Al-Mustaqbal University
College of Sciences
Cyber Security Department

جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

كلية العلوم

قسم الامن السيبراني

Lecture: (6)

Subject: Database Systems
Level: Second
Lecturer: Asst. Lecturer Qusai AL-Durrah

Study Year: 2024-2025

View of Data

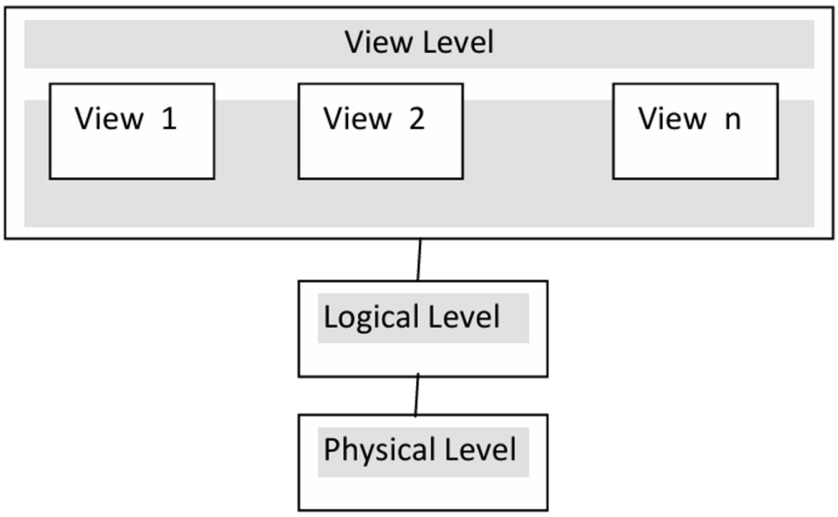
A database system is a collection of interrelated data and a set of programs that allow users to access and modify these data. A major purpose of a database system is to provide users with an abstract view of the data. That is, the system hides certain details of how the data are stored and maintained.

Data Abstraction

For the system to be usable, it must retrieve data efficiently. The need for efficiency has led designers to use complex data structures to represent data in the database. Since many database-system users are not computer trained, developers hide the complexity from users through several levels of abstraction, to simplify users' interactions with the system:

- **Physical level**: The lowest level of abstraction describes how the data are actually stored. The physical level describes complex low-level data structures in detail.
- **Logical level**: The next-higher level of abstraction describes what data are stored in the database, and what relationships exist among those data. The logical level thus describes the entire database in terms of a small number of relatively simple structures. Although implementation of the simple structures at the logical level may involve complex physical-level structures, the user of the logical level does not need to be aware of this complexity. This is referred to as physical data independence. Database administrators, who must decide what information to keep in the database, use the logical level of abstraction.
- **View level**: The highest level of abstraction describes only part of the entire database. Even though the logical level uses simpler structures, complexity remains because of the variety of information stored in a large database. Many users of the database system do not need all this information; instead, they need to access only a part of the database. The view level of abstraction exists to

simplify their interaction with the system. The system may provide many views for the same database.



The three levels of abstraction

Cardinality

Cardinality is a constraint on a relationship specifying the number of entity instances that a specific entity may be related to via the relationship.

The "n" represents an "arbitrary number of instances", and the "1" represents "at most one instance". We interpret the cardinality specifications with the following business rule statements:

• The "n" indicates that the same Product entity can be specified on "any number of" Invoice Lines.

• The "1" indicates that an Invoice Line entity specifies "at most one" Product entity.

One-to-One Relationships

One-to-one relationships have 1 specified for both cardinalities, and do not seem to arise very often. To illustrate a one-to-one, we require very specific business rules.

One-to-Many Relationships

This type of relationship has 1 and n specified for cardinalities, and is very common in database designs. Suppose we have customers and orders and the business rules:

• An order is related to one customer

• A customer can have any number (zero or more) of orders.

We say there is a one-to-many relationship between customer and order, and we draw this as:

Many-to-Many Relationships

Many-to-many relationships have "many" specified for both cardinalities, and are also very common. However, should you examine a data model in some business, there is a good chance you will not see any many-to-many relationships on the diagram. In those cases, the data modeler has resolved the many-to-many relationships into two one to many relationships. Suppose we are interested in courses and students and the fact that students register for courses: Any student may take several courses, A course may be taken by several students. This situation is represented with a many-to-many relationship between Course and Student:

ER Model to Relational Model

ER Model, when conceptualized into diagrams, gives a good overview of entity-relationship, which is easier to understand. ER diagrams can be mapped to relational schema, that is, it is possible to create relational schema using ER diagram. We cannot import all the ER constraints into relational model, but an approximate schema can be generated.

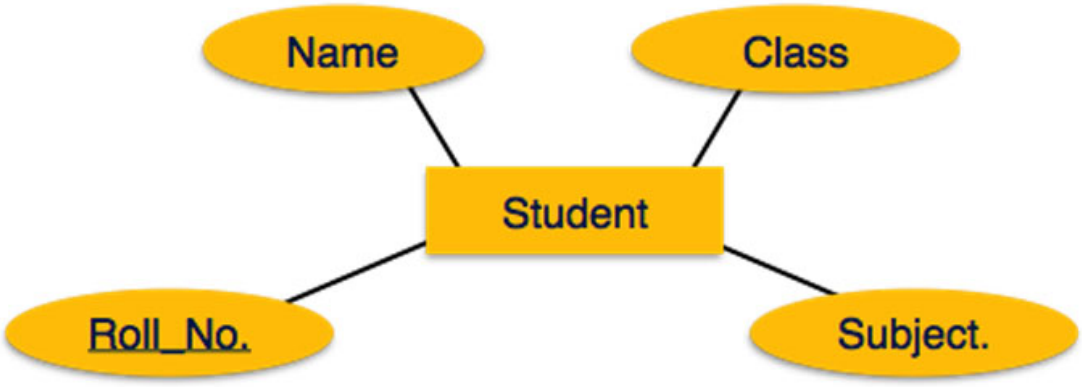
There are several processes and algorithms available to convert ER Diagrams into Relational Schema. Some of them are automated and some of them are manual. We may focus here on the mapping diagram contents to relational basics.

