UNIT I – INTRODUCTION TO DBMS – 3 HRS

Data

Data is a collection of raw facts and figures which is isolated, un-organized, unrelated, and unorganized, but able to organize into useful information

Information

Information is the processed data i.e. after processing the data we can get information.

DBMS Overview

- A **database management system (DBMS)** is a software package designed to define, manipulate, retrieve and manage data in a database.
- A DBMS generally manipulates the data itself, the data format, field names, record structure and file structure. It also defines rules to validate and manipulate this data.
- A DBMS relieves users of framing programs for data maintenance. Fourth− generation query languages, such as SQL, are used along with the DBMS package to interact with a database.
- Some other DBMS examples include: MySQL, SQL Server, Oracle, dBASE, FoxPro, MS−Access etc.

Database

- A database is a collection of information that is organized so that it can be easily accessed, managed and updated. Data is organized into rows, columns and tables, and it is indexed to make it easier to find relevant information.

Why databse?

Tradition file systems have been used from beginning for managing data in a computer. In traditional file system, the data can be duplicated, they cannot be shared and security and integrity cannot be maintained very well. It is also very difficult to search for data. So, it is not easy to work on a file system where everything is kept in a file.

Component of DBMS:

1. Bit (smallest unit of memory, 0 and 1)
2. Character (collection of 8 bits, represent a alphabet, number, or any sign and symbol)
3. Field (data item/ data element / attribute)
4. Record (collection of related field information)
5. File (collection of related records)
6. Database (collection of integrated files)

Problem with file processing:

In file processing system, data needed for each user application was stored in independent data files.

Some problems are:

1. Data duplication (same data stored in several file)
2. Data dependence (organization of file depend on physical location on storage, hardware, and application software used to access those files depend on one another)
3. Lack of data integration (difficult to provide information since data are stored in independent files)

DBMS vs. Flat File

SN	DBMS	Flat File Management System
1	Multi−user access	It does not support multi−user access
2	Design to fulfill the need for small and large businesses	It is only limited to smaller DBMS system.
3	Remove redundancy and Integrity	Redundancy and Integrity issues
4	Expensive. But in the long term Total Cost of Ownership is cheap	It's cheaper
5	Easy to implement complicated transactions	No support for complicated transactions

Characteristics of DBMS

- Provides security and removes redundancy (duplication)
- Insulation between programs and data abstraction
- Support of multiple views of the data
- Sharing of data and multiuser transaction processing
- DBMS allows entities and relations among them to form tables.
- It follows the ACID concept (Atomicity, Consistency, Isolation, and Durability).
- DBMS supports multi−user environment that allows users to access and manipulate data in parallel.

Objective of DBMS

1. Eliminate redundant data.
2. Make access to the data easy for the user.
3. Provide for mass storage of relevant data.
4. Protect the data from physical harm and un−authorized systems.
5. Allow for growth in the data base system.
6. Make the latest modifications to the data base available immediately.
7. Allow for multiple users to be active at one time.
8. Provide prompt response to user requests for data.

Importance of DBMS

- A database management system is important because it manages data efficiently and allows users to perform multiple tasks with ease.
- A database management system stores, organizes and manages a large amount of information within a single software application. Use of this system increases efficiency of business operations and reduces overall costs.
- Database management systems are important to businesses and organizations because they provide a highly efficient method for handling multiple types of data.
- Some of the data that are easily managed with this type of system include: employee records, student information, payroll, accounting, project management, inventory and library books. These systems are built to be extremely versatile.
- Without database management, tasks have to be done manually and take more time. Data can be categorized and structured to suit the needs of the company or organization.
- Data is entered into the system and accessed on a routine basis by assigned users. Each user may have an assigned password to gain access to their part of the system. Multiple users can use the system at the same time in different ways.

Advantages of DBMS

- DBMS offers a variety of techniques to store & retrieve data
- DBMS serves as an efficient handler to balance the needs of multiple applications using the same data
- Application programmers never exposed to details of data representation and storage.
- A DBMS uses various powerful functions to store and retrieve data efficiently.
- Offers Data Integrity and Security
- The DBMS implies integrity constraints to get a high level of protection against prohibited access to data.
- A DBMS schedules concurrent access to the data in such a manner that only one user can access the same data at a time
- Reduced Application Development Time

Disadvantages of DBMS

- Cost of Hardware and Software of a DBMS is quite high which increases the budget of your organization.
- Most database management systems are often complex systems, so the training for users to use the DBMS is required.
- In some organizations, all data is integrated into a single database which can be damaged because of electric failure or database is corrupted on the storage media
- Use of the same program at a time by many users sometimes lead to the loss of some data.
- DBMS can't perform sophisticated calculations

Application of DBMS

SN	Sector	Use of DBMS
1	Banking	For customer information, account activities, payments, deposits, loans, etc.
2	Airlines	For reservations and schedule information.
3	Universities	For student information, course registrations, colleges and grades.
4	Telecommunication	It helps to keep call records, monthly bills, maintaining balances, etc.
5	Finance	For storing information about stock, sales, and purchases of financial instruments like stocks and bonds.
6	Sales	Use for storing customer, product & sales information.
7	Manufacturing	It is used for the management of supply chain and for tracking production of items. Inventories status in warehouses.
8	HR Management	For information about employees, salaries, payroll, deduction, generation of paychecks, etc.

Types of Database Management Systems

There are several types of databases. Here is a list of seven common types of Databases.

