SI Course 5 - Entity Association Modeling

1- Entity Relationship Modeling

1-1 Entity Relationship Model

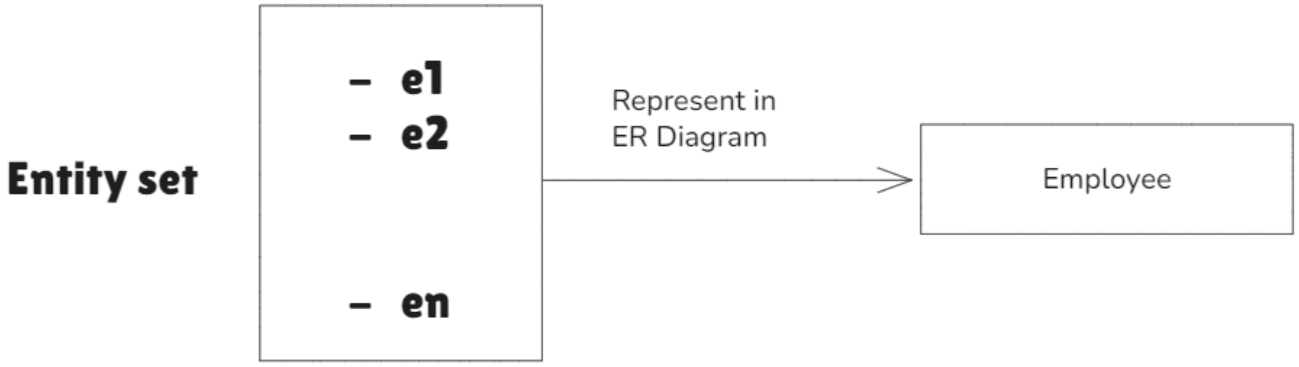
- ER models, also called ER schemas, are represented by ER diagrams.
- ER modeling is based on two concepts:
- Entities, defined as tables containing specific information (data)
- Relationships, defined as the associations or interactions between entities

We'll use the **COMPANY database** throughout the rest of this chapter to illustrate the **Entity-Relationship (ER) model** concepts. This database tracks **employees**, **departments**, and **projects**.

- Key Data Points
  - **Departments**: Each has a unique ID, a name, an office location, and an assigned managing employee.
  - **Projects**: Departments control multiple projects, each with a unique name, a unique number, and a budget.
  - **Employees**: Each has a name, ID number, address, salary, and birthdate. Employees belong to one department, have a direct supervisor, and can work on several projects (recording the start date for each).
  - **Dependents**: We track each employee's dependents, including their name, birthdate, and relationship to the employee.

1-2 Entities:

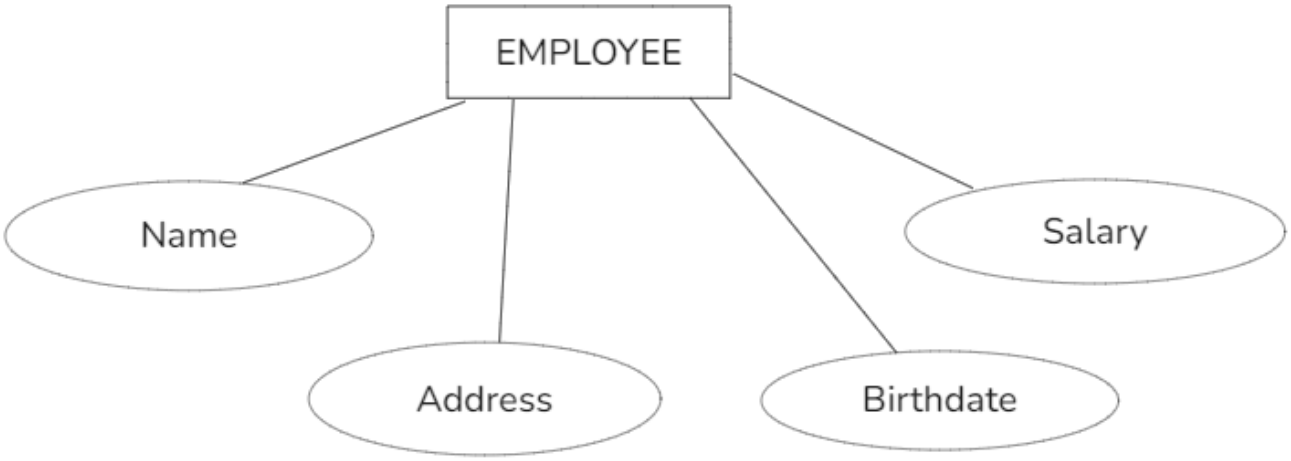
- An entity is a real-world object with an independent existence that can be distinguished from other objects.
- It can represent:
- an object with a real physical existence (Student)
- an object with a conceptual existence (Lecture)
- An entity type defines a collection of similar entities.
- An entity set is a collection of entities of an entity type at a given time.
- In an entity-relationship diagram (ERD), an entity type is represented by a name in a box.



