



# UNIT – I

# INTRODUCTION TO DBMS



# DATABASE MANAGEMENT SYSTEMS

## SYLLABUS FOR IV-SEMESTER

L:T:P(Hrs./week): 3:0:0	SEE Marks :60	Course Code : <b>U20PC420IT</b>
Credits : 3	CIE Marks :40	Duration of SEE : 3 Hours

<b>COURSE OBJECTIVES</b>	<b>COURSE OUTCOMES</b> <i>On completion of the course, students will be able to</i>
Apply the concepts of database management systems and design relational databases.	<ol style="list-style-type: none"><li>1. Understand functional components of the DBMS and develop ER model for a given problem and map ER it to Relational model</li><li>2. Devise queries using Relational Algebra and SQL</li><li>3. Design a normalized database schema using different normal forms.</li><li>4. Apply indexing and hashing techniques for effective data retrieval.</li><li>5. Understand transaction processing, concurrency control and recovery techniques</li></ol>



## **UNIT – I:**

**Introduction to DBMS:** Overview, File system vs DBMS, Advantages of DBMS, Database System Applications, Relational Databases, Object – Based and Semi-structured Databases, Data Storage and Querying, Database Architecture, Database Users and Administrators.


**Database Design and the E-R Model:** Overview of the Design Process, The E-R Model, Constraints, E-R Diagrams, E-R Design Issues, Weak Entity Sets, Extended E-R Features.

## **UNIT – II:**

**Relational Model:** Structure of Relational Databases, Reduction to Relational Schemas, Other Aspects of Database Design. Relational Algebra: Fundamental Relational-Algebra Operations, Additional Relational – Algebra Operations, Extended Relational -Algebra Operations, Null Values, Modification of the Databases.

**Structured Query Language:** Data Definition, Basic Structure of SQL Queries, Set Operations, Aggregate Functions, Null Values, Nested Sub queries, Complex Queries, Views, Joined Relations.





## **UNIT – III:**

**Advanced SQL and PLSQL:** SQL Data Types and Schemas, Integrity Constraints, Authorization, SQL functions, procedural SQL, embedded SQL, cursors, ODBC and JDBC, triggers

**Schema Refinement:** Features of Good Relational Design, Functional-Dependency Theory, Decomposition Using Functional Dependencies, Normalization, First, Second, Third Normal Forms, Dependency Preservation – Boyce/Codd Normal Form, Multi-valued Dependencies and Fourth Normal Form – Join Dependencies and Fifth Normal Form.

## **UNIT – IV:**

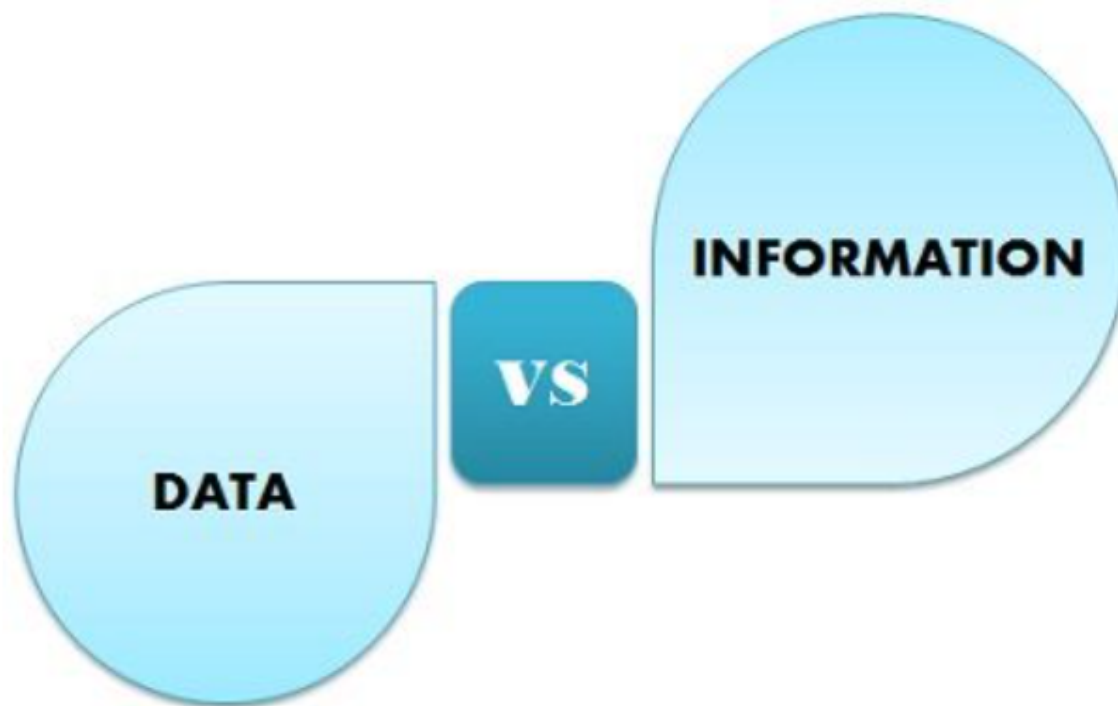
**Indexing and Hashing:** Basic Concepts, Ordered Indices, B+-tree Index Files, B-tree Index Files, Multiple-Key Access, Static Hashing, Dynamic Hashing, Comparison of Ordered Indexing and Hashing, Bitmap Indices, Index Definition in SQL.

