DATABASE and E-R MODELING

YouTube video https://www.youtube.com/watch?v=X7v0O8yiUuY

Definition	Key Concepts
Data	
Binary Data	
Database	
Database Strcture	
Main Purpose	
Real World Examples	
Common Software	
DBMS	

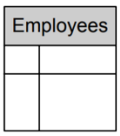
What is a Database?

A **database** is an organized collection of structured information, or data, that is stored in a computer system. One of the most critical aspects of any database is how that data is organized. To understand this organization, we use a modelling approach called **Entity-Relationship (E-R) modelling**, which is a type of "semantic modelling"—an approach that tries to understand and represent the actual meaning behind the data. **Database E-R modelling** is a way to organize information in a computer system by focusing on the meaning of the data. It uses a diagram (ERD) to communicate how a database is structured, making it easier for users to understand what information is stored and how it all fits together.

1. Entities: The "Things"

An **entity** is essentially any object or "thing" that can be clearly identified and about which you want to store information.

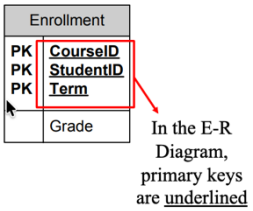
- **Examples:** People (employees), places (business locations), tangible items (equipment), or even events (sales).
- **In a Diagram:** Entities are represented as **rectangles**.



2. Attributes: The "Details"

Attributes are the specific characteristics or details that describe an entity.

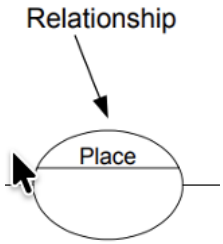
- **Example:** For a "Student" entity, attributes might include their Name, Address, and Phone Number.
- **Primary Key:** This is a special attribute (or a combination of them) that is unique to every single instance. For example, no two students would have the same Student ID.
- **In a Diagram:** Primary keys are usually identified by being **underlined**.



3. Relationships: The "Connections"

Relationships show how different entities are associated with one another.

- **Example:** A relationship exists between "Students" and "Courses" because students enroll in courses.
- **In a Diagram:** These are shown as **ellipses** (ovals) connected by lines to the entities they link



◊ **Degree**

The **degree** refers to the number of entities involved in a relationship:

- **Unary (Recursive):** One entity is related to itself (e.g., an Employee manages other Employees) .
- **Binary:** Two different entities are involved.
- **Ternary:** Three different entities are involved.

◊ **Cardinality**

Cardinality describes the numerical nature of a relationship—essentially, how many instances of one entity can relate to instances of another.

- **One-to-One (1:1):** One customer is related to exactly one account.
- **One-to-Many (1:N):** One advisor can have many students, but each student has only one advisor.
- **Many-to-Many (M:N):** Many students can take many different courses.
- **Symbols:** In diagrams, "one" is often shown as a single vertical line, and "many" is shown as a "crowfoot" symbol.

Cardinality: Constraints on Connections

Cardinality defines the numerical rules for a relationship:

- **Maximum Cardinality:** The highest number of instances of one entity that can be linked to another (e.g., 1:1, 1:N, or M:N).
- **Minimum Cardinality (Optionality):** The lowest number of instances required. A minimum of **zero** means the relationship is optional, while **one** means it is mandatory.
- **Representation:** "One" is shown as a single vertical line, and "Many" is shown as a "crowfoot" symbol.

Creating a ER Database

Creating a database using the **Entity-Relationship (E-R) modelling process** involves a series of structured activities designed to understand and represent the meaning of data.
Here are the specific steps to create a database model as described in the provided document:

Step 1: Identify Entities and Attributes

The first and most critical step is to identify your **Entities**—the objects, people, or concepts about which you need to store data.

- **Identify Entities:** List as many entities as you think are important to be precise. It is easier to remove unnecessary entities later than to add them at the end.
- **Define Attributes:** Identify the characteristics of each entity (e.g., a "Student" entity has attributes like "Name" and "Phone Number").
- **Distinguish Between the Two:** To verify if something is truly an entity, check its attributes. If an entity has no attributes of its own, it is likely just an attribute of another entity.

Step 2: Choose Primary Keys

Every entity must have a **Primary Key**—an attribute that uniquely identifies each instance so that no two records are exactly the same.

- **Select or Create:** If a natural primary key (like an ID number) isn't obvious, you must create a new attribute specifically for identification.
- **Key Requirements:** The chosen key must be unique, always have a valid value, and serve only to identify the entity without containing other "useful" information.

Step 3: Determine Relationships

Next, identify the **Relationships**, which are the associations between your entities.

- **Find Connections:** Look at your source documents; entities that appear in the same section are likely related.
- **Assign Names:** Give each relationship a meaningful name to clarify how the entities interact. If a name is hard to find, combine the names of the two entities involved.

Step 4: Define Cardinality

Cardinality describes the rules for how many instances of one entity can relate to another.

- **Maximum Cardinality:** Determine the most connections possible (e.g., One-to-One, One-to-Many, or Many-to-Many).
- **Minimum Cardinality (Optionality):** Determine if a connection is required (one) or optional (zero).
- **Symbols:** In diagrams, "One" is shown as a single vertical line, and "Many" is shown as a "crowfoot" symbol.

Step 5: Review and Refine

The final step is to compare your completed E-R diagram against the original requirements to ensure nothing was missed.

- **Verify Requirements:** Ensure the diagram can satisfy all the information needs of the system.
- **Exclude Computed Data:** Do not store data that can be calculated (like "Total Price"). These should be omitted from the database to save space and prevent errors .
- **Add Relationship Attributes:** If an attribute (like "Quantity") doesn't belong to a single entity but describes the connection between them, attach it directly to the relationship in your diagram.