ERD with entity Employee

1-3 Attributes

- Each entity is described by a set of attributes (e.g., Employee = (Name, Address, Date of Birth (Age), Salary)).
- Each attribute has a name and is associated with an entity and a domain of allowed values. However, information about the attribute domain is not presented on the ERD.

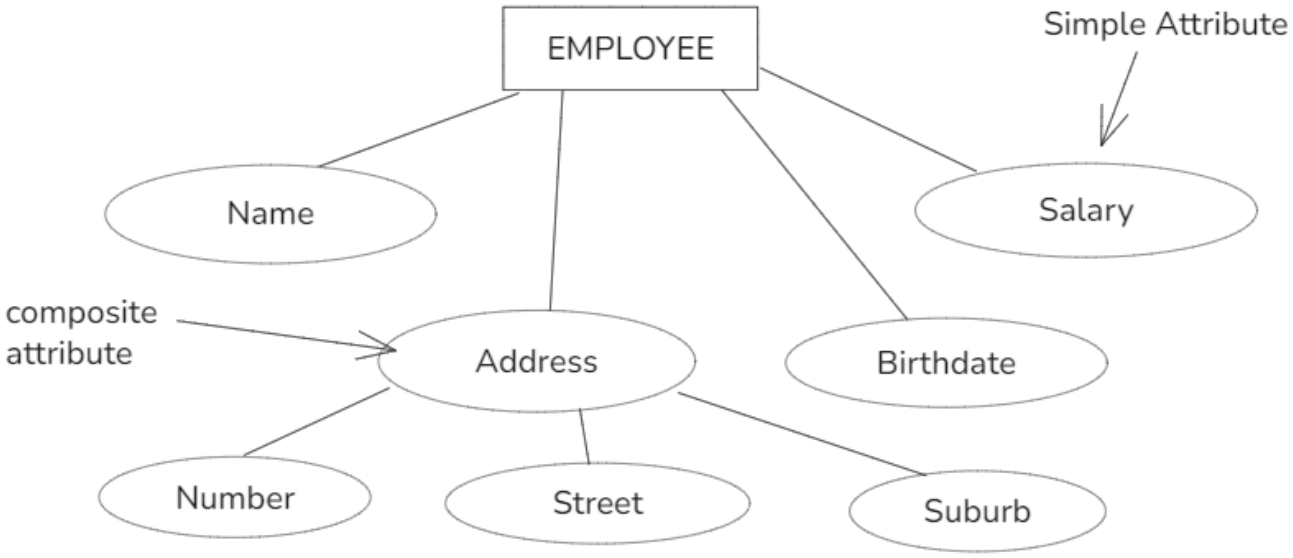


How attributes are represented in an ERD

1-3-1 Attribute Types

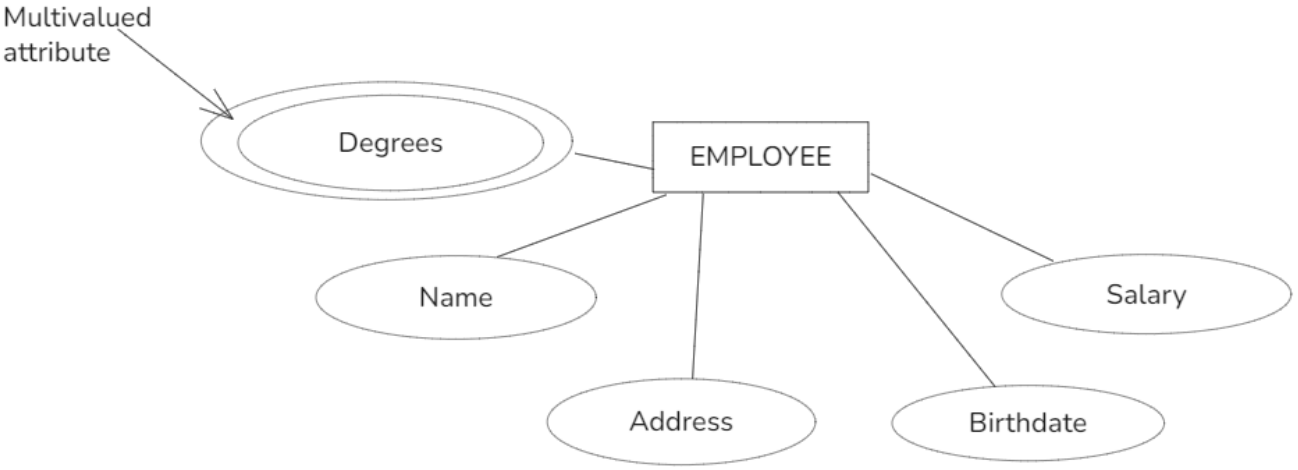
There are a few types of attributes you should know:

- **Simple Attributes:** These are attributes taken from atomic value domains; they are also called single-valued attributes. In a company database, an example of this would be: Name={John}; Age={23}.
- **Composite Attributes:** These are attributes that consist of a hierarchy of attributes.
- In the figure below, the address can consist of the number, the street, and the suburb. This would therefore be written like this: → Address={59 + 'Meek Street' + 'Kingsford'}.