1. Hierarchical databases
2. Network databases
3. Relational databases

4. Object-oriented databases
5. Graph databases
6. ER model databases
7. Document databases
8. NoSQL databases

Data Models

Data Model a collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints. A data model provides a way to describe the design of a database at the physical, logical, and view levels.
The data models can be classified into four different categories:

Relational Model

- ✓ The relational model uses a collection of tables to represent both data and the relationships among those data.
- ✓ Each table has multiple columns, and each column has a unique name. Tables are also known as relations. Each table contains records of a particular type. Each record type defines a fixed number of fields, or attributes.
- ✓ The columns of the table correspond to the attributes of the record type. The relational data model is the most widely used data model, and a vast majority of current database systems are based on the relational model.
- ✓ Data is organized into tables (relations) with rows and columns. Tables can be linked using primary and foreign keys.
  Example: A university database with tables for Students, Courses, and Enrollments, where Students and Courses are linked through Enrollments.

Entity-Relationship Model.

Used for conceptual database design, it represents data using entities and relationships.. An entity is a thing or object in the real world that is distinguishable from other objects. The entity−relationship model is widely used in database design.
Example: A bookstore database with entities like Customer, Order, and Product, and relationships such as Customer places Order and Order contains Product.

Object-Based Data Model.

Object−oriented programming (especially in Java, C++, or C#) has become the dominant software−development methodology. This led to the development of an object−oriented data model that can be seen as extending the E−R model with notions of encapsulation, methods (functions), and object identity. The object−relational data model combines feature of the object−oriented data model and relational data model. Integrates object-oriented programming principles with databases, representing data as objects with attributes and methods. Supports inheritance, encapsulation, and polymorphism.
Example: A library system where Book is an object with attributes like Title and Author, and derived classes like Textbook and Novel inherit from Book.

Semistructured Data Model.

The semistructured data model permits the specification of data where individual data items of the same type may have different sets of attributes. Represents data that does not fit neatly into tables but has some organizational structure. Often used with document-oriented databases. Data is stored in formats like JSON or XML.
Example: A NoSQL database with documents storing data in JSON format, such as a Customer document with nested fields like Orders.

Historically, the **network** data model uses graph structures with nodes (representing records) and edges (representing relationships), allowing multiple parent and child relationships. and the **hierarchical** data model organizes data in a tree-like structure, where each record has a single parent and can have multiple children. preceded the relational data model. These models were tied closely to the underlying implementation, and complicated the task of modeling data. As a result, they are used little now, except in old database code that is still in service in some places.

Database Languages

A database system provides a **data-definition language** to specify the database schema and a **data-manipulation language** to express database queries and modifying the data.

Data-Manipulation Language(DML)

A **data-manipulation language (DML)** is a language that enables users to access or manipulate data as organized by the appropriate data model.

Examples of DML:

- **SELECT** – is used to retrieve data from a database.
- **INSERT** – is used to insert data into a table.
- **UPDATE** – is used to update existing data within a table.
- **DELETE** – is used to delete records from a database table.

There are basically two types:

- ✓ **Procedural DMLs** require a user to specify what data are needed and how to get those data.
- ✓ **Declarative DMLs (also referred to as nonprocedural DMLs)** require a user to specify what data are needed without specifying how to get those data.

Data-Definition Language (DDL)

DDL or Data Definition Language actually consists of the SQL commands that can be used to define the database schema.DDL is used to define and manage the structure of database objects like tables, indexes, and schemas. It includes commands for creating, altering, and deleting these objects.

Examples of DDL commands:

- **CREATE** – is used to create the database or its objects (like table, index, function, views, store procedure and triggers).
- **DROP** – is used to delete objects from the database.
- **ALTER**– is used to alter the structure of the database.
- **TRUNCATE**– is used to remove all records from a table, including all spaces allocated for the records are removed.
- **COMMENT** – is used to add comments to the data dictionary.
- **RENAME** – is used to rename an object existing in the database.

Data Control Language (DCL)

DCL includes commands such as GRANT and REVOKE which mainly deals to control access to data within the database.

Examples of DCL commands:

- **GRANT**–gives user's access privileges to database.
- **REVOKE**–remove user's access privileges given by using the GRANT command.

Transaction Control Language (TCL)

TCL commands is used to manage transactions within the database, ensuring data integrity and consistency. Examples of TCL commands:

- **COMMIT**– commits a Transaction, Saves all changes made during the current transaction.
- **ROLLBACK**– rollbacks a transaction in case of any error occurs.
- **SAVEPOINT**– sets a save point within a transaction, so you can later roll back.
- **SET TRANSACTION**– specify characteristics for the transaction.

Data Models

Data Model a collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints. A data model provides a way to describe the design of a database at the physical, logical, and view levels.

The data models can be classified into four different categories:

Relational Model

- ✓ The relational model uses a collection of tables to represent both data and the relationships among those data.
- ✓ Each table has multiple columns, and each column has a unique name. Tables are also known as relations. Each table contains records of a particular type. Each record type defines a fixed number of fields, or attributes.
- ✓ The columns of the table correspond to the attributes of the record type. The relational data model is the most widely used data model, and a vast majority of current database systems are based on the relational model.
- ✓ Data is organized into tables (relations) with rows and columns. Tables can be linked using primary and foreign keys.
  Example: A university database with tables for Students, Courses, and Enrollments, where Students and Courses are linked through Enrollments.

Entity-Relationship Model.

Used for conceptual database design, it represents data using entities and relationships.. An entity is a thing or object in the real world that is distinguishable from other objects. The entity−relationship model is widely used in database design.

Example: A bookstore database with entities like Customer, Order, and Product, and relationships such as Customer places Order and Order contains Product.

Object-Based Data Model.

Object−oriented programming (especially in Java, C++, or C#) has become the dominant software−development methodology. This led to the development of an object−oriented data model that can be seen as extending the E−R model with notions of encapsulation, methods (functions), and object identity. The object−relational data model combines feature of the object−oriented data model and relational data model. Integrates object-oriented programming principles with databases, representing data as objects with attributes and methods. Supports inheritance, encapsulation, and polymorphism.

