Week-1 Section B

AKSHAY KUMAR (S20170010006)
DEEPESH BHAGERIA(S20170010041)
HARSHIT SINGH(S20170010055)
KEVIN JOHN(S20170010070)
PARTH PATWA(S20170010106)

Basic Definitions:-

Database:- A database is a collection of related data organised in a way that can be easily accessed, managed and updated or in other words it is a collection of interrelated data which helps in efficient
retrieval,insertion and deletion of data.

Database can be software based or hardware based, with one sole purpose, storing data.

During early computer days ,data was collected and stored on tapes,which were mostly write only.

Database can be of any size and complexity.

Data:- Known facts that can be recorded and have an implicit meaning.Generally it's raw and unprocessed .

Data becomes information when it is processed.

Mini-world:- Some part of real world about which data is stored in a database is called the miniworld or the universe of discourse.For examples, student grades and transcripts at a university.

Database Management System (DBMS):- A DBMS is a software that allows creation, definition and manipulation of database, allowing users to store, process and analyse data easily.

For example: MySql,Oracle,SQL Server,IBM DB2,PostgreSQL,Amazon simpleDB

Larry Ellison, the co-founder of Oracle was amongst the first few, who realised the need for a software based Database Management System.

Database system:- The DBMS software together with the data itself.sometime,the application also included.

Impact of Databases and Database Technology

Businesses:-Banking,Insurance,Retail,Transportation,
Healthcare,Manufacturing

Example: Doctor's offices and healthcare organizations, among others, store extensive amounts of patient data for easy accessibility.The databases behind this collection of information are large and complex and secure protected data. Healthcare.gov uses NoSQL database to manage their health insurance
information.

Service industries: Financial, Real-estate, Legal, Electronic Commerce, Small businesses

Example: From the stock market to local bank, databases are abundant across the financial world. Tracking the vast amount of information behind the world's daily transactions requires extremely powerful databases. This includes financial models that analyze that data to predict future activity.

Education: Resources for content and Delivery.

More recently: Social Networks, Environmental and Scientific Applications, Medicine and Genetics

Example:Every social media platform stores reams of user information in databases used to recommend friends, businesses, products, and topics to the end user. This cross-referencing of data is immensely complex and uses highly reliable and capable database software. For example,MySQL is used in Facebook data centers.

Personalized applications: based on smart
mobile devices.

Types of Databases and Databases Applications

Traditional Applications:

- Numeric and Textual Databases

More Recent Applications:

- Multimedia Databases
- Geographic Information Systems (GIS)
- Biological and genome Databases
- Data Warehouses
- Mobile Databases
- Real-time and Active Databases

Recent Developments

• Social Networks started capturing a lot of information about people and about communications among people like posts, tweets, photos, videos in systems such as Facebook,Twitter,Linked-In

• Search Engines, Google, Bing, Yahoo: collect their own repository of web pages for searching purposes

• New Technologies are emerging from the so-called non-database software vendors to manage vast amounts of data generated on the web.

• Big Data storage systems involving large clusters of distributed computers.

• NoSQL (Non-SQL, Not Only SQL) systems

• A large amount of data now resides on the “cloud” which means it is in huge data centers using thousands of machines.

List of Databases used by Big Companies:

Big Companies usually use a variety of databases, because different databases serve different purposes.

• The king of scalability, Google, uses BigTable.

• Facebook uses Hive(Data warehouse for Hadoop,supports tables and a variant of SQL called hiveQL) and Cassandra (Multi-dimensional,distributed key-value store) for Facebook’s private messaging.

• Yahoo uses modified PostgreSQL.

• Youtube uses MySQL but they are moving to Google’s BigTable.

• Myspace uses SQL Server.

• Twitter uses (their own version of) MySQL for tweets and users, and their own special kind of graph database.

• Microsoft uses SQL Server.

• Flickr uses MySQL.

• Amazon uses DynamoDB.It’s a NoSQL database.

Bigtable:-Bigtable is a distributed, column-oriented data store created by Google Inc. to handle very large amounts of structured data associated with the company's Internet search and Web services operations.

