Database I

(60012301-1)

Lecture 1: Introduction to Databases

Dr. Obead Alhadreti

Outline

- Course overview
- Data Storage Approaches
- Database definition and components
- Database management systems

Course Description

- This course provides *foundational* database knowledge and covers topics related to the **conceptual design of database** based on the functional requirements for organizations.

- It presents the basics of **information storage** and **management**, from the conceptual modelling of an organization's data requirements using the **relational model**, through to the implementation of these requirements with tools such as **SQL** and techniques such as **normalization**.

Course Learning Outcomes

1. Explain difference between **file systems** and **database systems.**

2. Differentiate between **entity relationship** and **normalization** in relational model construction.

3. Create a **Conceptual Data Model**.

Course Learning Outcomes

4. Design a **Relational Database Model**.
5. Write **SQL** queries.
6. Installing **database servers.**

Topics to be covered

#	Topic list
1	File Systems VS DB Systems
2	Data Modelling: Entity Relationship Diagram
3	ERD to Relational Mapping
4	Normalization: 1NF, 2NF, 3NF, BCNF
5	Relational Algebra Operations
6	SQL: Data Definition Language
7	SQL: Data Manipulation Language
8	Disk Storage, Basic File Structures, File Indexing Techniques

Course Elements

1. Lectures

Can be downloaded from BB

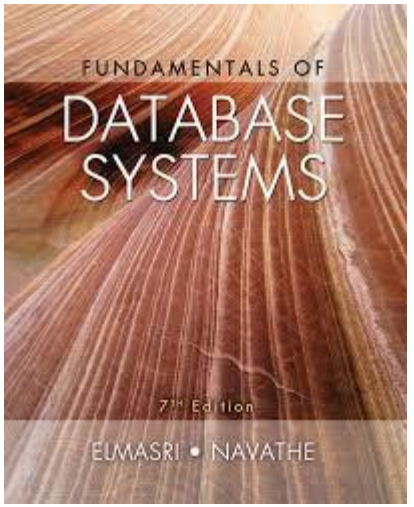
2. Lab Exercises

3. Project

Apply your knowledge in concrete use case

Textbook

- **Fundamentals of Database Systems**
- By: Ramez Elmasri , Shamkant B. Navathe
- 7th edition
- Published in 2015



Assessment Methods

- Quizzes (10%)
- Mid-term Exam (20%)
- A Group Project (30%)
- Final Exam (40%)

Motivations for Data Storage

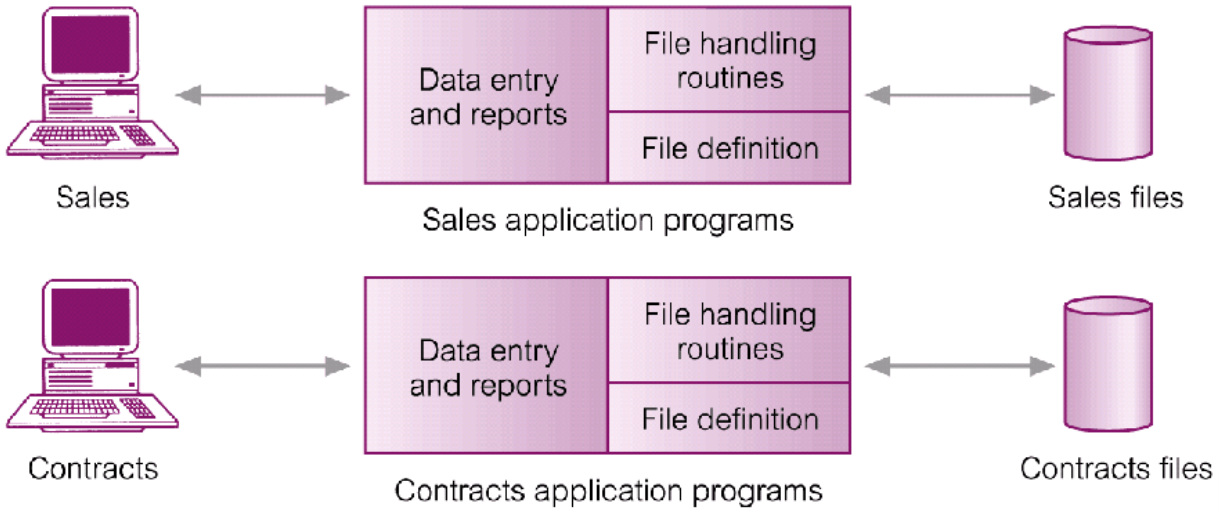
- **Data:**
  - Known facts that can be recorded and have an implicit meaning.

