Unit 1: Introduction to Database Systems
Part 1

Introduction

Introduction

- Data are the known facts that can be recorded and have an implicit meaning. Ex: name, regno, mobileno, age, year etc.
- A database is a collection of logically related data.
- Properties of a database:
  - It represents some aspect of real world.
  - It is a logically coherent collection of data with some inherent meaning.
  - It is designed, built and populated with data for a specific purpose.
- A database can be of any size and complexity.

Ex:1: A university database

- Entities such as students, faculty, courses
- Relationships between entities such as students' enrollment in courses, faculty teaching courses

Ex2: A Hospital database

- Entities such as doctors, patients, nurses, wards
- Relationships between entities such as doctors visiting patients, patients in rooms.

- A Database Management System (DBMS) is a software package (collection of programs) that enables users to create, store and maintain a database.
- The DBMS is a general purpose software system that facilitates the process of defining, constructing, manipulating, and sharing databases among various users and applications.

Ex: MySQL, Oracle DBMS, MS SQL Server, MS Access etc.

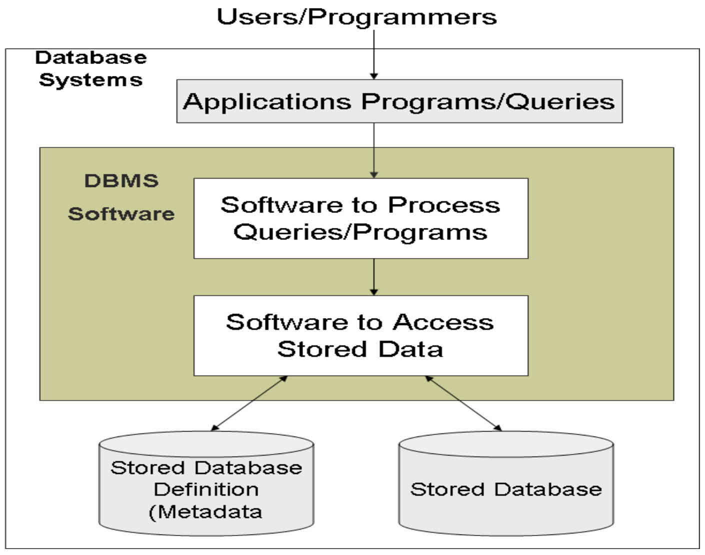
Functionalities of DBMS

- A DBMS is a *general purpose* software system facilitating each of the following (with respect to a database):
- **Defining a database**
  - specifying data types, structures, and constraints of the data to be stored in the database.
- **Constructing the database**
  - the process of storing the data on some storage medium (e.g., magnetic disk) that is controlled by the DBMS
- **Manipulating the database**
  - querying the database to retrieve specific data, updating the database to reflect changes, and generating reports
- **Sharing a database**
  - allowing multiple users and programs to access the database "simultaneously"
- **Maintaining the database**
  - allowing the system to evolve as requirements change over time

Protection includes system protection and security protection.

- **System protection**
  - preventing database from becoming corrupted when hardware or software failures occur
- **Security protection**
  - preventing unauthorized or malicious access to database.

An Example of Database based Application:


Users/Programmers
Database Systems
Applications Programs/Queries
DBMS Software
Software to Process Queries/Programs
Software to Access Stored Data
Stored Database Definition (Metadata
Stored Database

Steps in Database Design

1st Step: Requirements analysis

- Database designers interview prospective database users to understand and document their data requirements
- Two types of requirements
  - Functional requirements
  - Database requirements

2nd Step: Conceptual design

- Create conceptual schema using high level conceptual data model such as ER modeling
- Conceptual schema is a description of the data requirements of the users and includes entity types, relationships, constraints.
- Conceptual schema do not include implementation details and can be used to communicate with non technical users.
- It can be used to ensure that all users data requirements are met and no conflict exists

3rd Step: Logical design or data model mapping

- Conceptual schema is transformed from the high level data model into the implementation data model.
- Actual implementation of the database using a commercial DBMS

4th Step: Physical design

- Internal storage structures, access paths, and file organizations for the database files are specified
- In parallel with all steps, application programs are designed and implemented as database transactions corresponding to the high level transaction specifications.

File System vs a DBMS

DBMS	**File System**
DBMS is a collection of data. In DBMS, the user is not required to write the procedures.	File system is a collection of data. In this system, the user has to write the procedures for managing the database.
DBMS gives an abstract view of data that hides the details.	File system provides the detail of the data representation and storage of data.
DBMS provides a crash recovery mechanism, i.e., DBMS protects the user from the system failure.	File system doesn't have a crash mechanism, i.e., if the system crashes while entering some data, then the content of the file will lost.
DBMS provides a good protection and security mechanism.	It is very difficult to protect a file under the file system.
DBMS contains a wide variety of sophisticated techniques to store and retrieve the data.	File system can't efficiently store and retrieve the data.

