large columns together.

**RDMS**

* It calls Relational Database management System because it based on relational model introduced by E.F. Codd.
* Most commonly used and contains tables and each table has its primary key.

**Tables/Relation**

* Everything in relational database is stored in the form of tables/relation.
* It is collections of related data entries and contains rows and columns to store data.
* Doesn’t contain duplicates tuples.
* All attributes are atomic means stored single valued attributes.

**Row/Tuple/Record**

* contains specific information of each entry in the table.
* No two tuples are identical to each other.
* All tuples have same format in a table.

**Column/Attribute**

* contains all information associated with specific field.

**Degree –** Total no. of attributes/columns in a table.

**Cardinality –** Total no. of tuples/rows in a table.

**Domains –** possible values each attribute can contains.

**NULL VALUE –** specify that the field is blank.

**Data Integrity**

* **Entity Integrity –** specifies that there should be no duplicates rows in a table.
* **Domains Integrity –** enforced valid entry for given column by restricting the type, format, range values.
* **Referential Integrity -** specifies that rows cannot be deleted, which are used by other records.
* **User-defined Integrity -** It enforces some specific business rules defined by users.

**DBMS vs RDBMS**

**DBMS**

* DBMS applications store data as file
* Normalization is not present.
* Not apply any security with regards to data manipulation.
* Does not support distributed database.
* Deal with small data and support single users.
* E.g. - file system and XML.

**RDBMS**

* RDMS applications store data as tables.
* Normalization is present.
* Define Integrity constraint for the purpose of ACID property.
* Support distributed database.
* Deal with large amount of data and support multiple users.

**File System vs DBMS**

**File System**

* User have to writes procedures for managing the database.
* Data sharing is not easy as data is distributed in many files and may have different formats.
* Provide data details representation and storage of data.
* Not having security mechanisms.
* Not having recovery mechanism if system crashes then data is lost.
* Cheaper.
* Design by diff-diff users so having lot of duplicate data.

**DBMS**

* User not required to write the procedures.
* Data sharing is easy due to the centralized approach.
* Can provide abstract view of data by hiding internal details.
* It has good protection mechanisms.
* Provides crash recovery mechanisms.
* Costly.
* Due to the centralized database the problem of redundancy and inconsistency is controlled.

**View of Data/Three Schema Architecture**

* The major purpose of DBMS is to provide users with an abstract view of the data. That is, the system hides certain details of how the data is stored and maintained.
* To simplify user interaction with the system, abstraction is applied through several levels of abstraction.
* The main objective of three level architecture is to enable multiple users to access the same data with a personalized view while storing the underlying data only once.
* **Physical level / Internal level**
* The lowest level of abstraction describes how the data are stored.
* Low-level data structures used.
* It has Physical schema which describes physical storage structure of DB.
* Talks about: Storage allocation (N-ary tree etc), Data compression & encryption etc.
* Goal: We must define algorithms that allow efficient access to data.
* **Logical level / Conceptual level**
* The conceptual schema describes the design of a database at the conceptual level, describes what data are stored in DB, and what relationships exist among those data.
  + User at logical level does not need to be aware about physical-level structures.
  + DBA, who must decide what information to keep in the DB use the logical level of abstraction.
  + Goal: ease to use.
* **View level / External level**
  + Highest level of abstraction aims to simplify users’ interaction with the system by providing different view to different end-user.
  + Each view schema describes the database part that a particular user group is interested and hides the remaining database from that user group.
  + At the external level, a database contains several schemas that sometimes called as subschema. The subschema is used to describe the different view of the database.
  + At views also provide a security mechanism to prevent users from accessing certain parts of DB.

**Instances and Schemas**

* The collection of information stored in the DB at a particular moment is called an instance of DB.
* The overall design of the DB is called the DB schema.
* Schema is structural description of data. Schema doesn’t change frequently. Data may change frequently.
* DB schema corresponds to the variable declarations (along with type) in a program.
* We have 3 types of Schemas: Physical, Logical, several view schemas called subschemas.
* Logical schema is most important in terms of its effect on application programs, as programmers construct apps by using logical schema.
* Physical data independence, physical schema change should not affect logical schema/application programs.

**Data Models**

* Provides a way to describe the design of a DB at logical level.
* Underlying the structure of the DB is the Data Model
* a collection of conceptual tools for describing data, data relationships, data semantics & consistency constraints.
* E.g. ER model, Relational Model, object-oriented model, object-relational data model etc.

**Database Languages**

* Data definition language (DDL) to specify the database schema.
* Data manipulation language (DML) to express database queries and updates.
* Practically, both language features are present in a single DB language, e.g., SQL language.
* **DDL**

We specify consistency constraints, which must be checked, every time DB is updated.

* **DML**

Data manipulation involves

* Retrieval of information stored in DB.
* Insertion of new information into DB.
* Deletion of information from the DB.
* Updating existing information stored in DB.

Query language, a part of DML to specify statement requesting the retrieval of information.

**How is Database accessed from Application programs**?

* Apps (written in host languages, C/C++, Java) interacts with DB.
* E.g., Banking system’s module generating payrolls access DB by executing DML statements from the host language.
* API is provided to send DML/DDL statements to DB and retrieve the results.
* Open Database Connectivity (ODBC), Microsoft “C”.
* Java Database Connectivity (JDBC), Java.

**Database Administrator (DBA)**

* A person who has central control of both the data and the programs that access those data.
* Functions of DBA
* Schema Definition
* Storage structure and access methods.
* Schema and physical organization modifications.
* Authorization control.
* Routine maintenance
* Periodic backups.
* Security patches.
* Any upgrades.

**DBMS Application Architectures**

Client machines, on which remote DB users work, and server machines on which DB system runs.

* **T1 Architecture**
* The client, server & DB all present on the same machine.
* **T2 Architecture**
* App is partitioned into 2-components.
* Client machine, which invokes DB system functionality at server end through query language statements.
* API standards like ODBC & JDBC are used to interact between client and server.
* **T3 Architecture**
* App is partitioned into 3 logical components.
* Client machine is just a frontend and doesn’t contain any direct DB calls.
* Client machine communicates with App server, and App server communicated with DB system to access data.
* Business logic, what action to take at that condition is in App server itself.
* T3 architecture are best for WWW Applications.

**ACID Properties**

1. **Atomicity**
   * Transaction is atomic in nature either the transaction is completely executed or not executed at all.
   * E.g. A have 30$ and B have 100$, B have to send 10$ to A, in this transaction having two operation one is debiting from B and second is Credit in A. If the transaction is failed after first operation, then it is not atomic.
2. **Consistency**
   * Global State of the database before the transaction execution and after the execution is same/preserve.
   * Means data remains consistent before the transaction and after.
3. **Isolation**
   * No Transaction is affected by other Transaction that are run parallelly to it.
4. **Durability**
   * Changes made by transaction is preserved in database.