  - Day-to-day business processes executed by individuals and organisations require both present and historical data. Therefore, data storage is essential for organisations and individuals. Data supports business functions and aids in business decision-making.

File-based Approach

- Data are stored and processed using file-based approach.
- **In file-based approach**, data is stored in one or more **separate** computer files.
- Data is then processed by computer programs – **applications.**

File-based Approach (continued)



Sales Files

PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo)

PrivateOwner (ownerNo, fName, lName, address, telNo)

Client (clientNo, fName, lName, address, telNo, prefType, maxRent)

Contracts Files

Lease (leaseNo, propertyNo, clientNo, rent, paymentMethod, deposit, paid, rentStart, rentFinish, duration)

PropertyForRent (propertyNo, street, city, postcode, rent)

Client (clientNo, fName, lName, address, telNo)

File-based Approach (continued)

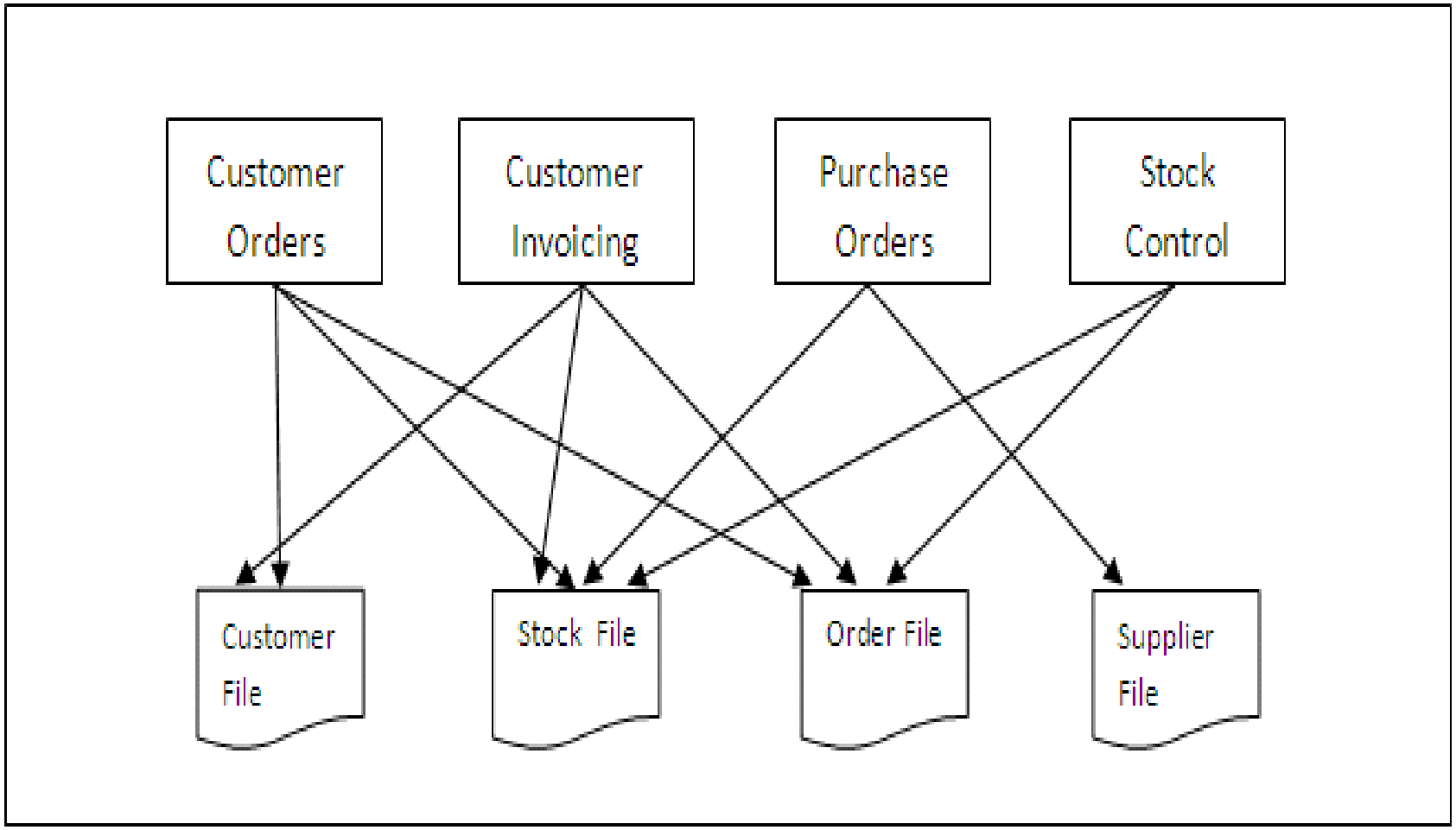
- **Problems/Limitations include but not limited to..**
  - Data Redundancy means storing the same data multiple times.
  - Data Inconsistency means that different files may contain different information of a particular object or person.
  - Definition of data was embedded in application programs, rather than being stored *separately* and *independently*.
  - No control over access and manipulation of data beyond that imposed by application programs.

