ADITYA ENGINEERING COLLEGE (A)

DATABASE MANAGEMENT SYSTEMS
1. INTRODUCTION

By

Dr. S Rama Sree
Professor in CSE & Dean(Academics)
Aditya Engineering College(A)
Surampalem.

CONTENTS

- Database system
- Database applications
- Characteristics (Database Vs File System) – Drawbacks of File System
- Advantages of Database systems
- Database Users(Actors on Scene, Workers behind the scene)
- Data Models; Concepts of Schema, Instance and data independence
  Three tier schema architecture for data independence
- Centralized and Client Server architecture for the database
- Database system structure

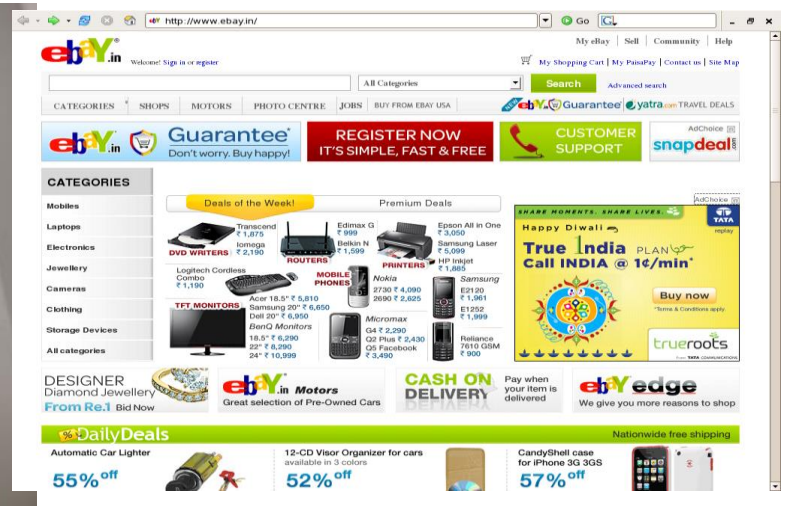
Definitions

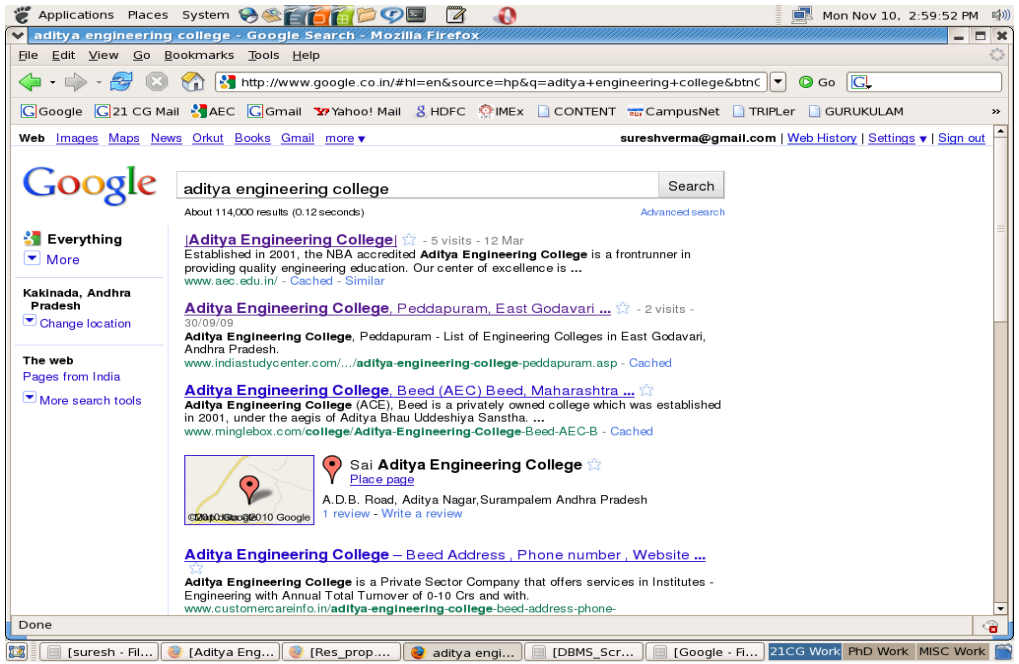
- A database-management system(DBMS) is a collection of interrelated data and a set of programs to access those data.
- The collection of data, usually referred to as the database, contains information relevant to an enterprise.
- The primary goal of DBMS is to provide a way to store and retrieve database information that is both convenient and efficient.
- It consists of schemas, tables, queries, reports, views, security.
- DBMS- It is a software designed in maintaining & utilizing large collections of data.
- RDBMS-It is a database management system based on the relational model. Ex:-Microsoft Access, Informix.