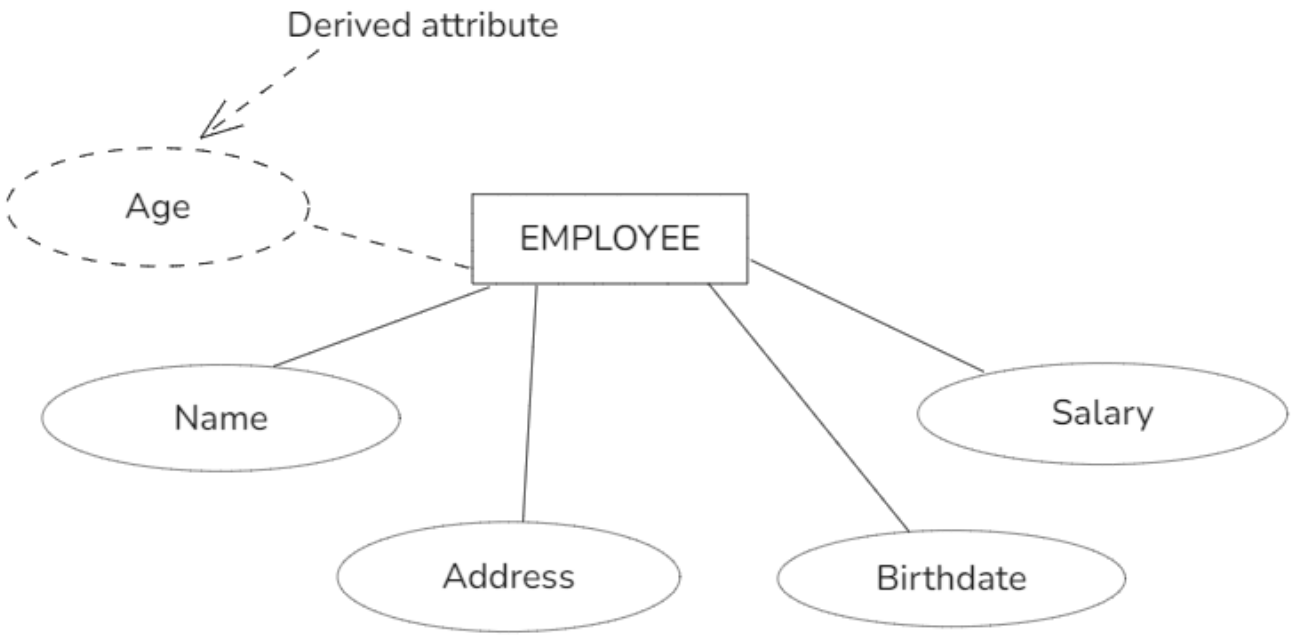
An exemple of composite attributes

- **MultivaluedAttributes:** Multivalued attributes are attributes that have a set of values defined for each entity. An example of a multivalent attribute in the database in the following figure is an employee's degrees: BSc, MIT, PhD.



How attributes are represented in an ERD

- **Derived Attributes:** Derived attributes are attributes that contain values calculated from other attributes. An example of this can be seen in the following figure. Age can be derived from the Date of Birth attribute. In this case, the date of birth is called a stored attribute, which is physically stored in the database.



Example of a derived attribute

1-4 Keys

An important constraint on an entity is the key. A key is an attribute or group of attributes whose values can be used to uniquely identify an individual entity within a set of entities.

1-4-1 Types of Keys

There are several types of keys. They are described below:

- **Candidate Key:** A candidate key is a simple or composite key that is unique and minimal. It is unique because no two rows in a table can have the same value at any time. It is minimal because every column is required to achieve uniqueness.
  - Example: From the COMPANY database, if the entity is Employee (EID, First Name, Last Name, SIN, Address, Phone, BirthDate, Salary, DepartmentID), possible candidate keys are:
    • EID, SIN
    • First Name and Last Name – assuming there is no one else in the company with the same name