Example: A library system where Book is an object with attributes like Title and Author, and derived classes like Textbook and Novel inherit from Book.

Historically, the **network** data model uses graph structures with nodes (representing records) and edges (representing relationships), allowing multiple parent and child relationships. and the **hierarchical** data model organizes data in a tree-like structure, where each record has a single parent and can have multiple children. preceded the relational data model. These models were tied closely to the underlying

implementation, and complicated the task of modeling data. As a result, they are used little now, except in old database code that is still in service in some places.

RDBMS

A **relational database** is a type of database. It uses a structure that allows us to identify and access data in relation to another piece of data in the database. Often, data in a relational database is organized into tables.

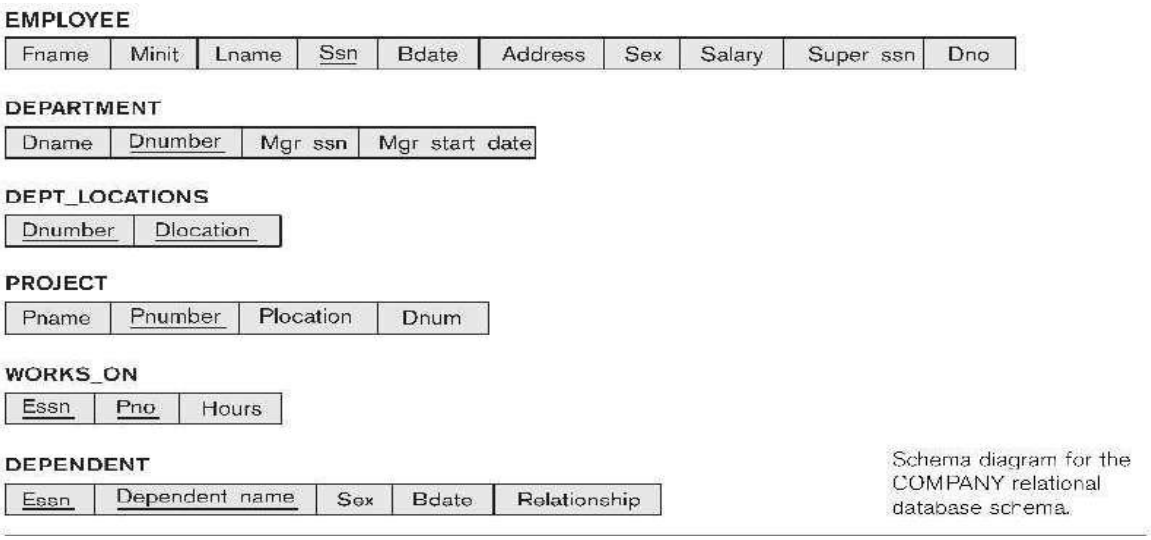
CUSTOMERS	
customer_id	customer_name
101	John Doe
102	Bruce Wayne

ORDERS			
order_id	customer_id	order_date	amount
555	101	12/24/09	$156.78
556	102	12/25/09	$99.99
557	101	12/26/09	$75.00

Database Schema

- ✓ A **database schema** is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. A database schema defines its entities and the relationship among them. It's the database designers who design the schema to help programmers understand the database and make it useful.
- ✓ A database schema can be divided broadly into two categories −
- ✓ **Physical Database Schema** − This schema pertains to the actual storage of data and its form of storage like files, indices, etc. It defines how the data will be stored in a secondary storage.
- ✓ **Logical Database Schema** − This schema defines all the logical constraints that need to be applied on the data stored. It defines tables, views, and integrity constraints.

Schema Diagram for Employee Database



Schema diagram for the COMPANY relational database schema.

Enitity, attribte, relationship

Entity: An entity is a name of 'thing' or 'person' or 'object' in the real-world that is distinguishable from all other object.

For example: book, student, employee, account, etc

Attribute: The descriptive property possed by each member of an entity set.
For example: student_id, name, roll_no, address, grade, image, marks, etc

The value for each attribute are defined in terms of properties such as data_type, domain,and default value.

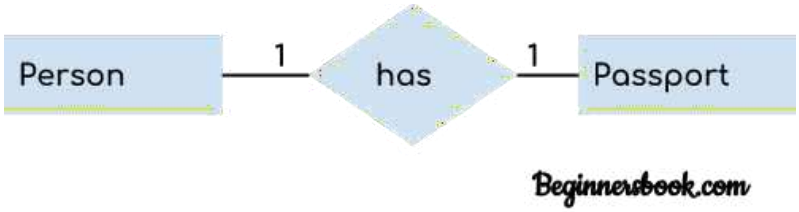
Data type: what type of data is stored in that attribute, like, integer, string, picture, etc
Domain: Domain for each attribute is a set of permitted values such as the mark of a subject cannot have a value greater than the full marks.
Default value: The value that will be recorded if not specified by the user such as 0 can be the default marks in case not specified.

Relationship
The association between entities is called relationship.
For example: student takes a class, a professor teaches a class, and a department employs a professor, so on.

Type of relationship:
1. 1 – 1 relationship
2. 1 – M relatioship
3. M – 1 relationship
4. M – M relationship

a) One to One

- ✓ When a single instance of an entity is associated with a single instance of another entity then it is called one to one relationship. For example, a person has only one passport and a passport is given to one person.



b) One to Many

- ✓ When a single instance of an entity is associated with more than one instances of another entity then it is called one to many relationships. For example – a customer can place many orders but an order cannot be placed by many customers.

c) Many to One

- ✓ When more than one instances of an entity is associated with a single instance of another entity then it is called many to one relationship. For example – many students can study in a single college but a student cannot study in many colleges at the same time.

d) Many to Many

- ✓ When more than one instances of an entity are associated with more than one instances of another entity then it is called many to many relationships.
- ✓ For example, a can be assigned to many projects and a project can be assigned to many students.