**Transactions:** ACID properties, Transaction States, Implementation of Atomicity and Durability, Concurrent Executions, Serializability, Recoverability, Testing for serializability.

## **UNIT – V:**

**Concurrency Control:** Lock-Based Protocols, Timestamp-Based Protocols, Validation-Based Protocols, Multiversion Schemes, Deadlock Handling.

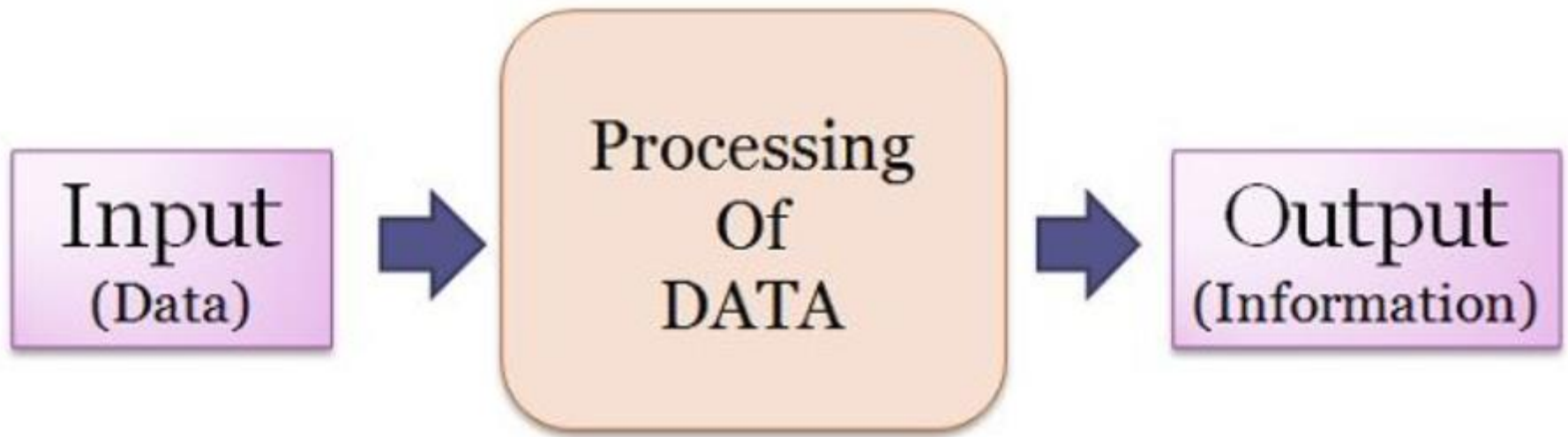
**Recovery System :** Failure Classification, Storage Structure, Log-Based Recovery, Aries ,Media recovery.