    • Last Name and DepartmentID – (no two different employees with these same values)
- **Composite Key:** A composite key is composed of two or more attributes, but it must be minimal.
- **Primary Key:** The primary key is a candidate key selected by the database designer to be used as an identification mechanism for the entire entity set. It must uniquely identify the tuples in a table and not be null. The primary key is indicated in the ER model by underlining the attribute.

    - Example:
        - Employee(EID, First Name, Last Name, SSN, Address, Phone, Year of Birth, Salary, DepartmentID)
- **Secondary Key:** A secondary key is an attribute used strictly for retrieval purposes (can be composite), for example: Phone and Last Name.
- **Alternate Key:** Alternate keys are any candidate keys not chosen as the primary key.
- **Foreign Key:** A foreign key (FK) is an attribute in one table that references the primary key in another table OR it can be null. Foreign and primary keys must be of the same data type.
- **Null**
    - A null is a special symbol, independent of data type, that means either unknown or inapplicable. It does not mean zero or empty. Characteristics of null include:
    - No data entry
    - Not allowed in the primary key
    - Should be avoided in other attributes
    - Can represent
        - An unknown attribute value
        - A known, but missing, attribute value
        - A "not applicable" condition
    - Can create problems when using functions such as COUNT, AVERAGE, and SUM
    - Can create logical problems when relational tables are linked