Keys / identify row

A key is an attribute that is used to identify a particular record in a database. A key may be composed of more than one attribute.

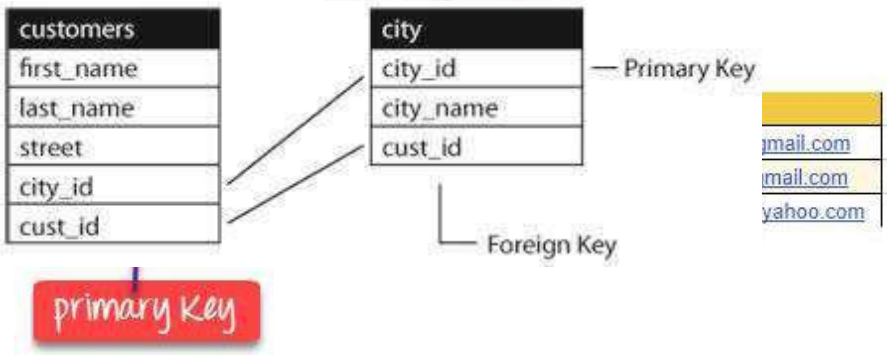
- **Super Key**: A set of one or more attributes that can uniquely identify a tuple/row in a table.

Example:

EmpSSN	EmpNum	Empname
9812345098	AB05	Shown
9876512345	AB06	Roslyn
199937890	AB07	James

In the above−given example, **EmpSSN** and **EmpNum** are **superkeys**.

- **Candidate Key**: A super key with no repeated attribute is called **candidate key**. Primary key should be selected from the candidate keys. Every table must have at least a single candidate key.
- **Primary Key**: A specific candidate key chosen to uniquely identify tuples in a table.
- **Foreign Key**: An attribute in one table that is a primary key in another table, used to establish relationships between the tables.



- **Alternate key**: In the case of two or more candidate keys, only one of them serves as the primary key. An alternate key is the candidate key which is not used as primary key.

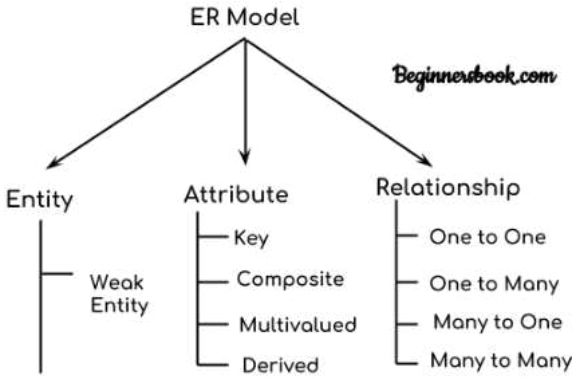
ER diagram

StudID, Roll No, Email are qualified to become a primary key. But since StudID is the primary key, Roll No, Email becomes the alternative key.

StudID	Roll No	First Name	LastName	Email
1	11	Tom	Price	abc@gmail.com
2	12	Nick	Wright	xyz@gmail.com
3	13	Dana	Natan	mno@yahoo.com

The **ER model** defines the conceptual view of a database. It works around real−world entities and the associations among them. At view level, the ER model is considered a good option for designing databases.

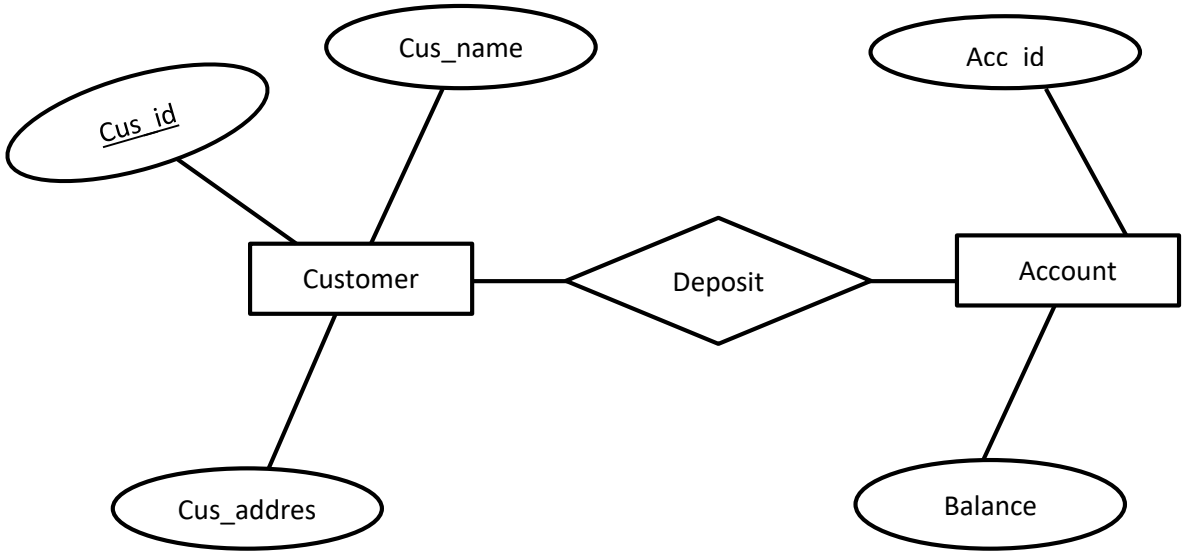
Components of a ER Diagram



Components of ER Diagram

- ✓ **Rectangle** : Represents Entity sets.
- ✓ **Ellipses or oval** : Attributes
- ✓ **Diamonds** : Relationship Set
- ✓ **Lines** : They link attributes to Entity Sets and Entity sets to Relationship Set

Example: An ER diagram for relation customer deposits amount in his account

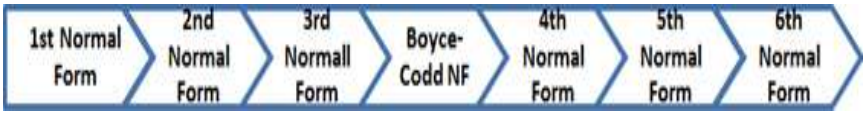


Normalization:

Database normalization, or simply normalization, is the process of decomposing a big table into much smaller and simple tables in order to reduce data redundancy and improve data integrity.

