Chapter 1
Introduction

Outlin

- Introduction
  - Characteristics of Database
  - Database System Applications
  - Comparison of File System with Database Approach

Introduction

- A database-management system (DBMS) is a **collection of Inter-related data** and a set of programs to access those data.
- The collection of data, usually referred to as the **database**, contains information relevant to an enterprise.
- **DBMS Goal** : provide a way to store and retrieve database information that is both convenient and efficient.
- Database systems are designed to manage large bodies of information.
- Management of data involves both defining structures for storage of information and providing mechanisms for the manipulation of information.

Introduction (continued)

- In addition, the database system must ensure the safety of the information stored, despite system crashes or attempts at unauthorized access.
- If data are to be shared among several users, the system must avoid possible anomalous results.

Main Characteristics of the Database System Approach

1. **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.

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

Example of a simplified database catalog

RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
….	….	…..
….	….	…..
….	….	…..
Prerequisite_number	XXXXNNNN	PREREQUISITE

Note: Major_type is defined as an enumerared type with all known majors. XXXXNNNN is used to define a type with four alpha characters followed by four digits

Figure 1.3
An example of a database catalog for the database in Figure 1.2.

Main Characteristics of the Database System Approach (contd..)

2. **Insulation between programs and data:**
   - The structure of data files is stored in the DBMS catalog separately from the access programs, call this property as **program-data independence**
   - Allows changing data structures and storage organization without having to change the DBMS access programs.
3. **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

Main Characteristics of the Database System Approach (contd..)

4. 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.

(a)

TRANSCRIPT

Student_name	Student_transcript				
	Course_number	Grade	Semester	Year	Section_id
Smith	CS1310	C	Fall	08	119
	MATH2410	B	Fall	08	112
Brown	MATH2410	A	Fall	07	85
	CS1310	A	Fall	07	92
	CS3320	B	Spring	08	102
	CS3380	A	Fall	08	135

(b)

COURSE_PREREQUISITES

Course_name	Course_number	Prerequisites
Database	CS3380	CS3320
		MATH2410
Data Structures	CS3320	CS1310

Main Characteristics of the Database System Approach (contd..)

5. **Sharing of data and multi-user transaction processing:**
   - *Concurrency control* within the DBMS guarantees that each **transaction** is correctly executed or aborted
   - Allowing a set of **concurrent users** to retrieve from and to update the database.
   - *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-System Applications

- **Enterprise Information**
  - Sales: For customer, product, and purchase information.
  - Accounting: For payments, receipts, account balances, assets, and other accounting information.
  - Human resources: For information about employees, salaries, payroll taxes, and benefits, and for generation of paychecks
- **Manufacturing**
  - For management of the supply chain and for tracking production of items in factories, inventories of items in warehouses and stores, and orders for items.

Database-System Applications

- **Banking and Finance**
  - Banking: For customer information, accounts, loans, and banking transactions.
  - Credit card transactions: For purchases on credit cards and generation of monthly statements.
  - Finance: For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds; also for storing real-time market data to enable online trading by customers and automated trading by the firm.
- **Universities**
  - For student information, course registrations, and grades
- **Airlines**
  - For reservations and schedule information. Airlines were among the **first to use databases** in a geographically distributed manner.

Database-System Applications

- **Telecommunication**
  - For keeping records of calls, texts, and data usage, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.

- **Web-based services**
  - Social-media: For keeping records of users, connections between users(such as friend/follows information), posts made by users, rating/like information about posts, etc.
  - Online retailers: For keeping records of sales data and orders as for any retailer, but also for tracking a user's product views, search terms, etc., for the purpose of identifying the best items to recommend to that user.

Database-System Applications

- **Document databases**
  - For maintaining collections of new articles, patents, published research papers, etc.
- **Navigation systems**
  - For maintaining the locations of varies places of interest along with the exact routes of roads, train systems, buses, etc.

Advantages of DBMS over File System

- **Controlling redundancy in data storage** and in development and maintenance efforts
  - Sharing of data among multiple users
- **Restricting unauthorized access** to data
- Providing **persistent storage** for program objects
- Providing **Storage Structures** (e.g. indexes) for efficient Query Processing
- Providing **backup and recovery services**
- Providing **multiple interfaces** to different classes of users
- Representing **complex relationships** among data
- Enforcing **integrity constraints** on the database

Comparison of File System with Database Approach

File System	DBMS
File system is a collection of data. Any management with the file system, **user has to write the procedures**	DBMS is a collection of data and **user is not required to write the procedures** for managing the database.
File system gives **the details of the data representation and storage of data**.	DBMS provides an **abstract view of data** that hides the details.
In File system **storing and retrieving of data cannot be done efficiently.**	DBMS is efficient to use since there are wide varieties **of sophisticated techniques to store and retrieve the data.**
Concurrent access to the data in the file system has many problems like : Reading the file while deleting some information, updating some information	DBMS **takes care of concurrent access** using some form of locking.

Comparison of File System with Database Approach(contd..)

File System	DBMS
File system doesn't provide crash recovery. Eg. While we are entering some data into the file if system crashes then content of the file is lost	DBMS has **crash recovery mechanism**, DBMS protects user from the effects of system failures.
Protecting a file under file system is very difficult.	DBMS has **a good protection mechanism.**

Module 1.2

Outline

- **Data Abstraction**
- **Data Independence**
- **DBMS System Architecture**
- **Database Administrator**
- **DBMS Languages**
- **Database Users**

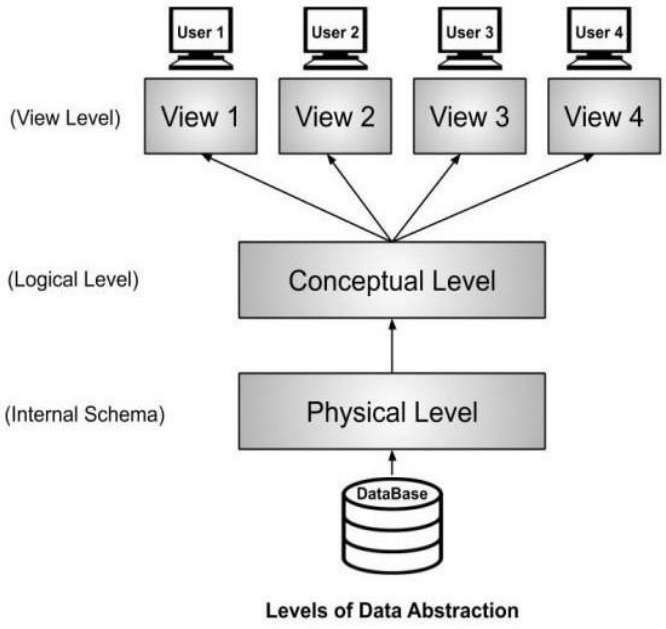
Data Abstraction

- Data Abstraction refers to the **process of hiding data organization and storage details from the user.**
- 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.**

