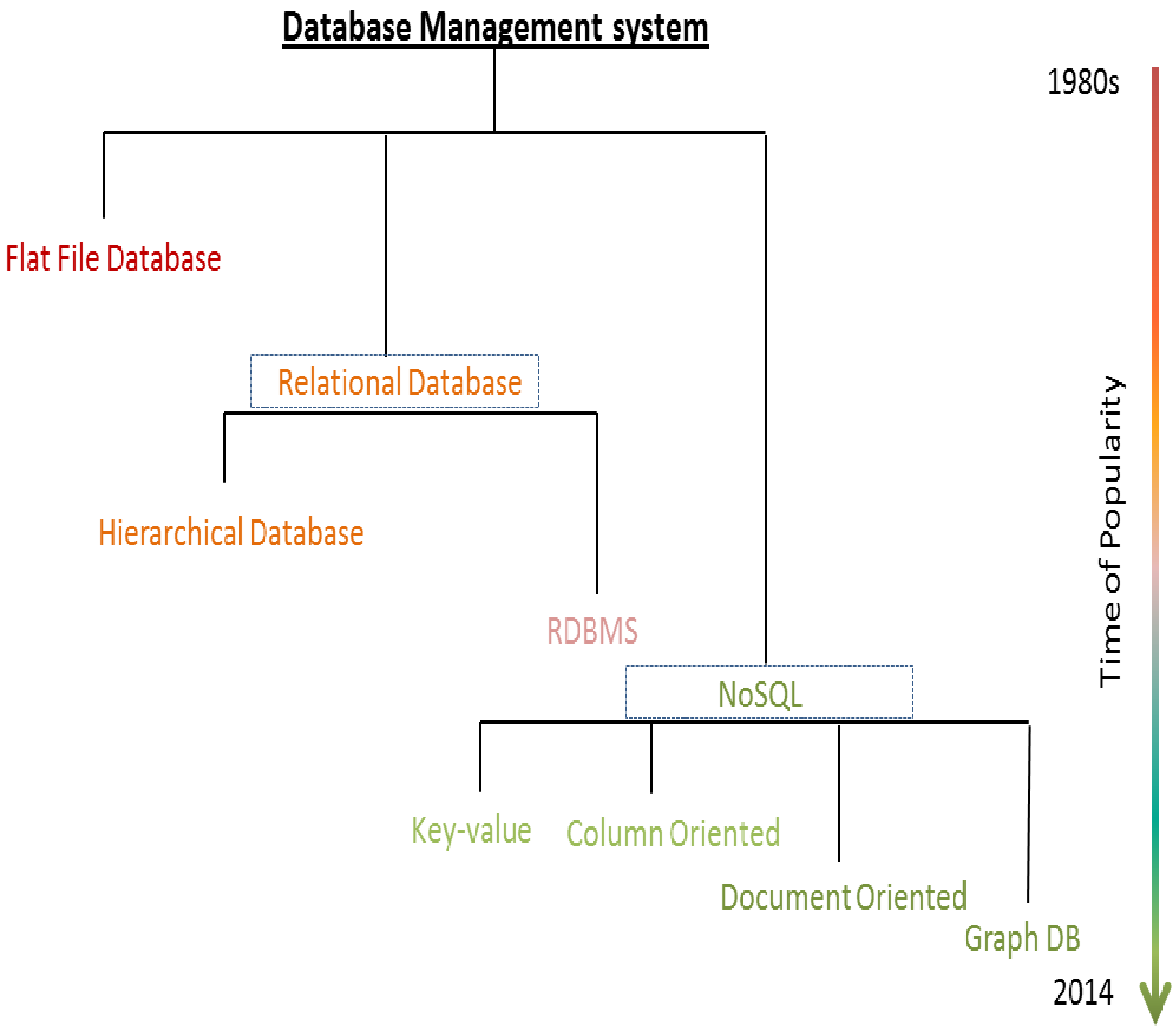