**Data** is raw, unanalyzed, unorganised, unrelated, uninterrupted material which is used to derive information, after analyzation. On the other hand, **Information** is perceivable, interpreted as a message in a particular manner, which provides meaning to data.



Information = Data + Meaning





A **database** is a collection of data, typically describing the activities of one or more related organizations. For example, a university database might contain information about the following:

- \* Entities such as students, faculty, courses, and classrooms.
- \* Relationships between entities, such as students' enrollment in courses, faculty teaching courses, and the use of rooms for course.



A **database management system**, or **DBMS**, is a collection of interrelated data and a set of programs to access those data. The primary goal of a DBMS is to provide a way to *store* and *retrieve* database information that is both *convenient* and *efficient*.

**Management of data involves both:**

- i) Defining structures for storage of Information.
- ii) Providing mechanisms for manipulation of information.



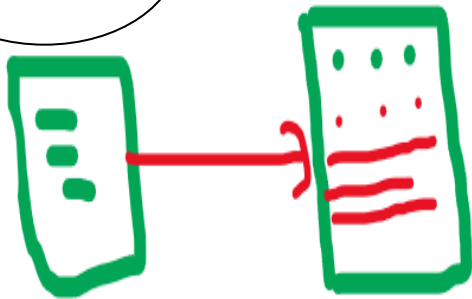


# Why DBMS?

1. i want to store student details in a file

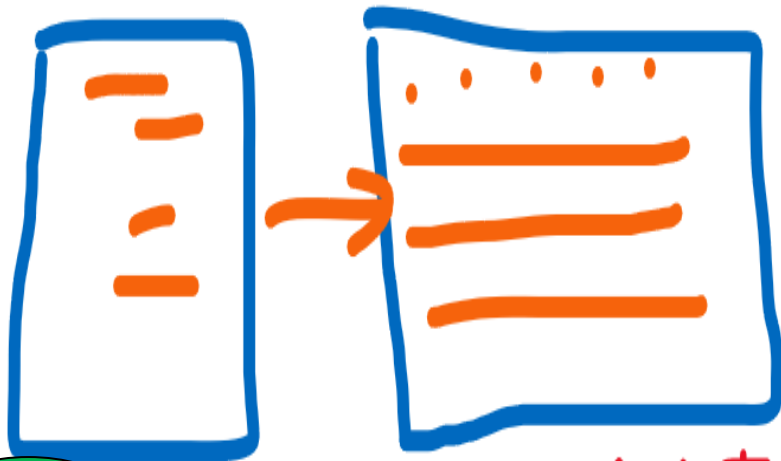
One.c

stud.txt



2. write a program 1.c to perform operations on stud.txt

this file contains student records in this format (stud id , student name , marks ) add new student, delete student with given rollnum, update marks of given student, display marks of student with given rollnum



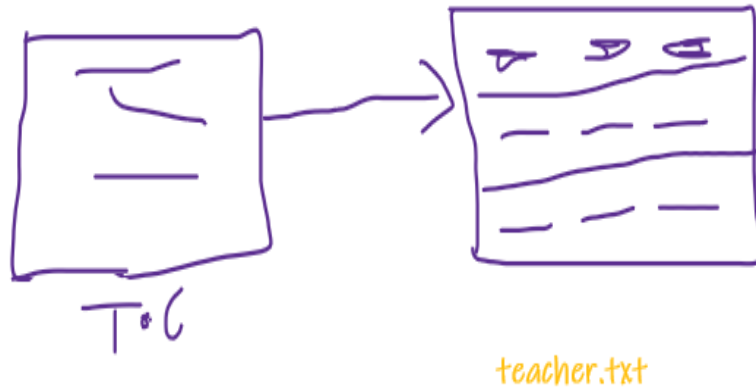
now i want to add phone number , addresss

