IT-212
Database Systems
Lecture-01

Dr. Muhammad Asif Khan
Assistant Professor

Department of Computer Science & IT
Sarhad University of Science and Information Technology,
Peshawar

About the Course

- **Course Title:** Database Systems
- **Course Code:** IT-212
- **Credit Hours:** 3-1
- **Program:** BS (CS/SE)
- **Pre-requisite:** Programing and Data structures concepts
- **Semester / Sections:** 3rd
- **Textbooks:**
  1. Database System Concepts by Abraham Silberschatz, Henry F. Korth, and S. Sudarshan. (Latest Edition).
  2. Fundamentals of Database Systems by Ramez Elmasri and Shamkant B. Navathe. (Latest Edition).
- **Reference Books:**
  1. Database Management Systems" by Raghu Ramakrishnan and Johannes Gehrke
  2. SQL and Relational Theory: How to Write Accurate SQL Code" by C.J. Date

Course Description

- This course provides a comprehensive introduction to database systems, focusing on foundational principles and real-world applications.
- Students will explore essential concepts such as data modeling, relational databases, and SQL, along with advanced topics like transaction management, normalization, and database security.
- The course is designed to build practical skills in database design, implementation, and management, preparing students to work with databases in various environments.

Course Objectives

- **Upon completing this course, students will be able to:**
  1. Understand the fundamental concepts and architecture of database systems.
  2. Develop skills in database design, including data modeling and normalization techniques.
  3. Write and execute SQL queries for data retrieval, insertion, update, and deletion.
  4. Apply principles of transaction management and concurrency control to ensure data integrity.
  5. Implement security and access control mechanisms for database protection.

Learning Outcomes

1. Describe the role and importance of database systems in various applications.
2. Design and normalize databases to maintain data integrity and reduce redundancy.
3. Use SQL to interact with databases, retrieving and manipulating data efficiently.
4. Manage transactions, handle concurrency issues, and recover from failures.
5. Ensure database security and manage user access to sensitive information.
6. Analyze and solve real-world problems using a well-designed database solution.

Course Contents at a Glance (1)

- Introduction to Database Concepts
- Database Environment
- The Relational Model
- Relational Algebra
- SQL: Data Definition
- SQL: Data Manipulation
- Database Planning, Design, and Administration
- Fact-Finding Techniques

Course Contents at a Glance (2)

- Entity-Relationship Model
- Enhanced E-R Model
- Normalization
- Conceptual, Logical, and Physical Database Design
- Transaction Management
- Backup
- Security

Today's Topics

- Some common uses of database systems
- Characteristics of file-based systems
- Problems with file-based approach
- The Database (DB)
- Database Management System (DBMS)
- Typical functions of a DBMS
- Major components of the DBMS environment
- Personnel involved in the DBMS environment
- Advantages and disadvantages of DBMSs
- History of the development of DBMSs

Examples of Database Applications

- Student Management Systems
- Customer Relationship Management (CRM)
- Inventory Management Systems
- Healthcare Management Systems
- Financial and Banking Systems
- E-commerce Platforms
- Human Resource Management Systems (HRMS)
- Social Media Platforms
- Library Management Systems
- Logistics and Supply Chain Management

Manual Filing Systems

- A manual file system is a traditional way of organizing and managing data without computers, relying entirely on physical files and folders.
- This system typically involves paper documents stored in filing cabinets, folders, or binders and is organized using a specific method (e.g., alphabetically, by date, or by category).
- For example, a manual file system might work for a small business or library.

File-Based Systems (1)

- A file-based system is an older way of managing data by storing it in flat files on the file system rather than using a database management system (DBMS).
  - Library Management System (File-Based Example): In a simple file-based library management system, data is organized using flat files, with different files storing different types of information. For example;

File-Based Systems (2)

- **Books File**:
  - Stores information about each book in the library, such as title, author, ISBN, and publication date.
  - Each line might represent a single book, separated by delimiters like commas or tabs.
- **Members File:**
  - Stores member information, like member ID, name, contact details, and membership status.
- **Transactions File:**
  - Tracks books borrowed by members, with information such as the book ID, member ID, issue date, and return date.
- **Staff File:**
  - Keeps records of library staff, including staff ID, name, position, and contact information.