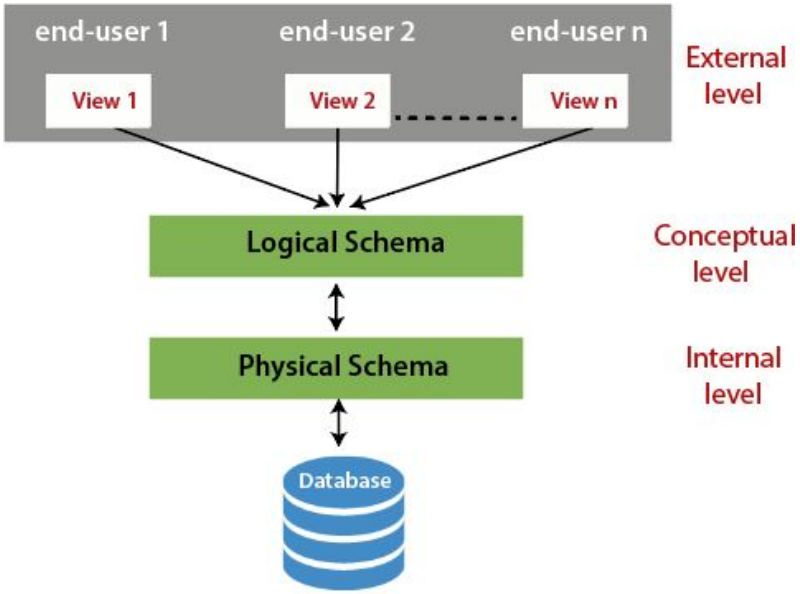
Levels of Abstraction

- These three levels of data abstraction / DBMS Architecture Diagram: Level

1. View Level/ External level
2. Conceptual Level/ Logical level
3. Physical Level/ internal level



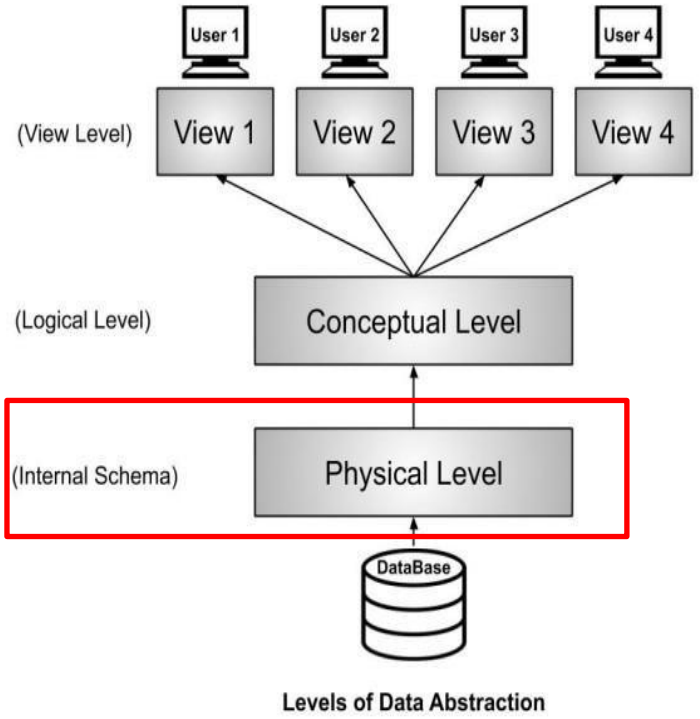
Levels of Data Abstraction



DBMS Architecture

1. Physical Level or Internal Level

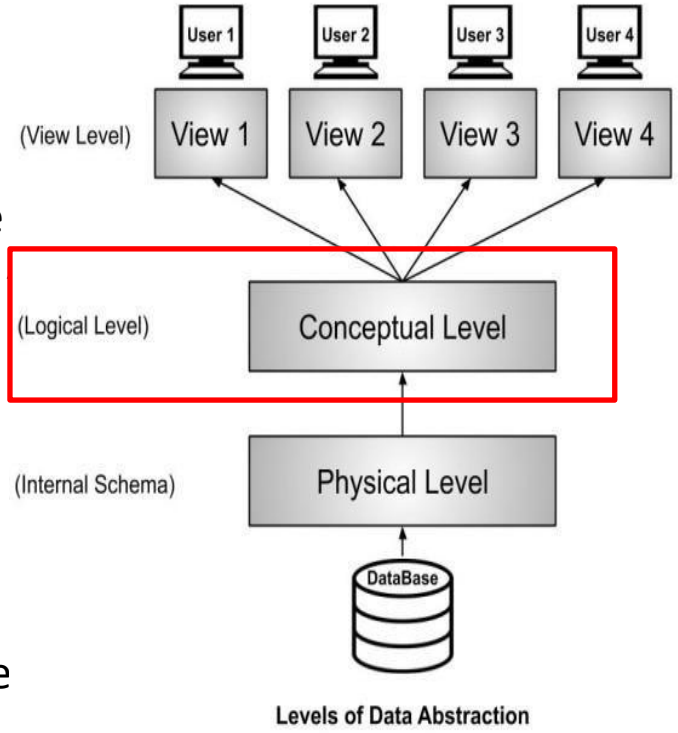
- This is the lowest level of data abstraction.
- It tells us **how the data is actually stored in physical memory (hard disks, magnetic tapes etc).**
- The access methods like sequential or random access and file organization methods like B+ trees, hashing used for the organization purpose
- Developers would know usability, size of memory, and accessing frequency which makes easy to design this level.
- Blocks of storage and the amount of memory used for these purposes is kept hidden from the user.



