Chapter 1: Introduction to database management system

Prepared By: Dr. Vipul Vekariya

.

Chapter 1: Introduction

- Purpose of Database Systems
- Database Languages
- Relational Databases
- Database Design
- Data Models
- Database Internals
- Database Users and Administrators
- Overall Structure
- History of Database Systems

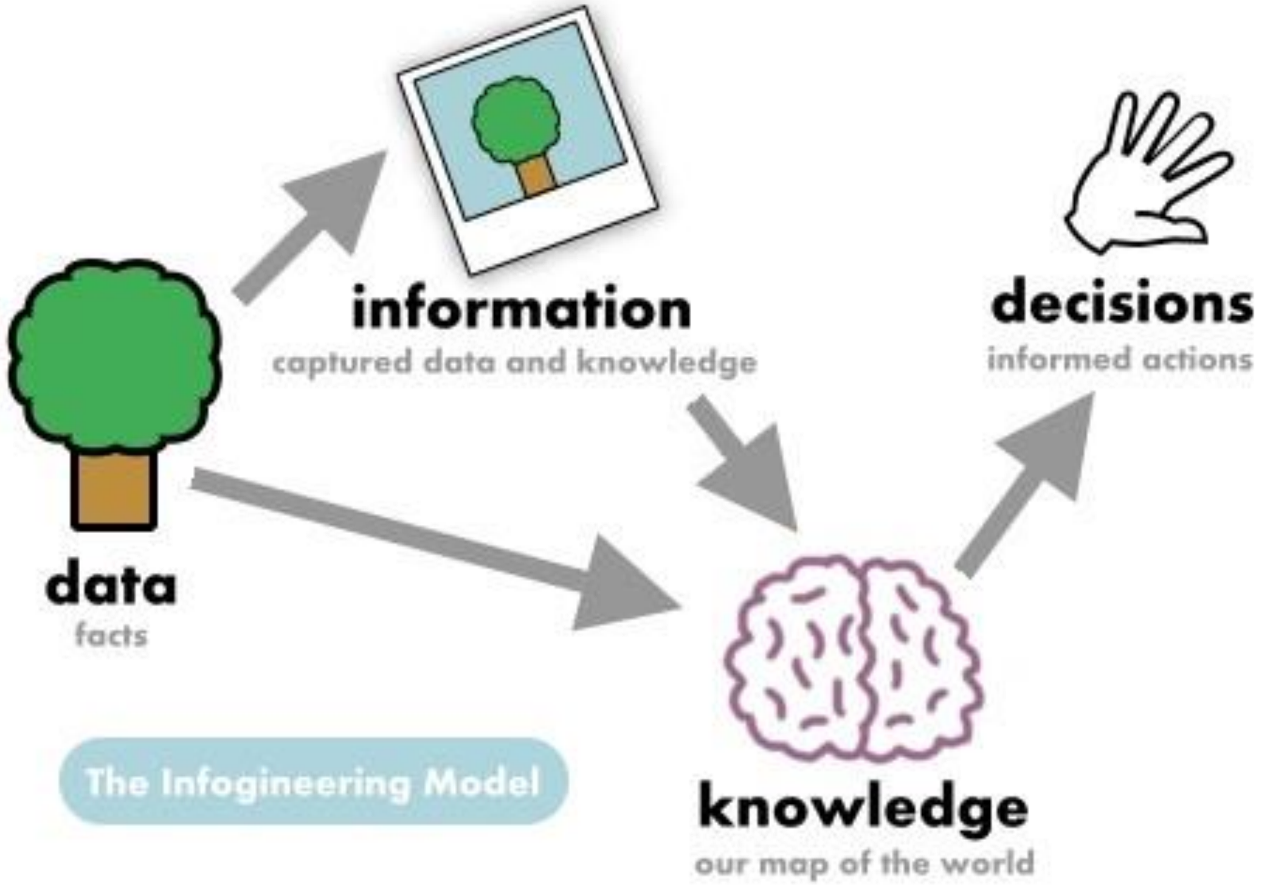
Data Vs Information

- Data are row or isolated facts from which the required information is produced.
- Data are distinct piece of information ,usually formatted in special way.
- They are binary computer representation of stored logical entities.
- Data can exit in variety of the form.
- Number or text on a paper , bit or byte stored in computer, fact stored in persons mind.
- Data also can be an object such as document, photographic image and video data.

DATA VS INFORMATION(CONT.)

- Data and information are closely related.
- Information is processed, organized, or summarize data.
- Data are processed to create information which is meaningful to the user.
- Information is used for make decision or to interpret the data to get the meaning.
- Database may contain either data or information(or both), according to organization needs.

Information model



Data Warehouse

- Data warehouse is a collection of data designed to support management in the decision making process.
- It is subject oriented, integrated, time variant collection of data.
- It is a unique kind of database which focus on business intelligence and decision making process.
- It contains wide Varity of data that represent coherent picture of business condition at a single point of time.

Why Database?

- Compactness : no paper works (no Bunch of file)
- Speed: machine can retrieve data faster then human.
- Less drudgery: maintain files by hand is eliminating.
- Currency: accurate and up to date information is available on demand.
- Protection : data can be better protected against unintentional loss and unlawful access.

Database Management System (DBMS)

- A DBMS is collection of logically related data stored together that is defined to meet the information needs of an organization.
- A DBMS is a collection of persistent data that is used by application system of some given enterprise.
- It is also say that, is a collection of interrelated data stored together without harmful or unnecessary redundancy.
- Finally it is just computerized record keeping system.
- A DBMS is a collection of interrelated data and a set of program to access those data.

Purpose of Database Systems

- In the early days, database applications were built directly on top of file systems

- Drawbacks of using file systems to store data:
  - Data redundancy and inconsistency
  - Difficulty in accessing data
  - Integrity problems
    - Integrity constraints (e.g. account balance > 0) become "buried" in program code rather than being stated explicitly
    - Hard to add new constraints or change existing ones

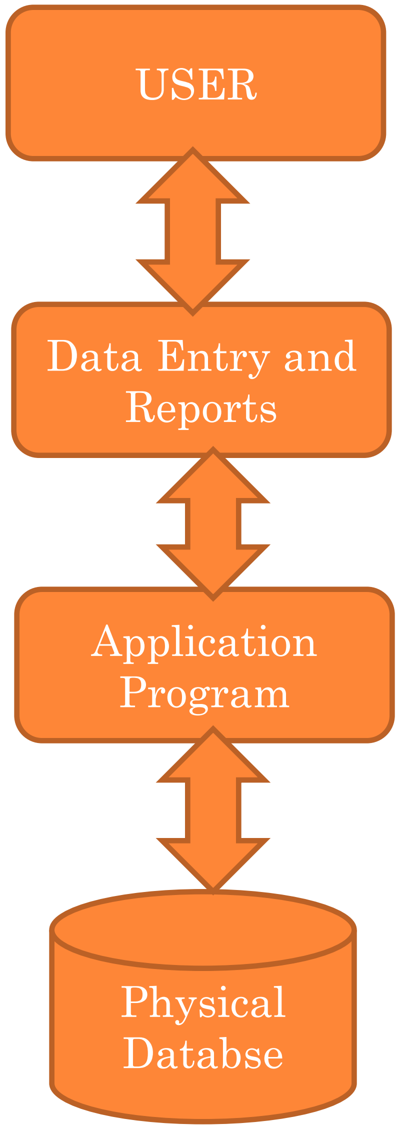
  - Atomicity of updates
    - Failures may leave database in an inconsistent state with partial updates carried out
    - Example: Transfer of funds from one account to another should either complete or not happen at all

Purpose of Database Systems (Cont.)