1-5 relationships

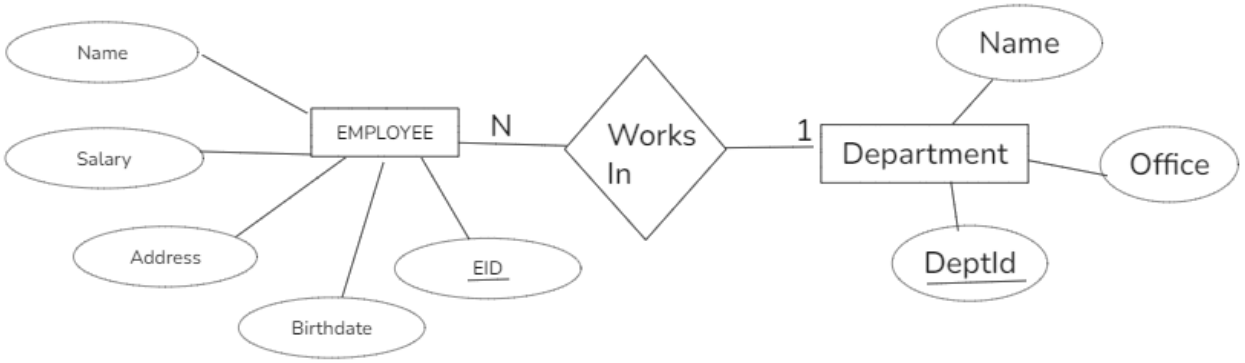
- relationships (relationships) are the glue that binds tables together. They are used to connect related information between tables.

1-5-1 Strength of relationships

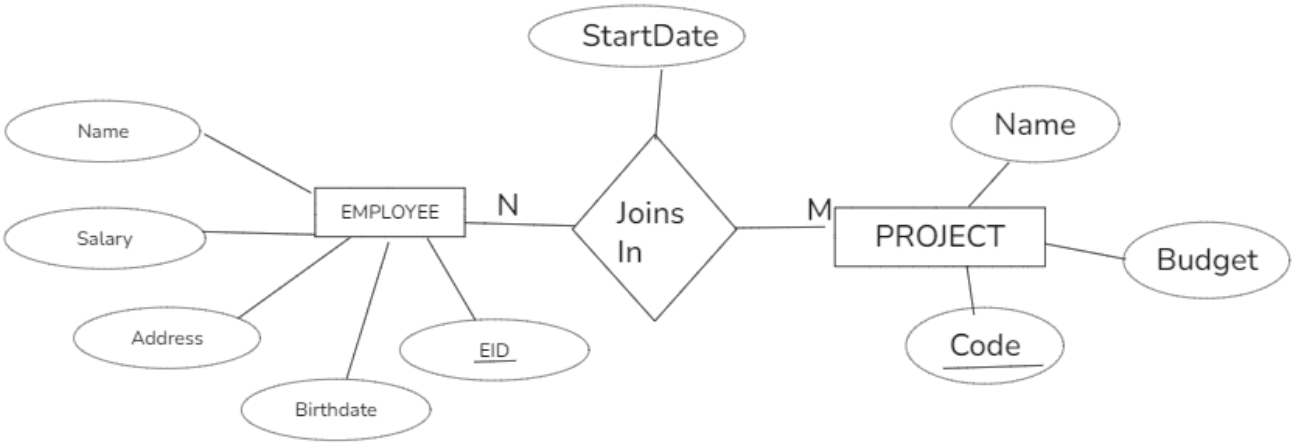
- The strength of an relationship is based on how the primary key of a related entity is defined.
- A weak, or non-identifying, relationship exists if the primary key of the related entity does not contain a component of the primary key of the parent entity.
- Customer(CustID, CustName)
- Order(OrderID, CustID, Date)
- A strong, or identifying, relationship exists when the primary key of the related entity contains the primary key component of the parent entity. Examples include:
- Course(CrsCode, DeptCode, Description)
- Class(CrsCode, Section, ClassTime...)