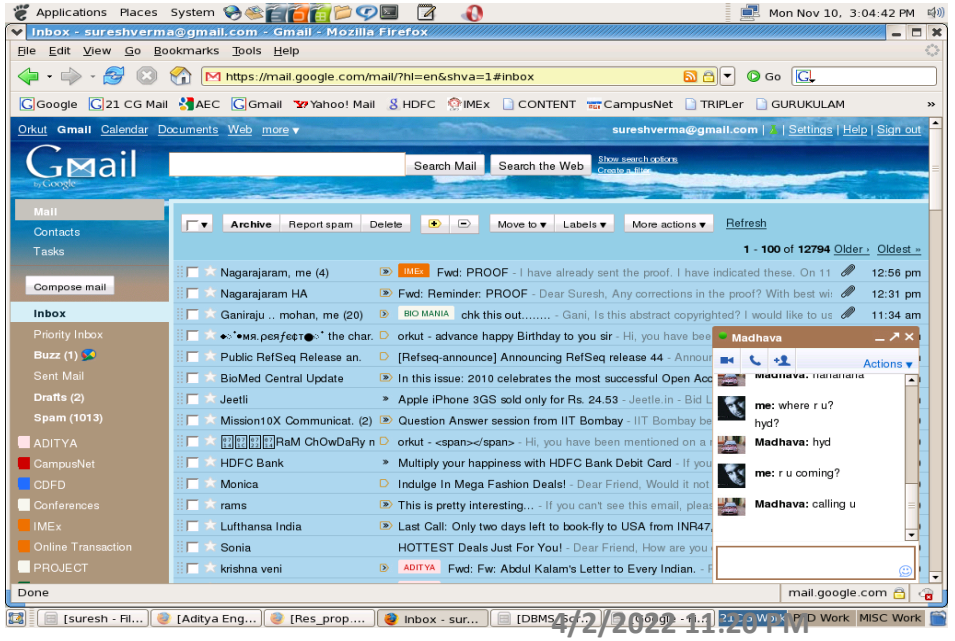
Applications

- Banking: all transactions
- Airlines: reservations, schedules
- Universities: student information, registration, grades
- Credit card transactions: purchase pf cards, monthly statemets
- Telecommunications: Calls, bills, communication networks
- Finance: sales, stocks, bonds
- Sales: customers, products, purchases
- Online retailers: order tracking, customized recommendations
- Manufacturing: production, inventory, orders, supply chain
- Human resources: employee records, salaries, tax deductions

Applications







Main Characteristics of the Database

1) Self Describing nature of database system

DBMS catalog- structure of file, type, storage, constraints

Information stored in catalog-meta data

2) Insulation between programs & data, data abstraction

Traditional file processing- structure of data files embedded in application programs

DBMS- structure of data files is stored in catalog separately from programs---**program-data independence**

Users define operations on data

Operations can have operation name, data type of its arguments

Implementation of operation is specified separately without changing interface—**program-operation independence**

Main Characteristics of the Database

3) Support of Multiple views of the data

Each user require a different perspective or view of the database

View contain virtual data that is derived from database files

Multiuser- users have a variety of distinct applications must provide facilities for defining multiple views.

4) Sharing of data & multi-user transaction processing

Mulituserr-allow multiple users to access the database at the same time

DBMS include concurrency control software-several users trying to update the same data in a controlled manner.

A Multiuser DBMS software is to ensure concurrent transactions operate correctly & efficiently.

Transaction-Executing program that includes one or more access such as reading or updating of database records.

Drawbacks of File Systems

- **Data redundancy and inconsistency -** Multiple file formats, duplication of information in different files.
  - phone no of customer when changed, it should be modified both in Savings account record and Checking accounts record, Same data stored in multiple places called Data Redundancy. When Redundant data is stored it occupies higher storage. If any one of the redundant record is forgotten to modify it leads to Data Inconsistency.
- **Difficulty in accessing data -** Need to write a new program to carry out each new task.
  - In banking sector, to find out who live within a particular postal code area, A new application program to be written other wise it is difficult to access the details. Conventional File System do not allow to retrieve data in a convenient and efficient manner.
- **Data isolation -** multiple files and different file formats leads to data isolation.
  - Writing new application programs to appropriate data is difficult, because the files have different file formats.

Drawbacks of File Systems

- **Integrity problems –** Data stored must satisfy certain types of Integrity constraints .

  Developers should implement these constraints while writing the programs. It is hard to add new constraints or change existing ones.

  Account balance should not fall below the prescribed amount (for ex: Rs. 500) When the min Balance increased to Rs. 1000 then new application program to be added which is difficult.

- **Concurrent access by multiple users -** Concurrent access by multiple users needed more faster and complicated programs, because uncontrolled concurrent accesses can lead to inconsistencies.

  • Two people reading a balance and updating it at the same time, other could not access the balance field, till one complete the access and modify it.

- **Security problems -** Hard to provide user access to some of the data, but not all (at least sensitive data).

  •Balance of the account should know to only the owner of the account, but not to others.

Drawbacks of File Systems

- **Atomicity Problems:** A computer system, like any other mechanical or electrical device, is subject to failure. In many applications, it is crucial that, if a failure occurs, the data be restored to the consistent state that existed prior to the failure.

  E.g. Funds transfer must be atomic. – it must happen in its entirety or not at all.

It is difficult to ensure atomicity using File system.

Why Use a DBMS?

- Data independence and efficient access.
- Reduced application development time.
- Data integrity and security
- Data administration.
- Concurrent access, recovery from crashes.

ADVANTAGES OF DBMS

- **Data Independence**: Application programs should not, ideally, be exposed to details of data representation and storage, The DBMS provides an abstract view of the data that hides such details
- **Efficient Data Access**: A DBMS utilizes a variety of sophisticated techniques to store and retrieve data efficiently. This feature is especially important if the data is stored on external storage devices
- **Data Integrity and Security:** If data is always accessed through the DBMS, the DBMS can enforce integrity constraints. For example, before inserting salary information for an employee, the DBMS can check that the department budget is not exceeded. Also, it can enforce access controls that govern what data is visible to different classes of users

- **Data Administration**: When several users share the data, centralizing the administration of data can offer significant improvements. Experienced professionals who understand the nature of the data being managed, and how different groups of users use it, can be responsible for organizing the data representation to minimize redundancy and for fine-tuning the storage of the data to make retrieval efficient
- **Concurrent Access and Crash Recovery**: A DBMS schedules concurrent accesses to the data in such a manner that users can think of the data as being accessed by only one user at a time. Further, the DBMS protects users from the effects of system failures.

- **Reduced Application Development Time:** Clearly, the DBMS supports important functions that are common to many applications accessing data in the DBMS. This, in conjunction with the high-level interface to the data, facilitates quick application development. DBMS applications are also likely to be more robust than similar stand-alone applications because many important tasks are handled by the DBMS (and do not have to be debugged and tested in the application)

RDBMS Softwares

Database Users

- Naive Users
- Application Programmers
- Sophisticated Users
- Specialized Users

Database Users

- **Naive Users –** these are the users who use the existing application to interact with the database. For example, online library system, ticket booking systems, ATMs etc which has existing application and users use them to interact with the database to fulfill their requests.