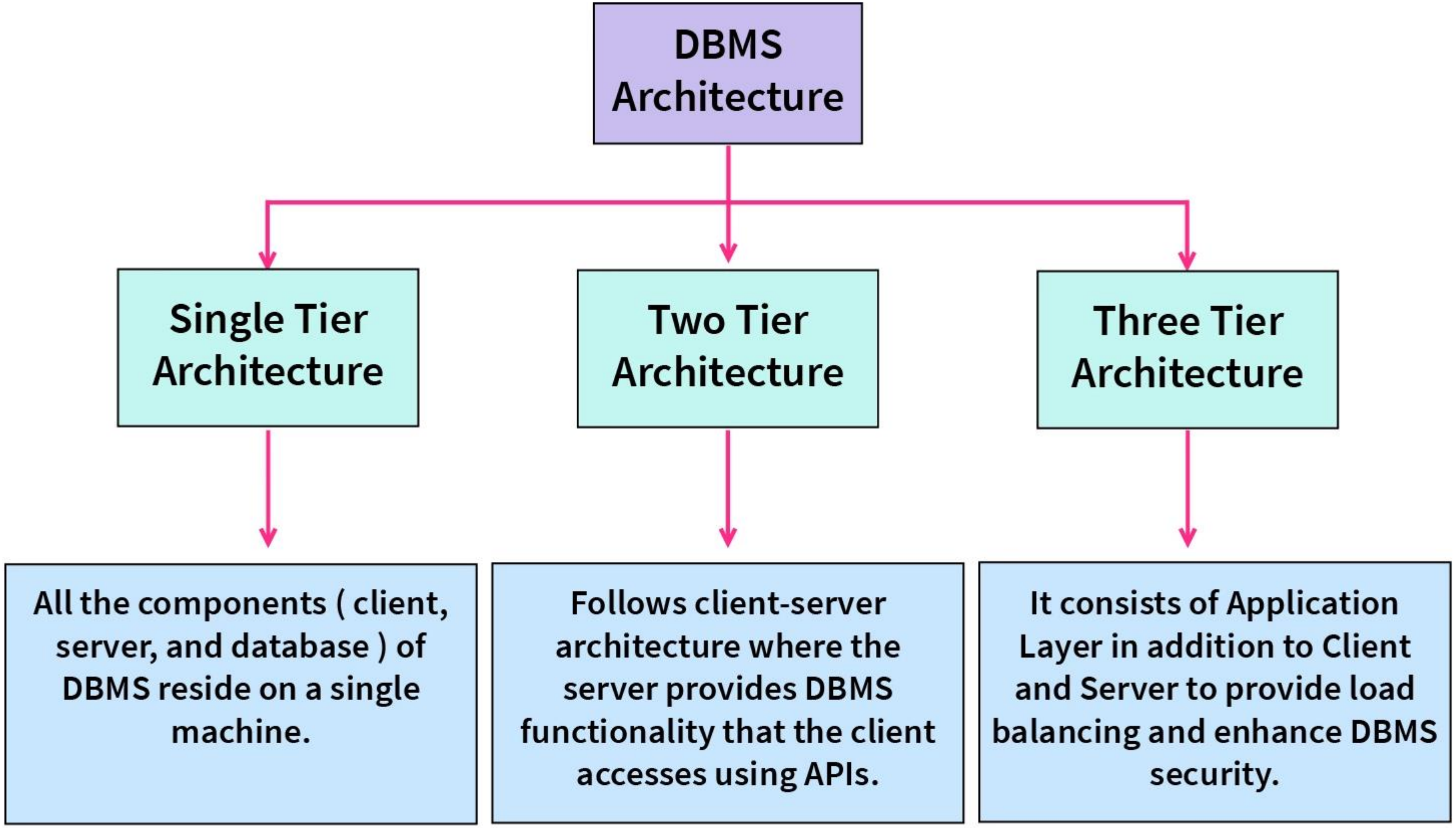
INTRODUCTION AND DATA MODELING