Shared File Approach

- Data (files) is shared between different applications.
- Data redundancy problem is alleviated.
- Data inconsistency problem across different versions of the same file is solved.

- Other problems:
  - Rigid data structure: If applications have to share files, the file structure that suits one application might not suit another.
  - Physical data dependency: If the structure of the data file needs to be changed in some way, this alteration will need to be reflected in all application programs that use that data file.
  - No support of concurrency control: While a data file is being processed by one application, the file will not be available for other applications or for ad hoc queries.

Shared File Approach (continued)

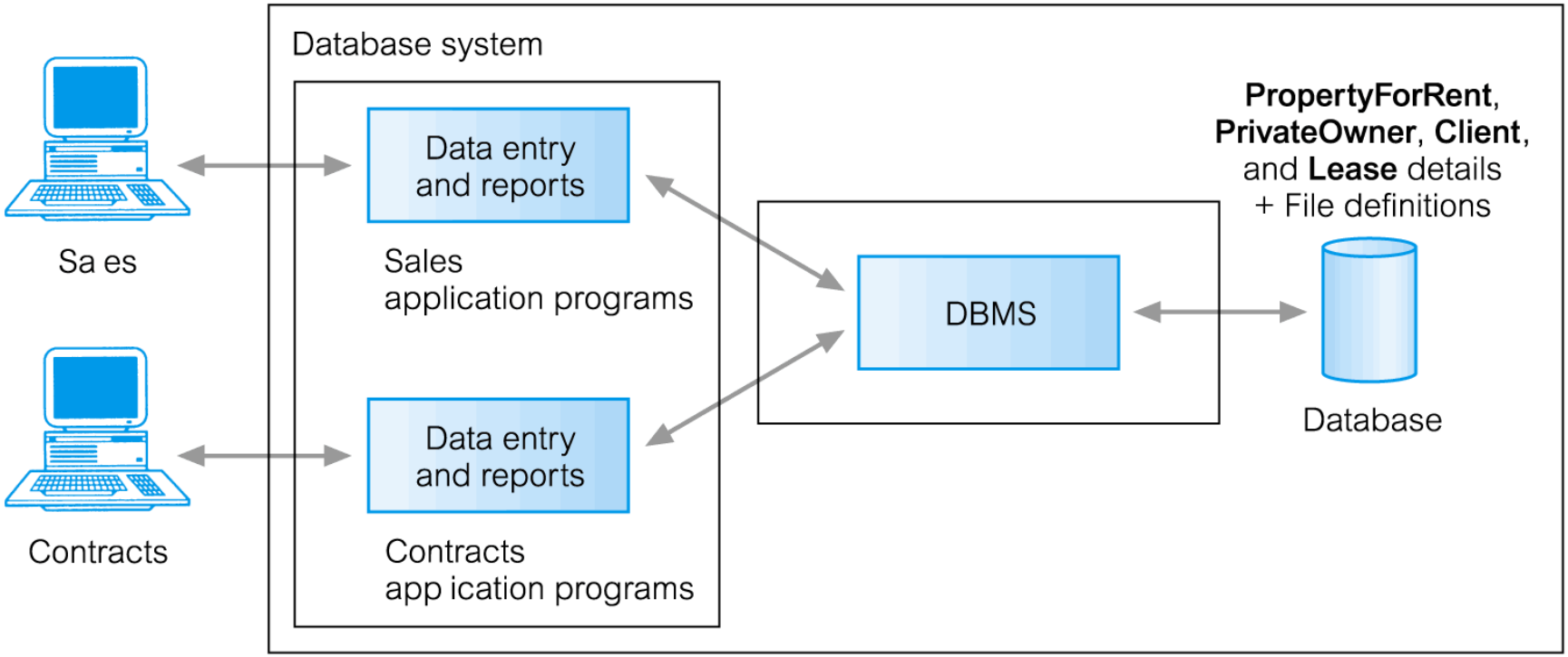


Database Approach

- The database approach is an improvement on the shared file solution.