Levels of Data Abstraction

2. Conceptual or Logical Level

- The conceptual level **describes the structure of the whole database**.
- This level acts as a middle layer between the physical storage and user view.
- It explains what data to be stored in the database what relationship exists among those data, and what the data types are.
- There is only one conceptual schema per database.
- Database administrator and the programmers use the logical level of abstraction to decide what information to keep in a database



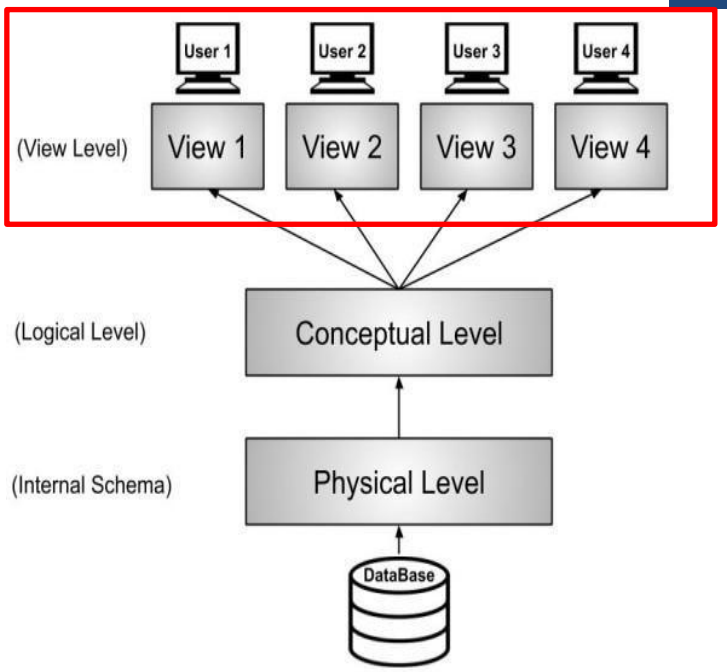
Levels of Data Abstraction

3. External or View Level

- This is the highest level of database abstraction
- This level describes user interaction with database system
- Many users are not aware of technical details of the system, and also they need not access whole information from database

Hence it is necessary to provide a simple and short interface for such users as per their requirements

- Multiple views can be created for same database for multiple users



Levels of Data Abstraction

Levels of Abstraction

Example: Storing all employees information in employee table

- **Physical level**: Records can be described as blocks of storage(bytes, gigabytes, terabytes etc) in memory. All these details are usually hidden from developer
- **Logical Level:** Records can be described as fields/ attributes along with their datatypes
  - Relationship between these fields can be implemented logically
  - Usually developer works at this level because they have all such knowledge required
- **View Level**: Each user interacts with help of GUI and enters details at screen
  - User is not aware of how the data is stored and what data is stored, such details are hidden from them

Data Independence

- Deals with independence between the way the data is structured and program that manipulate it
- Data independence is not possible in file processing systems, as file structure and application programs are tightly coupled together
- Data independence is possible in DBMS only because **application programs do not deal with data in database directly**
- Three level DBMS illustrates two types of data independence :
  - **Physical Data Independence**
  - **Logical Data independence**

Data Independence(contd..)

- **Physical Data Independence**
  - Ability to modify physical schema without causing application programs to be rewritten
  - Modifications at the physical level are occasionally necessary to improve performance
  - We can change physical storage/level without affecting conceptual or external view of data
  - New changes are absorbed by mapping techniques

Data Independence(contd..)

- **Logical Data Independence:**
  - Ability to modify logical schema without causing application program to be rewritten
  - Modifications at logical level are necessary whenever the logical structure of database is altered
  - Logical data independence means if we add some new columns or remove columns from table then user view or program should not change
- Application program need not know:
  - Ordering of data fields in a record
  - Ordering of records in a file
  - Size of each record
  - Size of each field
  - Format and type of each data item
  - Whether file is sorted or not

Logical Data Independence(contd..)

For example :

- Consider two users A & B. Both are selecting the fields "EmployeeNumber" and "EmployeeName".
- If user B adds a new column (e.g. salary) to his table, it will not affect the external view for user A, though the **internal schema of the database has been changed for both users A & B.**
- Logical data independence is more difficult to achieve than physical data independence, since application programs are heavily dependent on the logical structure of the data that they access.

DBMS System Architecture

- **The three tier architecture** is most widely used architecture in today's world.
- In this architecture the user layer, business layer and data layer are implemented independently by three different applications.
- The data required by the business logic exists in database server.
- In three tier architecture all layers interact with each other independently.

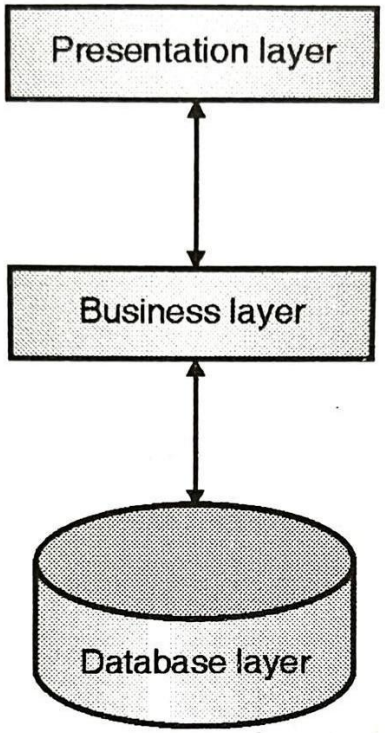


Fig. 1.7.1 : Three tier architecture

DBMS System Architecture (Continued)

- Below example is the online shopping diagram for 3 Tier architecture.
- Here frontend used is Dot Net environment while for backend database MS SQLServer 2008 is used.