Need of Database Management Systems (DBMS

- Data Organisation
- Data Security
- Data Integrity
- Data Sharing

EVOLUTION OF DATA BASE MANAGEMENT SYSTEM





DBMS Architecture

Single Tier Architecture

Two Tier Architecture

Three Tier Architecture

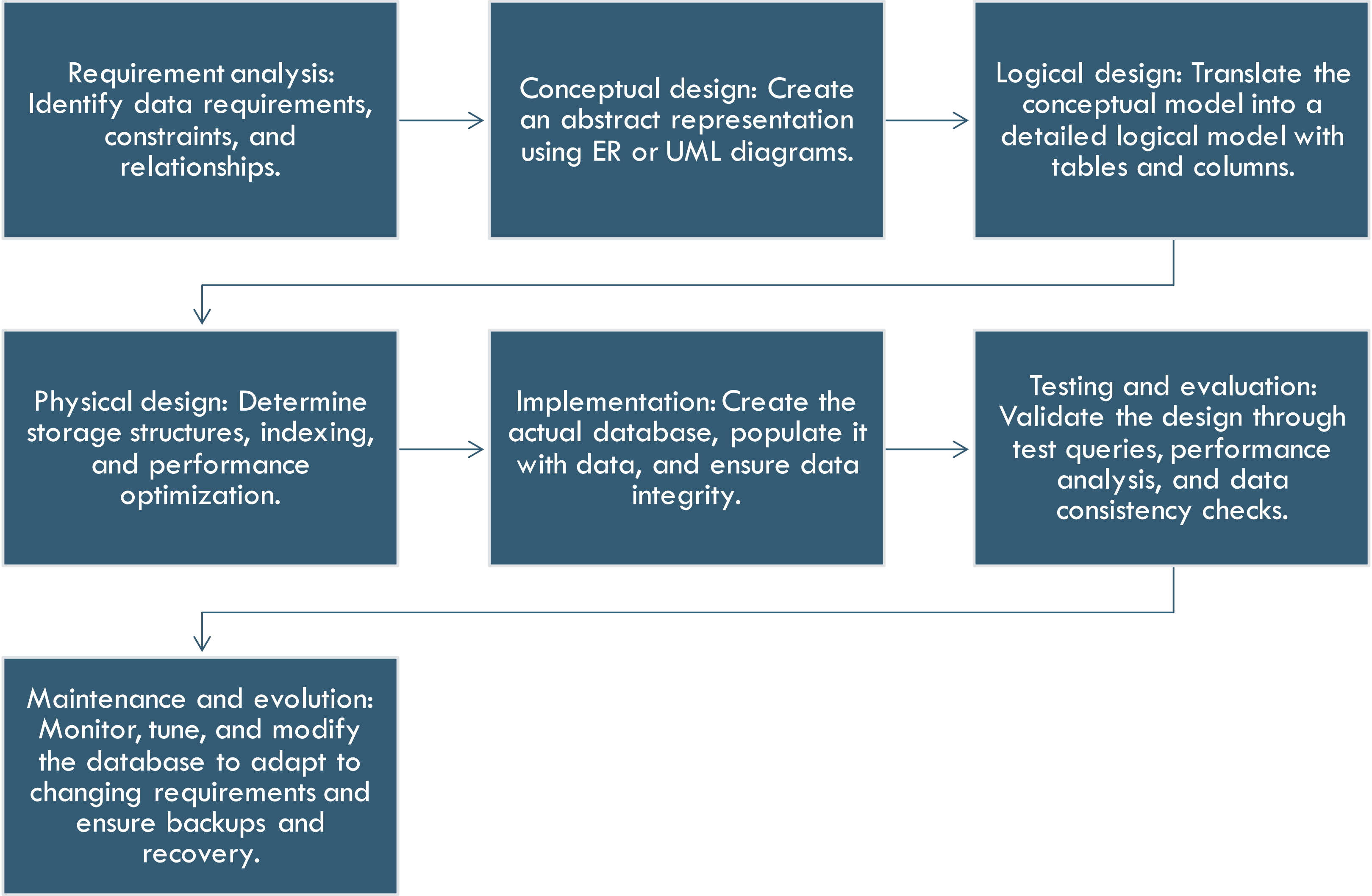
All the components (client, server, and database) of DBMS reside on a single machine.