- The use of a database management system (DBMS) provides facilities for querying, data security and integrity, and allows simultaneous access to data by a number of different users.

Database Approach



PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo)
PrivateOwner (ownerNo, fName, lName, address, telNo)
Client (clientNo, fName, lName, address, telNo, prefType, maxRent)
Lease (leaseNo, propertyNo, clientNo, paymentMethod, deposit, paid, rentStart, rentF nish)

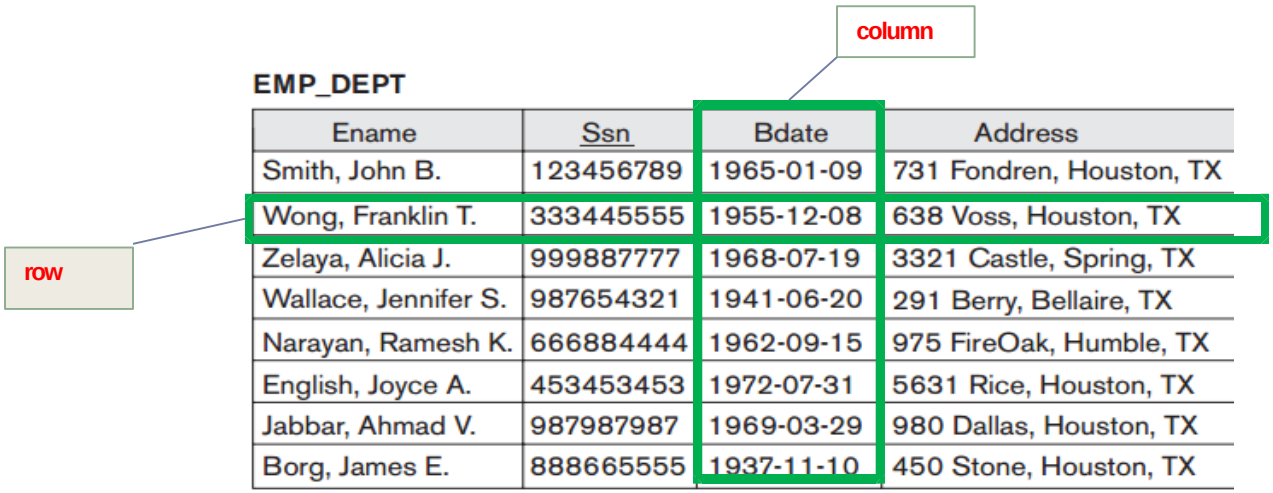
What is a Database (DB)?

- A database is a collection of related data.

- Properties:
  - A database represents some aspects of the *real* world.
  - A database is a logically coherent collection of data with some inherent meaning.
  - A database is designed, built, and populated with data for a specific purpose.

What is the main component of a Database (DB)?