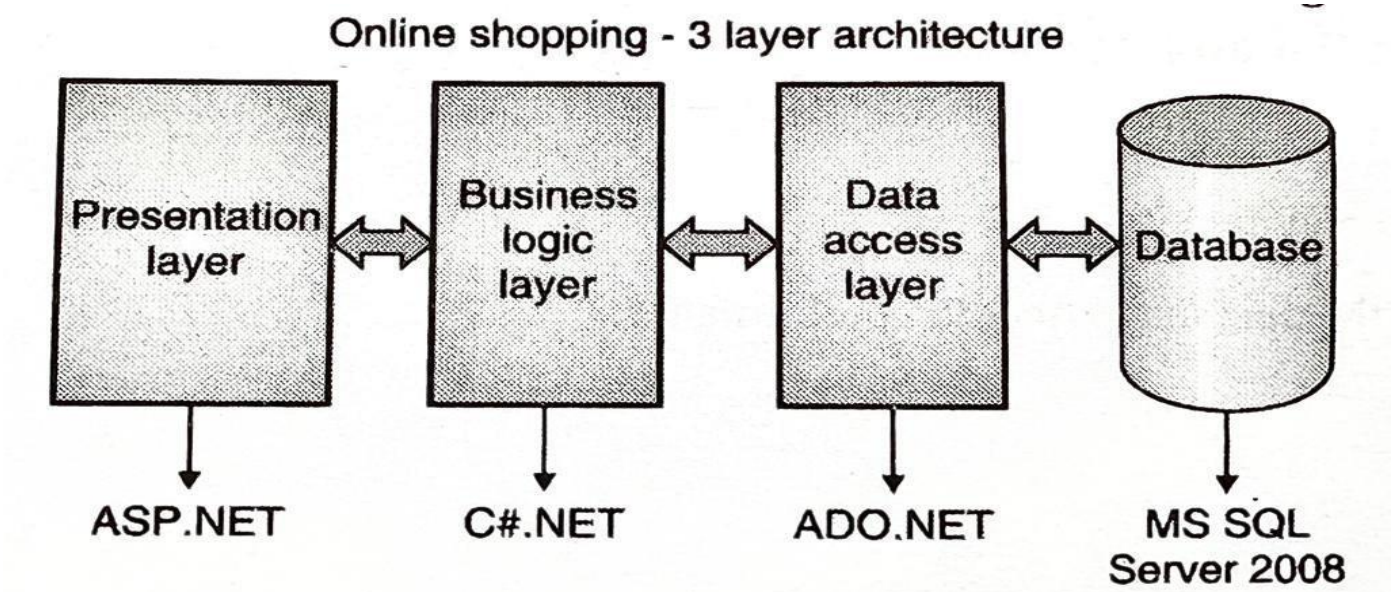


Fig. 1.7.2 : Online shopping diagram for 3-tier architecture

DBMS System Architecture (continued)

1. **Presentation layer:**

- This layer consist of **user interface designed** for the interaction with end user. This layer is created in ASP.Net.
- It includes the screens which will be used by the **end user for shopping**.
- Theses screen show the **products with details as per their categories**.
- User can **select the product to purchase and add them into cart**.
- This design is created with advanced controls available in ASP.Net

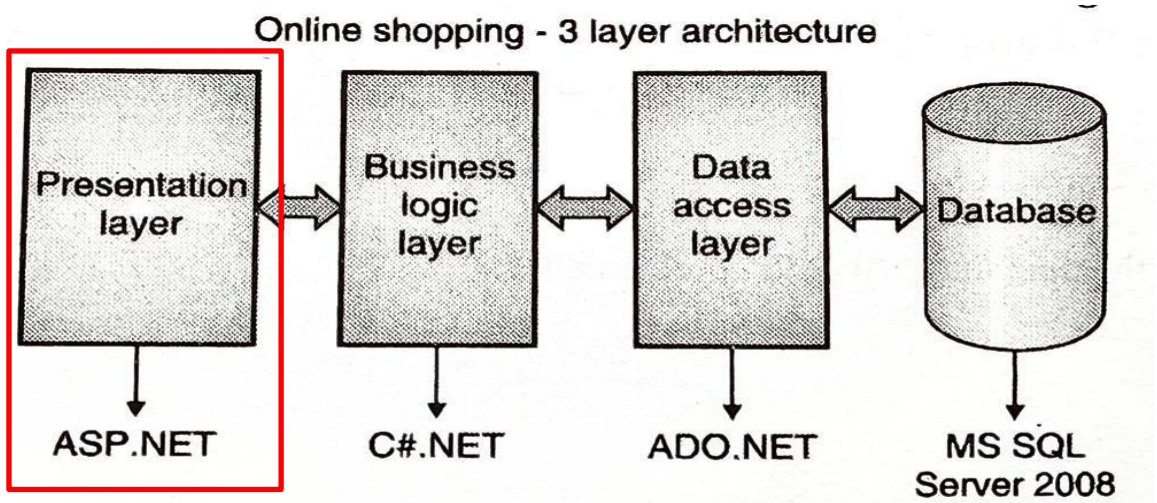


Fig. 1.7.2 : Online shopping diagram for 3-tier architecture

DBMS System Architecture (continued)

2. Business Layer:

- This layer consist of validation checking code related to product selection of user.
- Accidently user may select wrong number of products to purchase.
- For example if any user is giving order to purchase 10000 TV sets, then the order should be validated.
- This logical code is implemented using the C# Net.

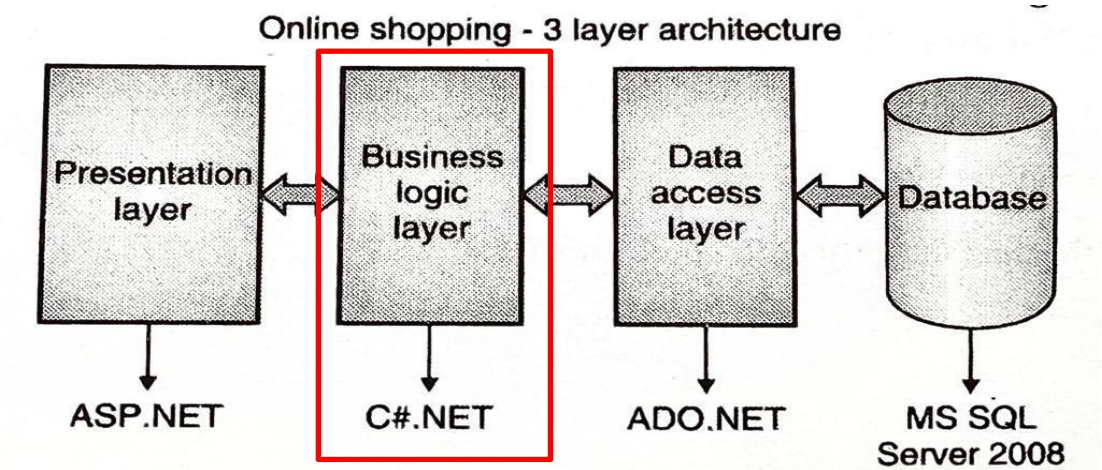


Fig. 1.7.2 : Online shopping diagram for 3-tier architecture

DBMS System Architecture (Continued)

3. Data Layer:

- This layer contains the code interacting with database on the server.
- For example accessing product details from database, inserting transaction details of user order in database etc.
- This database handling is implemented using the ADO.Net

Here all the three layers work independently and efficiently.