Follows client-server architecture where the server provides DBMS functionality that the client accesses using APIs.

It consists of Application Layer in addition to Client and Server to provide load balancing and enhance DBMS security.

DBMS ARCHITECTURE

DATABASE DESIGN PROCESS



Requirement analysis: Identify data requirements, constraints, and relationships.

Conceptual design: Create an abstract representation using ER or UML diagrams.

Logical design: Translate the conceptual model into a detailed logical model with tables and columns.

Physical design: Determine storage structures, indexing, and performance optimization.

Implementation: Create the actual database, populate it with data, and ensure data integrity.

Testing and evaluation: Validate the design through test queries, performance analysis, and data consistency checks.

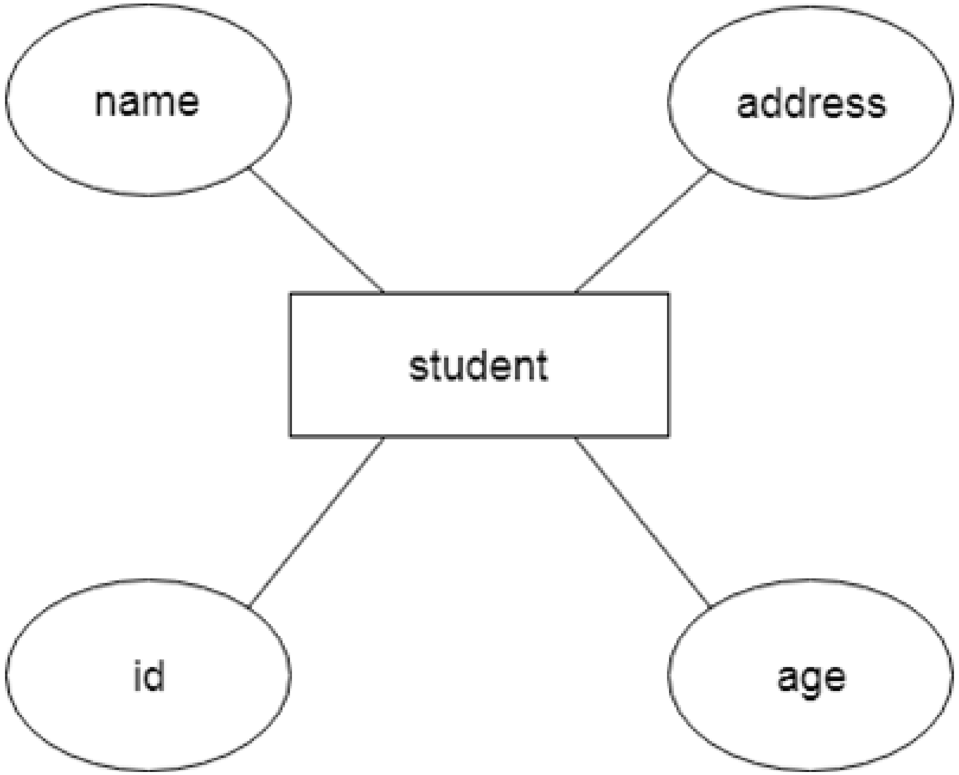
Maintenance and evolution: Monitor, tune, and modify the database to adapt to changing requirements and ensure backups and recovery.

WHAT IS DATA MODEL?