- *A table (**Relation**) – one or many - is a data structure for representing related **Entities**.*
  - *Consists of columns (represent **Attributes**) and rows (represent **Entities** or **tuples**).*
  - *A **column** represents a same property of a same data type for all entities.*
  - *A **row** represents a set of properties of different data types for a specific entity*



Types of Databases and Database Applications

- Traditional Applications:
  - Numeric and Textual Databases
- More Recent Applications:
  - Multimedia Databases
  - Geographic Information Systems (GIS)
  - Data Warehouses
  - Real-time and Active Databases
  - Many other applications

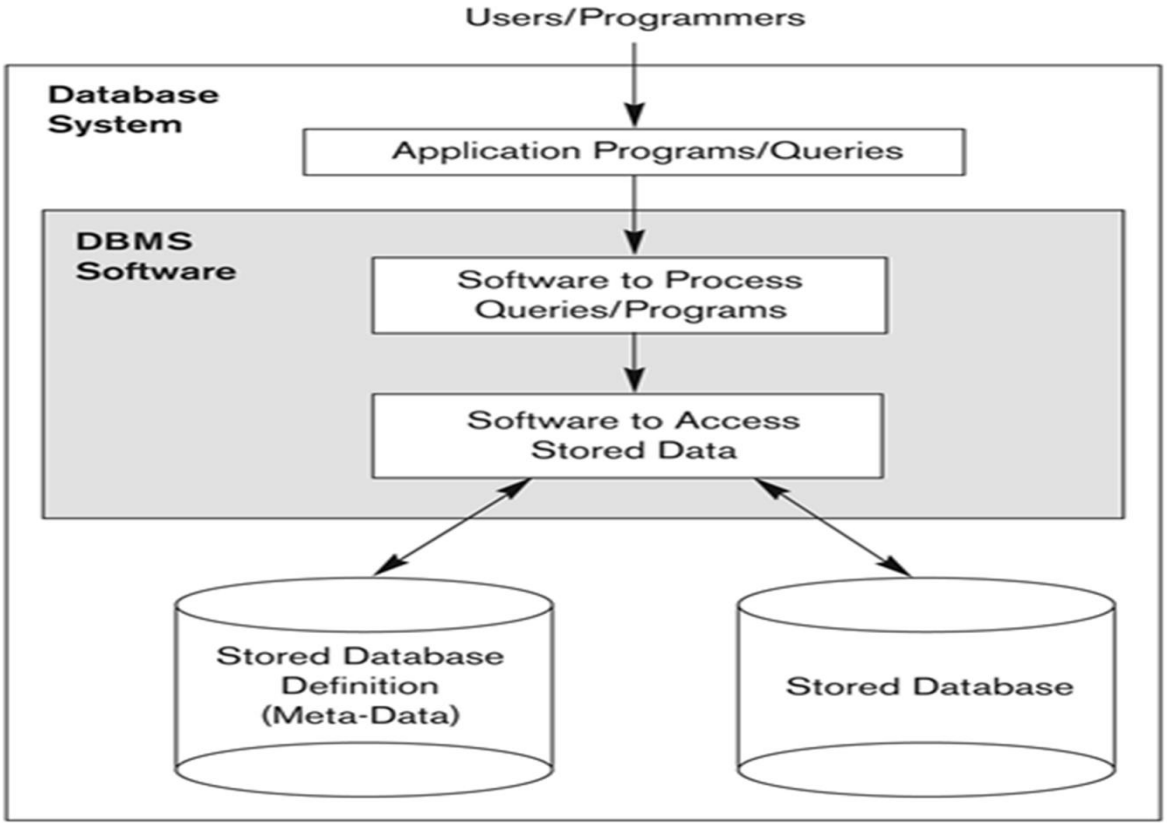
Basic Definitions

- ***Mini-world:***
  - Some part of the real world about which data is stored in a database. *For example*, student grades and transcripts at a university.

- ***Database Management System (DBMS):***
  - A software package/ system to facilitate the creation and maintenance of a computerized database.

Basic Definitions

- ***Database System:***
  - The DBMS software together with the data itself. Sometimes, the applications are also included.



Typical DBMS Functionality

- Define a particular database in terms of its data types, structures, and constraints.
- Construct or Load the initial database contents on a secondary storage medium.
- Manipulating the database:
  - **Retrieval**: Querying, generating reports
  - **Modification**: Insertions, deletions and updates to its content
    - Accessing the database through Web applications
- Processing and Sharing by a set of concurrent users and application programs.

Typical DBMS Functionality

- Other features:
  - Protection or Security measures to prevent unauthorized access
  - Presentation and Visualization of data
  - Maintaining the database and associated programs over the lifetime of the database application
    - Called database, software, and system maintenance

Example of a database

- **Mini-world for the example:**
  - Part of a UNIVERSITY environment.