now i have to change the program to suit to this data in file

two.c

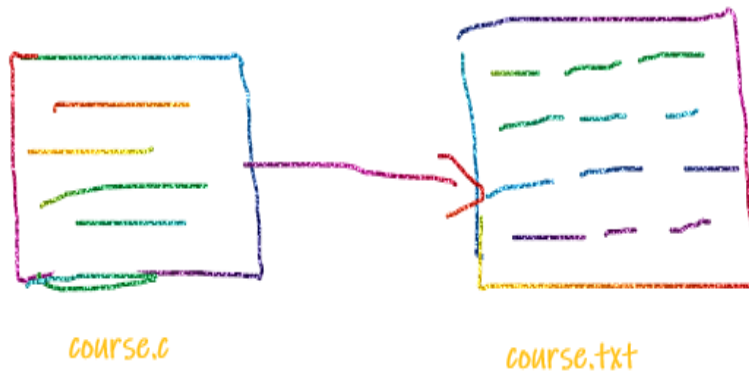
stud.txt

now i want store teacher data



1. now i create teacher.txt(teacherID,name,designation,department , salary)
2. write program in c similar to two.c called T.c , but now for teacher information to perform operations...

now i want store course data



1. now i create course .txt (courseID,name,credits,semester,year,dept\_offering)
2. write program in c similar to two.c call it course.c , but now for course information to perform operations...

your reflections ?now



We observe that

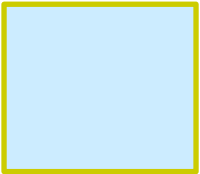
1. we are writing similar code to
2. Perform similar operations
3. On similar data structures
4. Also that data can have several features

So what is the Solution ?

You want some software that lets you

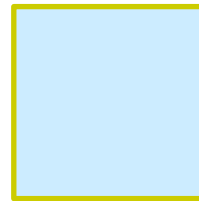
- to create any information without writing code all by yourself
- Add new information
- Delete unnecessary information
- Update the information
- i.e., you want some help to manage your information / data