1-5-2 relationship Types

- **One-to-Many relationship (1:M)**



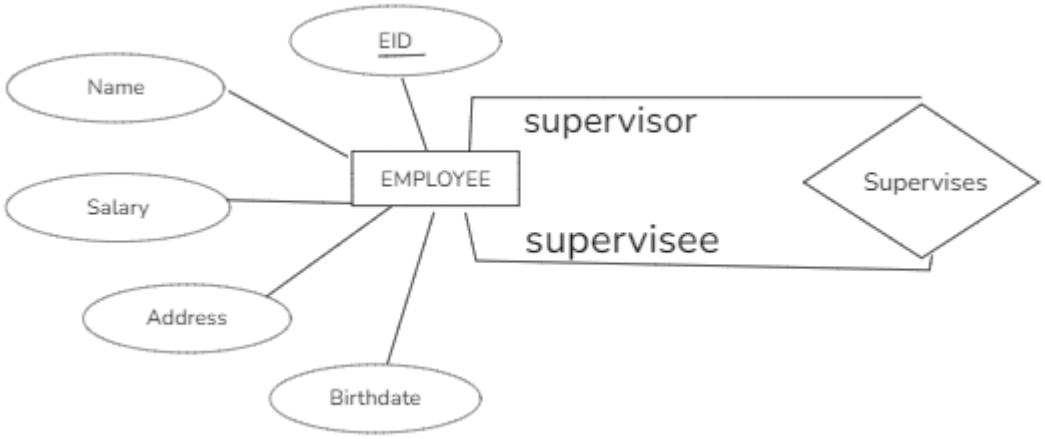
- Relational Schema:
  - EMPLOYEE (EID, Name, Address, Birthdate, Salary, DeptId)
  - DEPARTMENT (DpetId, Name, Office)
- **one-to-one (1:1) relationship**
  - Example:
    - In the COMPANY database, there is the Employees table, in which an employee is associated with a spouse. And a spouse is associated with an employee.
- **many-to-many (M:N) relationship**
  - Cannot be implemented as such in the relational model.
  - Can be changed into two 1:M relationships.
  - Can be implemented by decomposing to produce a set of 1:M relationships.
  - Involves the implementation of a composite entity.
  - Creates two or more 1:M relationships.
  - The composite entity table must contain at least the primary keys of the original tables.
  - The linking table contains multiple occurrences of the foreign key values.
  - Additional attributes can be assigned as needed.
  - It can avoid the problems inherent in an M:N relationship by creating a composite entity or a bridge entity. For example, an employee can work on multiple projects OR a project can have multiple employees working on it, depending on business rules. Or, a student can have multiple classes and a class can contain multiple students.



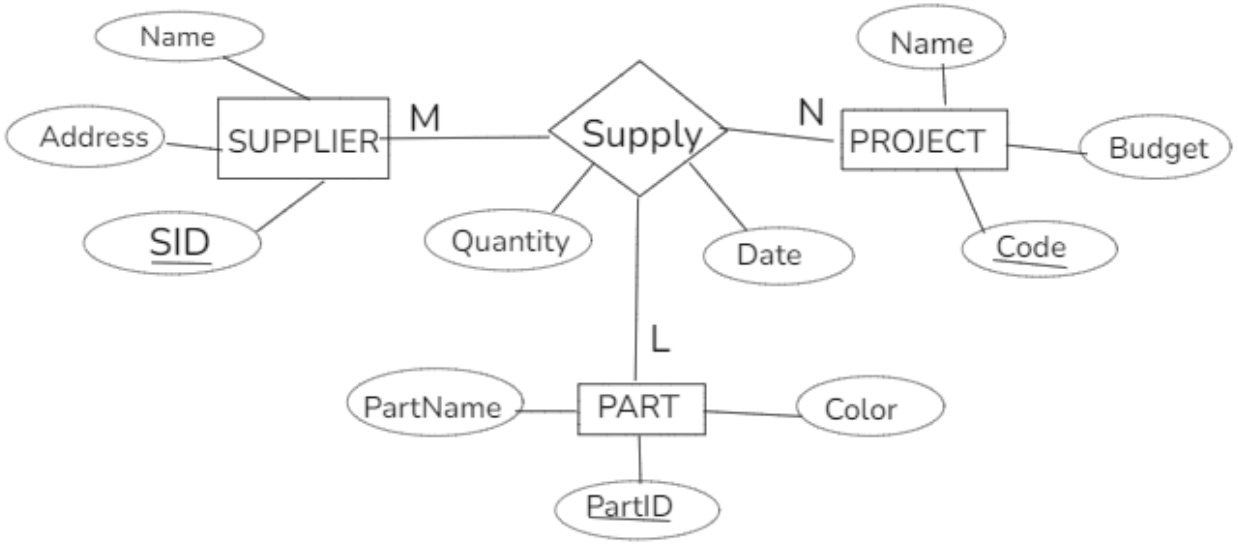
- Relational Schema:
  - EMPLOYEE (EID, Name, Address, Birthdate, Salary, DeptId)
  - PROJECT (Code, Name, Budget)
  - JOIN(EID,Code, StartDate)