- **Some mini-world entities:**
  - STUDENTs
  - COURSEs
  - SECTIONs (of COURSEs)
  - (academic) DEPARTMENTs
  - INSTRUCTORs.

Example of a database

- **Some mini-world relationships:**
  - SECTIONs are of specific COURSEs
  - STUDENTs take SECTIONs
  - COURSEs haveprerequisite COURSEs
  - INSTRUCTORs teach SECTIONs
  - COURSEs are offered by DEPARTMENTs
  - STUDENTs major in DEPARTMENTs

- **Note:** The above entities and relationships are typically expressed in a conceptual data model, such as the ENTITY- RELATIONSHIP data model.

Example of a database

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

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Fig 2 A database that stores student and course information

Main Characteristics of the Database Approach

- **Self-describing nature of a database system**
  - A DBMS **catalog** stores the description of a particular database (data structures, types, and constraints)
    - The description is called **meta-data**.
  - This allows the DBMS software to work with different database applications.

- **Insulation between programs and data**
  - Called **program-data independence**.
  - Allows changing data structures and storage organization.

Example of 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

Fig3: An example of a database catalog (for the database in fig 2)

Main Characteristics of the Database Approach (continued)

- **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.

- **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.

Main Characteristics of the Database Approach (continued)

- **Sharing of data and multi-user transaction processing**
  - 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 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 (Continued)

- **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.
  - **End-users:** They use the data for queries, reports and some of them update the database content.

Advantages of Using the DBMS Approach

- Controlling redundancy in data storage and in development and maintenance efforts.
  - The data is shared by all users

- Restricting unauthorized access to data.
  - The data security and privacy can be managed and ensured because the data entry in the database occurs once only and is protected by the security measures.

- Providing persistent storage for program Objects and data structure.

Advantages of Using the DBMS Approach (continued)

- Providing Storage Structures 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.
- Need to reduce long lead times and high cost in new application development
- Lots of data shared through out the organization

Historical Development of Database Technology

- Early Database Applications:
  - The Hierarchical and Network Models were introduced in mid 1960s and dominated during the seventies.
  - A bulk of the worldwide database processing still occurs using these models, particularly, the hierarchical model.

- Relational Model based Systems:
  - Relational model was originally introduced in 1970, was heavily researched and experimented within IBM Research and several universities.
  - Relational DBMS Products emerged in the early 1980s.

Historical Development of Database Technology (continued)

- Object-oriented and emerging applications:
  - Object-Oriented Database Management Systems (OODBMSs) were introduced in late 1980s and early 1990s to cater to the need of complex data processing.
  - Many relational DBMSs have incorporated object database concepts, leading to a new category called *object-relationa*l DBMSs (ORDBMSs)
  - *Extended relational* systems add further capabilities (multimedia data, XML, and other data types)

Historical Development of Database Technology (continued)

- Data on the Web and E-commerce Applications:
  - Web contains data in HTML (Hypertext markup language) with links among pages.
  - This has given rise to a new set of applications and E-commerce is using new standards like XML (eXtended Markup Language).
  - Script programming languages such as PHP and JavaScript allow generation of dynamic Web pages that are partially generated from a database.
    - Also allow database updates through Web pages

Extending Database Capabilities

- New functionality is being added to DBMSs in the following areas:
  - Scientific Applications
  - XML (eXtensible Markup Language)
  - Image Storage and Management
  - Audio and Video Data Management
  - Data Warehousing and Data Mining
  - Spatial Data Management
  - Time Series and Historical Data Management

When not to use a DBMS

- Main inhibitors (costs) of using a DBMS:
  - High initial investment and possible need for additional hardware.
  - Overhead for providing generality, security, concurrency control, recovery, and integrity functions.

- When a DBMS may be unnecessary:
  - If the database and applications are simple, well defined, and not expected to change.
  - If there are stringent real-time requirements that may not be met because of DBMS overhead.
  - If access to data by multiple users is not required.

When not to use a DBMS

- When no DBMS may suffice:
  - If the database system is not able to handle the complexity of data because of modeling limitations.
  - If the database users need special operations not supported by the DBMS.