DBMS takes care of Concurrent access of data using some form of locking.	In the File system, concurrent access has many problems like redirecting the file while other deleting some information or updating some information.
Data is redundancy is less	Data redundancy is more
Data is consistent	Data is inconsistent

Characteristics of a Database system

- Self-describing nature of a database system
- Insulation between programs and data, and data abstraction
- Support of multiple views of the data
- Sharing of data and multiuser transaction processing

1. Self describing nature of a Database system

- A complete description of the database structure and constraints is stored in the DBMS catalog called metadata.
- A general purpose DBMS Software package is not written for a specific database application; it must refer to the catalog to know the structure of the files in a specific database.

2. Insulation between programs and data, and data abstraction

In DBMS

- DBMS provides an abstract view of the data that hides the details. A single repository of data is maintained that is used by various users.
- A DBMS provides users with a conceptual representation of data that does not include the details of how data is stored and how the operations are implemented.
    - Program-data independence
    - Program-operation independence
- Insulation between programs and data, and data abstraction
- **Program-data independence:**
    - The definition and description of the database is stored in the catalog (known as metadata); to change description of databases, database software is not changed. Insulation between programs and data, and data abstraction
- **Program-operation independence**:
    - In object-oriented relational system, an operation is specified in two parts:
    - The interface of an operation name and data types of its arguments.
    - The implementation of the operation is specified separately and can be changed without affecting the interface.

    - User application programs can operate on the data by invoking on these operations through their names and arguments, regardless of how the operations are implemented.

Support of multiple views of the data

- A database has many users, different users may have different requirements i.e. require a different view of the database.
- A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored.

Sharing of data and multiuser transaction processing

- A multiuser DBMS must allow multiple users to access the database at the same time, if data for multiple related applications is integrated and maintained in a single database
- DBMS includes concurrency control software to ensure that several users updating the same data, should update in a controlled manner so that the result of the updates is correct.

Actors on the Scene

- Database Administrator
- Database Designers
- End Users
    - Casual end users
    - Naive/Parametric end users
    - Sophisticated end users
    - Stand-alone users
- System Analysts and Application Programmers

Database Administrator (DBA)

- DBA is the chief administrator, who oversees and manages the database system.

Duties include

1. Security & Authorization
    - authorizing users to access the database, coordinating/monitoring its use
2. Data availability and recovery failures
    - restoring the data, maintaining the log files
3. Database tuning
    - acquiring hardware/software for upgrades, etc.
4. Monitors all the data base resources.
5. Troubleshoot problems.

Database Designers

- They are responsible for identifying the data to be stored and for choosing appropriate structures to represent and store this data.

- They also define views for different categories of users. The final design must be able to support the requirements of all the user sub-groups.

End-users

- These are persons who access the database for querying, updating, and report generation. They are main reason for database's existence!

1. Casual end users

- They use database occasionally, needing different information each time.
- They use query language to specify their requests.
- They are middle- or high-level software literate managers.

2. Naive/Parametric end users:

- Typically the biggest group of users; frequently query/update the database using standard front end based application that have been carefully programmed and tested in advance.
- Examples:
  - bank tellers check account balances, post withdrawals/deposits
  - reservation clerks for airlines, hotels, etc., check availability of seats/rooms and make reservations.

3. Sophisticated end users

- They are engineers, scientists, business analysts who implement their own applications to meet their complex needs.

4. Stand-alone users

- They use "personal" databases,
- A tax program user that creates his or her own internal database by using ready-made package.

System Analysts and Application Programmers

- They develop the packages that facilitate the data access for end users using the host language DBMS packages, other DBMS related software tools like report writers.

Workers behind the Scene

Database system designers & implementers

- They design and implement the DBMS modules and interfaces as a software packages.
- DBMS consists of many components e.g. for implementing the catalog, processing query language, processing the interface, accessing and buffering data, controlling concurrency and handling data recovery and security

Tool Developers

- They design and implement tools – the software packages that facilitate database modeling design.
- These tools include packages for database design, performance monitoring, natural language or graphical interfaces, prototyping, simulation, and test data generation.

- Tools can be purchased separately which are developed by different vendors.

Operators and maintenance personnel

- They are responsible for the actual running and maintenance of the hardware and software environment for the database system.

Advantages of using the DBMS approach

1. Controlled Redundancy

- In the file processing approach, each user defines and implements the files needed and software applications to manipulate those files.
- Various files are likely to have different formats and programs may be written in different languages and same information may be duplicated in several files.
- Data redundancy leads to
    - wasted storage space,
    - duplication of effort (when multiple copies of a datum need to be updated),
    - a higher likelihood of the introduction of inconsistency.
- Database design stores each logical data item at one place to ensure consistency and saves storage.
- But sometimes, controlled redundancy is necessary to improve the performance.
- Database should have capability to control this redundancy & maintain consistency by specifying the checks during database design.