- Drawbacks of using file systems (cont.)
  - Concurrent access by multiple users
    - Concurrent accessed needed for performance
    - Uncontrolled concurrent accesses can lead to inconsistencies
      - Example: Two people reading a balance and updating it at the same time
  - Security problems
    - Hard to provide user access to some, but not all, data
    - Poor data control
    - Limited data sharing
- Database systems offer solutions to all the above problems

Database System Environment

- Data
- Hardware
- Software
- User(people)



Benefit of DBMS

- The data can be shared
- Redundancy can be reduced
- Inconsistency can be avoided
- Transaction support can be provided
- Integrity can be maintained
- Security can be enforced
- Conflicting requirement can be balanced
- Standards can be enforced.
- Increased concurrency
- Improved backup and recovery services
- Improved data quality

INSTANCES AND SCHEMAS

- Similar to types and variables in programming languages

- **Schema** – the overall design of the database is called database schema.
  - Example: The database consists of information about a set of customers and accounts and the relationship between them)
  - The plan of the database is known as schema.
  - Analogous to type information of a variable in a program

- **Instance** – the collection of information stored in the database at a particular point in time.
  - Analogous to the value of a variable

DATA MANIPULATION LANGUAGE (DML)

- A DML is a language that enable user to access or manipulate the data.
- DML also known as query language
- Retrieval of information
- Insertion of new information
- Deletion of information
- Modification of information

- Two classes of languages
  - **Procedural** – user specifies what data is required and how to get those data
  - **Declarative (nonprocedural)** – user specifies what data is required without specifying how to get those data

- SQL is the most widely used query language

Data Definition Language (DDL)

- Specification notation for defining the database schema

  Example: **create table** *account* (
  account_number **char**(10),
  branch_name **char**(10),
  balance **integer**)

  - Database schema
  - Data *storage and definition* language
    - Specifies the storage structure and access methods used
  - Integrity constraints
    - Domain constraints
    - Referential integrity (e.g. *branch_name* must correspond to a valid branch in the *branch* table)
  - Authorization

Database Design

The process of designing the general structure of the database:

- Logical Design – Deciding on the database schema. Database design requires that we find a “good” collection of relation schemas.
  - Business decision – What attributes should we record in the database?
  - Computer Science decision – What relation schemas should we have and how should the attributes be distributed among the various relation schemas?
- Physical Design – Deciding on the physical layout of the database

Data Models

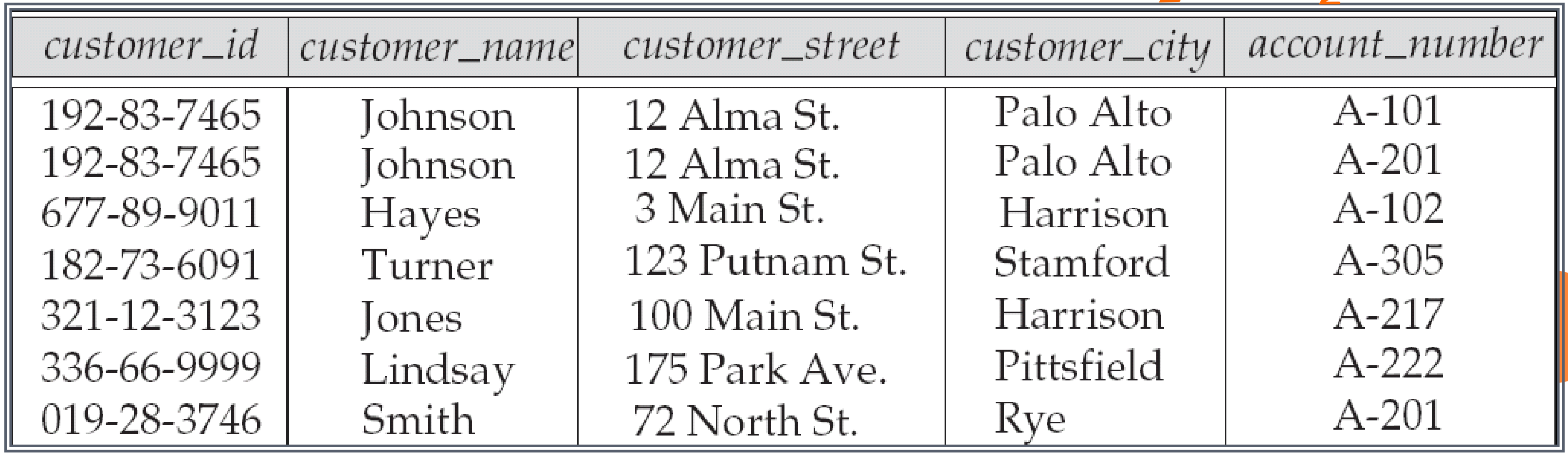
- A collection of tools for describing
  - Data
  - Data relationships
  - Data semantics
  - Data constraints
- It is a way to describe the design of the database.
- Types of data model:
- Relational model
- Entity-Relationship data model (mainly for database design)
- Object-based data models (Object-oriented and Object-relational)

- Network model
- Hierarchical model

Relational Model

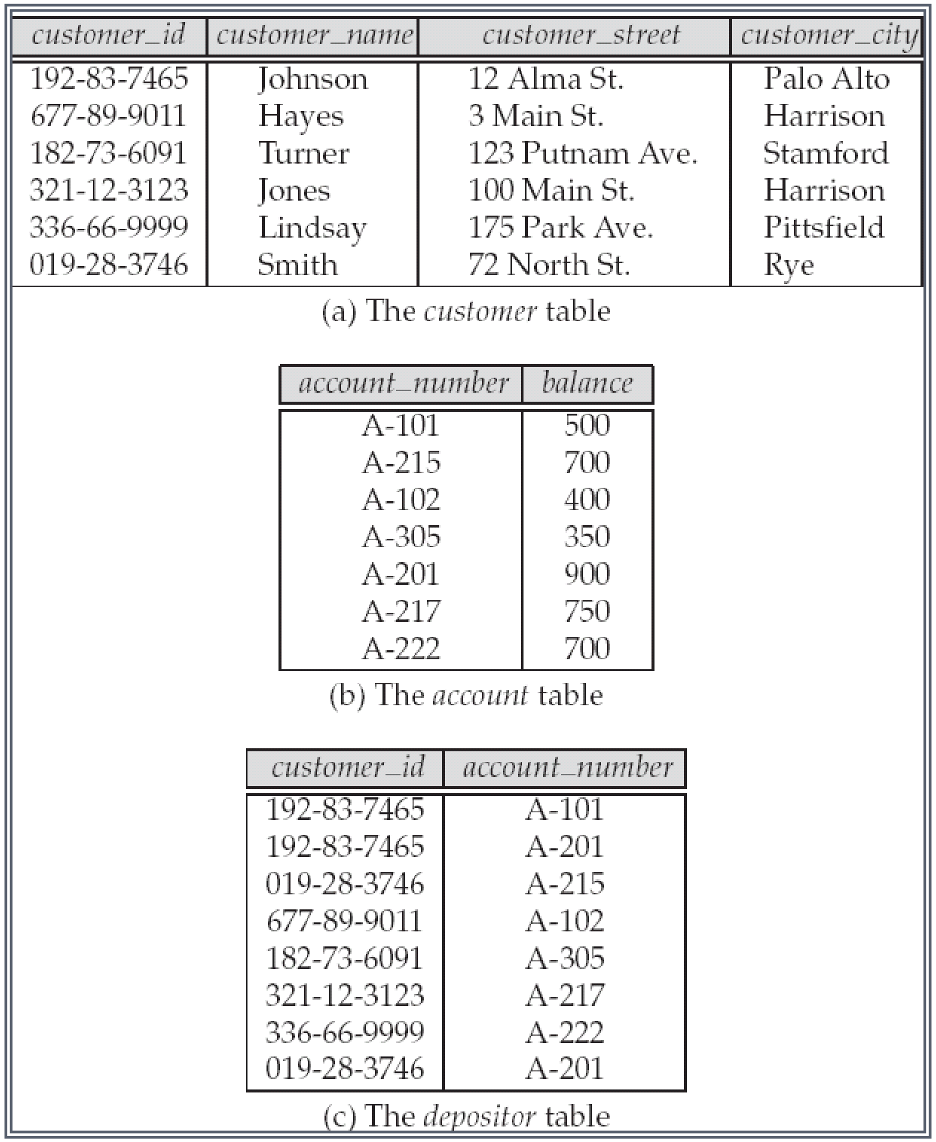
- Dr. E.F. Codd of IBM research first introduced Relational data model.
- The relational model use a collection of table to represent both data and relationship among those data.
- Each table has multiple column.
- Each column has unique name.
- The relational data model is the most widely used the data model.