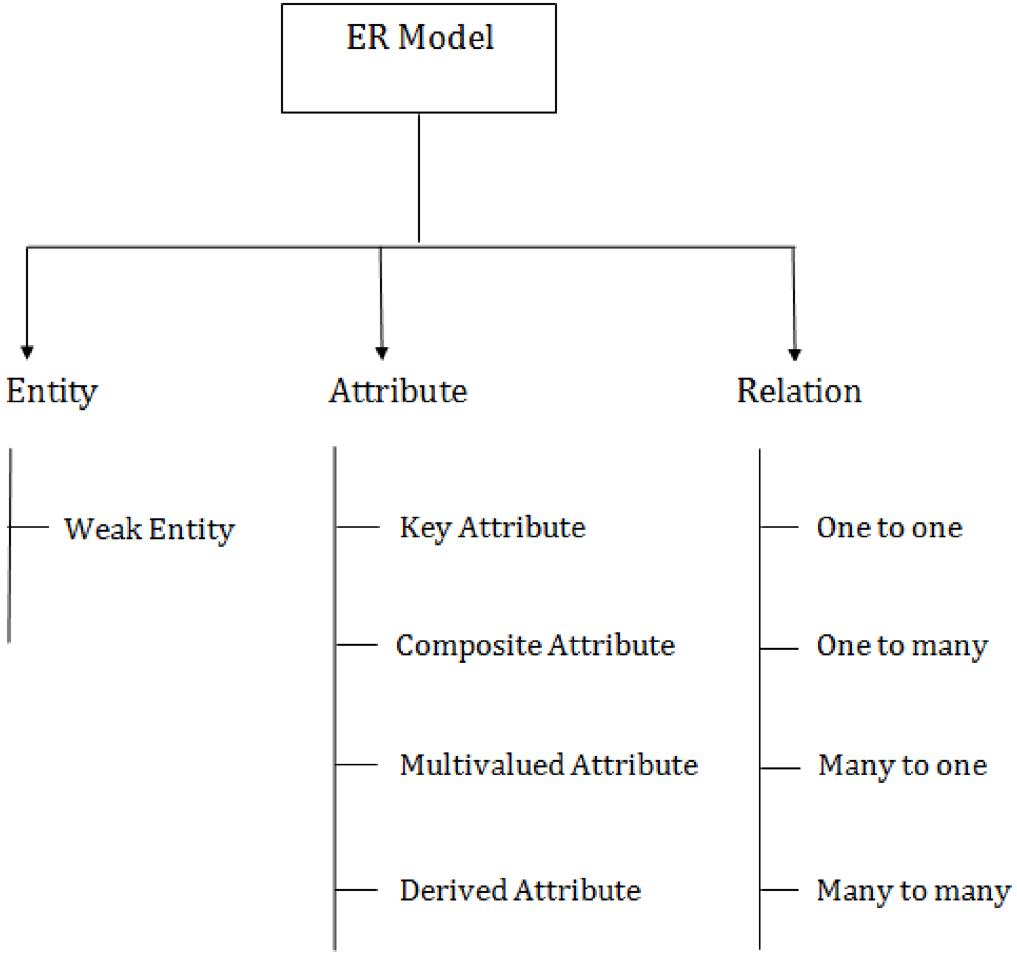
- Data Model is a collection of conceptual tools for describing data, data relationships, data semantics and consistency constraint.
- A data model is a conceptual representation of data structures required for data base and is very powerful in expressing and communicating the business requirements.
- A data model visually represents the nature of data, business rules governing the data, and how it will be organized in the database

ER MODEL

- The Entity Relationship (ER) Model is a conceptual modeling technique used to represent entities, attributes, and relationships in a database system. It provides a visual representation of the structure and organization of data.
- It is a high-level data model. This model is used to define the data elements and relationship for a specified system.
- It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.
- In ER modeling, the database structure is portrayed as a diagram called an entity-relationship diagram.