File-Based Systems (3)

- Early attempt to Computerize the manual filing system
- Collection of application programs that perform services for the end users (e.g. reports).
- Each program defines and manages its own data.

- Consider **PropHome** example for file-based systems
  - **Sales Department :** responsible for selling and renting of properties
  - **Contract Department:** responsible for handling lease agreements

File-Based Systems (4)

Sales Department

- **PropertyForRent:** (propertyNo, street, city, postcode, type, rooms, rent)
- **Client:** (clientNo, firstName, lastName, telNo, preftype, maxRent)
- **PrivateOwner:** (ownerNo, firstName, lastName, address, telNo)

Contract Department

- **Lease:** (leaseNo, propertyNo, clientNo, rent , paymentMethod, deposit, paid, rentStart, rentFinish, duration)
- **PropertyForRent:** (propertyNo, street, city, postcode, type, rooms, rent)
- **Client:** (clientNo, firstName, lastName, telNo, preftype, maxRent)

File-Based Systems (5)

PropertyForRent

propertyNo	street	city	postcode	type	rooms	Rent_PKR
PA14	…	Peshawar	…	House	6	65K
PL94	…	Lahore	…	Flat	4	40K
PG45	…	Mardan	…	Flat	3	35K

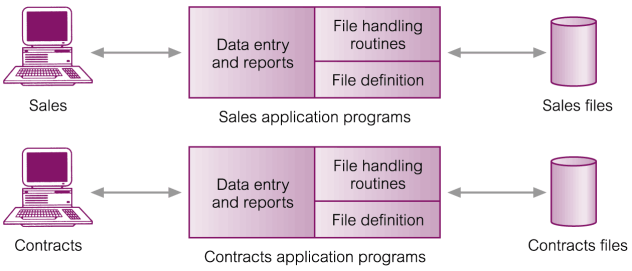
PrivateOwner

ownerNo	firstName	lastName	address	telNo
C-46	Aslam	Khan	Karachi	021-8235579
C-87	Jahan	Bakht	Lahore	042-5698342

Client

clientNo	firstName	lastName	telNo	prefType	maxRent
CR76	Salman	Akram	091-1234567	Flat	50K
CR56	Abid	Jamal	051-1204589	Flat	35K

File-Based Processing



Sales Department

- **PropertyForRent:** (propertyNo, street, city, postcode, type, rooms, rent)
- **Client:** (clientNo, firstName, lastName, telNo, preftype, maxRent)
- **PrivateOwner:** (ownerNo, firstName, lastName, address, telNo)

Contract Department

- **Lease:** (leaseNo, propertyNo, clientNo, rent , paymentMethod, deposit, paid, rentStart, rentFinish, duration)
- **PropertyForRent:** (propertyNo, street, city, postcode, type, rooms, rent)
- **Client:** (clientNo, firstName, lastName, telNo, preftype, maxRent)

Limitations of a File-Based System (1)

- **Separation and isolation of data**
  - Each program maintains its own set of data.
  - Users of one program may be unaware of potentially useful data held by other programs.
- **Duplication of data**
  - Decentralized approach taken by each department.
  - Same data is held by different programs.
  - Wasted space and potentially different values and/or different formats for the same item.
- **Data dependence**
  - File structure is defined in the program code.
- **Incompatible file formats**
  - Programs are written in different languages, and so cannot easily access each other's files.

Limitations of a File-Based System (2)

- **Fixed Queries of application programs**
  - Programs are written to satisfy particular functions.
  - Any new requirement needs a new program.
- **Data Inconsistency**
  - Changes in one file (like change in address) won't update in related files.
- **Poor Scalability and Performance**
  - As the number of files grows, searching and retrieving data becomes slower.

Why Database Approach?

- **Because:**
  - 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.
- **Result:**
  - The **Database** and **Database Management System (DBMS)**.