- **Application Programmers -** Computer professionals who write application programs. Programmers choose many tools to develop user interfaces. RAD(Rapid application development) tools enable an application programmer to construct forms and reports.

Database Users

- **Sophisticated Users –** These users interact with the system without writing programs. Instead, they form their requests in a database query language. They submit each such query toa Query processor. Analysts who submit the queries to explore data fall in this category. Eg: They are database developers, who write SQL queries to select/insert/delete/update data. They directly interact with the database by means of query language like SQL.

- **Specialized Users-**They are sophisticated users who write specialized database applications that do not fit into the traditional data-processing framework. Eg: The systems that store data with complex data types like graphics data, audio data etc.

Database Administrator

- One of the main reasons for using DBMS is to have central control of both the data and the programs that access those data.
- A person who has such central control over the system is called a database administrator (DBA).
- **DBA functions include**

1) **Schema Definition :** The DBA creates the original database schema by executing a set of data definition statements in the DDL
2) **Storage structure and access-method definition**

3) Schema and physical-organization modification. The DBA carries out changes to the schema and physical organization to reflect the changing needs of the organization, or to alter the physical organization to improve performance.

4) Granting of authorization for data access. By granting different types of authorization, the database administrator can regulate which parts of the data- base various users can access. The authorization information is kept in a special system structure that the database system consults whenever some- one attempts to access the data in the system.

5) Routine maintenance. Examples of the database administrator's routine maintenance activities are:

- Periodically backing up the database, either onto tapes or onto remote servers, to prevent loss of data in case of disasters such as flooding.
- Ensuring that enough free disk space is available for normal operations, and upgrading disk space as required.
- Monitoring jobs running on the database and ensuring that performance is not degraded by very expensive tasks submitted by some users.

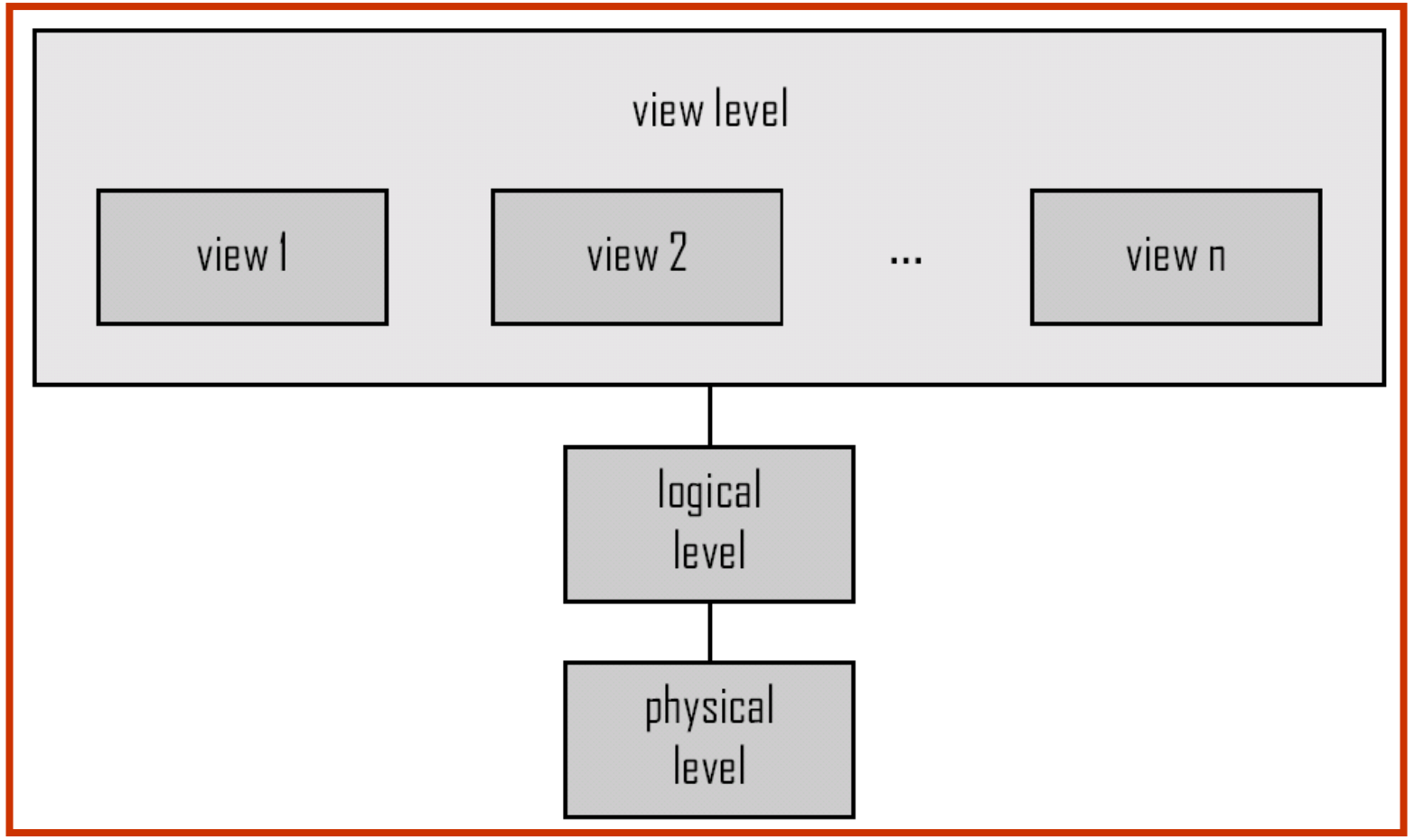
View of Data

Abstraction - Hiding irrelevant details from user and providing abstract view of data to the users. System users are not computer trained and hence developers hide the complexity from users through several levels of abstraction, to simplify users interactions with the system.