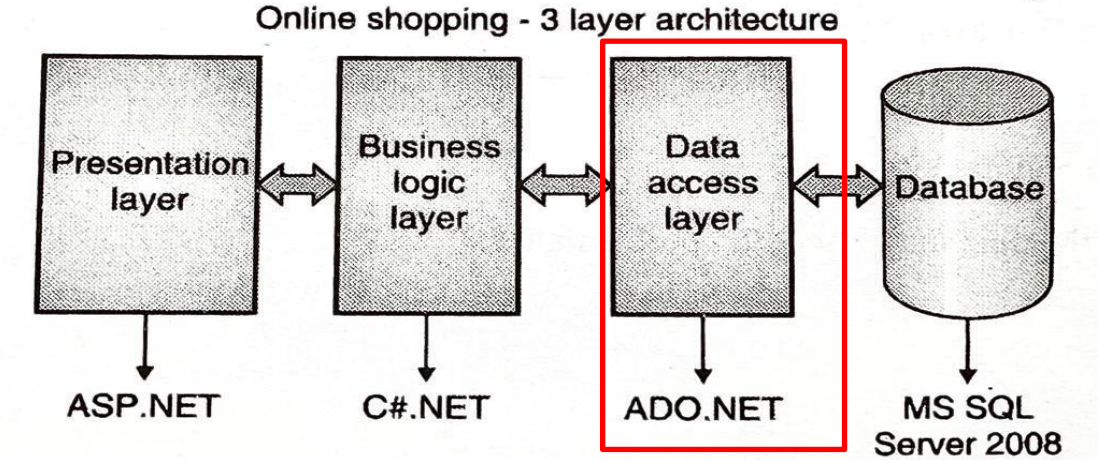


Fig. 1.7.2 : Online shopping diagram for 3-tier architecture

Three tier Architecture: Advantages

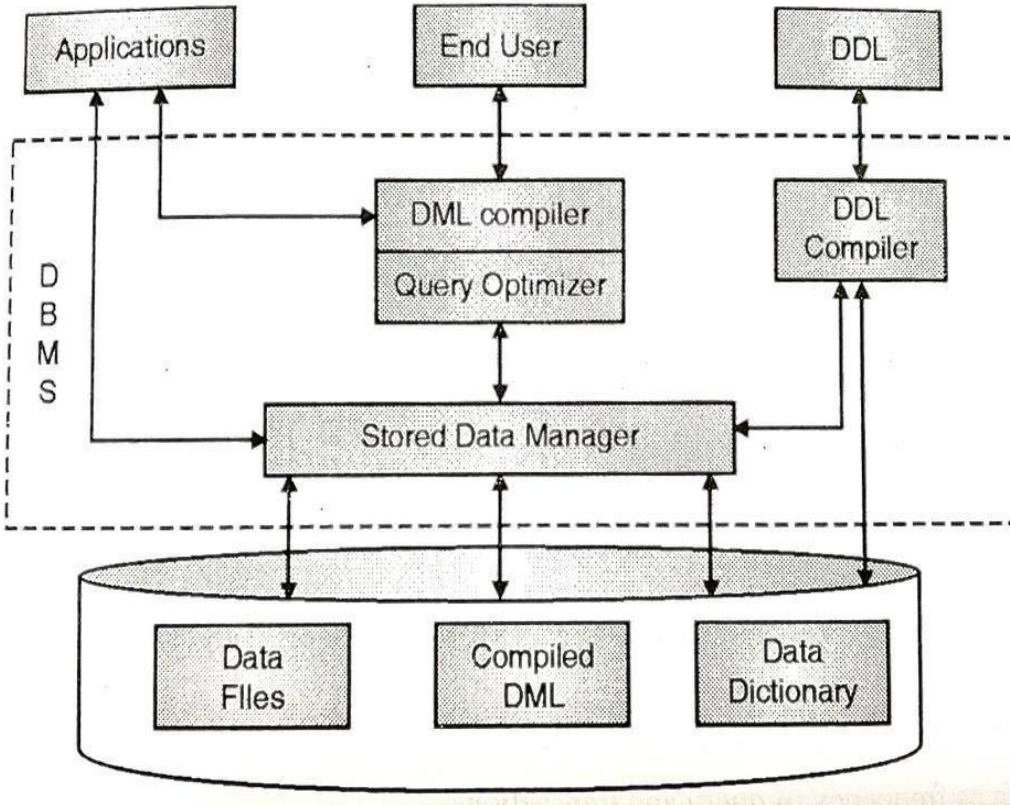
- In three tier architecture we can **manage the data independently**.
- Can make the **changes in presentation layer without affecting other two tiers.**
- As each tier is independent it is **possible to use different groups of developers**
- It is **most secure** since the client doesn't have direct access to the database layer.
- When **one tier fails there is no data loss**, because you are always secure by accessing the other tier.
- Due to distributed deployment of application server**, scalability is increased.**
- It is **reusable.**
- It is **robust and secure** due to multiple layers.

Three tier Architecture: Disadvantages

- It is more complex structure.
- More difficult to set up and maintain it.
- The physical separation of the tiers may affect the performance.

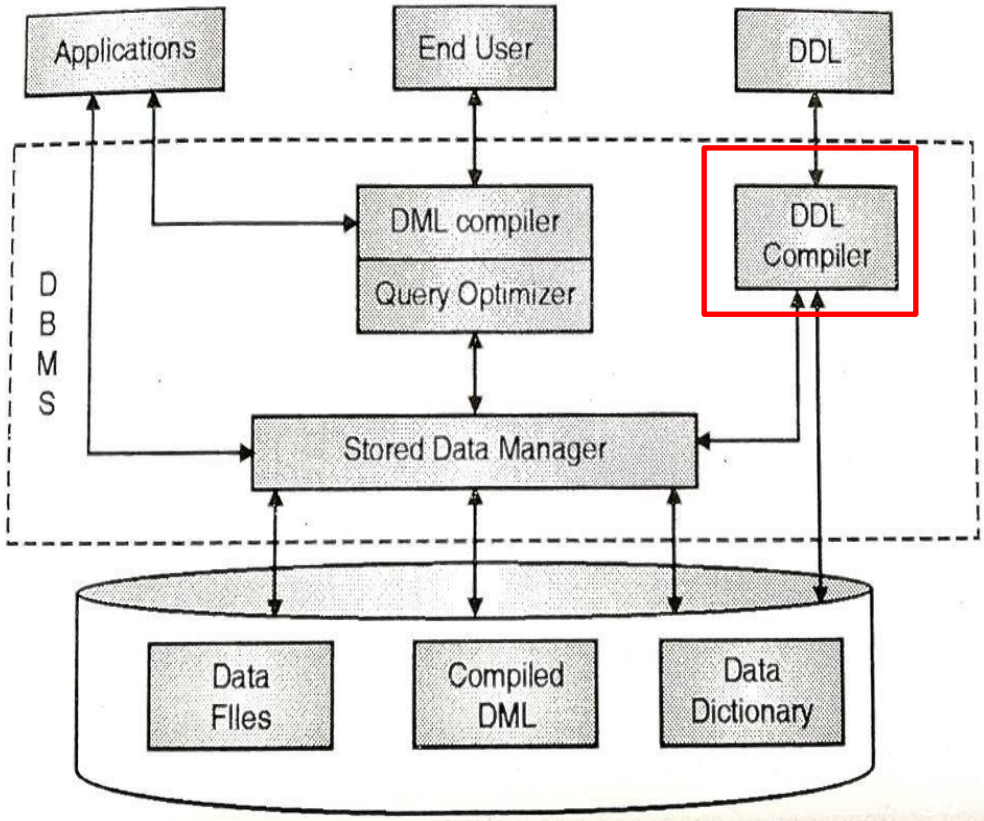
Components of DBMS

- DBMS (Database Management System) **acts as an interface between the user and the database**
- The user requests the DBMS to perform various operations **retrieve, insert, delete and update on the database**
- The components of DBMS perform these requested operations on the database and provide necessary data to the users



