# Chapter - 1

## Introduction

- Text Books:
  - 1. "Database System Concepts" 4th Edition, By Abraham Silberschatz, Henry F.Korth and S.Sudarshan, McGraw Hill Publication
  - 2. "Murach's MySQL" 3rd Edition, By Joel Murach, Mike Murach & Associates
  - 3. "Getting Started with NoSQL", By Gaurav Vaish, Packt Publishing

#

# Database Management System (DBMS)

- It is a Collection of interrelated data
- Set of programs to access the data
- Goal of dbms is to provide a way to store and retrieve the data efficiently and conveniently
- DBMS contains information about a particular enterprise
- DBMS provides an environment that is both *convenient* and *efficient* to use.
- Database Applications:
  - Banking: all transactions
  - Airlines: reservations, schedules
  - Universities:  registration, grades
  - Sales: customers, products, purchases

# Purpose of Database System

- In the early days, database applications were built on top of file systems and different application program required for manipulation of the data of the file.

- Drawbacks of using file systems to store data:

1. Data redundancy and inconsistency
   - Different files have different format formats, and different application written in different languages so duplication of information in different files.
   - Redundancy leads to higher storage and access cost.
   - It may lead to data inconsistency.

2. Difficulty in accessing data
   - Need to write a new program to carry out each new task For every new query there need to write new program

#

- File processing do now allow data to be retrieved in a convenient and efficient manner.

3  Data isolation — multiple files and formats

- data are scattered in various files, and files may be in different formats, writing new application programs to retrieve the appropriate data is difficult

4  Integrity problems

- Integrity constraints  (e.g. account balance > 0) become part of program code
- Hard to add new constraints or change existing ones when data are available in several different files.

5  Atomicity of updates

- It is difficult to ensure atomicity in file processing system
- While database follows the Atomicity.
- E.g. transfer of funds from one account to another should either complete or not happen at all.

# 6  Concurrent access by multiple users

- Concurrent accessed needed for performance.
- If multiple users are updating the same data simultaneously it will result in inconsistent data state.
- In file processing system it is very difficult to handle this using program code.
- Uncontrolled concurrent accesses can lead to inconsistencies.
  - E.g. two people reading a balance and updating it at the same time.

#

# 7  Security problem

- Every data should not be accessed by every one.

- So in file system – it is difficult to manage security of data with the application programs .

- Database systems offer solutions to all the above problems.

# Levels of Abstraction

– A major purpose of a database system is to provide users with an *abstract* view of the data. That is, the system hides certain details of how the data are stored and maintained

– Physical level describes how a record/data (e.g., customer) is actually stored.

– Logical level: describes what data stored in database, and the relationships exists among the data.

```
type customer = record
            name : string;
            street : string;
            pin no : integer;
end;
```
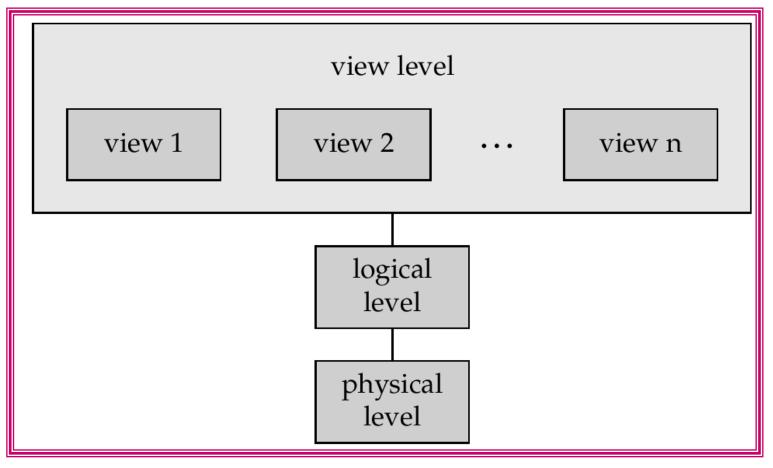
#

– View level:

– highest level of abstraction

– application programs hide details of data types. Views can also hide information (e.g., salary) for security purposes. [Only part of the database can be accessed by users].