Attributes



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

A Sample Relational Database



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

(a) The *customer* table

account_number	*balance*
A-101	500
A-215	700
A-102	400
A-305	350
A-201	900
A-217	750
A-222	700

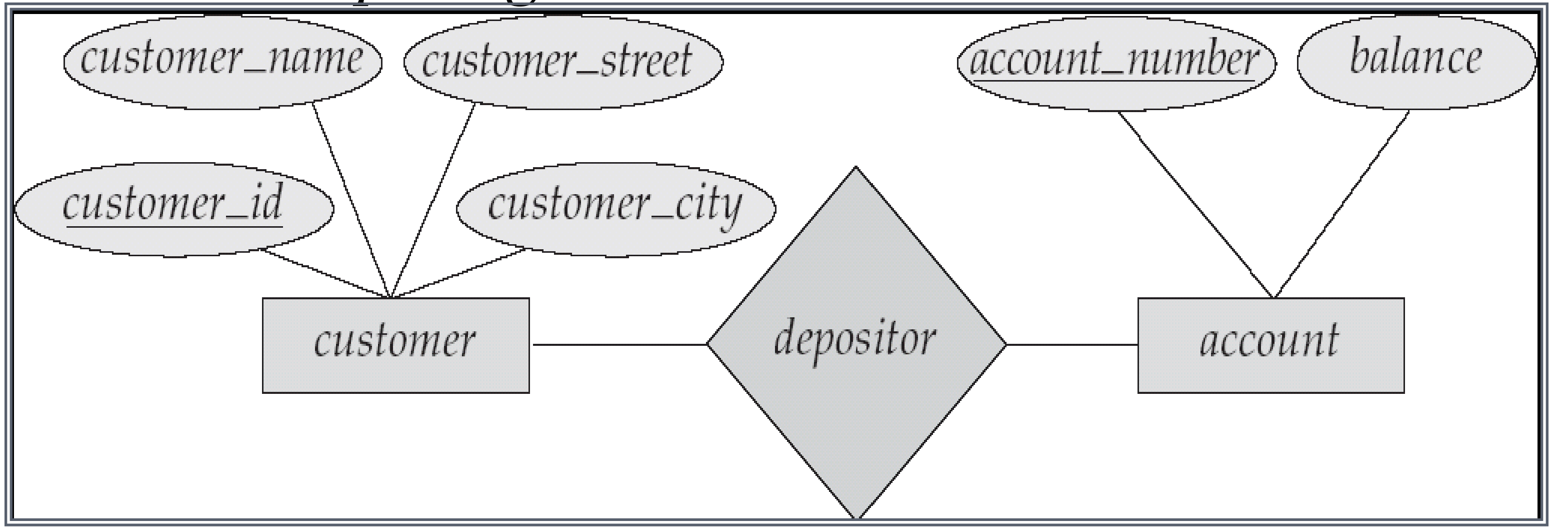
(b) The *account* table

customer_id	*account_number*
192-83-7465	A-101
192-83-7465	A-201
019-28-3746	A-215
677-89-9011	A-102
182-73-6091	A-305
321-12-3123	A-217
336-66-9999	A-222
019-28-3746	A-201

(c) The *depositor* table

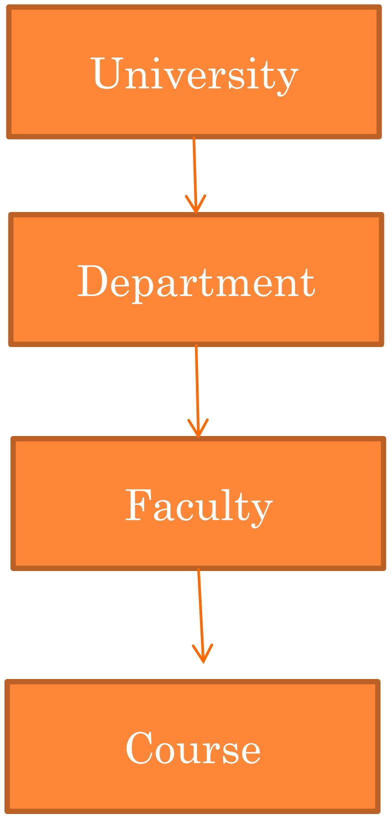
The Entity-Relationship Model

- Models an enterprise as a collection of *entities* and *relationships*
  - Entity: a "thing" or "object" in the enterprise that is distinguishable from other objects
    - Described by a set of *attributes*
  - Relationship: an association among several entities
- Represented diagrammatically by an *entity-relationship diagram:*