There are 3 levels of abstraction

- **Physical level**:-It is the lowest level of data abstraction. It describes **how data is actually stored** in database. The physical level describes complex low level data structures in detail.
- **Logical level**:- It is the next higher level of data abstraction. It describes what data are stored in the database, what relationships exist among those data. The database administrators who must decide what information to be kept in the database, use the logical level of abstraction.
- **View level**:- The highest level of data abstraction. This level describes the user interaction with database system. Many users of the database system do not need all this information instead they need to access only a part of the database the view level of abstraction exists to simplify their interaction with the system the system may provide many views for the same database

View of Data



- **Example:**

```
type customer = record
        customer-id : string;
        customer-name : string;
        customer-street : string;
        customer-city : string;
end;      // This code defines a new record type called customer with four fields. Each field has a name and a type .
```

A banking enterprise may have several such record types, including

- account, with fields account-number and balance
- employee, with fields employee-name and salary

At the physical level, a customer, account, or employee record can be described as a block of consecutive storage locations. This info is not known to users.

At the logical level, each such record is described by a type definition, as in the previous code segment, and the interrelationship of these record types is defined as well. Programmers using a programming language work at this level of abstraction. Similarly, database administrators usually work at this level of abstraction.

Finally, **at the view level**, computer users see a set of application programs that hide details of the data types. Similarly, at the view level, several views of the database are defined, and database users see these views.

Instances and Schemas

- Databases change over time as information is inserted and deleted. The collection of information stored in the database at a particular moment is called an instance of the database.
- The overall design of the database is called the database schema. Schemas are changed infrequently
- Database systems have several schemas, partitioned according to the levels of abstraction.
- The **physical schema** describes the database design at the physical level
- The **logical schema/Conceptual** describes the database design at the logical level. A data definition language (DDL) is used to define the external and conceptual schema.
- A database may also have several schemas at the **external/view level**, sometimes called subschemas, that describe different views of the database.

DATA INDEPENDENCE

- A very important advantage of using a DBMS is that it offers data independence. That is, application programs are insulated from changes in the way the data is structured and stored.
- Data independence is achieved through use of the three levels of data abstraction; in particular, the conceptual schema and the external schema provide distinct benefits in this area.
- Relations in the external schema (view relations) are in principle generated on demand from the relations corresponding to the conceptual schema.
- It helps you to modify the schema at one level of the database system without altering the schema at the next higher level.

Data Independence

- Users can be shielded from changes in the logical structure of the data, or changes in the choice of relations to be stored. This property is called logical data independence.
- Logical level data independence

  we can change the relations, logical structure of data but external

  schema will remain same.
- The conceptual schema insulates users from changes in physical storage details. This property is referred to as physical data independence.
- Physical level data independence

  we can change the storage details without altering applications.

Data Models

- Underlying the structure of a database is the data model: a collection of conceptual tools for describing data, data relationships, data semantics and consistency constraints. It provides a way to describe the design of a database at physical, logical and view level.

Categories of data models

- Relational model
- Entity-Relationship data model
- Object-based data models
- Semi structured data model (XML)
- Other older models: Network model & Hierarchical model

Data Models

- **Relational model**:- It is the most widely used data model. It uses a collection of tables to represent both data and relationships among those data. Each table has multiple columns and rows. Each column has a unique name. Relational model is an example of a record based model.
- **Entity Relationship model**:- It is based on perception of a real world that consists of objects called entities and relationships among those objects. ER model is widely used in database design.
- **Object based data model**:-It is an extending the ER model with notations of encapsulation, methods, object identity. Object relational data model combines the features of object oriented data model and relational data model.

Data Models

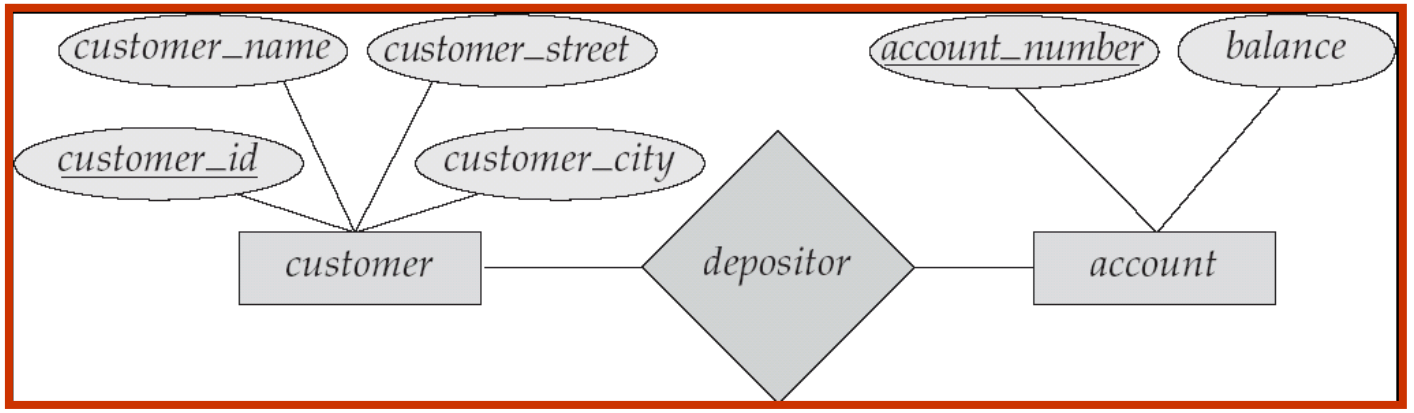
- **Semi structured data model**:-It permits the specification of data where individual data items of same type may have different set of attributes.