2. Restricting Unauthorized Access

- A DBMS provides a security and authorization subsystem, which is used by DBA to create user accounts and to specify restrictions on user accounts.
- File processing system provides password mechanism and very less security which is not sufficient to enforce security policies like DBMS.

3. Providing Persistent Storage for Program Objects

- Object oriented database systems are compatible with programming languages such as C++ and Java.
- A DBMS software automatically performs the conversion of a complex object which can be stored in object oriented DBMS, such an object is said to be persistent due to its survival after the termination of the program.

4. Providing Storage Structures for Efficient Query Processing

- The DBMS utilizes a variety of sophisticated techniques (view, indexes etc.) to store and retrieve the data efficiently that are utilized to improve the execution time of queries and updates.
- DBMS provides indexes and buffering for fast access of query result, the choice of index is part of physical database design and tuning.
- The query processing and optimization module is responsible for choosing an efficient query execution plan for each query submitted to the system.

5. Providing Backup & Recovery

- Data should be restored to a consistent state at the time system crash and changes being made
- If hardware or software fails in the middle of the update program, the recovery subsystem of DBMS ensures that update program is resumed at the point of failure.

6. Multiple user interfaces

- DBMS provides a variety of user interfaces for the users of varying level of technical knowledge.
- These includes query language for casual users, programming language interfaces for application programmers, forms and command codes for parametric users, menu driven interfaces and natural language interfaces for stand alone users etc

7. Representing Complex Relationships among data

- A DBMS must have the capability to represent a variety of complex relationship among the data, to define new relationships as they arise, and to retrieve and update the related data easily and efficiently.

8. Enforcing Integrity Constraints

- The DBMS have certain integrity constraints that hold on data.
- Some constraints can be specified to the DBMS at the time of defining data definitions and automatically enforced.
- Database does not allow violation of constraints at the time of updating the database.

Database System Concepts & Architecture

Introduction

- The architecture of DBMS packages has evolved as follows:
    - Early monolithic systems: whole DBMS package was one tightly integrated system
    - Modern DBMS packages those are modular in design with client/server system architecture.

Data Models, Schemas, and Instances

- A data model is a collection of concepts that can be used to describe the structure of a database and thus it provides the necessary means to achieve abstraction.
- Structure of a database includes data types, relationships, and constraints that should hold on the data.
- Most data models also include a set of basic operations for specifying retrievals and updates on the database.
- Data models also include the dynamic aspect or behavior of a database application.
- Concepts to specify behavior are fundamental to object oriented data models but are also being incorporated in more traditional relational data models (e.g. stored procedures)

Categories of Data Models

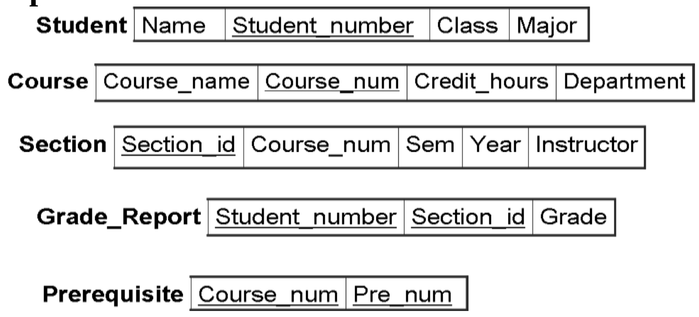
- A number of models for data representation have been proposed.

- **High level or conceptual data models**
    - provide concepts that how users perceive the data, e.g. ER model
    - object data model group (ODMG): a conceptual model for object oriented database
- **Low level or physical data models**
    - provide concepts that describe the details of how data is stored in the computer e.g. record formats, record orderings, access path, index etc.
- **Representational data models**
    - provide the concepts which is in between two extremes
    - they hide some details of data storage but can be implemented on a computer system directly
    - most frequently used in commercial relational data model (also called record-based data models)
    - other legacy data models: network and hierarchical models

Schemas, Instances, and Database State

- The description of a database is called the database schema which is specified during database design and is not expected to change frequently.
- Schema diagram displays structure of each record type but not the actual instances of records.
- Each object in the schema is known as a schema construct e.g. student table.
- A schema diagram displays only some aspects of a schema such as names of record types and data items, and some type of constraints; other aspects are not specified.