DynamoDB:- DynamoDB is a fully managed NoSQL database service that allows to create database tables that can store and retrieve any amount of data.

What is “Big Data” ?

• "Big Data are high-volume, high-velocity, and/or high-variety information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization” (Gartner 2012)

High Volume:- It refers to the large amount of data.

High Velocity:- It refers to the speed at which new data is generated.

High Variety:- It refers to the different types of data which are used in processing.

• Bottom line: Any data that exceeds the current capability of processing can be considered as “big”.

Why is “big data” a “big deal” ?

Private Sector

• Walmart handles more than 1 million customer transactions every hour, which is imported into databases estimated to contain more than 2.5 petabytes of data.

• Twitter handles 12 Terabytes of data Per Day.

• Facebook handles 40 billion photos from its user base.

• Google handles more than 40,000 web searches per second.

• Falcon Credit Card Fraud Detection System protects 2.1 billion active accounts world-wide

Lifecycle of Data: 4 “A”s

• Acquisition:- collecting the data.

• Aggregation:- compiling of information from databases.

• Analysis:- It is a process of inspecting,cleansing, transforming and modeling data with the goal of discovering useful information.

• Application:Main Purpose is retrieving information from the database.

What a DBMS Facilitates

• Define a particular database in terms of its data types, structures, and constraints.

• Construct or load the initial database contents on a secondary storage medium.

• Manipulating the database:

Retrieval: Querying, generating reports

Modification: Insertions, deletions and updates to its content

Accessing the database through Web applications

• Processing and sharing by a set of concurrent users and application programs – yet, keeping all data valid and consistent

• Reduced Redundancy: DBMS follows Normalisation which divides the data in such a way that repetition is minimum.

• Support Multiple user and Concurrent Access:DBMS allows multiple users to work on it(update, insert, delete data) at the same time and still manages to maintain the data consistency.

• Query Language:DBMS provides users with a simple Query language, using which data can be easily fetched, inserted, deleted and updated in a database.

Other DBMS Functionalities

DBMS may additionally provide:

• Protection or Security measures to prevent unauthorized access.
• “Active” processing to take internal actions on data.
• Seamless integration into the application programming languages which makes it very easier to add a database to almost any application or website.
• Presentation and visualization of data.
• ACID Properties DBMS follows the concepts of Atomicity,Consistency,Isolation, and Durability (normally shortened as ACID). These concepts are applied on transactions, which manipulate data in a database.
• Maintenance of the database and associated programs over the lifetime of the database application.
• DBMS supports transactions, which allows us to better handle and manage data integrity in real world applications where multi-threading is extensively used.
• Multiple views DBMS offers multiple views for different users.

Drawbacks of using file systems to store data

Data redundancy and inconsistency

- Multiple file formats, duplication of information in different files

No Backup and Recovery:

- File system does not incorporate any backup and recovery of data if a file is lost or corrupted

Difficulty in accessing data

- Need to write a new program to carry out each new task

Data isolation

- Multiple files and formats

Integrity problems

- Integrity constraints (e.g., account balance > 0) become “buried” in program code rather than being stated explicitly
- Hard to add new constraints or change existing ones

Atomicity of updates

- Failures may leave database in an inconsistent state with partial updates carried out
- Example: Transfer of funds from one account to another should either complete or not happen at all

Concurrent access by multiple users

- Concurrent access needed for performance
- Uncontrolled concurrent accesses can lead to inconsistencies
- Example: Two people reading a balance (say 100) and updating it by withdrawing money (say 50 each) at the same time

Security problems

- Hard to provide user access to some, but not all, data.
- File System may lead to unauthorized access to data.

Example of a Database(with a Conceptual Data Model)

Mini-world for the example:

- Part of a university environment.

Some mini-world entities:

- Students
- Courses
- Sections (of Courses)
- (Academic) Departments
- Instructors
- Clubs
- Examination cell
- Academic cell
- Library
- Classrooms
- Management

Example of a Database(with a Conceptual Data Model)

Some mini-world relationships:

- Sections are of specific Courses
- Students take Sections
- Courses have prerequisite Courses

- Instructors teach Sections
- Courses are offered by Departments
- Students major in Departments
- Instructor belong to Department
- Instructor offer Courses.
- Students takes Courses.
- Students participates in Clubs.