#

# View of Data

An architecture for a database system

- database system hides many of the lowest-level storage details from database programmers.
  - Database administrators, on the other hand, may be aware of certain details of the physical organization of the data.
- the logical level, each such record is described by a type definition.
- Programmers using a programming language work at this level of abstraction.
- view level, computer users see a set of application programs that hide details of the data types.

#

- hiding details(datatypes) of the logical level of the database.
- views also provide a security mechanism to prevent users from accessing certain parts of the database.

#

# Instances and Schemas

- Similar to types and variables in programming languages
- **Schema** – the logical structure of the database
  - The overall design of the database is called the database **schema**. Schemas
  - are changed infrequently
  - e.g., the database consists of information about a set of customers and accounts and the relationship between them)

#

- Physical schema: database design at the physical level

- Logical schema: database design at the logical level

- **Instance**
  - the actual content of the database at a particular point in time


- **Physical Data Independence**
  - The ability to modify the physical schema without changing the logical schema
  - Applications depend on the logical schema #

– Application programs are said to exhibit physical data independence if they do not depend on the physical structure.

In general, the interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others.

#

# Data Model

- A collection of tools for describing
  - data
  - data relationships
  - data semantics
  - data constraints
- Entity-Relationship model
- Relational model
- Other models:
  - object-oriented model contains the object, encapsulation ,methods.
  - Object –relations data model combine features of object models and relational data model.
  - XML: used to represent semi-structured data.
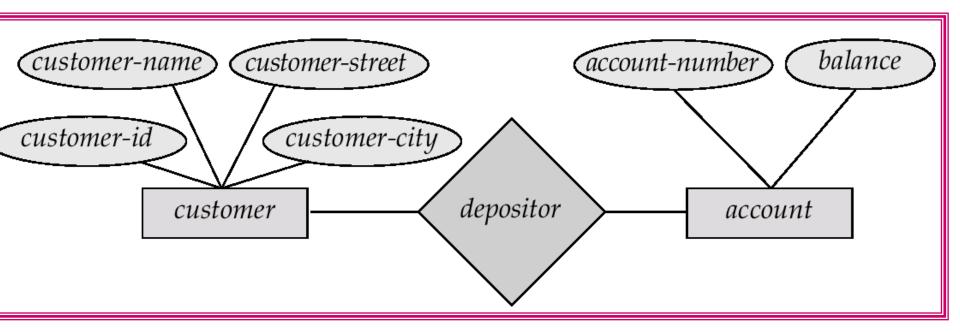  - Older models: network model and hierarchical model

#

# Entity Relationship Model

– E-R model based on  real world perception
– Entities (objects)
  • E.g. customers, accounts, bank branch
– Relationships between entities
  • E.g. Account A-101 is held by customer Johnson
  • Relationship set *depositor* associates customers with accounts
– Widely used for database design
– Database design in E-R model usually converted to design in the relational model which is used for storage and processing.

#

# Entity-Relationship Model

Example in the entity-relationship model

# Relational Model

- Example of tabular data in the relational model

| Customer-id | customer-name | customer-street | customer-city | account-number |
|---|---|---|---|---|
| 192-83-7465 | Johnson | Alma | Palo Alto | A-101 |
| 019-28-3746 | Smith | North | Rye | A-215 |
| 192-83-7465 | Johnson | Alma | Palo Alto | A-201 |
| 321-12-3123 | Jones | Main | Harrison | A-217 |
| 019-28-3746 | Smith | North | Rye | A-201 |

Collection of table to represent both data and the relation ships among those data.
it is at lover level of abstraction than the E-R model.