Old Models

- **Network model**:- It represents data as record types and also represents a limited type of 1:N relationship. One instance of record to many record instances using some pointer linking mechanisms.
- **Hierarchical model**:- It represents data as hierarchical tree structure. Each hierarchy represents a number of related records.

Data Models

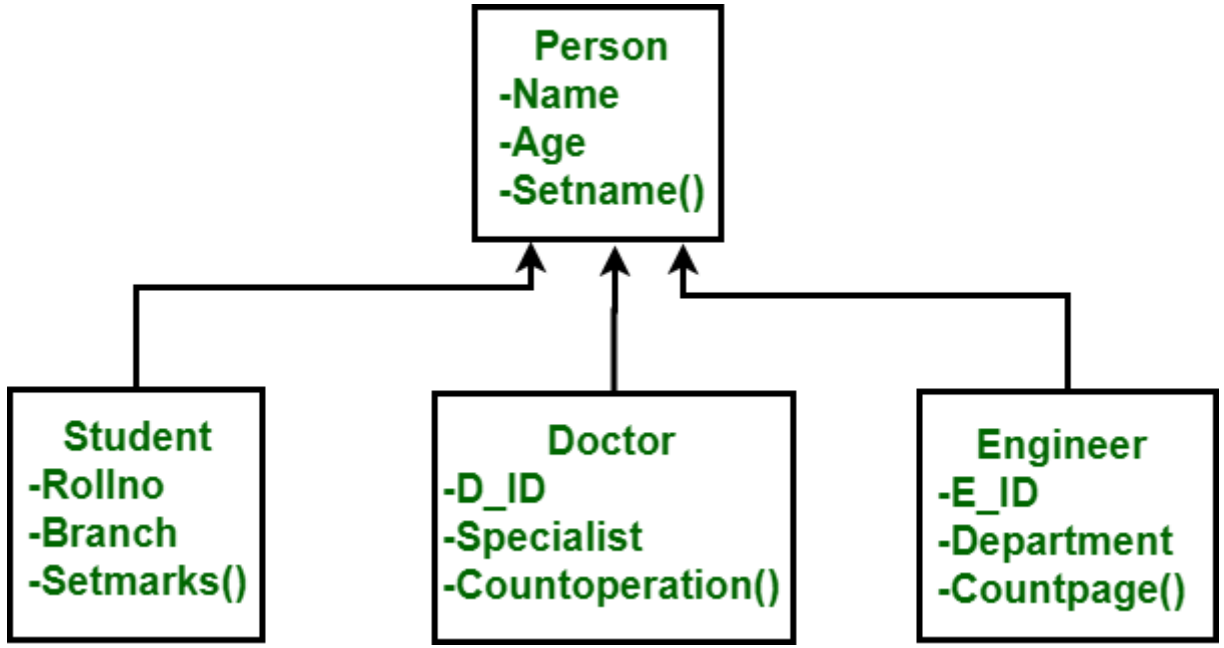
Sample Entity-Relationship (ER) model



Sample Relational model

customer_id	customer_name	customer_street	customer_city	account_number
192-83-7465	Johnson	12 Alma St.	Palo Alto	A-101
192-83-7465	Johnson	12 Alma St.	Palo Alto	A-201
677-89-9011	Hayes	3 Main St.	Harrison	A-102
182-73-6091	Turner	123 Putnam St.	Stamford	A-305
321-12-3123	Jones	100 Main St.	Harrison	A-217
336-66-9999	Lindsay	175 Park Ave.	Pittsfield	A-222
019-28-3746	Smith	72 North St.	Rye	A-201

Object Based Data Model



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 updates.
- In practice, the data definition and data manipulation languages are not two separate languages; instead they simply form parts of a single database language, such as the widely used SQL**(Structured Query Language)** language.

Data-Definition Language:

- We specify a database schema by a set of definitions expressed by a special language called a data-definition language (DDL).
- For instance, the following statement in the SQL language defines the account table:

  create table account (account-number char(10), balance integer)

- Execution of the above DDL statement creates the account table.
- In addition, it updates a special set of tables called the data dictionary or data directory.
- A data dictionary contains metadata—that is, data about data. The schema of a table is an example of metadata. A database system consults the data dictionary before reading or modifying actual data

Database Languages

Data-Manipulation Language:

- A data-manipulation language (DML) is a language that enables users to access or manipulate data as organized by the appropriate data model. The types of access are
  - The retrieval of information stored in the database
  - The insertion of new information into the database
  - The deletion of information from the database
  - The modification of information stored in the database
- There are basically two types of DML:
  - **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
- A query is a statement requesting the retrieval of information. The portion of a DML that involves information retrieval is called a query language