Main Characteristics of the Database Approach

Self-describing nature of a database system:

- A DBMS catalog stores the description of a particular database (e.g. data structures, types, and constraints)
- The description is called meta-data*.
- This allows the DBMS software to work with different database applications.

Insulation between programs and data:
- Called program-data independence.
- Allows changing data structures and storage organization without having to change the DBMS access programs

Data Abstraction:
- The Characteristics that allow program-data inpependence and program opertaion independence is data abstraction.
- A data model is used to hide storage details and present the users with a conceptual view of the database.
- Programs refer to the data model constructs rather than data storage details.

Support of multiple views of the data:
- Each user may see a different view of the database, which describes only the data of interest to that user.

Sharing of data and multi-user transaction processing:
- Allowing a set of concurrent users to retrieve from and to update the database.
- Concurrency control within the DBMS guarantees that each transaction is correctly executed or aborted.
- Recovery subsystem ensures each completed transaction has its effect permanently recorded in the database.
- OLTP (Online Transaction Processing) is a major part of database applications. This allows hundreds of concurrent transactions to execute per second.

Database Users

Users may be divided into:

- Those who actually use and control the database content, and those who design, develop and maintain database applications (called “Actors on the Scene”)

• Those who design and develop the DBMS software and related tools, and the computer systems operators (called “Workers Behind the Scene”).

Database Users – Actors on the Scene

• Database administrators: Database Administrator or DBA is the one who manages the complete database management system. DBA takes care of the security of the DBMS, it's availability, managing the license keys, managing user accounts and access etc.

• Database designers:This user group is involved in developing and desiging the parts of DBMS.

• End-users: They use the data for queries, reports and some of them update the database content. End-users can be categorized into:

Casual: Access database occasionally when needed

Naïve: Naive or parametric end users make up a sizable portion of database end users. Their main job function revolves around constantly querying and updating the database,using standard types of queries and updates— called canned transactions.

Sophisticated: end users include engineers, scientists, business analysts, and others who thoroughly familiarize themselves with the facilities of the DBMS so as to implement their applications to meet their complex requirements.

Stand-alone users:Maintain personal databases by using ready-made program packages that provide easy-to-use menu- or graphics-based interfaces.

• System Analysts : Determine the requirements of end users, especially naive and parametric end users, and develop specifications for standard canned transactions that meet these requirements.

• Application Programmers : Implement the specifications developed by system analyst as programs; then they test,debug,document and maintain these canned transactions.

• Business Analysts: There is an increasing need for such people who can analyze vast amounts of business data and realtime data (“Big Data”) for better decision making related to planning, advertising, marketing etc.

Database Users – Actors behind the Scene

• System Designers and Implementers : Design and implement modules and interfaces as a software package.

• Tool Developers: Design and implement tools for modeling and designing databases, performance monitoring, prototyping, test data generation, user interface creation, simulation etc.

• Operators and Maintenance Personnel: They manage the actual running and maintenance of the database system hardware and software environment.

Database Systems

Data Abstraction in DBMS Source

Database systems are made-up of complex data structures. To ease the user interaction with database, the developers hide internal irrelevant details from users. This process of hiding irrelevant details from user is called data abstraction.

The three levels of Abstraction are:

Physical Level: This is the lowest level of data abstraction. It describes how data is actually stored in database. You can get the complex data structure details at this level.

Logical Level: This is the middle level of 3-level data abstraction architecture. It describes what data is stored in database.

View Level: Highest level of data abstraction. This level describes the user interaction with database system.

For example: at the physical level records are described as blocks of storage (data) in memory. At Logical level these records can be described as fields and attributes along with their data types. At view level, user just interact with system with the help of GUI and enter the details at the screen.

Schema and Instances Source

Definition of schema: Design of a database is called the schema. Schema is of three types: Physical schema, logical schema and view

• The design of a database at physical level is called physical schema, how the data stored in blocks of storage is described at this level.

• Design of database at logical level is called logical schema, programmers and database administrators work at this level

• Design of database at view level is called view schema. This generally describes end user interaction with database systems.