An Example

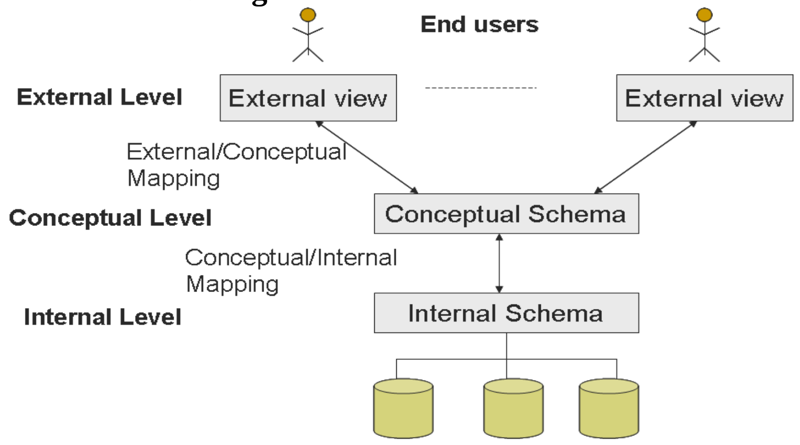


Student	Name	Student_number	Class	Major	
Course	Course_name	Course_num	Credit_hours	Department	
Section	Section_id	Course_num	Sem	Year	Instructor
Grade_Report	Student_number	Section_id	Grade		
Prerequisite	Course_num	Pre_num			

Three-Schema Architecture

- The schema in DBMS can be described at three levels:
    - Internal level has an internal schema
    - Conceptual level has a conceptual schema
    - External level includes a number of external schemas or user views
- The information about all three schemas is stored in the system catalog.

Three-Schema Architecture: Diagram



Internal Schema

- The internal schema specifies complete details of storage and access paths for the database.
- File organization on the disk should be decided e.g. hashing, indexing etc.
- The process of arriving at a good physical database schema is called physical database design.

Conceptual Schema

- The conceptual schema (or logical schema) describes the structure of the database.
- The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints.
- A representational data model is used to describe the conceptual schema when a database is implemented.
- The process of arriving at a good conceptual database schema is called conceptual database design.

Conceptual Schema (example)

Students (sid: string, sname: string, login: string, age: integer, gpa: real)
Faculty (fid: string, fname: string, sal: real)
Courses (cid: string, cname: string, credits: integer)
Rooms (rno: string, address: string, capacity: integer)
Enrolled (sid: string, cid: string, grade: string)
Teaches (fid: string, cid: string)
Meets_In (cid: string, rno: integer, time: string)

External Schema

- Allows the end users to use the database by supplying values, modifying data, view data.
- The external schemas allow data access to be customized at the level of individual users and hide rest of the details from the users.
- Any database has exactly one conceptual schema and one physical schema but may have many external schemas in view to support different users.
- Each external schema consists of a collection of one or more number of views or tables.

Data Independence

- One of the most important benefits of using a DBMS is its support for data independence.
- Applications are insulated from how data are structured and stored.
- Data independence is the capacity to change the schema at one level of a database without changing the schema at the next higher level.

Data Independence (types)

- ***Logical data independence:***
  - Protection of user views from changes in logical structure of data.
  - Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs.
    Ex: Add/Modify/Delete a new attribute, entity or relationship is possible without a rewrite of existing application programs, Merging two records into one, Breaking an existing record into two or more records etc.
- ***Physical data independence:***
  - Protection of logical structure from changes in physical structure of data.
  - Physical data independence is the capacity to change the internal schema without having to change conceptual schemas.
- Physical data independence exists in most databases.
- But logical data independence is hard to achieve.
  Ex: Using a new storage device like Hard Drive or Magnetic Tapes, Modifying indexes. Change of Location of Database from say C drive to D Drive etc.

Difference between Physical and Logical Data Independence

Logical Data Independence	**Physical Data Independence**
Logical Data Independence is mainly concerned with the structure or changing the data definition.	Mainly concerned with the storage of the data.
It is difficult as the retrieving of data is mainly dependent on the logical structure of data.	It is easy to retrieve.
It is difficult to achieve logical data independence.	it is easy to achieve physical data independence.
You need to make changes in the Application program if new fields are added or deleted from the database.	A change in the physical level usually does not need change at the Application program level.

Concerned with conceptual schema	Concerned with internal schema
Example: Add/Modify/Delete a new attribute	Example: change in compression techniques, hashing algorithms, storage devices, etc

Database Languages

- A DBMS has appropriate languages and interfaces to express database queries and updates.
- Database languages can be used to read, store, manage and update the data in the database.

- Data Definition Language (DDL)
- **Data Control Language**
- Transaction Control Language
- Data Manipulation Language (DML)
  - A high level or nonprocedural DML
  - A low level or procedural DML

Data Definition Language

- **DDL** stands for **D**ata **D**efinition **L**anguage. It is used to define database structure or pattern.
- It is used to create schema, tables, indexes, constraints, etc. in the database.
- Using the DDL statements, you can create the skeleton of the database.
- Data definition language is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints, etc

Here are some tasks that come under DDL:

- **Create:** It is used to create objects in the database.
- **Alter:** It is used to alter the structure of the database.
- **Drop:** It is used to delete objects from the database.
- **Truncate:** It is used to remove all records from a table.
- **Rename:** It is used to rename an object.