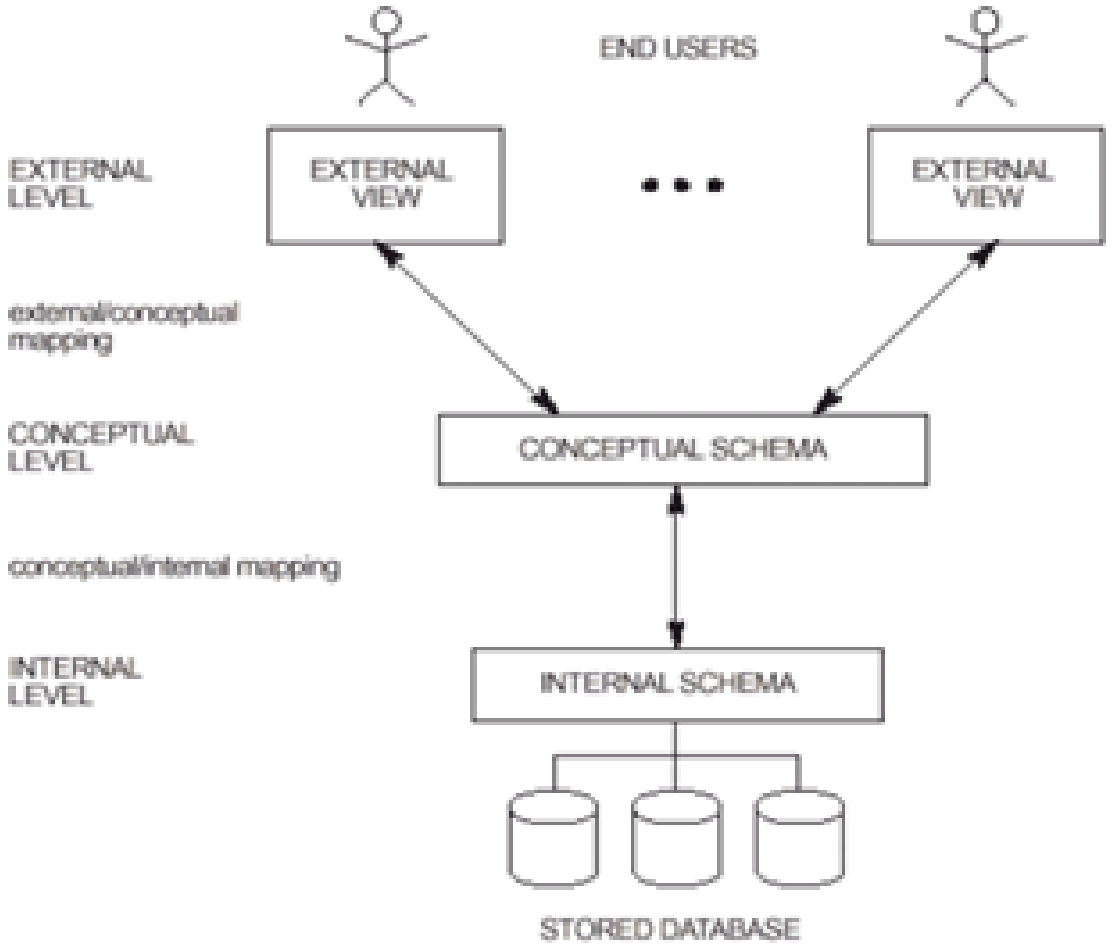
- The Query language of SQL is non-procedural
- This query in the SQL language finds the name of the customer whose customer-id is 192-83-7465:

select customer.customer-name from customer

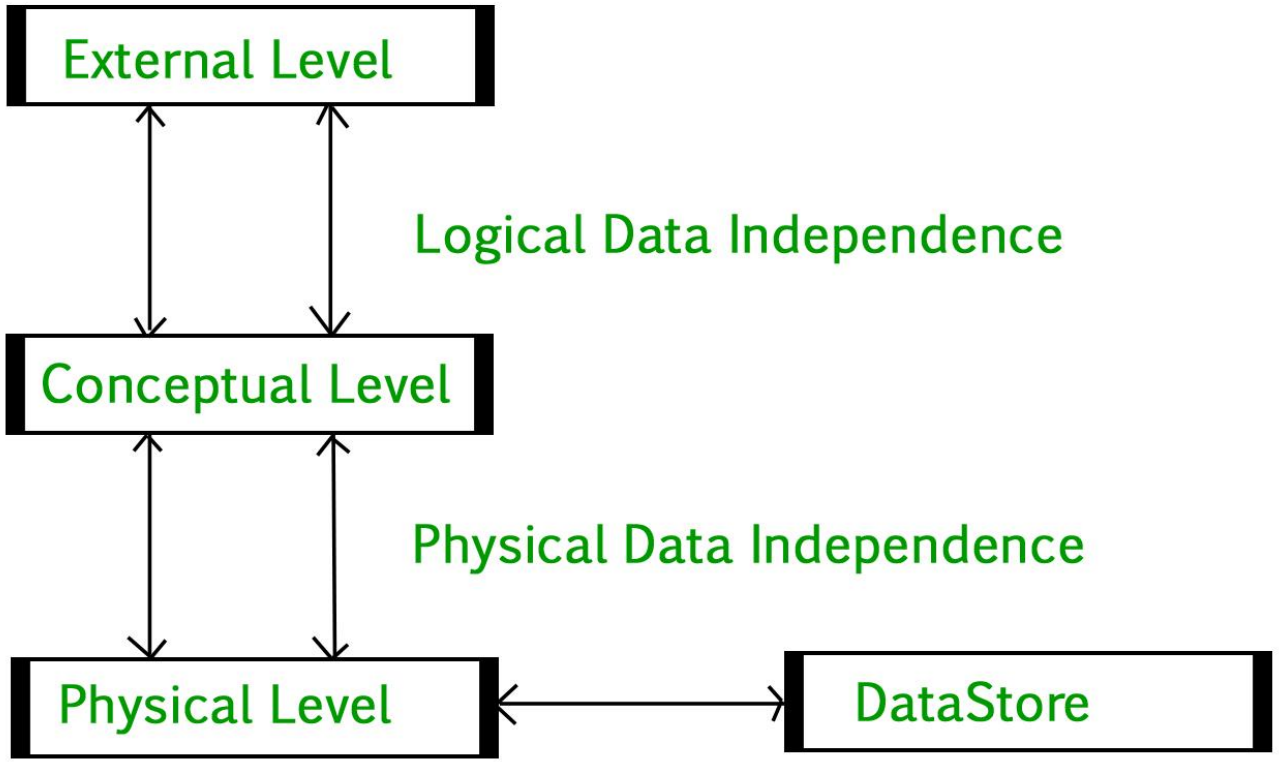
where customer.customer-id = 192-83-7465

- The query specifies that those rows from the table customer where the customer-id is 192-83-7465 must be retrieved, and the customer-name attribute of these rows must be displayed

Three Schema Architecture



Three- Tier Architecture for Data Independence



1.Physical Data Independence: Any change in the physical location of tables and indexes should not affect the conceptual level or external view of data.