Definition of instance: The data stored in database at a particular moment of time is called instance of database.

To put into perspective, database schema defines the variable declarations in tables that belong to a particular database; the value of these variables at a moment of time is called the instance of that database.

Source for DDL, DML, DCL and TCL

Data Definition Language

The statements or queries used to define schema description and how data represented in database.

Example commands: Create, Alter, Drop, Comment, Truncate, Rename .

Data Manipulation Language

The statements or queries used to manipulate the data in tables

Example commands: Update, Delete, Insert and Select.

Data Control Language

The statements or queries that mainly deals with the rights, permissions and other controls of the database system.
Example commands: Grant and R.
Transaction Control Language
The statements or queries deals with transaction control within the database
Example commands: Commit, Rollback, Savepoint, Set Transaction.

SQL Source
SQL is a standard language for accessing and manipulating databases. SQL stands for Structured Query Language. This was initially called SEQUEL (Structured English Query Language) which was changed to SQL for trademark issues.
The SQL language is subdivided into several language elements, including:

- Clauses, which are constituent components of statements and queries. (In some cases, these are optional.)[19]
- Expressions, which can produce either scalar values, or tables consisting of columns and rows of data
- Predicates, which specify conditions that can be evaluated to SQL three-valued logic (3VL) (true/false/unknown) or Boolean truth values and are used to limit the effects of statements and queries, or to change program flow.
- Queries, which retrieve the data based on specific criteria. This is an important element of SQL.
- Statements, which may have a persistent effect on schemata and data, or may control transactions, program flow, connections, sessions, or diagnostics.
- SQL statements also include the semicolon (";") statement terminator. Though not required on every platform, it is defined as a standard part of the SQL

Database Design
Database design is the organisation of data according to a database model. The designer determines what data must be stored and how the data elements interrelate. Database design involves classifying data and identifying interrelationships. This theoretical representation of the data is called an ontology.
There are two ways of ensuring the database is proper and viable:

- Entity relationship model: An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database.
- Database Normalization: Database normalization is the process of structuring a relational database in accordance with a series of so-called normal forms in order to reduce data redundancy and improve data integrity

XML
A metalanguage which allows users to define their own customized markup languages, especially in order to display documents on the Internet.
An XML database is a database that stores data in XML format. This type of database is suited for businesses with data in XML format and for situations where XML storage is a practical way to archive data, metadata and other digital resources.
Database Engine

A database engine (or storage engine) is the underlying software component that a database management system (DBMS) uses to create, read, update and delete (CRUD) data from a database. Most database management systems include their own application programming interface (API) that allows the user to interact with their underlying engine without going through the user interface of the DBMS.
Example: MyISAM, InfiniDB, InnoDB, RocksDB.
Database engines have 3 important components: (Source)

- Storage Manager: A storage manager is a program module which is responsible for storing, retrieving and updating data in the database.
- Query Processor: A query processor is one of the major components of a relational database which writes and reads data to and from storage media. It is responsible for parsing and translation, Optimization and evaluation of queries.
- Transaction manager: A transaction manager is a part of an application that controls the coordination of transactions over one or more resources. The transaction manager is responsible for creating transaction objects and managing their durability and atomicity.

Database users (Source)
Database users are the one who really use and take the benefits of database. There will be different types of users depending on their need and way of accessing the database.

- Application Programmers
- Sophisticated Users
- Specialized Users
- Stand-alone Users
- Native Users

Database administrators
Administration and maintenance of Database is taken care by database admins (DBA).
Roles of a DB Admin:

- Installing and upgrading the DBMS Servers
- Design and implementation
- Performance tuning
- Migrate database servers
- Backup and Recovery
- Security
- Documentation

Database Architecture
Database architecture uses programming languages to design a particular type of software for businesses or organizations. Database architecture focuses on the design, development, implementation and maintenance of computer programs that store and organize information for businesses, agencies and institutions.
The design of a DBMS depends on its architecture.