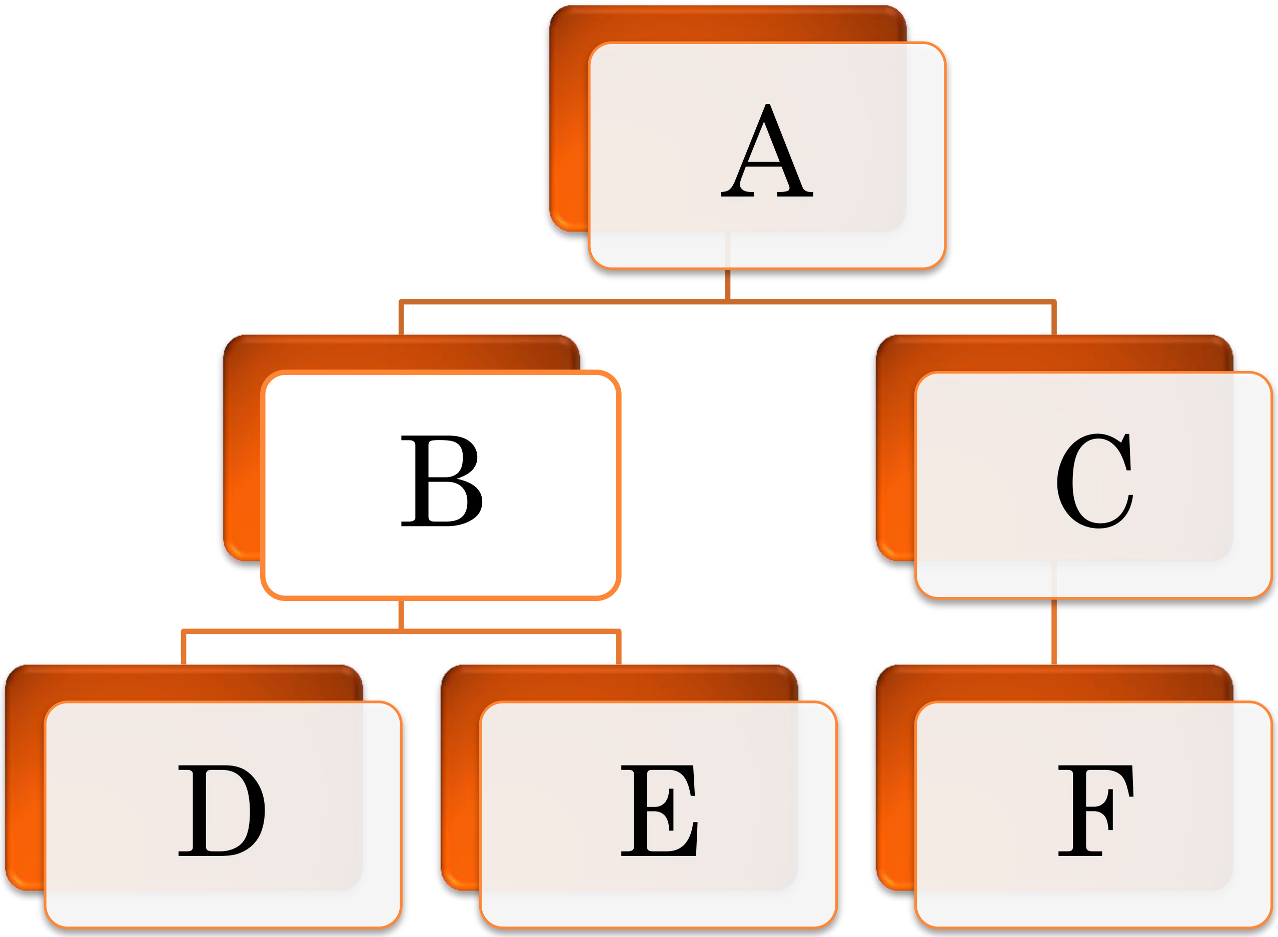
Hierarchical model

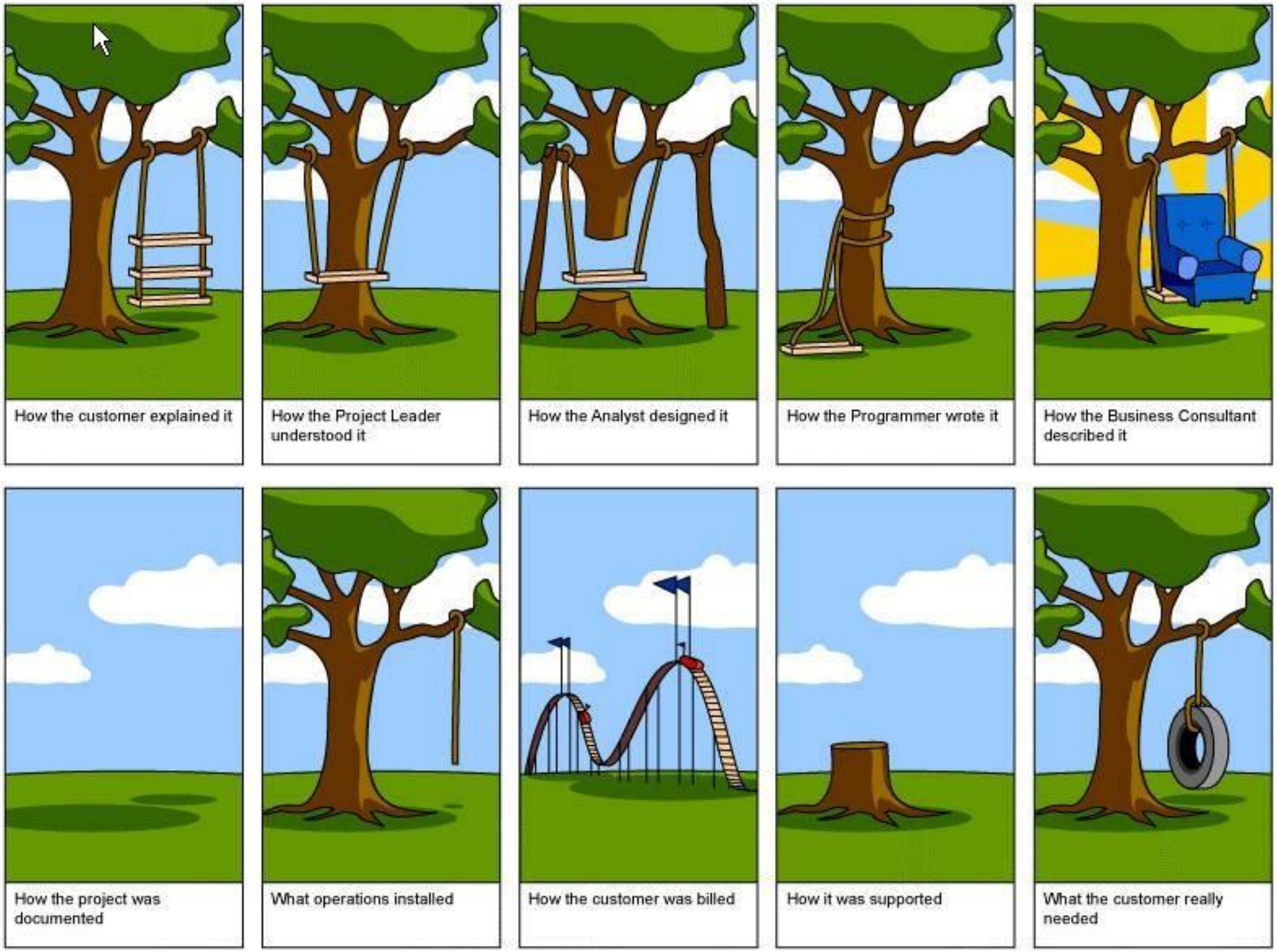
- This model is represented by an up- down tree.
- The user perceives the hierarchical database as a hierarchy of segments.
- A segments is equivalent of file system records type.



Network Data Model.

- It is similar to hierarchical model except that a record can have multiple parents.




How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

The Three level ANSI-SPARC Database Architecture

- Internal Level:
  - Is the one closest to the physical storage.
  - It is the one concerned with the way how the data is stored inside the system.
  - It describe the data structure, file structure, and access method to be used by the database.

- External Level:
  - Is the one closest to the user.
  - It is the one concerned with the way the data is seen by individual users.
  - In this level user seen only interested data for them.