- *Example mapping of a binary relationship type M:N
  - For each binary relationship M:N, identify two relationships.
  - A and B represent two types of entities participating in R.
  - Create a new relationship S to represent R.
  - S must contain the PKs of A and B. These together can be the PKs in table S OR these, along with another simple attribute in the new table R, can be the PKs.
  - The combination of the primary keys (A and B) will make the primary key of S.

- **Unit-wise relationship (recursive)**



- Relational Schema:
  - EMPLOYEE (EID, Name, Address, Birthdate, Salary, Super-EID)
- **ternary relationships



- Relational Schema:
  - SUPPLIER (SID, Name, Address)
  - PROJECT (Code, Name, Budget)

    - PART (PartID, PartName, Color)
    - Supply (SID, Code, PartID, Quantity, Date)

- *Example of mapping a ternary relationship type*
    - For each n-ary relationship (>2), create a new relationship to represent the relationship.
    - The primary key of the new relationship is a combination of the primary keys of the participating entities that hold the N (many) side.
    - In most cases of n-ary relationships, all participating entities have multiple sides.

2- Types of Entities

2-1 Independent Entities

- Independent entities, also called cores, are the backbone of the database.
- They form the foundation upon which other tables are built.
- Cores have the following characteristics:
    - They are the building blocks of a database.
    - The primary key can be simple or composite.
    - The primary key is not a foreign key.
    - They do not depend on another entity for their existence.

2-2 Dependent Entities:

- Dependent entities, also called derived entities, depend on other tables for their meaning.
- These entities have the following characteristics:
    - Dependent entities are used to link two cores together.
    - They are said to be dependent on two or more tables for their existence.
    - Many-to-many relationships become associative tables with at least two foreign keys.
    - They can contain other attributes.
    - The foreign key identifies each associated table.
    - There are three options for the primary key:
        1. Use a composite of foreign keys from associated tables if they are unique.
        2. Use a composite of foreign keys and a qualifying column.
        3. Create a new single primary key.

2-3 Characteristic Entities:

- Characteristic entities provide more information about another table.
- These entities have the following characteristics:

- They represent multivalued attributes.
- They describe other entities.
- They generally have a one-to-many relationship.
- The foreign key is used to further identify the characterized table.
- The options for the primary key are as follows:

2. Use a composite foreign key plus a qualifying column.
3. Create a new simple primary key.

- Example:
- Employee (EID, Name, Address, Age, Salary) – EID is the simple primary key.
- EmployeePhone (EID, Phone) – EID is part of a composite primary key. EID is also a foreign key.

Bibliographic References

- Watt, Adrienne, and Nelson Eng. *Database design*. 2nd Edition. BCcampus, 2014