ER diagrams mainly comprise of:

- Entity and its attributes
- Relationship, which is association among entities.

Mapping Entity

An entity is a real-world object with some attributes.



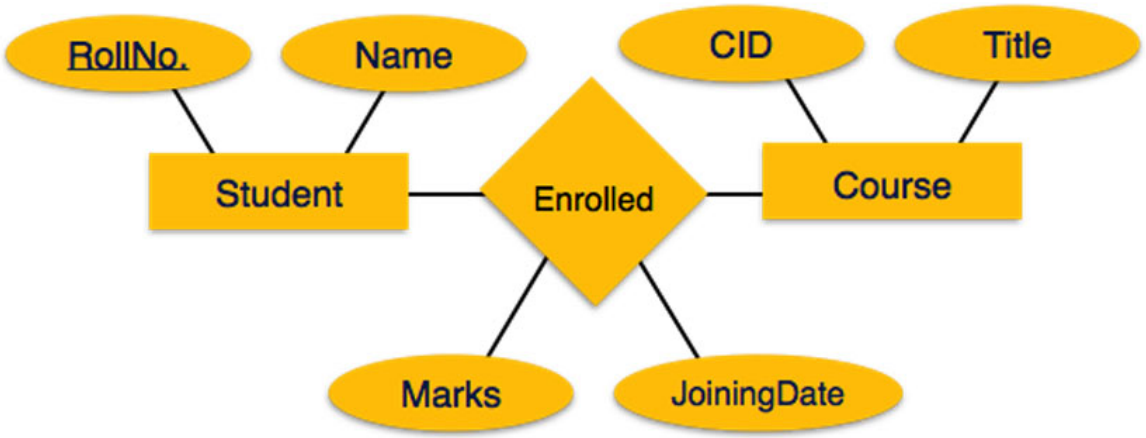
Mapping Process:

- Create table for each entity.
- Entity's attributes should become fields of tables with their respective data types.
- Declare primary key.

Student (relation)			
ID	Name	Class	Subject

Mapping Relationship

A relationship is an association among entities.



Mapping Process:

- Create table for a relationship.
- Add the primary keys of all participating Entities as fields of table with their respective data types.
- If relationship has any attribute, add each attribute as field of table.
- Declare a primary key composing all the primary keys of participating entities.
- Declare all foreign key constraints.

Enrolled (relation)			
ID	CID	Marks	JoiningDate

File System & DBMS

File System: A File Management system is a DBMS that allows access to single files or tables at a time. In a File System, data is directly stored in a set of files. It contains flat files that have no relation to other files (when only one table is stored in a single file, then this file is known as a flat file).

DBMS: A Database Management System (DBMS) is application software that allows users to efficiently define, create, maintain and share databases. Defining a database involves specifying the data types, structures and constraints of the data to be stored in the database. Creating a database involves storing the data on some storage medium that is controlled by DBMS. Maintaining a database involves updating the database whenever required to evolve and reflect changes in the Miniworld and also generating reports for each change. Sharing a database involves allowing multiple users to access the database. DBMS also serves as an interface between the database and end users or application programs. It provides control access to the data and ensures that data is consistent and correct by defining rules on them.

An application program accesses the database by sending queries or requests for data to the DBMS. A query causes some data to be retrieved from the database.

Advantages of DBMS over File system:

- **Data redundancy and inconsistency:** Redundancy is the concept of repetition of data i.e. each data may have more than a single copy. The file system cannot control the redundancy of data as each user defines and maintains the needed files for a specific application to run. There may be a possibility that two users are maintaining the data of the same file for different applications. Hence changes made by one user do not reflect in files used by second users, which leads to inconsistency of data. Whereas DBMS controls redundancy by maintaining a single repository of data that is defined once and is accessed by many users. As there is no or less redundancy, data

remains consistent.

- **Data sharing:** The file system does not allow sharing of data or sharing is too complex. Whereas in DBMS, data can be shared easily due to a centralized system.

- **Data concurrency:** Concurrent access to data means more than one user is accessing the same data at the same time. Anomalies occur when changes made by one user get lost because of changes made by another user. The file system does not provide any procedure to stop anomalies. Whereas DBMS provides a locking system to stop anomalies to occur.

- **Data searching:** For every search operation performed on the file system, a different application program has to be written. While DBMS provides inbuilt searching operations. The user only has to write a small query to retrieve data from the database.

- **Data integrity:** There may be cases when some constraints need to be applied to the data before inserting it into the database. The file system does not provide any procedure to check these constraints automatically. Whereas DBMS maintains data integrity by enforcing user-defined constraints on data by itself.

- **System crashing:** In some cases, systems might have crashed due to various reasons. It is a bane in the case of file systems because once the system crashes, there will be no recovery of the data that's been lost. A DBMS will have the recovery manager which retrieves the data making it another advantage over file systems.

- **Data security:** A file system provides a password mechanism to protect the database but how long can the password be protected? No one can guarantee that. This doesn't happen in the case of DBMS. DBMS has specialized features that help provide shielding to its data.

- **Backup:** It creates a backup subsystem to restore the data if required.

- **Interfaces:** It provides different multiple user interfaces like graphical user interface and application program interface.

- **Easy Maintenance:** It is easily maintainable due to its centralized nature.