- Conceptual Level:
  - is a level of indirection between other two.
  - All the database entities and the relationship among them are included.

Levels of Abstraction

- **Physical level:** describes how a record (e.g., customer) is stored.
  - storage space allocation for data storage.
  - record description for storage with storage size.
  - record placement
  - Data compression and data encryption technique.

- **Logical level:** describes data stored in database, and the relationships among the data.

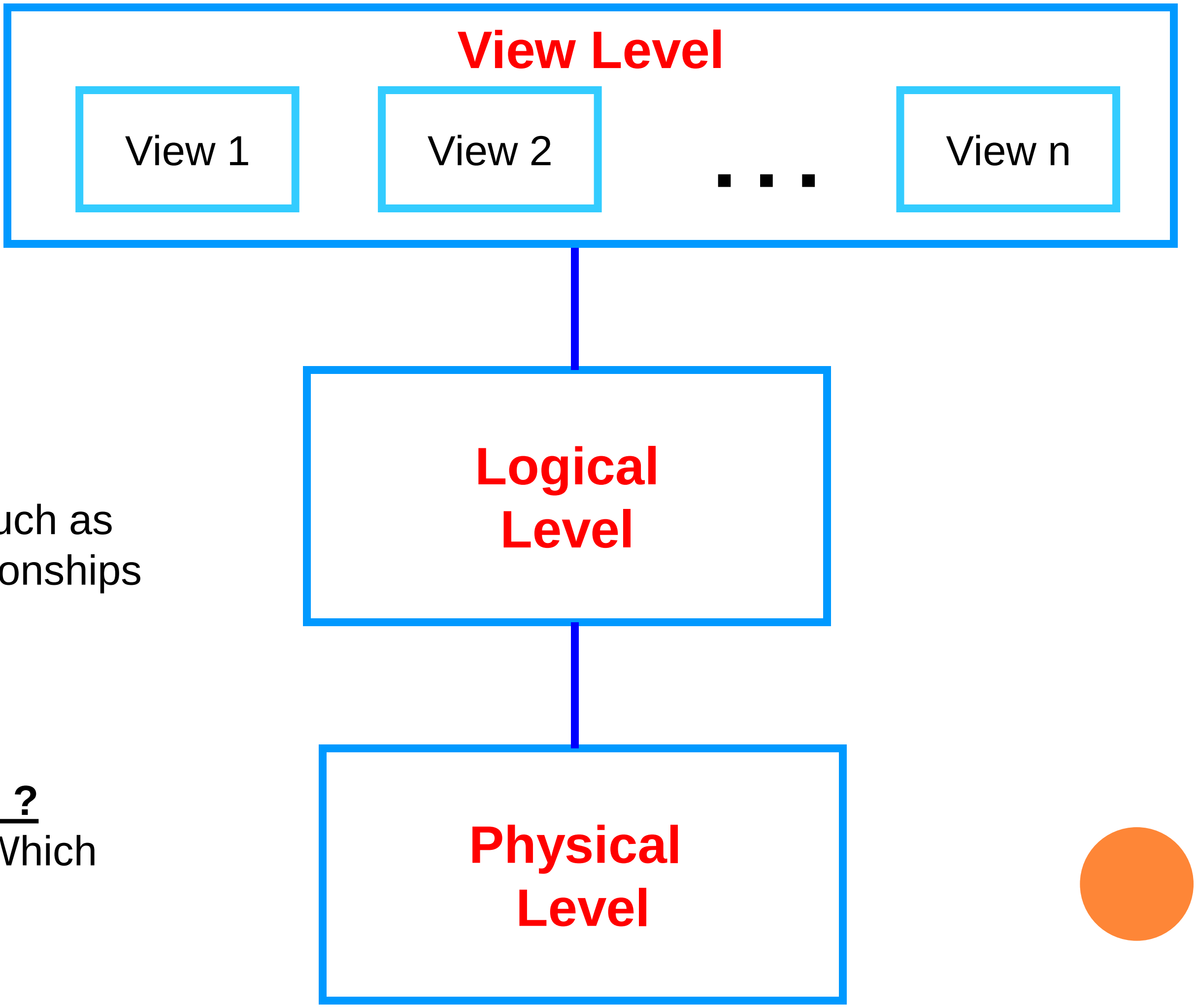
```
type customer = record
        customer_id : string(20);
        customer_name : string(45);
        customer_street : string(20);
        customer_city : string(20);
    end;
```

  - Constraint on data.
  - Check to retain data consistency and integrity
  - security information

- **View level:** application programs hide details of data types. Views can also hide information (such as an employee's salary) for security purposes.

Data Abstraction

<u>What data users and application programs see ?</u>



<u>What data is stored ?</u>
describe data properties such as data semantics, data relationships

<u>How data is actually stored ?</u>
e.g. are we using disks ? Which file system ?

Database Management System Internals

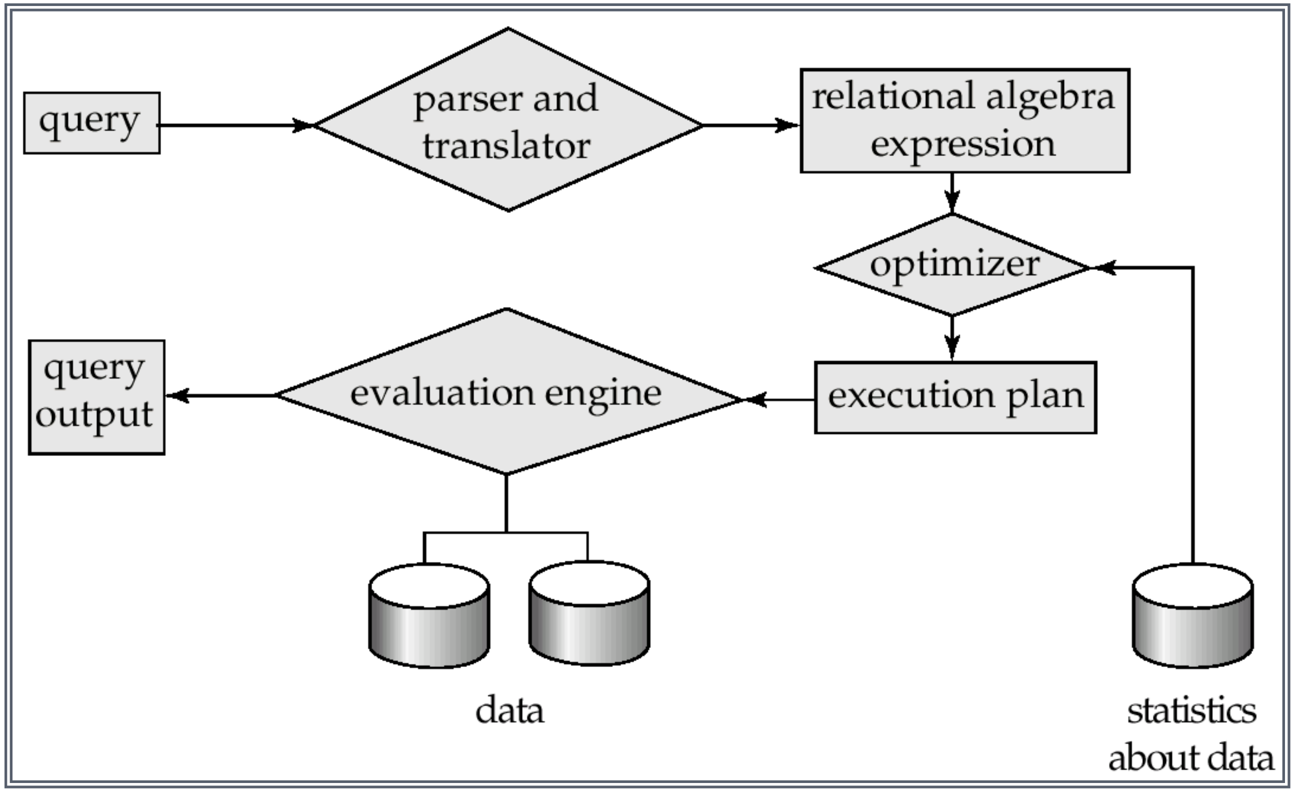
- Storage management
- Query processing
- Transaction processing