#

# A Sample Relational Database

| customer-id | customer-name | customer-street | customer-city |
|---|---|---|---|
| 192-83-7465 | Johnson | 12 Alma St. | Palo Alto |
| 019-28-3746 | Smith | 4 North St. | Rye |
| 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

# Data Languages

- Data Definition language(DDL)
- Data Manipulation language(DML)

#

- **Data definition language**
  - It is Specification notation for defining the database schema
  - E.g.
    create table *account* (
    	*account-number*    char(10),
    	*balance*                integer)
  - DDL compiler generates a set of tables stored in a *data dictionary*
  - Data dictionary contains metadata (i.e., data about data)  . database schema
  - Database system consult the data dictionary before reading or modifying actual data.
  - Data *storage and definition* language
    - language in which the storage structure and access methods used by the database system , are specified.

- It define the implementation details of database schema which is hidden form the user.

– DDL provide the facilities to specify consistency constraints.

#

- **Data manipulation Language**
  - Language for accessing and manipulating the data organized by the appropriate data model
    - DML also known as query language
  - Two classes of languages
    - Procedural – user specifies what data is required and how to get those data .
    - Nonprocedural – user specifies what data is required without specifying how to get those data.
  - SQL is the most widely used query language.

#

# SQL

– Query language is part of DML for information retrieval

– SQL: widely used non-procedural language

– E.g. find the balances of all accounts held by the customer with customer-id 192-83-7465

       **select**   *account.balance*
       **from**    *depositor, account*
       **where**  *depositor.customer-id* = '192-83-7465' **and**
             *depositor.account-number =*

*account.account-number*

#

– Application program interface (e.g. ODBC/JDBC) which allow SQL queries to be sent to a database and retrieve results

– Usually DML precompilers converts DML statements to procedure calls

#

# Database Users

- Four different type of users according to way they expect to interact with the system are designed for different types of users.

- Different type of user interface

1. Naïve users

  - Unsophisticated users who interact with system using application programs that have been written previously

  - E.g. people accessing database over the web, bank tellers, clerical staff

  - Generally forms are used as interface and reports are generated for such users.

#

2. Application programmers

– Computer professionals who write application programs to access database

– RAD tool can be used for accessing database and allow to construct forms and reports without writing a code.

– Fourth generation languages provide control structures that can be combined with DML statements

# Database Users

3. **Sophisticated users**

- Interact with database without writing programs
- They will use query language and submit to query processor

- Analyst are included in this category.
- OLAP (Online Analytical Processing ) tool simplifies task of analyst.
- Data mining is another technique for finding patterns in data.

# 4. Specialized users

- Sophisticated users that write specialized database applications that do not fit into the traditional data processing framework

- E.g. application for knowledgebase system, expert system, graphics applications etc

- They can develop their own data base applications according to their requirement.

#

# Database Administrator

– Person who can control data and programs to access those data is called as database administrator

– Coordinates all the activities of the database system; the database administrator has a good understanding of the enterprise's information resources and needs.

– Database administrator's duties include:

  • Schema definition : create original database schema

  • Execute DDL

  • Storage structure and access method definition

  • Database tables or indexes are stored in the following ways: Flat files,Heaps,B+ Tree et

  • Schema and physical organization modification

#

- Granting of authorization for data access
  - The DBA provides different access rights to the users according to their level.
  - The authorization information is kept in a special system structure that the database system consults whenever someone attempts to access the data in the system.
- Routine maintenance
  - Periodically backing up database
  - Ensuring enough free disk space is available and upgrading disk space
  - Monitoring jobs running on database and ensuring performance is not degraded

#

# Transaction Management System

– A transaction is a collection of operations that performs a single logical function in a database application

– It should have **atomicity** and **consistency**

  • *If database was consistent when transaction started, it must* be consistent when transaction terminates successfully

  • But temporarily inconsistency must be allowed during execution of transaction where handling of failure is necessary

– Durability

  • After successful execution, values must persist despite of failure.

  • Ensuring atomicity and durability is responsibility of **transaction management component** of database system.

#

- Database system must have **failure recovery** which ensures that failure has no effect on state of database

- Restore database to state that existed prior to occurrence of failure.