Database

- A structured collection of data that is stored and accessed electronically.
  - Characteristics: Organized, easily accessible, and manageable
- Shared collection of logically related data, and a description of this data, designed to meet the information needs of an organization.
  - Logically related data comprises entities, attributes, and relationships of an organization's information.
- Metadata provides description of data to enable program-data independence.

Importance of Database

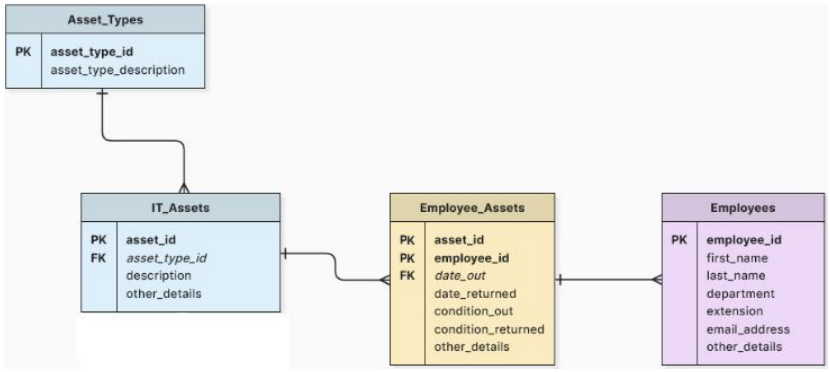
1. **Data management**
   - Efficiently store, retrieve, and manage data.
2. **Decision-making**
   - Supports business intelligence and analytics.
3. **Automation**
   - Enables applications to operate smoothly and efficiently

- Large organizations (e.g., NADRA) rely on database to manage vast amounts of information.

- **Real-World Examples of Databases**
  - **E-commerce:** Amazon's product database.
  - **Social Media:** Facebook's user data.
  - **Healthcare:** Electronic health records (EHR) systems

Components of a Database

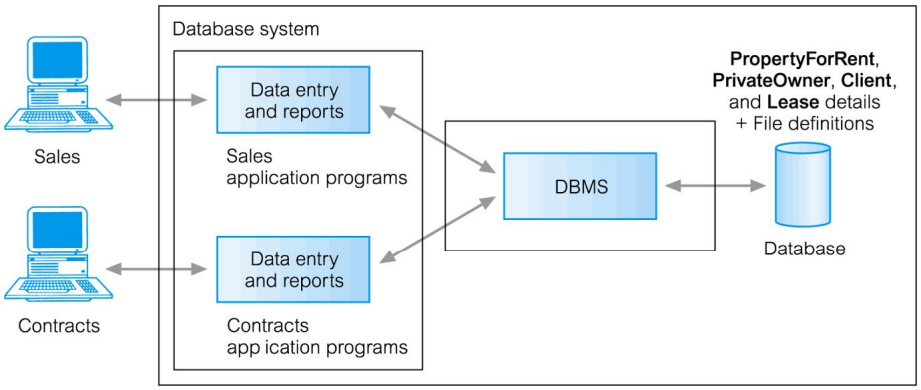
- **Tables**: Structured data organized in rows and columns.
- **Relationships**: How tables interact with each other.
- **Schema**: The framework how data is organized



Database Management System (DBMS) (1)

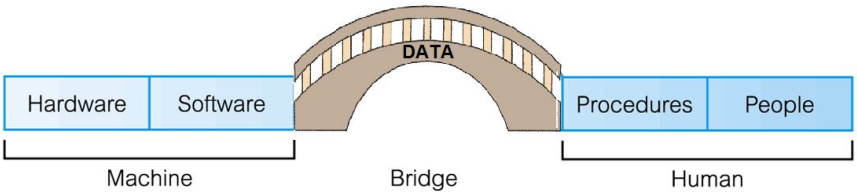
- A software system that enables users to define, create, maintain, and control access to the database.
  - Examples: MySQL, Oracle Database, Microsoft SQL Server, MongoDB
- DBMS Functions
  - Data Definition & Organization: Defining data structures and types.
  - Data Manipulation: CRUD operations (Create, Read, Update, Delete).
  - Data Security: Protecting data from unauthorized access.
  - Backup and Recovery: Ensuring data integrity and availability
  - Concurrency Control
- **(Database) application program:** A computer program that interacts with database by issuing an appropriate request (SQL statement) to the DBMS.