Storage Management

- **Storage manager** is a program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.
- The storage manager is responsible to the following tasks:
  - Interaction with the file manager
  - Efficient storing, retrieving and updating of data
  - Authorization and integrity manager: check for integrity constraint and check for authority of user.
  - Transaction manager: responsible for database remains in consistent state when system failure. And concurrent transaction execution.
  - File manager: manages the allocation of space on disk storage.
  - Buffer manager: is responsible for fetching data from disk storage in to main memory and deciding what data to cache in memory.
  - Data dictionary: which store metadata (data about data)
  - Indices: which can provide fast access.

Query Processing

1. Parsing and translation
2. Optimization
3. Evaluation



Query Processing (Cont.)

- DDL Interpreter: which interpreter DDL statements and record the definition in the data dictionary.
- DML Complier: Which translate DML statements in to low level instruction.
- Query Evolution Engine: which execute low level instruction generated by DML complier.

Transaction Management

- A **transaction** is a collection of operations that performs a single logical function in a database application
- **Transaction-management component** ensures that the database remains in a consistent (correct) state despite system failures (e.g., power failures and operating system crashes) and transaction failures.
- **Concurrency-control manager** controls the interaction among the concurrent transactions, to ensure the consistency of the database.

Database Users

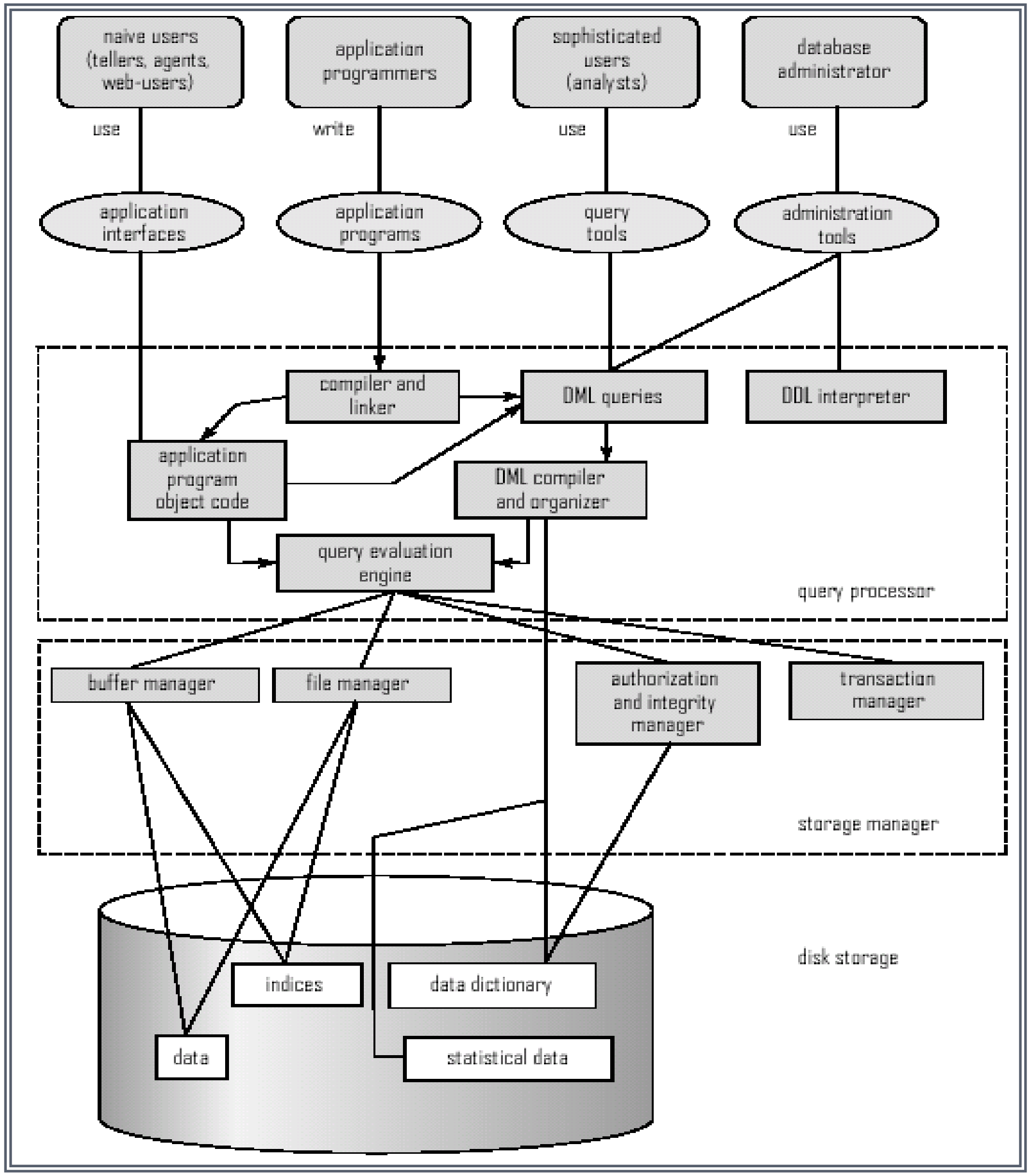
Users are differentiated by the way they expect to interact with the system

- **Application programmers** – is computer professional who write application program and access database.
- **Sophisticated users** –(without writing program) form requests in a database query language
- **Specialized users** – write specialized database applications that do not fit into the traditional data processing framework
- **Naïve users** – invoke one of the permanent application programs that have been written previously
  - Examples, people accessing database over the web, bank tellers, clerical staff

Database Administrator

- Coordinates all the activities of the database system
  - has a good understanding of the enterprise's information resources and needs.
- Database administrator's duties include:
  - Schema Definition
  - Storage structure and access method definition
  - Granting users authority to access the database
  - Periodically Backing up data
  - Ensuring that enough free disk space is available or not.
  - Monitoring performance and responding to changes
    - Database tuning

Overall System Structure



DATA DICTIONARY

- Data dictionary (or data repository) or system catalog is an important part of the DBMS.
- It contains data about data (or metadata).
- It means that it contains the actual database descriptions used by the DBMS.
- In most DBMSs, the data dictionary is active and integrated. It means that the DBMS checks the data dictionary every time the database is accessed.
- The data dictionary contains the following information.
- Description of schema of database.
- Schemas, mappings and constraints
- Description of the database user, their responsibilities and rights.
- Descriptions of statistics such as frequencies of queries and transactions and access counts to different portion of database. .
- Description about physical database design, such as storage structures, access paths ,file and record size etc.
- Descriptions about users of DBMS and their access rights.