- When database is concurrently accessed, consistency is to be maintained

- Responsibility of concurrency control manager to control interaction of concurrent transactions

- All database systems may not have feature like backup and recovery, concurrency etc

#

# Database System Structure

– Database system is divided into modules for handling different functions

– Main two modules:

1. Storage Manager

   • It will organize database system structure such that movement of between disk and memory is less.

2. Query Processor

   • Simplifies access of data

#

- **Storage Manager**
  - Storage manager is program module that provides interface between low-level data stored in database with application programs and queries submitted to system.
  - The Storage manager translates the various DML statements into low level .

  - The storage manager is responsible for:
    - Interaction with file manager since raw data is stored as file in system.
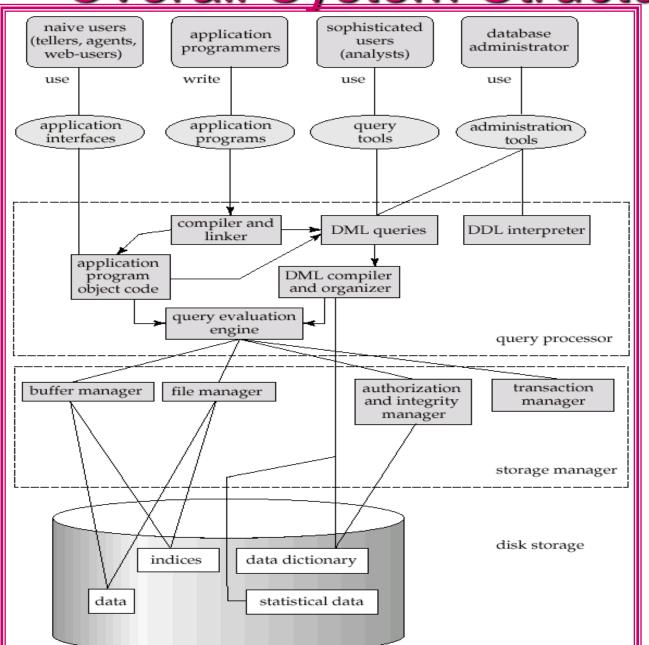
#

- it include components like
- Authorization and integrity manager
  - It test for the authority and integrity constraints.
- Transaction manager
  - Ensure that the database remain in consistent state even system failures.
  - Concurrent transaction executing without conflicting.
- File manager
  - Manages the space on disk
- Buffer manager
  - responsible for fetching data form disk to main memory

- The Query Processor
  - It include
  - DDL interpreter : it records the definitions in the data dictionary.
  - DML : translate DML into the language in which query engine understand.
  - Query evaluation engine :
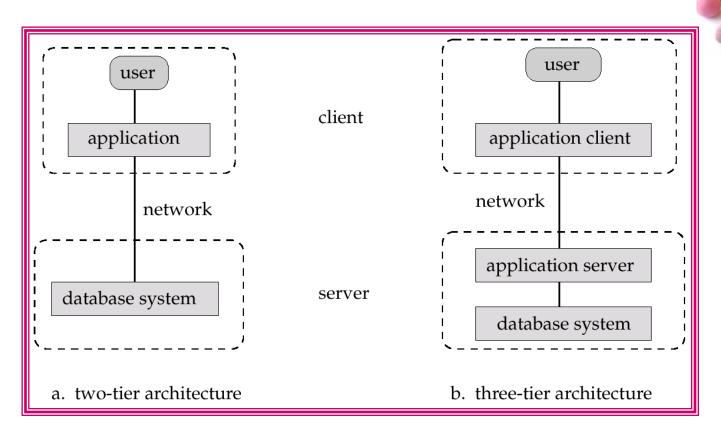    - execute instruction generated by DML compiler.

#

# Overall System Structure

# Application Architectures



a. two-tier architecture
b. three-tier architecture

- **Two-tier architecture**:  E.g. client programs using ODBC/JDBC to
   communicate with a database
- **Three-tier architecture**: E.g. web-based applications, and applications built using "middleware"