- Centralised
- Client-Server
- Parallel
- Distributed

DAY-2

DATA MODELS

Data Model: A database model shows the logical structure of a database, including the relationships and constraints that determine how data can be stored and accessed. Individual database models are designed based on the rules and concepts of whichever broader data model the designers adopt. Most data models can be represented by an accompanying database diagram. Source

Structure and Constraints: Constraints in a relational database define its structure. Every relation has some conditions that must hold for it to be a valid relation. These conditions are called Relational Integrity Constraints. There are three main integrity constraints:

- Key Constraint : There must be at least one minimal subset of attributes in the relation, which can identify a tuple uniquely. This minimal subset of attributes is called key for that relation. If there are more than one such minimal subsets, these are called candidate keys.
- Domain Constraint: Attributes have specific values in real-world scenario. For example, age can only be a positive integer. The same constraints have been tried to employ on the attributes of a relation. Every attribute is bound to have a specific range of values. For example, age cannot be less than zero.
- Referential Integrity Constraints: Referential integrity constraints work on the concept of Foreign Keys. A foreign key is a key attribute of a relation that can be referred in other relation. Referential integrity constraint states that if a relation refers to a key attribute of a different or same relation, then that key element must exist. Source

Operations: These refer to the operations that can be performed on the databases. It includes both, basic and user defined operations; for example basic operations may include inserting, deleting, updating etc. User may define their own operations to perform specific actions on the data, like calculating average or total.

CATEGORIES OF DATA MODELS Source

Conceptual Data Model: A conceptual data model identifies the highest-level relationships between the different entities. Features of conceptual data model include:

- Includes the important entities and the relationships among them.
- No attribute is specified.
- No primary key is specified.

Physical Data Model: It represents how the model will be built in the database. A physical database model shows all table structures, including column name, column data type, column constraints, primary key, foreign key, and relationships between tables. Features of a physical data model include:

- Specification all tables and columns.
- Foreign keys are used to identify relationships between tables.
- Denormalization may occur based on user requirements.
- Physical considerations may cause the physical data model to be quite different from the logical data model.
- Physical data model will be different for different RDBMS. For example, data type for a column may be different between MySQL and SQL Server.

Implementation (representational) data models: It describes the data in as much detail as possible, without regard to how they will be physical implemented in the database. Features of a logical data model include:

- Includes all entities and relationships among them.
- All attributes for each entity are specified.
- The primary key for each entity is specified.
- Foreign keys (keys identifying the relationship between different entities) are specified.
- Normalization occurs at this level.

DATABASE SCHEMA Source

A database schema represents the logical configuration of all or part of a relational database. It can exist both as a visual representation and as a set of formulas known as integrity constraints that govern a database. These formulas are expressed in a data definition language, such as SQL. As part of a data dictionary, a database schema indicates how the entities that make up the database relate to one another, including tables, views, stored procedures, and more.
There are two main kinds of database schema:

- A logical database schema conveys the logical constraints that apply to the stored data. It may define integrity constraints, views, and tables.
- A physical database schema lays out how data is stored physically on a storage system in terms of files and indices.

DATABASE INSTANCE

It is a snapshot of a database as it existed at a particular time. Thus, database instances can change over time, whereas a database schema is usually static, since it's difficult to change the structure of a database once it is operational.

Initial Database State: Instance of the database when it is originally loaded in the system.
Valid State: Instance of the database that satisfies all constraints.

THREE SCHEMA ARCHITECTURE

This framework is used to describe the structure of a specific database system. The three schema architecture is also used to separate the user applications and physical database. The three schema architecture contains three-levels. It breaks the database down into three different categories.

Internal Level

- The internal level has an internal schema which describes the physical storage structure of the database.
- The internal schema is also known as a physical schema.
- It uses the physical data model. It is used to define that how the data will be stored in a block.
- The physical level is used to describe complex low-level data structures in detail.

Conceptual Level

- The conceptual schema describes the design of a database at the conceptual level. Conceptual level is also known as logical level.
- The conceptual schema describes the structure of the whole database.
- The conceptual level describes what data are to be stored in the database and also describes what relationship exists among those data.
- In the conceptual level, internal details such as an implementation of the data structure are hidden.
- Programmers and database administrators work at this level.