For example, Suppose we design a school database. In this database, the student will be an entity with attributes like address, name, id, age, etc. The address can be another entity with attributes like city, street name, pin code, etc and there will be a relationship between them.



ER Model

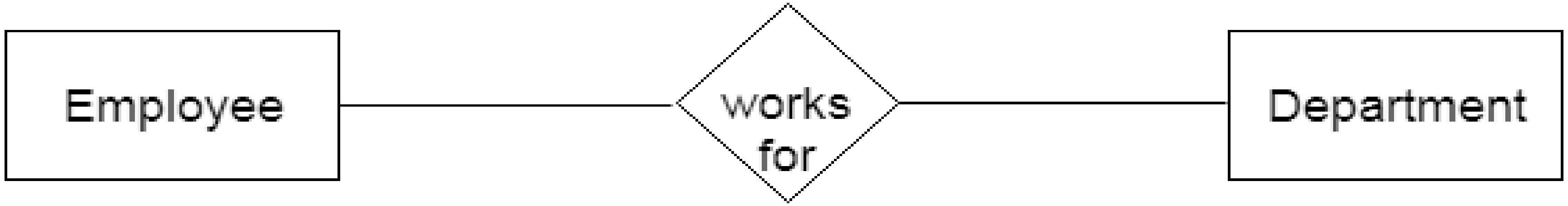
Entity
- Weak Entity

Attribute
- Key Attribute
- Composite Attribute
- Multivalued Attribute
- Derived Attribute