- ✓ **Data redundancy** is the existence of data that is additional to the actual data.
- ✓ **Data integrity** is the maintenance of, and the assurance of the accuracy and consistency of, data over its entire life−cycle.

The most important and widely used normal forms are:



Advantages of Normalization

- A smaller database can be maintained as normalization eliminates the duplicate data. Overall size of the database is reduced as a result.
- As databases become lesser in size, the passes through the data becomes faster and shorter thereby improving response time and speed.
- Avoid redundant fields or columns.
- More flexible data structure i.e. we should be able to ad new rows and data values easily
- Easier to maintain data structure i.e. it is easy to perform operations and complex queries can be easily handled.

Disadvantages of Normalization

- Database systems are complex, difficult, and time−consuming to design.
- Substantial hardware and software start−up costs.

- Initial training required for all programmers and users.
- On Normalizing the relations to higher normal forms i.e. 4NF, 5NF the performance degrades.
- It is very time consuming and difficult process in normalizing relations of higher degree.
- Careless decomposition may lead to bad design of database which may leads to serious problems.

First Normal Form(1NF)

A relation or table is said to be in 1NF

1. Each column in a table should not contain multiple values.
2. Attribute Domain should not change
3. Unique name for Attributes/Columns
4. Order doesn't matters when data is store.

How to make 1NF?

If any row data is repeated again and again in same table, then such attributes are removed either into separate table or decomposed into different rows.

Table : Student (Un-normalized table)

Roll_no	Name	Class	Subject	Marks
15	Ram	XI	English Math Nepali	50 95 60
12	Hari	XII	English Computer Accountancy	52 70 85

Table: Student After 1NF Normalized Table

Roll_no	Name	Class	Subject	Marks
15	Ram	XI	English	50
15	Ram	XI	Math	95
15	Ram	XI	Nepali	60
12	Hari	XII	English	52
12	Hari	XII	Computer	70
12	Hari	XII	Accountancy	85

Second Normal Form(2NF)

A relation is said to be second normal form:

1. If it is 1NF
2. Each attribute is functionally dependent on the entire primary key.
3. Remove **partial key dependencies** i.e. each attribute in the table must depend on whole key, not just the part of it.

For example: in above table, primary key is the combination of "**Roll_No + Class**".

Name depends on **Roll_No** and **Class**
Subject depends only on **Class**, not on **Roll_No**
Marks depends on **Subject** and **Name**

Table : Student

Name	Roll_No	Class
Ram	15	XI
Hari	12	XII

Table: Subjects

Subject	Class
English	XI
Math	XI
Nepali	XI
English	XII
Computer	XII
Accountancy	XII

Table: Marks

Name	Subject	Marks
Ram	English	50
Ram	Math	95
Ram	Nepali	60
Hari	English	75
Hari	Compter	25
Hari	Computer	45

3rd Normal Form
A relation is said to be 3rd normal form:

1. If it is in 2nd NF
2. All attributes that doesnot dependent upon primary key must be eliminated.
3. It doesnot contain any transitive dependency on the primary key.

A transitive dependency is the one in which, among 3 attributes A, B and C , if $A \rightarrow B$, $B \rightarrow C$ then $A \rightarrow C$.

Table: Student

Student_Id	Name	Roll_No	Class_Id
1	Ram	15	11
2	Hari	12	12

Table: Subjects

Student_Id	Subject
101	English
102	Math
103	Nepali
201	English
202	Computer
203	Accountancy

Table: Marks

Student_Id	Subject_ID	Marks
1	101	50
1	102	95
1	103	60
2	201	52
2	202	70
2	203	85

Table: Class

Class_Id	Class
11	XI
12	XII

Second example of Normalization process

Table: Un-normalized Data

E_code	Dept	Proj_code	Hours
E101	Systems	P27 P51 P20	90 101 60
E305	Sales	P27	109
E508	Admin	P51 P27	NULL 72

Table: A table in 1NF

E_code	Dept	Proj_code	Hours
E101	Systems	P27	90
E101	Systems	P51	101
E101	Systems	P20	60

E305	Sales	P27	109
E508	Admin	P51	NULL
E508	Admin	P27	72

Table: A table in 2NF

E_code	DEPT
E101	Systems
E305	Sales
E508	Admin

E_code	Proj_code	Hours
E101	P27	90
E101	P51	101
E101	P20	60
E305	P27	10
E508	P51	NULL

Consider the following table where primary key here is E_code. The attribute Dept is dependent on E_code. The attribute Dept_head is dependent on Dept. Dept_head is the code of department head. Since there is an indirect dependence on the primary key.