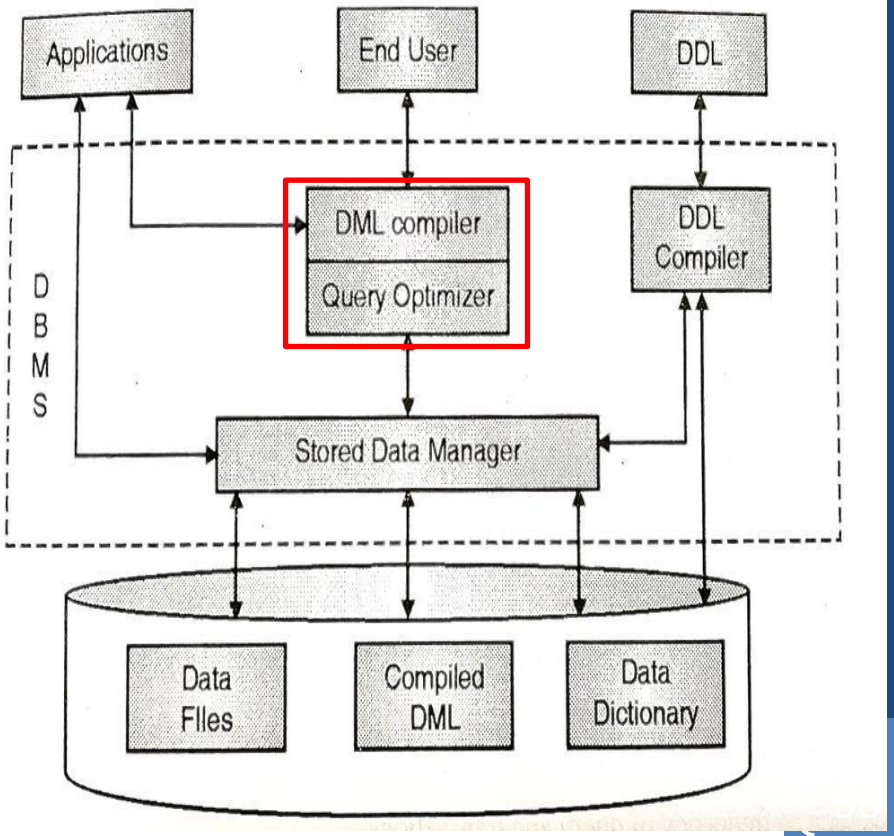
Components of DBMS

- **DDL Compiler**
  - The DDL Compiler converts DDL commands into set of tables containing metadata stored in a data dictionary.
  - The metadata information is
    - name of the files
    - data items
    - storage details of each file
    - mapping information and constraints etc.



Components of DBMS

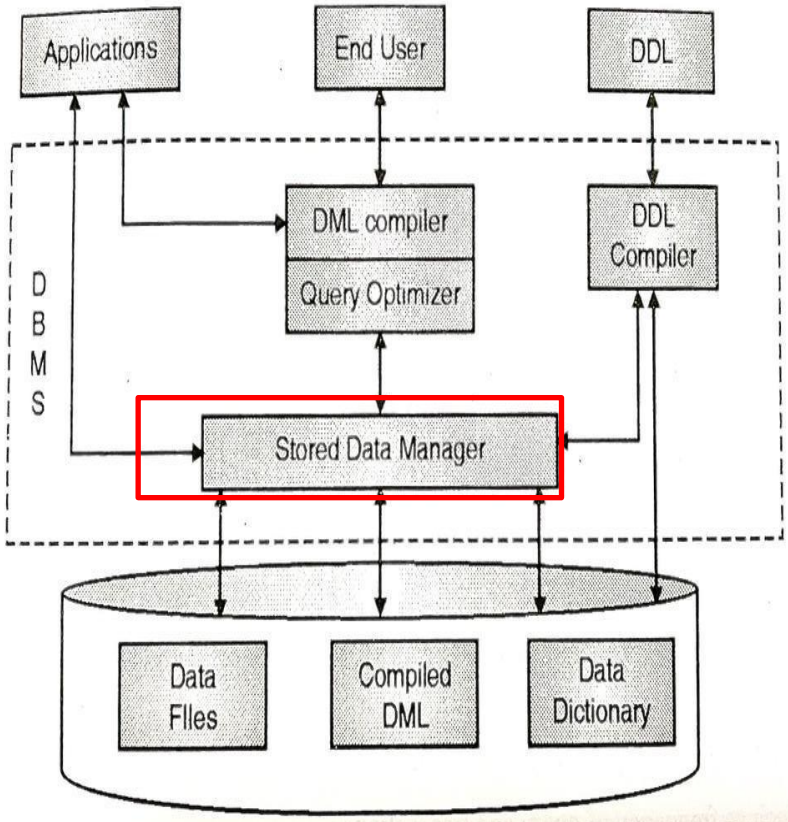
- **DML Compiler and Query optimizer**
  - The DML commands such as retrieve, insert, update, delete etc. from the application program are sent to the DML compiler for compilation.
  - It converts these commands into object code for understanding of database.
  - The object code is then optimized in the best way to execute a query by the query optimizer and then send to the data manager.