External Level

- 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.
- An external schema is also known as view schema.
- Each view schema describes the database part that a particular user group is interested and hides the remaining database from that user group.
- The view schema describes the end user interaction with database systems.

Source

DATA INDEPENDENCE

A database system normally contains a lot of data in addition to users' data. For example, it stores data about data, known as metadata, to locate and retrieve data easily. It is rather difficult to modify or update a set of metadata once it is stored in the database. But as a DBMS expands, it needs to change over time to satisfy the requirements of the users. If the entire data is dependent, it would become a tedious and highly complex job.

Metadata itself follows a layered architecture, so that when we change data at one layer, it does not affect the data at another level. This data is independent but mapped to each other.

Logical Data Independence: Logical data is data about database, that is, it stores information about how data is managed inside. For example, a table (relation) stored in the database and all its constraints, applied on that relation. Logical data independence is a kind of mechanism, which liberalizes itself from actual data stored on the disk. If we do some changes on table format, it should not change the data residing on the disk.

Physical Data Independence: All the schemas are logical, and the actual data is stored in bit format on the disk. Physical data independence is the power to change the physical data without impacting the schema or logical data. For example, in case we want to change or upgrade the storage system itself – suppose we want to replace hard-disks with SSD – it should not have any impact on the logical data or schemas. Source

DAY-3

Activities involving databases and designing are divided into 2 categories :
1)concept level design of databases - it involves schema construction for a database application
2)design of databases - it focuses on programs and interfaces that access the database. It is a part of software engineering.

Database design can be done using Entity Relationship (ER) Diagrams, UML and other design tools.
What is an ER diagram?
An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how "entities" such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs.
Example:

ER model concepts

ER Diagrams are composed of entities, relationships and attributes. They also depict cardinality, which defines relationships in terms of numbers.

Entity

It is a thing with distinct and independent existence. Examples: a customer, student, car or product. Typically shown as a rectangle.

Entity type: A group of definable things, such as students or athletes, whereas the entity would be the specific student or athlete. Other examples: customers, cars or products.

Entity set: Same as an entity type, but defined at a particular point in time, such as students enrolled in a class on the first day. Other examples: Customers who purchased last month, cars currently registered in Florida. A related term is instance, in which the specific person or car would be an instance of the entity set.

Entity categories: Entities are categorized as strong, weak or associative. A strong entity can be defined solely by its own attributes, while a weak entity cannot. An associative entity associates entities (or elements) within an entity set.

Entity keys: Refers to an attribute that uniquely defines an entity in an entity set. Entity keys can be super, candidate or primary. Super key: A set of attributes (one or more) that together define an entity in an entity set. Candidate key: A minimal super key, meaning it has the least possible number of attributes to still be a super key. An entity set may have more than one candidate key. Primary key: A candidate key chosen by the database designer to uniquely identify the entity set. Foreign key: Identifies the relationship between entities.

Attribute

It is a property or characteristic of an entity. Often shown as an oval or circle. A specific entity will have a value for each of its attributes.

Value set

Each attribute has a value set (or data type) associated with it –

e.g. integer, string, date, enumerated type.

Each simple attribute is associated with a value set.

A value set specifies the set of values associated with an attribute.

E.g., Last name has a value which is a character string of up to 20 characters.

Descriptive attribute: A property or characteristic of a relationship (versus of an entity.)

Attribute categories: Attributes are categorized as simple, composite, derived, as well as single-value or multi-value.

Simple: Means the attribute value is atomic and can't be further divided, such as a phone number.

Composite: Sub-attributes spring from an attribute. Example Name.

Derived: Attributed is calculated or otherwise derived from another attribute, such as age from a birthdate.

Multi-value: More than one attribute value is denoted, such as multiple phone numbers for a person.

Single-value: Just one attribute value. The types can be combined, such as: simple single-value attributes or composite multi-value attributes.