E_code	Dept	Dept_head
E101	Systems	E901
E305	Sales	E906
E402	Sales	E906
E508	Admin	E908
E607	Finance	E909
E608	Finance	E909

Therefore above table is reduce to following two tables:

Table: In 3NF

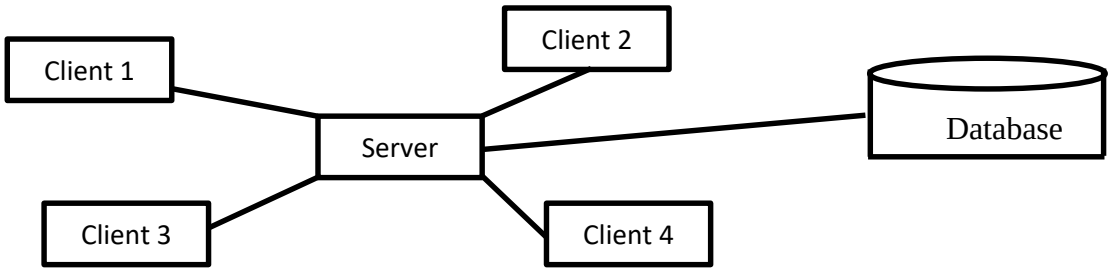
E_code	Dept
E101	Systems
E305	Sales
E402	Sales
E508	Admin
E607	Finance
E608	Finance

Dept	Dept_head
Systems	E901
Sales	E906
Admin	E908
Finance	E909

Centralized Database Vs. Distributed Database

Centralized Database System

Centralized database systems are those that run on a single computer system. A compter system may be a single user or multi user. In this system, users have access to computer resources via hundreds of remote communication devices including on-line terminals used to input data and printers to obtain reports. Highly skilled, technically trained specialists are required for this processing. This processing mode offers control of data processing expenses and also better security, control, and protection of data.



Advantages:

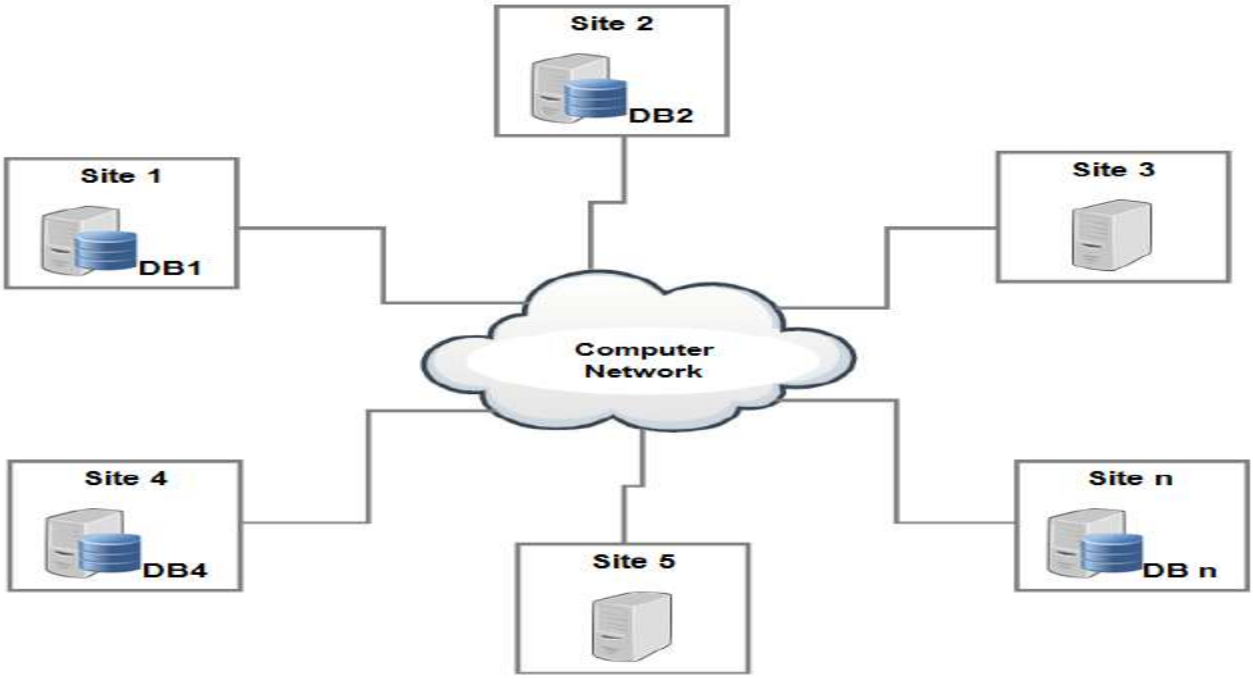
- It is simple to implement and manage.
- It has less or no risk of data loss.
- It has a low cost of implementation and operation.
- It is suitable for a small organization with limited data and users.
- High security can be provided.

Disadvantages:

- Failure of the server will completely down the whole computer svstem.
- It is not reliable as a distributed database.
- The performance and service will degrade when the nunnber of users increases.
    - It is not appropriate for big organizations, which have a large mutliper of users and computers.

Distributed Database system

In a distributed database system, the database is stored on several computers. Distributed databases are databases located at different places. They facilitate easy access to data from any location. It has the problem of control of security, increased the problem of control of the database, and needs more computer resources. It can manage people with functional knowledge. This mode of processing incurs more expense with limited security, control, and protection of data.



Advantages:

1. Faliure of a site doesnot effect the whole computer system.
2. It is much reliable as compared to the centralized database system.
3. The performance and service are better than centralized database system.
4. It is suitable for a big organization with a large amount of data, database users, and computers.
5. Due to the use of multiple servers, it can support a large number of users and computers at the same time.

Disadvantage

1. It is complex to implement and manage.
2. It has a high risk of data theft, hacking, etc.
3. It has a high cost of implementation and operation.

Difference between

Centralized Databse System	Distributed Database System
Data reside in a single location.	The data reside in several locations.
Files are kept on the basis of the location of disk drives and names.	Files are kept on the basis of names rather than location.
It does not contain several sites or nodes (i.e. does not contain several computers)	It contains several computers and communicates with one another through communication media
Once database system fails, it doesnot operates.	If one site fails, the remaining sites may be able to continue operating.
No risk of data loss	High risk of data theft, hacking, etc
Suitable for a single organization like school, college, industries, etc. of a location.	Suitable for a large organization spread in different geographical locations.
Low cost.	High cost.