ANSWER is DBMS

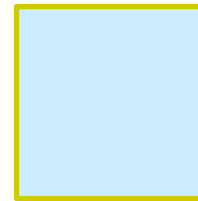


teacher .table  
with add, create,  
delete ,update commands

instead of

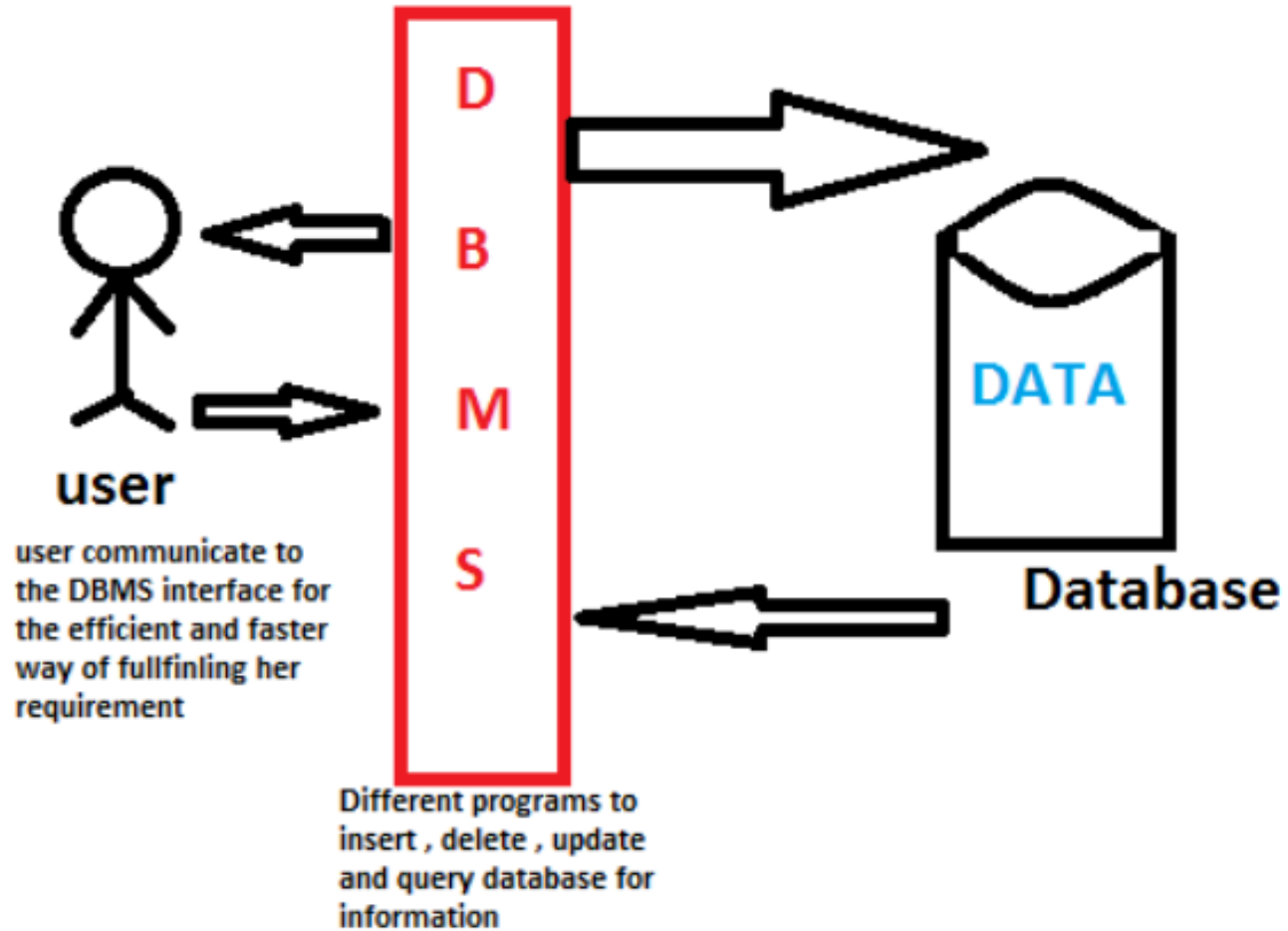


teacher.c



teacher.txt







# Introduction

- Introduction to Database Applications -University database Example
- The Need for Databases
  - disadvantages of File Processing systems
  - Advantages of DBMS
- Levels of abstraction
- Views of data
- Schemas and instances
- The database architecture



# Database Management System (DBMS)

- DBMS contains information about a particular enterprise
  - Collection of interrelated data
  - Set of programs to access the data
  - An environment that is both *convenient* and *efficient* to use
- Database Applications:
  - Banking: transactions
  - Airlines: reservations, schedules
  - Universities: registration, grades
  - Sales: customers, products, purchases
  - Online retailers: order tracking, customized recommendations
  - Manufacturing: production, inventory, orders, supply chain
  - Human resources: employee records, salaries, tax deductions
- Databases can be very large.
- Databases touch all aspects of our lives



# University Database Example

- Application program examples
  - Add new students, instructors, and courses
  - Register students for courses, and generate class rosters
  - Assign grades to students, compute grade point averages (GPA) and generate transcripts
- 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
  - ❑ Multiple file formats, duplication of information in different files, programmers styles –different , various copies of same data no longer agree
- ❑ Difficulty in accessing data
  - ❑ Need to write a new program to carry out each new task
- ❑ Data isolation
  - ❑ Data scattered in various files, Multiple files and formats
- ❑ 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