Database Management System (DBMS) (2)



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

Components of DBMS Environment



- **Hardware**: Can range from a PC to a network of computers.
- **Software**: DBMS, operating system, network software (if necessary) and also the application programs.
- **Data**: Used by the organization and a description of this data called the schema.
- **Procedures**: Instructions and rules that should be applied to the design and use of the database and DBMS.
- **People:** Users of the system.

DBMS Generations

1st Generation: File-Based Systems (1960s)

- Data stored in flat files; no DBMS.
- Manual data management; prone to redundancy and inconsistency.

2nd Generation: Hierarchical and Network Models (1970s)

- **Hierarchical Model**: Tree-like structure (e.g., IBM IMS).
- **Network Model**: Graph-based relationships (e.g., CODASYL DBTG).
- Improved structure but rigid and complex querying.

3rd Generation: Relational DBMS (1980s–Present)

- Data stored in tables; SQL for manipulation.
- Normalization, ACID properties, and indexing.
- Examples: Oracle, MySQL, SQL Server.

4th Generation: Post-Relational & Modern DBMS (2000s–Present)

- **NoSQL**: Unstructured data (e.g., MongoDB).
- **Cloud Databases**: Scalable and managed services (e.g., BigQuery).
- **Hybrid Models**: Multi-model support (e.g., Cosmos DB).
- Trends: AI-driven optimization, big data integration, distributed systems.

Summary of DBMS Generations

Generation	Key Technology	Focus	Examples
1st Generation (1960s)	File-based systems	Basic data storage	COBOL file systems
2nd Generation (1970s)	Hierarchical & Network Models	Structured data management	IMS, CODASYL DBTG
3rd Generation (1980s–Present)	Relational DBMS	Standardized, robust data	Oracle, MySQL, SQL Server
4th Generation (2000–Present)	NoSQL, Cloud, Hybrid, AI-driven	Scalability, flexibility	MongoDB, BigQuery, Cosmos DB

Advantages of DBMSs

- Control of data redundancy
- Data consistency
- More information from the same amount of data
- Sharing of data
- Improved data integrity (constraints)
- Improved security (authentication, rights)
- Enforcement of standards (data formats, naming conventions, documentation etc.)
- Improved backup and recovery services

Disadvantages of DBMSs

- Complexity
- Size (disk space for DBMS)
- Cost of DBMS
- Additional hardware costs
- Cost of conversion
- Performance
- Higher impact of a failure

SQL: The Language of Databases

- SQL – Structured Query Language (SQL) for managing and manipulating databases.
  - Key Operations: SELECT, INSERT, UPDATE, DELETE

- **Major Components:**
  1. ***Data definition language (DDL).***
     - Permits specification of data types, structures and any data constraints.
     - All specifications are stored in the database.
  2. ***Data manipulation language (DML).***
     - General enquiry facility (query language) of the data.

The Database Approach

- **Controlled access to database may include:**
  - **A security system**
    - Which prevents unauthorized users accessing the database
  - **An integrity system**
    - Which maintains the consistency of stored data
  - **A concurrency control system**
    - Which allows shared access of the database
  - **A recovery control system**
    - Which restores the database to a previous consistent state in case of hardware or software failure
  - **A user-accessible catalog**
    - Which contains description of the data in the database

Views

- Allows each user to have his or her own view of the database.
- A view is essentially some subset of the database.
- Advantages:
  - Reduce complexity
  - Provide a level of security
  - Provide a mechanism to customize the appearance of the database
  - Present a consistent, unchanging picture of the structure of the database, even if the underlying database is changed