Data security

Security refers to the protection of data against unauthorized access. Not every user of the database system should be able to access all the data. For example, in a university database system, an accountant should be able to see only part of the database that has information about parent payment system. They do not need access to information about teacher system. Therefore security mechanism of a DBMS makes sure that only authorized users are given access to the data in the database. The means of ensuring that data is kept safe from corruption and that access to it is suitably controlled. It helps to ensure privacy and protect personal nformation.

Data security is concerned with the protection of data from the unauthorized access, alteration, or destruction of data. Misuse of data can be categorized as intentional or accidental.

Accidental loss of data may result from:

1. Crashes during transaction processing.
2. Logical errors in the program.
3. Due to the distribution of data over several computers.

Intentional loss of data may result from

1. Unauthorized reading of data.
2. Unauthorized modification of data.
3. Unauthorized destruction of data.

To protect the database, we must take security measures at several levels:

1. **Physical level:** The place or the site containing the computers systems must be physically secure against external intruders. Software security is not enough for the security of a system, physical security is a must.
2. **Human:** Users must be authorized carefully to reduce the chance of such users giving access to an intruder.
3. **Operating system**: Even though the database system is highly secured, weakness in the operating system may provide the pathway for unauthorized access.
4. **Network:** Since almost all database systems allow access through terminals or networks, software-level security within the network software is important as physical security.
5. **Database system:** Security should also be provided by the database system itself, some database users may be allowed to issue queries but may be forbidden to modify data.

Secure guideline of databse:

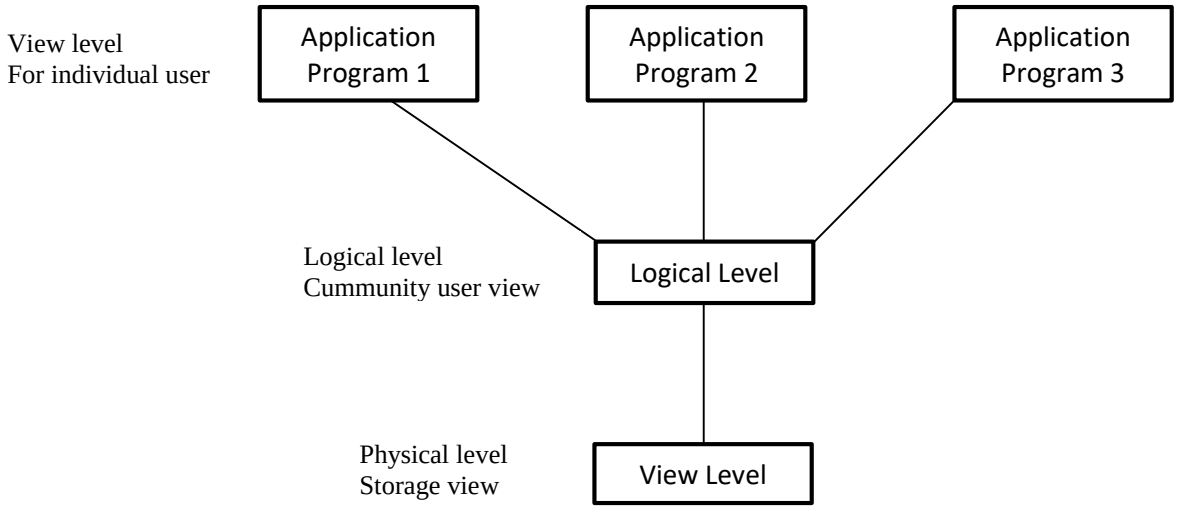
- Always maintain a log of your backup schedule.
- Occasionally store backups in a separate location.
- Spread your database across several disks.
- Always consider your existing hardware configuration and system workload.
- Determine the volatility and value of your data.
- Verify the integrity of your database regularly.
- Verify the integrity of your backup media after each backup.

Technique for achieve different levels of security:

1. Cryptographic tools can be used to defend the data in transit between systems, reducing the probability that data can be intercepted or modified in between.
2. Strong Authentication techniques can be employed to ensure that the two end-points taking part in transition are who they say they are.
3. Secure crypto-processors can be used to leverage physical security techniques into protecting the security of the computer system.
4. Chain of Trust techniques can be used to attempt to ensure that all the software loaded has been certified as authentic by the system designers.
5. Mandatory Access Control can be used to ensure that privileged access is withdrawn when privileges are retracted.
6. Capability & Access Control List techniques can be used to ensure privilege separation.
7. Anti-virus software deletes or disables viruses on the computer, protecting against them.
8. Firewalls, the hardware and/or software components protect the connputer system from intruders, not allowing anything to enter into it without correct markings.
9. Intrusion-detection System is used to detect people that are on the network but doing things that can be hazardous to the whole systern.
10. Backups are a way of securing the information by keeping another copy of all the required files and documents.

Data abstraction

A major purpose of a database system is to provide users with an abstract view of the data. That is the system hides certain details of how the data are stored and maintained. Since many database-systems users are not computer trained, developers hide the complexity from users through several levels of abstraction.



1. **Physical level:** It is the lowest level. It describes how the data are actually stored. At this level, low-level data structures are described in detail.
2. **Logical level:** It is the next higher level. It describes what data are stored in the databas and what

relationship exists among those data. The logical level of abstraction is used by the database administrator, who must decide what information is to be kept in the database.
3. **View level:** It is the highest level of data abstraction. It describes only a part of the entire database. Users interact with the system through the view level.

Database Administrator (DBA)

The data administrator is the person who makes the strategic and policy decisions regarding the data of the enterprise, and the database administrator (DBA) is a person who provides the necessary technical support for implementing those decisions. Thus, the DBA is responsible for the overall control of the system at a technical level.