In general, composite and multi-valued attributes may be nested arbitrarily to any number of levels, although this is rare.
For example, PreviousDegrees of a STUDENT is a composite
multi-valued attribute denoted by {PreviousDegrees (College, Year, Degree, Field)}
Multiple PreviousDegrees values can exist
Each has four subcomponent attributes:
College, Year, Degree, Field
Relationships
How entities act upon each other or are associated with each other. Think of relationships as verbs. For example, the named student might register for a course. The two entities would be the student and the course, and the relationship depicted is the act of enrolling, connecting the two entities in that way. Relationships are typically shown as diamonds or labels directly on the connecting lines.

Recursive relationship: The same entity participates more than once in the relationship.

Cardinality
Defines the numerical attributes of the relationship between two entities or entity sets. The three main cardinal relationships are one-to-one, one-to-many, and many-many.
A one-to-one example would be one student associated with one mailing address.
A one-to-many example (or many-to-one, depending on the relationship direction): One student registers for multiple courses, but all those courses have a single line back to that one student. Many-to-many example: Students as a group are associated with multiple faculty members, and faculty members in turn are associated with multiple students.

Cardinality views: Cardinality can be shown as look-across or same-side, depending on where the symbols are shown.

Cardinality constraints: The minimum or maximum numbers that apply to a relationship.

Relationship Type: It is the schema description of a relationship which identifies the relationship name and the participating entity types. It also identifies certain relationship constraints.
Relationship Set: It is the current set of relationship instances represented in the database

Degree of a relationship: It is the number of participating entity types.
In general, more than one relationship type can exist between the same participating entity types
Constraints on Relationships
Relationship types have certain constraints that limit the possible combination of entities that may participate in relationship. There are two main types of relationship constraints, cardinality ratio, and participation(Existence Dependency Constraint).

Cardinality Ratio for Binary Relationship
Binary relationships are relationships between exactly two entities.
The cardinality ratio specifies the maximum number of relationship instances that an entity can participate in.
The possible cardinality ratios for binary relationship types are: 1:1, 1:N, N:1, M:N.

Participation Constraints and Existence Dependencies
The participation constraint specifies whether the existence of an entity depends on its being related to another entity via the relationship type.
The constraint specifies the minimum number of relationship instances that each entity can participate in.
There are two types of participation constraints:

Total(one or more)
If an entity can exist, only if it participates in at least one relationship instance, then that is called total participation, meaning that every entity in one set, must be related to at least one entity in a designated entity set.
An example would be the Employee and Department relationship. If company policy states that every employee must work for a department, then an employee can exist only if it participates in at least one relationship instance (i.e. an employee can't exist without a department)
It is also sometimes called an existence dependency.

Partial(zero)
If only a part of the set of entities participate in a relationship, then it is called partial participation.
Using the Company example, every employee will not be a manager of a department, so the participation of an employee in the "Manages" relationship is partial.
Source

Recursive Relationship
A relationship between two entities of similar entity type is called a recursive relationship. For example, in a team there are many members but one of them is a leader and others are their followers.

Weak Entity Type
A weak entity is an entity that cannot be uniquely identified by its attributes alone; therefore, it must use a foreign key in conjunction with its attributes to create a primary key. The foreign key is typically a primary key of an entity it is related to.

In entity relationship diagrams, a weak entity set is indicated by a bold (or double-lined) rectangle (the entity) connected by a bold (or double-lined) type arrow to a bold (or double-lined) diamond (the relationship).

Notation for Constraint on Relationship

Cardinality ratio is shown by numbers on relationship edge(1:1, 1:n, or m:n).
Participation constraint is shown by single line for partial and double line for total.
ER Notation:

References :
Slides

https://www.studytonight.com/dbms/components-of-dbms.php
https://www.tutorialspoint.com/dbms/dbms_overview.htm
https://www.coursehero.com/file/p2auuuv/Naive-or-parametric-end-users-make-up-a-sizable-portion-of-database-end-users/
https://www.liquidweb.com/blog/ten-ways-databases-run-your-life/
https://www.geeksforgeeks.org/database-management-system-introduction-set-1/
http://www.ctp.bilkent.edu.tr/~russell/ctp225/Chapter3b.doc
https://www.lucidchart.com/pages/er-diagrams?a=0
https://d2slcw3kip6qmk.cloudfront.net/marketing/pages/chart/seo/ERD/discovery/erd-example.svg