DBMS Vs RDBMS

- DBMS : Data Base Management System is a process of managing data for efficient retrieval & storage of data.
- Ex: Sybase , FoxPro
- RDBMS : The database which is used by relations(tables) to acquire information retrieval are known as RDBMS
- EX: SQL, ORACLE,MY-SQLSERVER
- DBMS does not impose any constraints or security with regard to data manipulation. It is user or the programmer responsibility to ensure the ACID PROPERTY of the database
- In DBMS Normalization process will not be present.
- It supports Single User only
- It treats Data as Files internally
- It supports 3 rules of E.F.CODD out off 12 rules
- It requires low Software and Hardware Requirements.

RDBMS

- RDBMS is based on relational model, in which data is represented in the form of relations, with enforced relationships between the tables.
- RDBMS defines the integrity constraint for the purpose of holding ACID PROPERTY.
- In RDBMS, normalization process will be present to check the database table consistency.
- RDBMS helps in recovery of the database in case of loss of database due to system failure or any other reason.
- It supports multiple users.
- It treats data as Tables internally.
- It supports minimum 6 rules of E.F.CODD.
- It requires High software and hardware requirements.

DBMS Vs RDBMS

DBMS	RDMS
Data handle as a File oriented system	Data handle as a in form of table
DBMS does not support the client server Architecture	Most of the RDBMS support the client server architecture
DBMS does not support distributed databases	RDBMS support distributed database
There is no security of data	There are multiple level of security
DBMS may satisfy less then 7 rule of Dr. E. F. Codd	RDBMS satisfy more than 7 rule of Dr. E. F. Codd
	Naming Conventions
Field	Columns, Attributes
Record	Row, Tuple,Entity
File	Table, relation, Entity class

Data independency

- It is a major objective of DBMS in organization.
- It is a characteristics of DBMS to change the schema at one level without having to change the schema at the next higher level.
- Physical data independency
- Logical data independency

Physical data independency

- Immunity of the conceptual level to change in the internal schema is referred to a Physical data independency.
- Means change in the physical storage of the data.
- Change different file structure, storage structure, different storage devices, modify index or hashing algorithm.

- ## Logical data independency

- Immunity of the external level to change in the conceptual schema is referred to a logical data independency
- Change in logical structure of data.
- Addition and deletion of entities , change in relationship of table

Mapping

- The process of transforming request and result between three levels are called mapping.

Software Architecture

Prepared By: Vipul Vekariya

M.E(Computer)

File Sharing architecture :

- where the server downloads files from the shared location to the desktop environment.
- The requested user job is then run both logic and data in the desktop environment.
- The main problem with this setup is that file sharing gets strained when the number of online users grow significantly.
- Also network traffic got congested.
- Works if
- Shared usage is low
- Data transferred is low

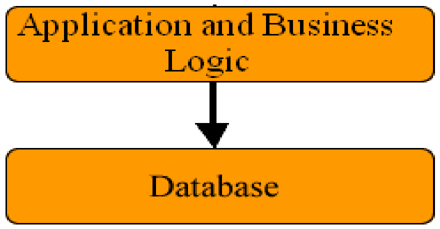
CLIENT/SERVER ARCHITECTURE(TWO TIER)

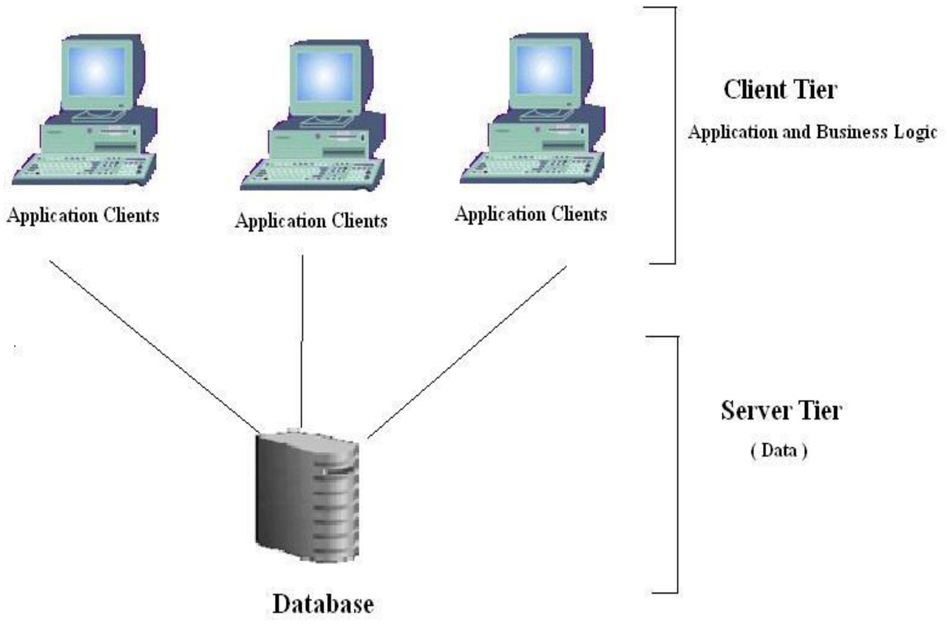
- Database server to replace File server
- Query based approach
- Reduced traffic, since required content is only transferred
- remote procedure calls or standard query language statements are being used by clients to communicate with servers.
- The application exists entirely on the client PC while the database sits out on a server machine.
- the full processing load is lying onto the PC while the more powerful server machine act as a traffic controller between the application and the database.
- arises a very high Hang-up on scalability, availability and performance on the system.
- Another main problem with any two-tiered client/server approach is that of maintenance.

Two Tier Architecture

Pictorial Representation Of 2-TIER

- It mainly comprises of two layers:



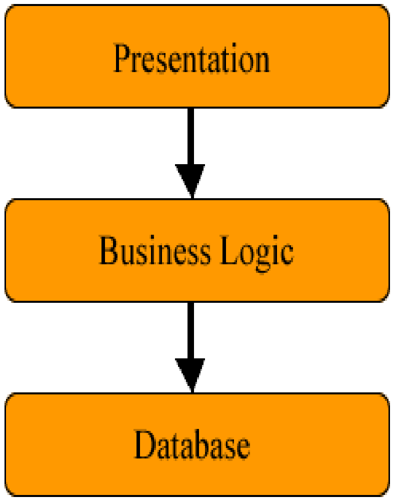


Three Tier Architecture

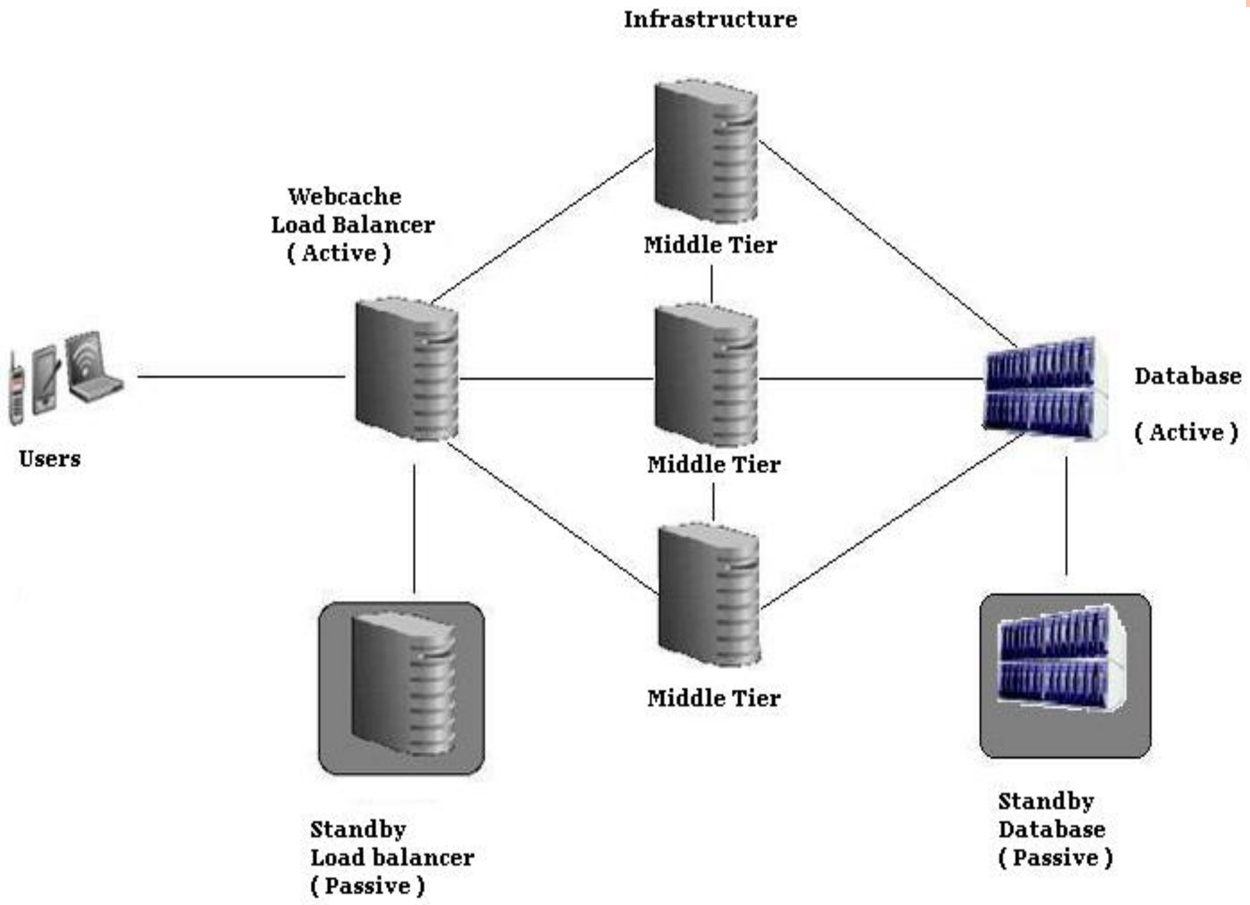
- In order to address the above-mentioned issues in an effective way, software developers community came with an evolutionary but effective and efficient solution referred to as "three-tier architecture".
- The first tier is called as the presentation layer and it normally consists of a graphical user interface.
- The middle tier consists of the application logic and the third tier is the data layer.
- The middle tier composes business application logic code, which the user calls upon through the front-end graphical user interface to retrieve the desired data.
- The middle tier can perform queuing, application execution, locating.
- The third tier contains the data that is needed for the application.. The data can be from any source of information such an enterprise database like Oracle, MS SQL Server, a set of XML documents.

New Three Tier Architecture:

- It mainly comprises of three layers



Pictorial Representation of New Environment



Presentation tier

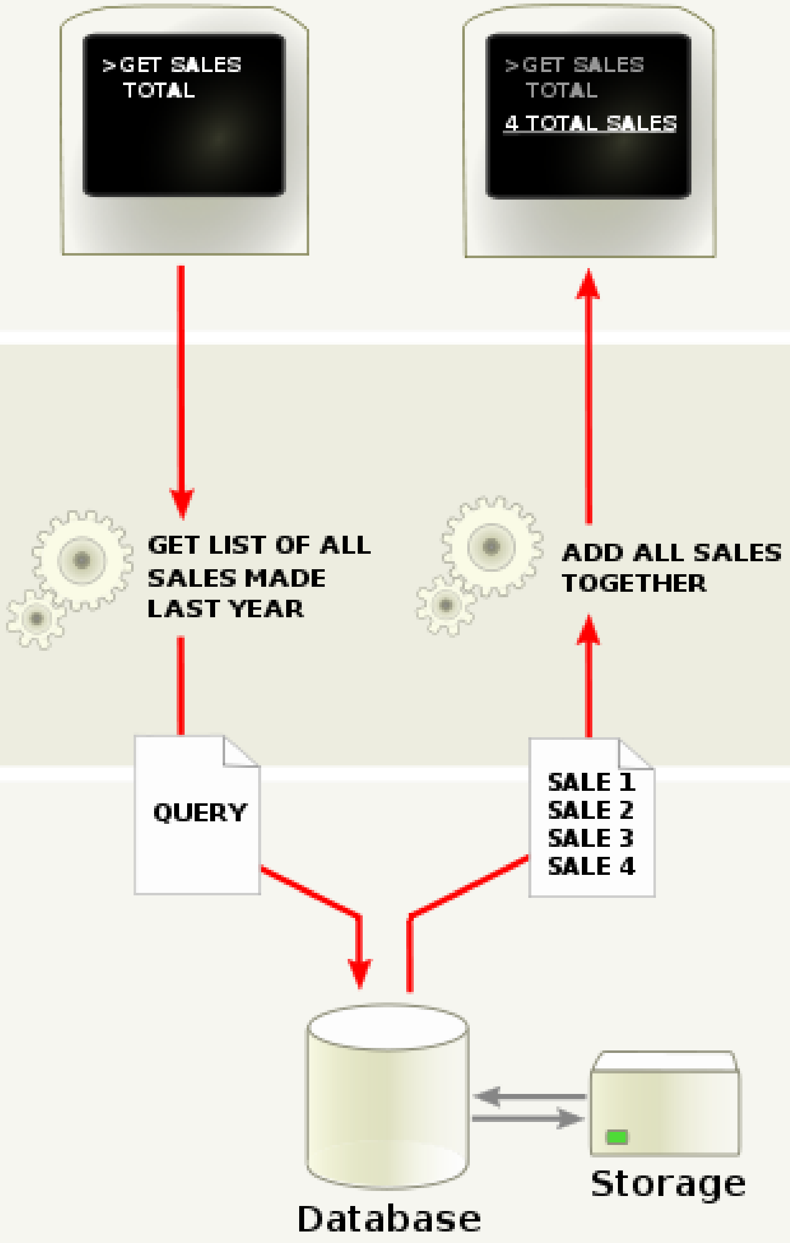
The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.



ADVANTAGE

- Improved Scalability, availability, and performance
- Improved flexibility, and extensibility of business systems
- Three-tier systems are more flexible and extensible than their two-tier systems as the business logic and services such as security, persistence, transactions etc reside on the middle-tier and transparent to the client applications.
- This tends to make these services being applied automatically to client requests and any changes made in the business logic code do not reflect on the clients in any way.
- Clustering makes three-tier systems more available and scalable because multiple servers and resources can support fail-over and balance the word loads of a growing client population.

End of Chapter 1