The functions (Role) of DBA are:

1. **Defining the internal schema:** The DBA must define how the data is to be represented in the stored database. This process is usually referred to as physical database design.
2. **Defining the external schema:** The DBA must also decide exactly what information is to be stored in the database. This process is usually called as a logical (sometimes conceptual) database design.
3. **Liaising with users**: It is the business of the DBA to liaise with users to ensure that the data they require is available, and to write the necessary external schemas.
4. **Defining security and integrity rules:** The DBA must define security rules for the protection of data against unauthorized users. S/he must also define integrity rules for checking the accuracy or validity of data.
5. **Defining backup and recovery procedures:** The DBA must define backup and recovery procedures because the damage to any portion of the system causes an error.
6. **Monitoring performance and responding to the changing requirements:** The DBA must monitor performance and make the appropriate adjustments as the requirements change.

Data Integrity

Data integrity or integrity constraint is a rule that restricts the values that may be present in the database. Integrity constraints ensure that changes made to the database by authorized users do not result in a loss of data consistency. Integrity constraints guard against accidental damage to the database by ensuring that authorized changes to the database do not result in a loss of data consistency.

The different types of data integrity are:

1. **Entity integrity:** Entity integrity is the rule that no column that is part of the primary key may accept null values. Entity integrity guarantees that each record will indeed have its own identity. In other words, entity integrity prevents the primary key from accepting null values and ensures that one record can be distinguished from another.
2. **Referential integrity:** The referential integrity rule states that if table A contains a foreign key that matches the primary key of table B, then values of this foreign key either must match the value of the primary key for some row in table B or must be null. In other words, referential integrity keeps us from making data-entry mistakes. It says, essentially that all the information in the two fields should match.
3. **Domain integrity:** Domain constraints specify the set of possible values that may be associated with an attribute. Such constraint may also allow the use of null values for particular attributes. Domain constraints are the most elementary form of integrity constraint. They are tested easily by the system, whenever a new data item is entered into a database.

Lab Work in SQL:

SQL stands for Structured Query Language. It is a database language. It is used to access and manipulate databases.

SQL statements can instruct the server to perform certain operations:

1. Data query: requesting specific information from the existing database.
2. Data manipulation: adding, deleting, changing, sorting, and other operations to modify the data, the values, or the visuals.
3. Data identity: defining data types, e.g. changing numerical data to integers. This also includes defining a schema or the relationship of each table in the database.
4. Data access control: providing security techniques to protect data, this includes deciding who can view or use any information stored in the database.

MySQL

MySQL is one of the most popular RDBMS software. It implements a client-server model. MySQL creates a database for storing and manipulating data, defining the relationship of each table. Clients can make requests by typing specific SQL statements on MySQL. The server application will respond with the requested information and it will appear on the clients' side. MySQL is popular due to the reasons like:

- Flexible and easy to use.
- High performance.
- An industry standard.
- Secure.

The SQL statement for creating, dropping, and altering database and table

XAMPP provides a GUI environment to perform any operations on the database. However, it also provides an option to use SQL statements to perform any operations. SQL statements are used in the SQL menu in phpMyAdmin. The SQL statements used in XAMPP also work well with most of the databases.

Creating a database:

Syntax: CREATE DATABASE databasename;

Example: CREATE DATABASE School;

Dropping the database (deleting the database):

Syntax: DROP DATABASE databasename;

Example: DROP DATABASE School;

Creating a table:

Syntax: CREATE TABLE table_name (column1 datatype, column2 datatype, column3 datatype, ...);
Example: CREATE TABLE Students (StudentID int, FName varchar(255), LName varchar(255), Address varchar(255), Class varchar(255));

Altering table (adding, deleting, or modifying columns in an existing table)

1. **Adding column**

Syntax: ALTER TABLE table_name ADD column_name datatype;

Example: ALTER TABLE Students ADD Email varchar(255);

2. **Deleting column**

Syntax: ALTER TABLE table_name DROP COLUMN column_name;

Example: ALTER TABLE Students DROP COLUMN Email;

3. **Modifying column (changing the data type of a column in a table)**

Sytnax: ALTER TABLE table_name MODIFY COLUMN column_name datatype;

Example: ALTER TABLE Students MODIFY COLUMN Class int;

4. **Deleting table**

Syntax: DROP TABLE table_name;

Example: DROP TABLE Students;

SQL statement to insert, select, update and delete data

1. **Inserting data**

Syntax: INSERT INTO table_name (column1, column2, column3, ...) VALUES (value1, value2, value3, ...);

Example: INSERT INTO Students (StudentID, FName, LName, Address, Class) VALUES ('101', 'Ram', 'Sharma', 'Pokhara', '7');

2. **Selecting data (selecting data from a database)**

Syntax: SELECT column1, column2, ... FROM table_name;

Example:

- SELECT * FROM Students; [This will select all the columns from the table Students]
- SELECT FNAME, LNAME FROM Students; [This will select only the First Name and Last Name from the table Students.]

3. **Selecting data using conditions**

Syntax: SELECT column1, column2, ... FROM table_name WHERE condition;

Example: SELECT FNAME, LNAME FROM Students; WHERE Address='Pokhara';

[This will select only the First Name and Last Name from the table Students with Address=Pokhara]

4. **Updating data (modifying the existing records in a table)**

Syntax: UPDATE table_name SET column1 = value1, column2 = value2,... WHERE condition;

[Where condition is optional]

Example: UPDATE Students SET Address= 'Janakpur' WHERE StudentID = 101;

Deleting data

Syntax: DELETE FROM table_name WHERE condition;

Example: DELETE FROM Students WHERE FName='Ram';