Roles in the Database Environment

- Data Administrator (DA)
  - Database planning
  - Development and maintenance of standards, policies and procedures
- Database Administrator (DBA)
  - Physical realization of the database
  - Physical database design and implementation
  - Security and integrity control
  - Maintenance of the operational system
  - Ensuring satisfactory performance of the applications for users
- Database Designers (Logical and Physical)
- Application Programmers
- End Users (naive and sophisticated)

History of Database Systems (1)

1. **Early Database Concepts (1950s–1960s)**
   - **File-Based Systems**:
     - In the 1950s, computers were first used for data storage in file systems.
     - Data was stored in flat files, requiring extensive programming to retrieve specific information.
     - File systems were rigid, lacked a clear structure, and were prone to redundancy and inconsistency.
   - **Navigational Databases:**
     - By the late 1960s, hierarchical and network database models emerged, improving the organization of data.
     - **Hierarchical Model**: Data was organized like a tree (parent-child relationships). Example: IBM's Information Management System (IMS) in 1966.
     - **Network Model**: Allowed more flexible relationships than the hierarchical model (e.g., CODASYL DBTG, 1969).
       - CODASYL: Conference on Data Systems Languages,
       - DBTG: Data Base Task Group

History of Database Systems (2)

2. **Relational Database Era (1970s)**
   - **Edgar F. Codd's Relational Model (1970):**
     - Codd, working at IBM, proposed the relational database model, introducing the concept of organizing data into tables (relations).
     - Data manipulation was based on mathematical set theory and predicate logic.
     - Key ideas included normalization to eliminate redundancy and ensure data integrity.
   - **First Relational Database Management Systems (RDBMSs):**
     - IBM's System R was the first prototype.
     - Oracle (1979) became the first commercial RDBMS, followed by products like IBM's DB2 and Microsoft's SQL Server.
3. **Emergence of SQL (Structured Query Language)**
   - SQL was developed as part of IBM's System R and became the standard language for relational databases.
   - In 1986, SQL was standardized by ANSI and later ISO.

History of Database Systems (3)

4. **The Rise of Object-Oriented Databases (1980s–1990s)**
   - With advancements in object-oriented programming, **Object-Oriented Database Management Systems (OODBMSs)** emerged.
   - These databases supported complex data types, inheritance, and encapsulation.
   - Popular systems included **ObjectStore** and **Versant**.
5. **Client-Server and Distributed Databases (1990s)**
   - The **client-server model** gained popularity, allowing multiple users to access centralized databases.
   - **Distributed Databases**:
     - Data was distributed across multiple locations, but users could access it as if it were stored in one place.
     - This improved scalability and fault tolerance.

History of Database Systems (4)

6. **NoSQL and Big Data (2000s)**
   - **NoSQL Databases:**
     - Designed to handle unstructured and semi-structured data.
     - Offered high scalability, flexibility, and performance for web-scale applications. Examples: MongoDB, Cassandra, and Redis.
   - **Big Data and Hadoop:**
     - The explosion of data led to the development of frameworks like Hadoop for distributed storage and processing of massive datasets.
7. **Cloud Databases and Database as a Service (2010s–Present)**
   - **Cloud-Based Databases:**
     - Databases moved to the cloud, offering flexibility, scalability, and reduced maintenance.
     - Examples: Amazon RDS, Google Cloud Spanner, and Azure SQL Database.
   - **Database as a Service (DBaaS):**
     - Providers manage the database infrastructure, allowing users to focus on development.

History of Database Systems (5)

8. **Current Trends**
   - **AI-Driven Databases:**
     - AI is now being used for query optimization, data analytics, and database self-management.
   - **Blockchain Databases:**
     - Emerging as a way to ensure secure, tamper-proof transactions and decentralized data storage.
   - **Hybrid and Multi-Model Databases:**
     - Support for combining relational, document, and graph models in a single system (e.g., ArangoDB, Cosmos DB).

Summary

- Some common uses of database systems
- Characteristics of file-based systems
- Problems with file-based approach
- The Database (DB)
- Database Management System (DBMS)
- Typical functions of a DBMS
- Major components of the DBMS environment
- Personnel involved in the DBMS environment
- Advantages and disadvantages of DBMSs
- History of the development of DBMSs

THANK YOU