Components of DBMS

Data Manager

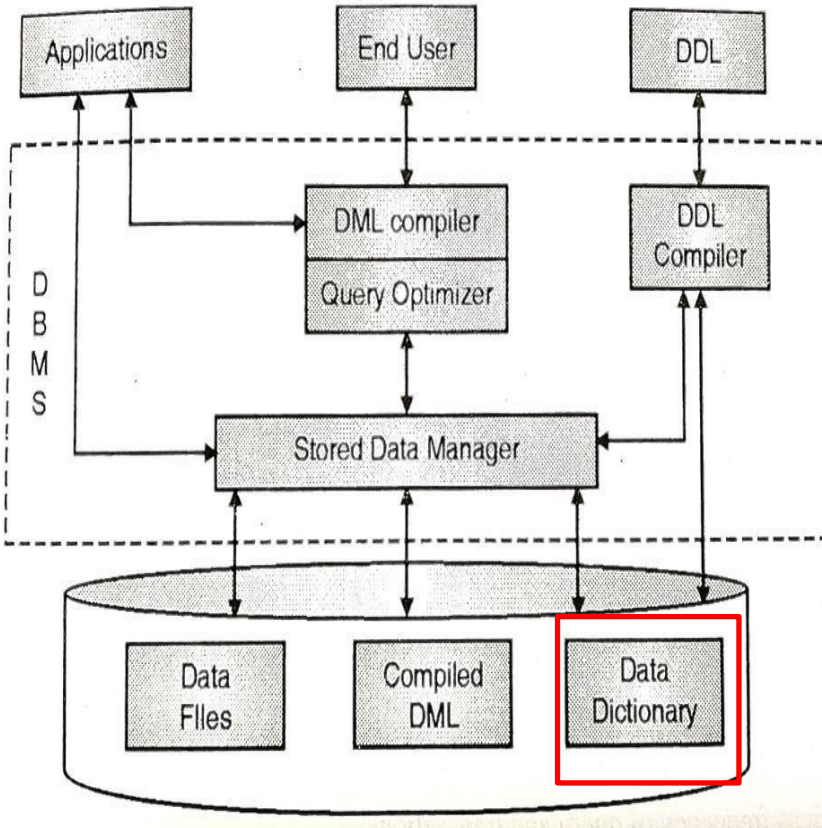
- The Data Manager is the central software component of the DBMS also knows as Database Control System
- The main functions of Data Manager are :
  - It converts the requests received from query optimizer to machine understandable form so it makes actual request inside the database
  - Controls DBMS information access that is stored on disk.
  - It controls handling buffers in main memory.
  - It enforces constraints to maintain consistency and integrity of the data.
  - It synchronizes the simultaneous operations performed by the concurrent users.
  - It also controls the backup and recovery operations.



Components of DBMS

Data Dictionary

- Repository of description of data in the database.
- It contains information about **Data** - names of the tables, names of attributes of each table, length of attributes, and number of rows in each table, Relationships between database transactions
- Constraints on data i.e. range of values permitted.
- Detailed information on physical database design such as storage structure, access paths, files and record sizes.
- Access Authorization - description of database users their responsibilities and their access rights.
- Usage statistics such as frequency of query and transactions.
- Data dictionary is used to actually control the data integrity and accuracy.
- It may be used as an **important part of the DBMS.**



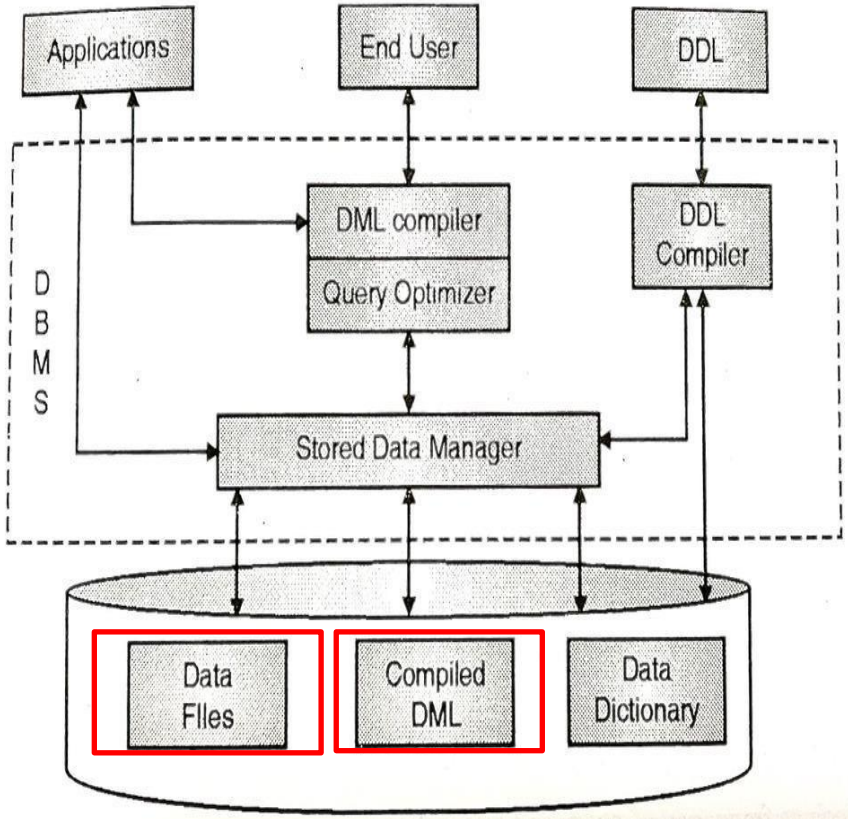
Components of DBMS

Data File

- It contains the data portion of the database i.e. it has the real data stored in it.
- It can be stored as magnetic disks, magnetic tapes or optical disks.