2. Data Manipulation Language

DML stands for **D**ata **M**anipulation **L**anguage. It is used for accessing and manipulating data in a database. It handles user requests.

Here are some tasks that come under DML:

- **Select:** It is used to retrieve data from a database.
- **Insert:** It is used to insert data into a table.
- **Update:** It is used to update existing data within a table.
- **Delete:** It is used to delete all records from a table.

Nonprocedural DML

  - It can be used on its own to specify complex database operations.
  - These statements can be entered interactively from a display monitor or terminal.
  - These can be embedded in a general purpose programming language where they can be extracted by a precompiler and processed by the DBMS.
  - This can specify and retrieve many records in a single DML statement, thus known as set-at-a-time or set-oriented DMLs (eg. SQL)
  - Also known as declarative language
  - **the user only specifies *what* data is needed.**

- **Procedural DML**
  - It must be embedded in a general purpose programming language.
  - This retrieves individual records from the database and processes each separately.
  - Thus it needs programming language constructs like loops.
  - This is also known as record-at-a-time (eg. DL/1)
  - **the user specifies *what* data is needed and *how* to get it**

3. Data Control Language

The Data Control Language (DCL) is used to control privilege in Database. To perform any operation in the database, such as for creating tables, sequences or views we need privileges.

- Grant - It gives user access privileges to a database.
- Revoke - It takes back permissions from the user.

4. Transaction Control Language

TCL is used to run the changes made by the DML statement. TCL can be grouped into a logical transaction.

Here are some tasks that come under TCL:

- **Commit:** It is used to save the transaction on the database.

- **Rollback:** It is used to restore the database to original since the last Commit.

Database Interfaces

User-friendly interfaces provided by a DBMS include:

- Menu-based Interfaces for Web Clients or Browsing
- Form-based Interfaces
- Graphical Interfaces
- Natural Language Interfaces
- Speech Input and Output
- Interfaces for parametric Users
- Interfaces for DBA

Menu-based Interfaces for Web Clients or Browsing

- These interfaces present the users with lists of options (called menus) that help the user in formulation of query request.
- The query is composed step by step by picking options from a menu that is displayed by the system.
- Pull-down menu is a popular technique in Web-based user interfaces.
- They are used in browsing interfaces, which allow a user to browse the content of a database in an exploratory and unstructured manner.

Form-based Interfaces

- Forms are designed and programmed for naïve users.
- User can fill up the form for new entries in database.
- User can also fill up few entries and rest of the matching entries are retrieved from the database.
- Many DBMSs have specification languages, which help programmers specify such forms.
- Oracle Forms, a component of Oracle product suite, provides an extensive set of features to design and build applications using forms.

Graphical Interfaces

- A GUI displays a schema to the user in programmatic form.
- Users can specify a query by manipulating the diagram.
- GUIs utilize both menus and forms.
- A pointing device, mouse, can be used.

Natural Language Interfaces

- A natural language interface has its own schema and a dictionary of important words.
- The natural language interface refers to the words in its schema and to the set of standard words in dictionary.

- If the interpretation is successful, the interface generates a high level query corresponding to the natural language request and submit it to DBMS for further processing.
- If interpretation is not successful, a dialogue is started with the user to clarify the request.

Speech Input and Output

- It provides limited use of speech as an input query and speech as an answer to a question or result of a request e.g. Telephone directory, Flight arrival/departure, and Bank account information etc.
- The speech input is detected using a library of predetermined words and used to set up the parameters that are supplied to the queries.
- For output, similar conversion from text or numbers into speech takes place.

Interfaces for parametric Users

- A special interface is implemented for each known class of naive users.
- Special function keys can be programmed to do repeated work fast.