# Drawbacks of using file systems to store data (Cont.)

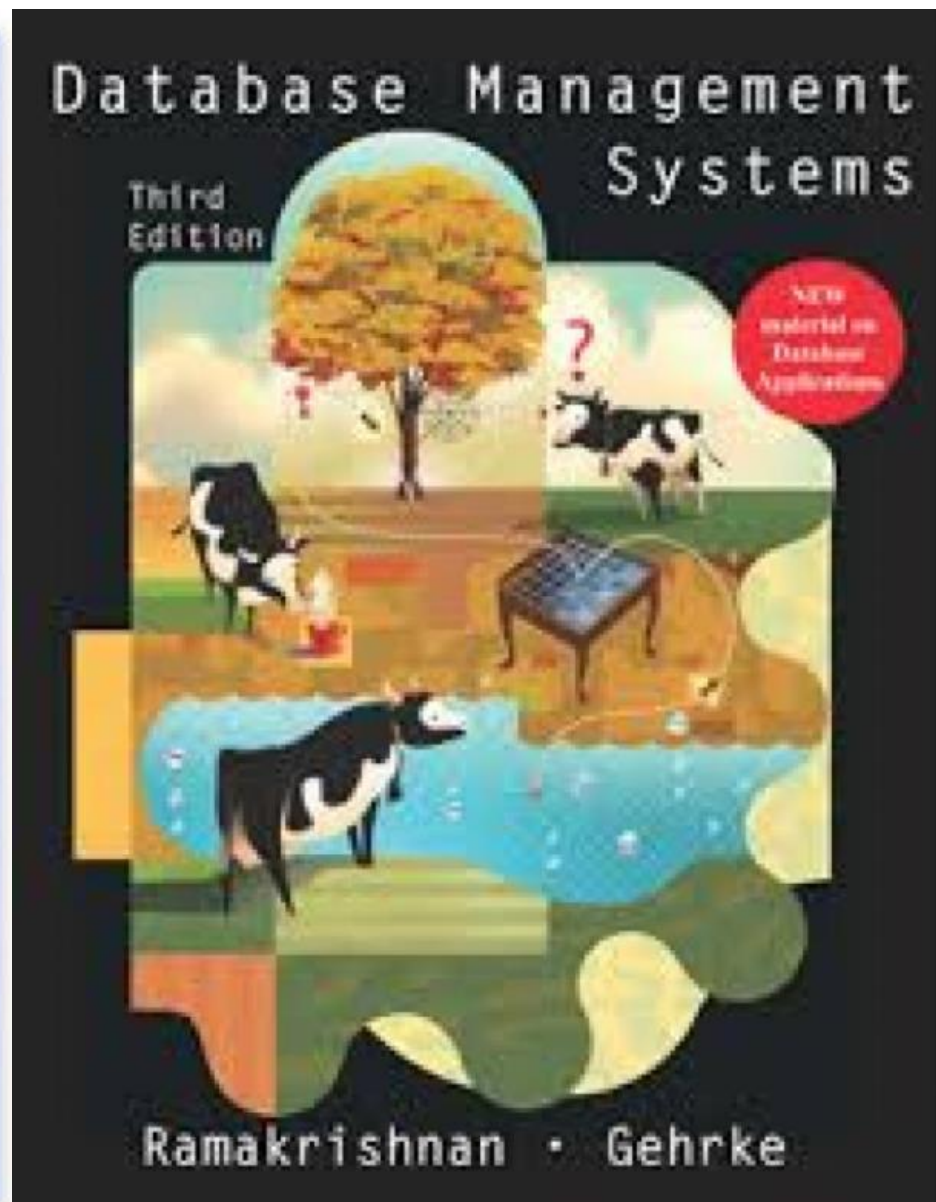
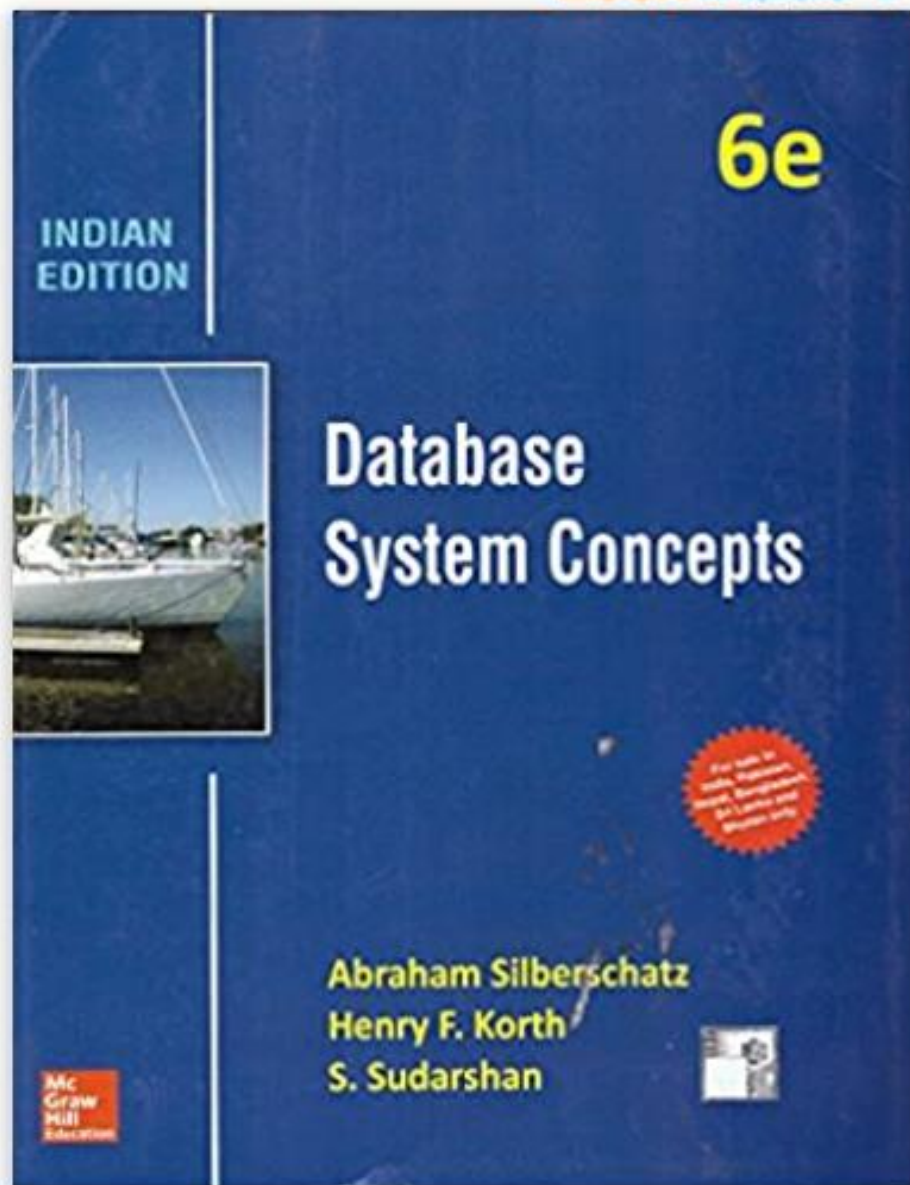
- Atomicity of updates
  - Failures may leave database in an inconsistent state with partial updates carried out
  - {TRANSFER 100 A->B, 1. A=A-100, 2. B=B+100}
  - A=200,B=300 (500) AFTER TR A=100, B=400(500)
  - Example: Transfer of funds from one account to another should either complete or not happen at all
- Concurrent access by multiple users
  - Concurrent access needed for performance
  - Uncontrolled concurrent accesses can lead to inconsistencies
    - ▶ Example: Two people reading a balance (say 100) and updating it by withdrawing money (say 50 each) at the same time
- Security problems
  - Hard to provide user access to some, but not all, data

**Database systems offer solutions to all the above problems**



Work for today :

- Identify the various DBMS SYSTEMS ,
- Who are their vendors
- How are they different
  
- R1 , R2 , R3... (5-6)
- D1 .. D2 ... D3...( ? NAMES )
- ? ? ? ( WHAT ARE THEIR SPECIALITIES )
  
- USER
  
- DBMS( )
  
- FILES







### **Learning Resources:**

1. Abraham Silberschatz, Henry F Korth, S. Sudarshan, Database System Concepts, 6th Edition, McGraw-Hill International Edition, 2011.
2. Raghu Ramakrishnan, Johannes Gehrke, Database Management Systems, Third Edition, McGraw-Hill International Edition, 2003.
3. Elmasri, Navathe, Somayajulu and Gupta, Fundamentals of Database System, 6<sup>th</sup> Edition, Pearson Education, 2011.
4. Patric O'Neil, Elizabeth O'Neil, Database-principles, programming, and performance, Morgan Kaufmann Publishers, 2001.
5. Peter Rob, Carlos coronel, Database Systems, (2007), Thomson.
6. <https://nptel.ac.in/courses/106105175/>