Relation
- One to one
- One to many
- Many to one
- Many to many

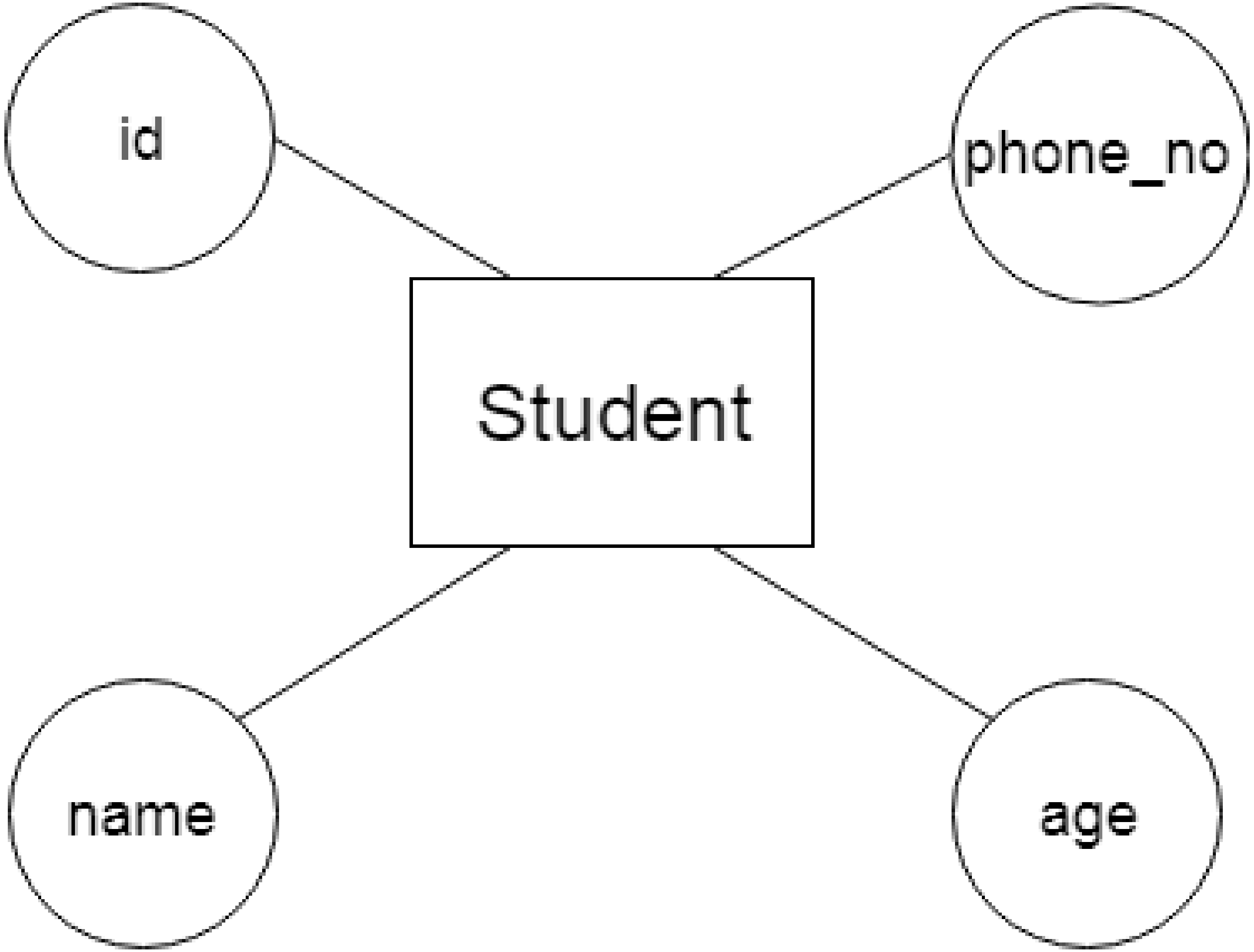
ENTITY

- An entity may be any object, class, person or place. In the ER diagram, an entity can be represented as rectangles.
- Consider an organization as an example- manager, product, employee, department etc. can be taken as an entity