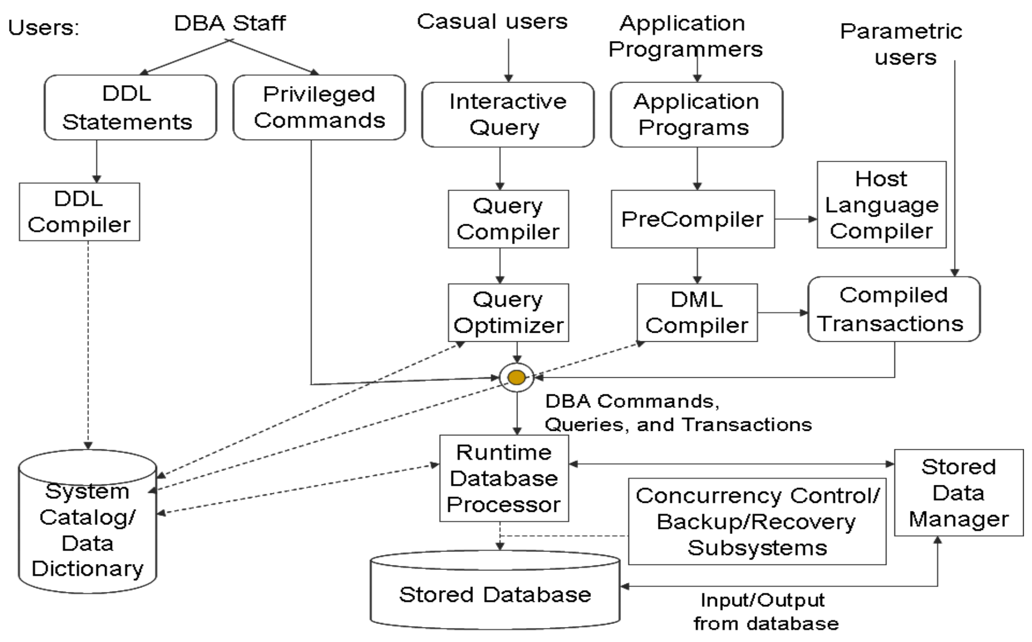
Interfaces for DBA

- DBA Staff can use privileged commands for creating accounts, setting system parameters, granting account authorization, changing a schema and reorganizing the storage structure of the database.

The Database System Environment

- The database and DBMS catalog are stored on disk.
- A stored data manager (module of DBMS) controls access to DBMS information that is stored on disk whether it is part of the database or catalog.
- Various users such as
  - DBA staff & casual users work with interactive interfaces to formulate queries.
  - Application programmers write programs using host language.
  - Parametric users make data entry to database through predefined forms.
- DBA staff works on defining the database and tuning it using DDL and other privileged commands.
- DDL compiler processes schema definitions, specified in the DDL, and stores description of the schema in the DBMS catalog.
- Queries are parsed, analyzed for correctness of the operations and names of data elements by query compiler that compiles them into an internal form.
- Internal query is subjected to query optimization by query optimization.
- The precompiler extracts DML commands from an application program and sends to DML compiler for compilation.
- Rest of the program is sent to host language compiler.

- The object codes for the DML commands and rest of the program are linked, forming a canned transaction whose executable code includes calls to runtime database processor.
- Parametric users supply the parameters to these canned transactions directly so they can run transactions repeatedly.
- Runtime database processor executes
  - Privileged commands
  - Executable query plans
  - Canned transactions with runtime parameters.
- It works with the system dictionary, stored data manager which in turn uses basic OS services for carrying out low level input/output operations between disk and memory.
- It also handles some aspects of management of buffers in main memory, some DBMS have their own buffer management.
- Concurrency control, back up and recovery manager are integrated into the working of the runtime database processor for purposes of transaction management.



Component Modules of a DBMS and their interactions

DBMS Architecture

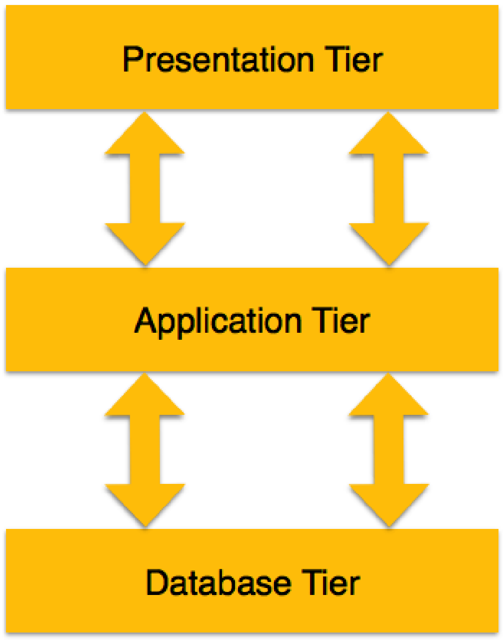
The architecture of a DBMS can be seen as either single tier or multi-tier. An n-tier architecture divides the whole system into related but independent **n** modules, which can be independently modified, altered, changed, or replaced.

In 1-tier architecture, the DBMS is the only entity where the user directly sits on the DBMS and uses it. Any changes done here will directly be done on the DBMS itself. It does not provide handy tools for end-users. Database designers and programmers normally prefer to use single-tier architecture.

If the architecture of DBMS is 2-tier, then it must have an application through which the DBMS can be accessed. In the two-tier architecture, applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: **ODBC**, **JDBC** are used. The user interfaces and application programs are run on the client-side. The server side is responsible to provide the functionalities like: query processing and transaction management. To communicate with the DBMS, client-side application establishes a connection with the server side

3-tier Architecture

A 3-tier architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database. It is the most widely used architecture to design a DBMS.