2.Conceptual Data Independence: The data at conceptual level schema and external level schema must be independent. This means a change in conceptual schema should not affect external schema. e.g.; Adding or deleting attributes of a table should not affect the user's view of the table.

Database Architecture

The database architecture consists of the following components

- Database system:- It resides along with its query processing languages. We have the relations that define the data and their constraints.
- Application:- It presents abstract view of the database and acts as a mediator between the end user and database. Multiple views of the database can be provided by the application.
- End users/Clients :- users who interact with the database directly or indirectly. Few users know nothing about existence of the database.
- We may have **Centralized DBMSs Architectures** or **Client/Server Architectures** for DBMSs

Centralized DBMSs Architectures

- The PCs and workstations consists of all the DBMS functionality, application program execution, and user interface processing on one machine. So, the DBMS itself is still a **centralized** DBMS.
- Architectures for DBMSs have followed trends similar to those for general computer system architectures.
- A DBMS is centralized if all the data is stored in a single computer site.
- A centralized DBMS can support multiple users but DBMS software & database reside totally at a single computer site.

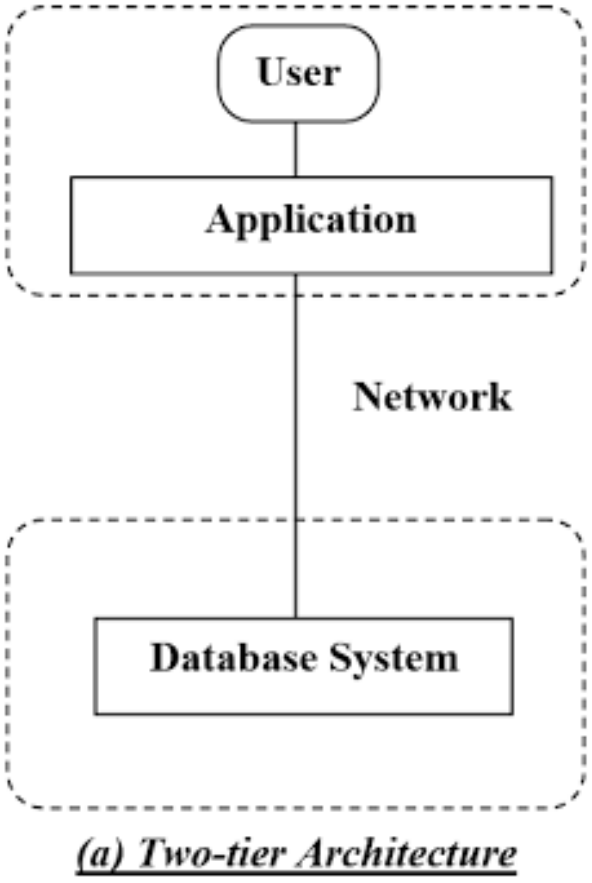
Client/Server Architectures for DBMSs

- Most users of a database at the site of the database system, but connect to it through a network.
- We can therefore differentiate between client machines - on which remote database users work, and server machines - on which the database system runs.
- The Client/Server Architectures for DBMSs are usually categorized as two-tier architecture or three-tier architecture.

- In a **two-tier architecture**, the application is partitioned into a component that resides at the client machine, which invokes database system functionality at the server machine through query language statements. Eg: JDBC interface is used for interaction between the client and server.

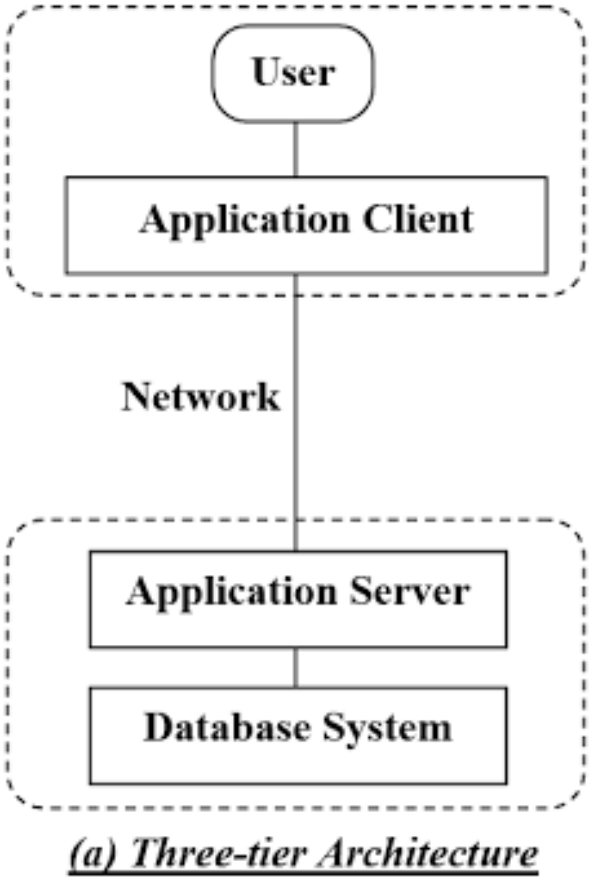
- In a **three-tier architecture**, the client machine acts as merely a front end and does not contain any direct database calls. Instead, it communicates with an application server, through an interface.
- The application server in turn communicates with a database system to access data. The business logic of the application, is embedded in the application server, instead of being distributed across multiple clients.
- Three-tier applications are more appropriate for large applications, and for applications that run on the World Wide Web.