Compiled DML

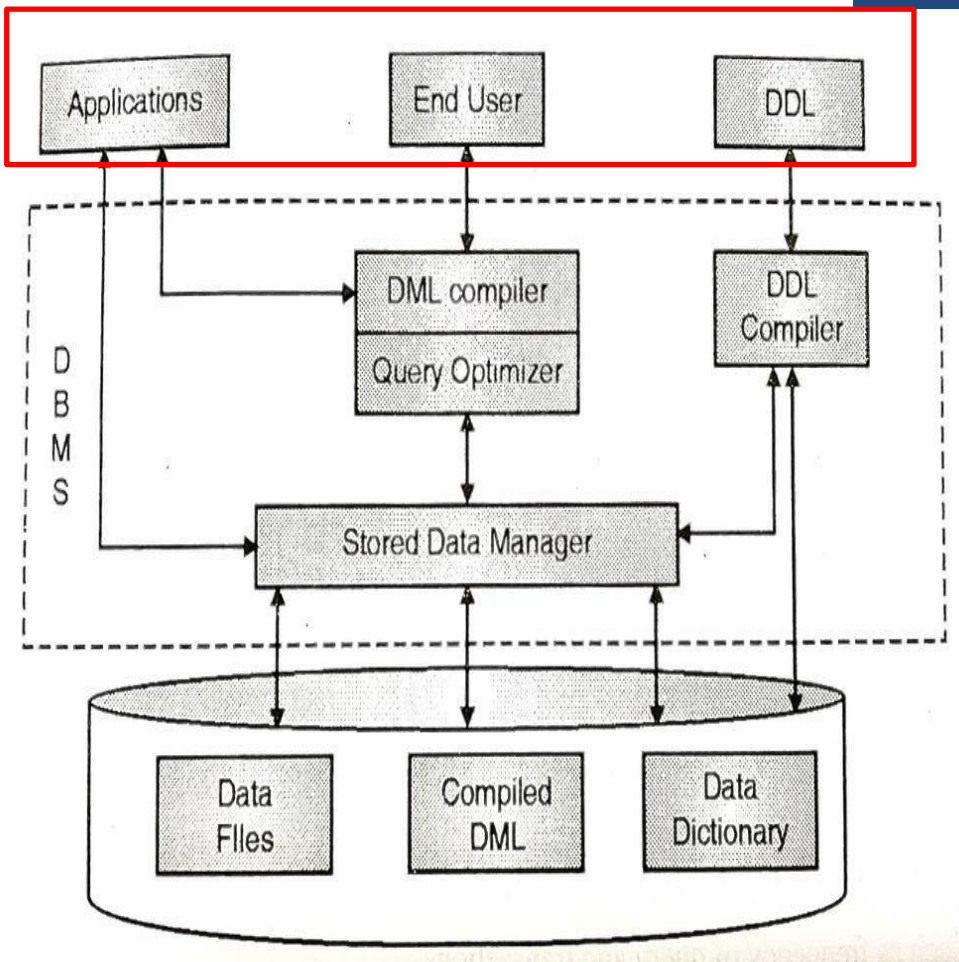
- The DML compiler converts the high level Queries into low level file access commands known as compiled DML.
- Some of the processed DM statements (insert, update, delete) are stored in it so that if there is similar requests, the data can be reused.



Components of DBMS

End Users

- They are the real users of the database.
- They can be
  - Database developers
  - Database Designers
  - Database administrator
  - Actual users of the database



Database Administrator

- A database administrator (DBA) is a person who directs or performs all activities related to maintaining a successful database environment.
- Responsibilities include
  - **Designing**
  - **Implementing and maintaining the database system**
  - **Establishing policies and procedures pertaining to the management**
  - **Security, maintenance**
  - **Training employees in database management and use**
- The life cycle of database begins with designing, and ends with final implementation.
- All the types of databases need to be designed accurately so that it should work without any problems.

Database Administrator

- After completion of design, it needs to be installed.
- After this step, users start using the database.

The database grows as the data grows in the database. When the size of database becomes huge, its performance may fall down.

- Also accessing the data from such huge database becomes challenge.

There will be unused memory in database, making the memory inevitably huge.

- These administration and maintenance of database is taken care by database Administrator - **DBA**.

Database Administrator

- The important **difference** between database user and database administrator is that:
  - **Database users** are only concerned with use of database as per their need
  - **Database administrator** has to manage and maintain the database.
- A DBA has many responsibilities. A good performing database is in the hands of DBA.

Database Administrator: Functions

- **Installing and upgrading the database server** and application tools
- **Allocating system storage and planning future storage requirements** for the database system
- **Modifying the database structure**, as necessary, from information given by application developers
- **Enrolling users and maintaining** system security
- **Ensuring compliance with database vendor license agreement**
- **Controlling and monitoring user access** to the database
- **Monitoring and optimizing the performance** of the database
- **Planning for backup and recovery** of database information
- **Maintaining archived data**

Database Administrator: Functions

- **Backing up and restoring databases**.
- **Contacting database vendor** for technical support.
- **Generating various reports** by querying from database as per need.
- **Managing and monitoring data replication.**

Database Administrator: Skillset

- Communication skills
- Knowledge of database Queries, Database Theory and Database Design
- Knowledge about the RDBMS itself, e.g. Microsoft SQL Server or MySQL
- Knowledge of Structured Query Language (SQL), e.g. SQL or Transact-SQL
- General understanding of distributed computing architectures, e.g. Client-server model
- General understanding:
  - operating system, e.g. Windows or Linux
  - storage technologies and networking
  - routine maintenance, recovery, and handling failover of a database

DBMS Languages

1. Data Definition Language (DDL)
2. Data Manipulation Language (DML)
3. Data Control Language(DCL)
4. Transaction Control Language(TCL)

DBMS Languages(contd..)

1. **Data Definition Language (DDL):**
   - Used by the DBA and database designers to specify the conceptual schema of a database.
   - In many DBMSs, the DDL is also used to define internal and external schemas (views)
2. **Data Manipulation Language (DML):**
   - Used to specify database retrievals and updates
   - **High Level or Non-procedural Language:**
     - For example, the SQL relational language
     - Are “set”-oriented and specify what data to retrieve rather than how to retrieve it.
   - **Low Level or Procedural Language:**
     - Retrieve data one record-at-a-time
     - Loops are needed to retrieve multiple records

DBMS Languages(contd..)

3. Data Control Language

- Grant privilege to a user using the GRANT statement.
- **GRANT**: Give privilege to access the database.
- **REVOKE:** Take back the privilege to access the database

4. Transaction Control Language

- Manage transactions in the Database using the Transaction Control Language:
- **COMMIT**: Save the work
- **SAVEPOINT**: Set a point in transaction to rollback later
- **ROLLBACK**: Restores since last commit

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 (contd..)

- **Actors on the scene**
  - **Database administrators:**
    - Responsible for authorizing access to the database, for coordinating and monitoring its use, acquiring software and hardware resources, controlling its use and monitoring efficiency of operations.
  - **Database Designers:**
    - Responsible to define the content, the structure, the constraints, and functions or transactions against the database. They must communicate with the end-users and understand their needs.

Database Users(contd..)

- **Logical designers**: identify data and relationships between them for application without taking into account how software will be used
- **Physical designers**: look at how database model can be best mapped to physical storage(hard disk)

- **System Analysts and Application Programmers:**
  - Determine the requirements of end users
  - ***System analysts*** develop specifications for application that meet user requirements
  - ***Application Programmers*** then implement these specifications as application programs is appropriately tested, documented and maintained

Database Users(contd..)

- **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 or Parametric**: they make up a large section of the end-user population.
      - Examples are bank-tellers or reservation clerks who do this activity for an entire shift of operations.
    - **Sophisticated:** These include business analysts, scientists, engineers, others thoroughly familiar with the system capabilities.
      - Many use tools in the form of software packages that work closely with the stored database

Database Users(contd..)

- **Workers behind the scene**
  - Do not use database as part of their job
  - **Tool developers** that develop tools for database design, performance monitoring, graphical interfaces and test data generation
  - **Operators and maintenance personnel as well as system administrator** who are responsible for actual running and maintenance of hardware and software environment

Thank You!!