- **Database (Data) Tier** − At this tier, the database resides along with its query processing languages. We also have the relations that define the data and their constraints at this level.
- **Application (Middle) Tier** − At this tier reside the application server and the programs that access the database. For a user, this application tier presents an abstracted view of the database. End-users are unaware of any existence of the database beyond the application. At the other end, the database tier is not aware of any other user beyond the application tier. Hence, the application layer sits in the middle and acts as a mediator between the end-user and the database.
- **User (Presentation) Tier** − End-users operate on this tier and they know nothing about any existence of the database beyond this layer. At this layer, multiple views

of the database can be provided by the application. All views are generated by applications that reside in the application tier.

Classification of Database Management Systems

Based on number of users:

- Single User Systems: with PCs
- Multiuser Systems: majority of DBMS

Based on number of sites over which database is distributed:

- Centralized DBMS: Data is stored at single computer site
- Distributed DBMS: Database and DBMS software are distributed over many sites connected by a computer network.
    - **Homogeneous:** Same DBMS software at multiple sites
    - **Heterogeneous:** Participating DBMS have a degree of local autonomy. This leads to federated Database in which participating DBMSs are loosely coupled.

Based on cost:

- Open Source DBMS: Main RDBMS products like MySQL, PostgresSQL are available as 30 days versions, Many vendors support with additional facilities and sell. Giant systems are sold in modular form according the configuration required.
- License based
    - Site license allow unlimited use of database system with any number of copies running at customer site.
    - License limits the number of concurrent users at a location.
- Standalone single user versions are sold per copy or included in desktop or laptop configuration, ACCESS
- Additional features can be made available in any of the above kind at extra cost

Based on types of access paths:

■Different File Structures

e.g. Inverted File Structure

Based on purpose of use:

- General Purpose: These DBMS systems can be used for variety of applications.
- Special Purpose: A DBMS system is designed for a special application e.g. Online transaction processing (OLTP)

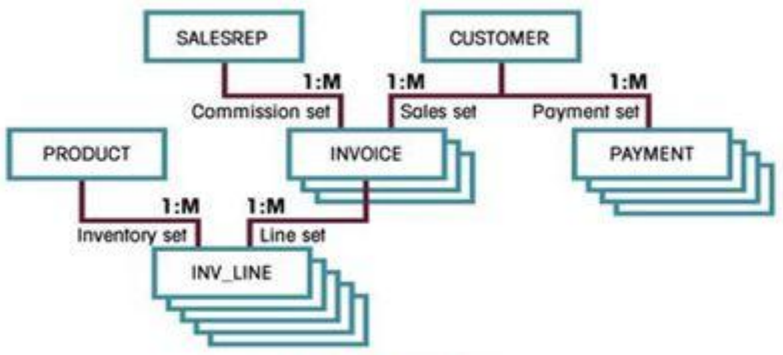
Based on data model:

Data models are used to show how the data is connected and stored inside a system. Data models mainly represent the relationship between the data.

- Hierarchical data model
- Network data model
- Object data model
- Relational data model

Network model

In the network data model, data model data are represented by collections of records. Relationships among data are represented by links. In this data model, graph data structure is used. It permits a record to have more than one parent.



Network data model

Advantages of Network data model

- The Network data model is also conceptually simple and easy to design.
- In the network data model relationships like one-to-one and many-to-many are exist.
- In the network data model without the owner, no member exists.
- In the network database terminology, a relationship is a set. Each set comprises two types of record an owner record and a member record.

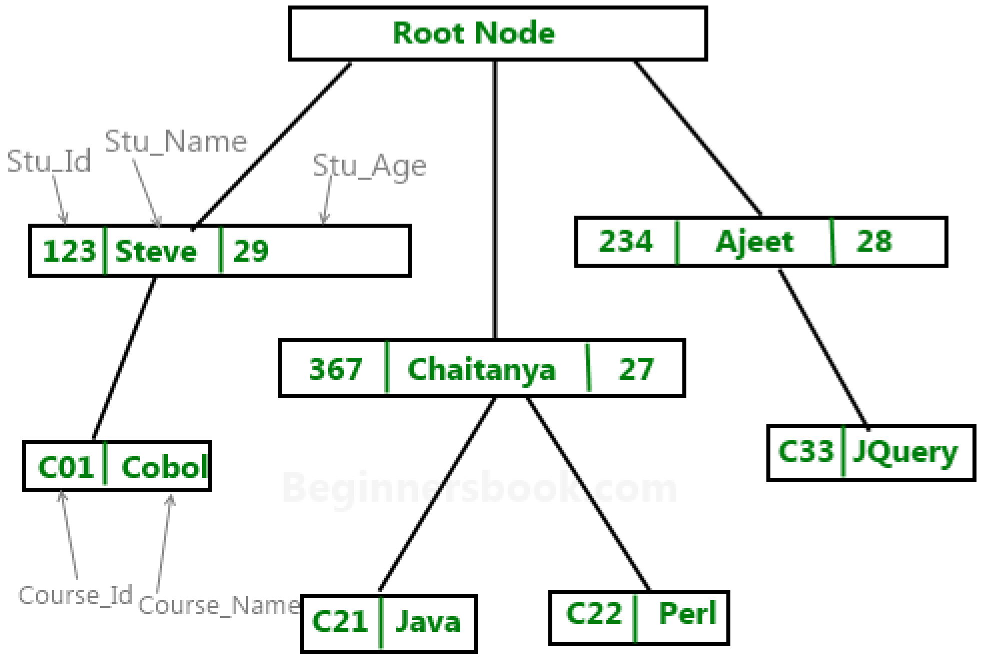
The Network Model has the following disadvantages:

- 1. System complexity

- 2. Lack of structural independence

Hierarchical data model

In the hierarchical data, model data are represented by collections of records. Relationships among data are represented by links. In this model, tree data structure is used. There are two concepts associated with the hierarchical model segments types and parent-child relationships.

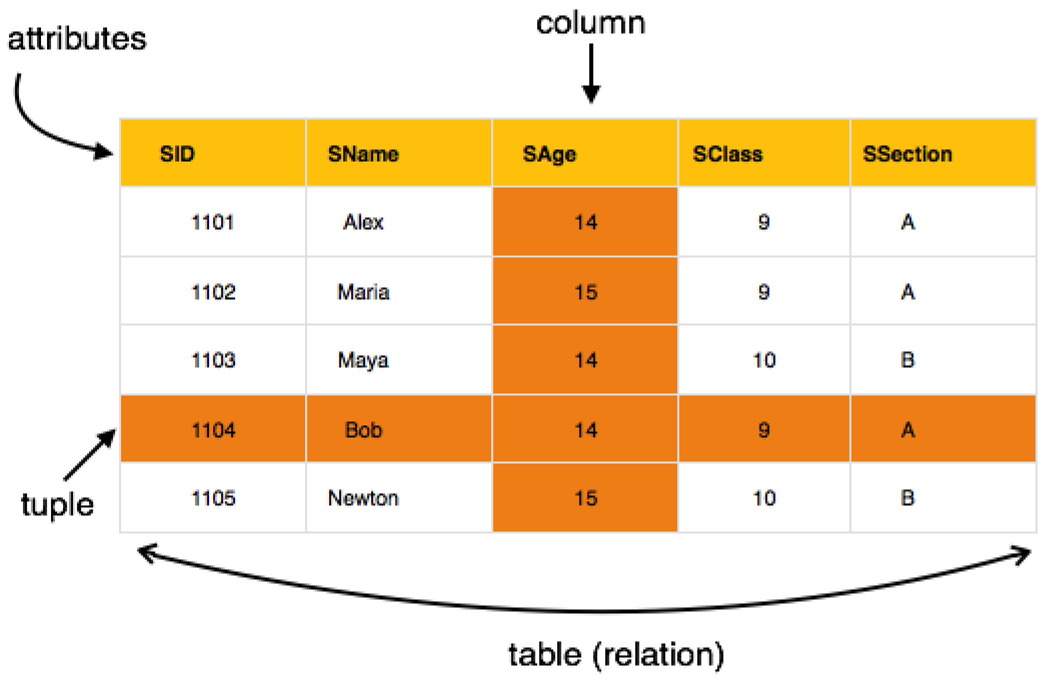


Advantages of Hierarchical data model

- Since the database is based on the hierarchical structure the relationships between the various layers are logically simple.
- The hierarchical data model was the first database that offered the data security that is provided by DBMS.
- The Hierarchical database model is based on the parent-child relationships.
- It is very efficient one when the database contains a large number of one-to-many relationships.

Relational data model

In relational model, the data and relationships are represented by collection of inter-related tables. Each table is a group of column and rows, where column represents attribute of an entity and rows represents records.



SID	SName	SAge	SClass	SSection
1101	Alex	14	9	A
1102	Maria	15	9	A
1103	Maya	14	10	B
1104	Bob	14	9	A
1105	Newton	15	10	B

Object Data Model:

The object-oriented model is based on a collection of objects. An object contains values stored in instances variable within the object. An object contains bodies of code that operate on the object.

Advantages of object oriented data model

- It represents relationships explicitly supporting both navigated and associative access to information.
- Object-oriented database systems are not suited for all applications.
- It is difficult to maintain when organizational information changes.