Database Architecture



client

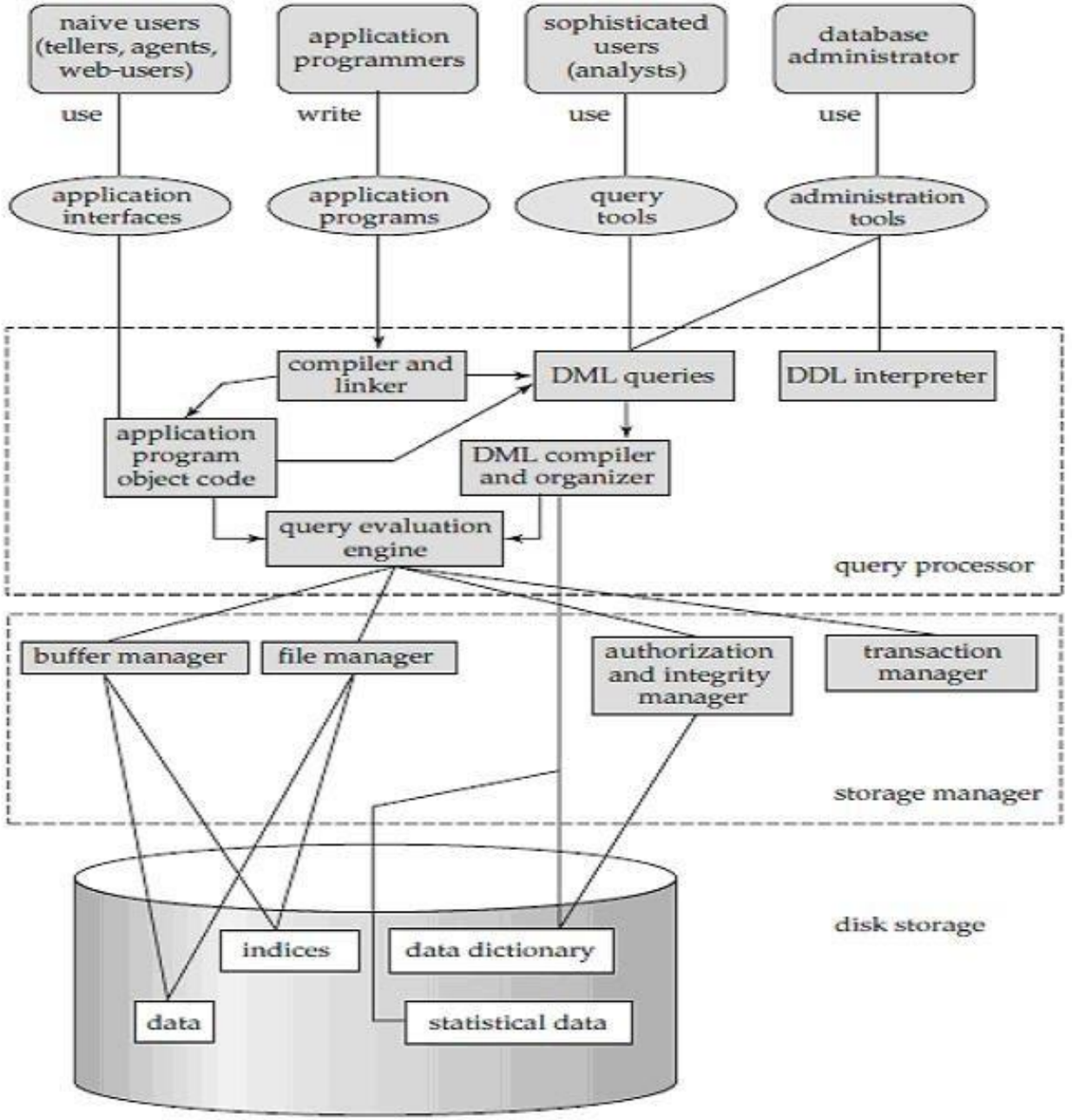
Server



(a) Two-tier Architecture

(a) Three-tier Architecture

DATABASE SYSTEM STRUCTURE



Database System Structure

1) **Types of users** – Naïve users, Application Programmers, Sophisticated users, Specialized users

2) Query Processor

The query processor components include

- **DDL interpreter**: It interprets DDL statements and records the definitions in the data dictionary.
- **DML compiler :** It translates DML statements in a query language into an evaluation plan consisting of low-level instructions that the query evaluation engine understands.

  A query can usually be translated into any of a number of alternative evaluation plans that all give the same result.

  The DML compiler also performs query optimization, i.e., it picks the lowest cost evaluation plan from among the alternatives.
- **Query evaluation engine :** It executes low-level instructions generated by the DML compiler.

Database System Structure

3) **Storage Manager**

Buffer Manager:- It is responsible for fetching data from disk storage into main memory and deciding what data to cache in main memory.

File manager:- Manages the allocation of space on disk storage & data structures used to represent information stored on disk.

Authorization & integrity manager:- This tests for the satisfaction of integrity constraints and checks the authority of users to access data.

Transaction Manager:- Ensure the database remains in a consistent state despite of system failures & that concurrent transaction executions proceed without conflicting.

Database System Structure

4) **Disk Storage**:-

Data Files:- Stores the database itself

Data Dictionary:- Stores metadata about the structure of database.

Indices:- This provide fast access to data items to hold particular values

Statistical data:- Stores Statistical information about the data in the database

References

Text Books:

- Database Management Systems, 3/e, Raghurama Krishnan, Johannes Gehrke, TMH.
- Database System Concepts,5/e, Silberschatz, Korth, TMH.

Reference Books:

- Introduction to Database Systems, 8/e C J Date, PEA.
- Database Management System, 6/e Ramez Elmasri, Shamkant B. Navathe, PEA
- Database Principles Fundamentals of Design Implementation and Management, Corlos Coronel, Steven Morris, Peter Robb, Cengage Learning.

Web Links:

- https://nptel.ac.in/courses/106/105/106105175/
- https://www.geeksforgeeks.org/introduction-to-nosql/
- https://beginnersbook.com/2015/05/normalization-in-dbms/


Thank
You