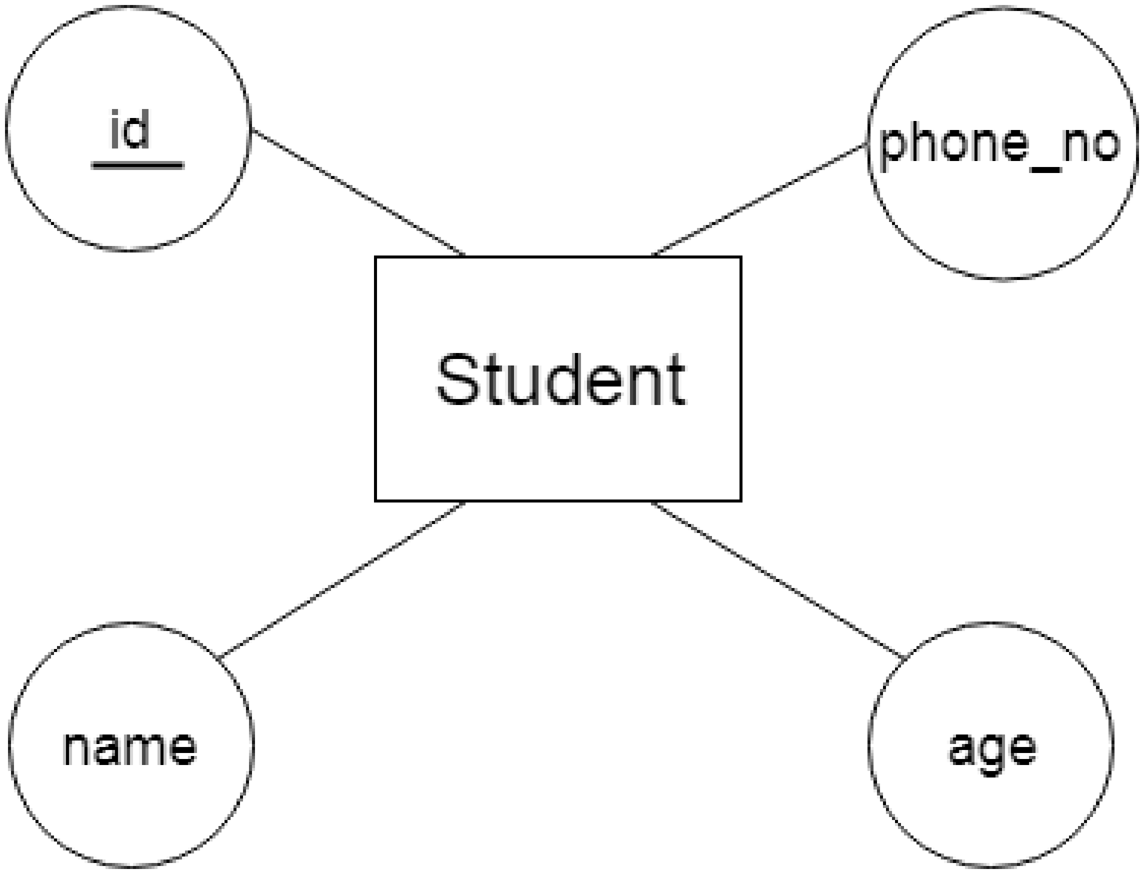
ATTRIBUTE

- The attribute is used to describe the property of an entity. Eclipse is used to represent an attribute.
- **For example**, id, age, contact number, name, etc. can be attributes of a student.



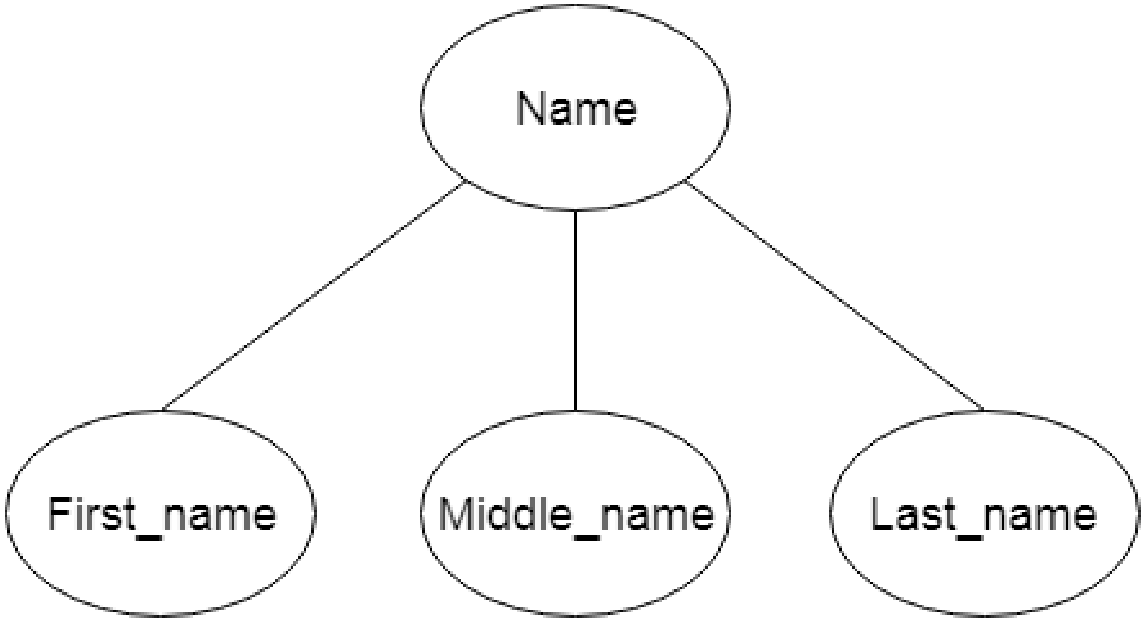
KEY ATTRIBUTE

The key attribute is used to represent the main characteristics of an entity. It represents a primary key. The key attribute is represented by an ellipse with the text underlined.



COMPOSITE ATTRIBUTE

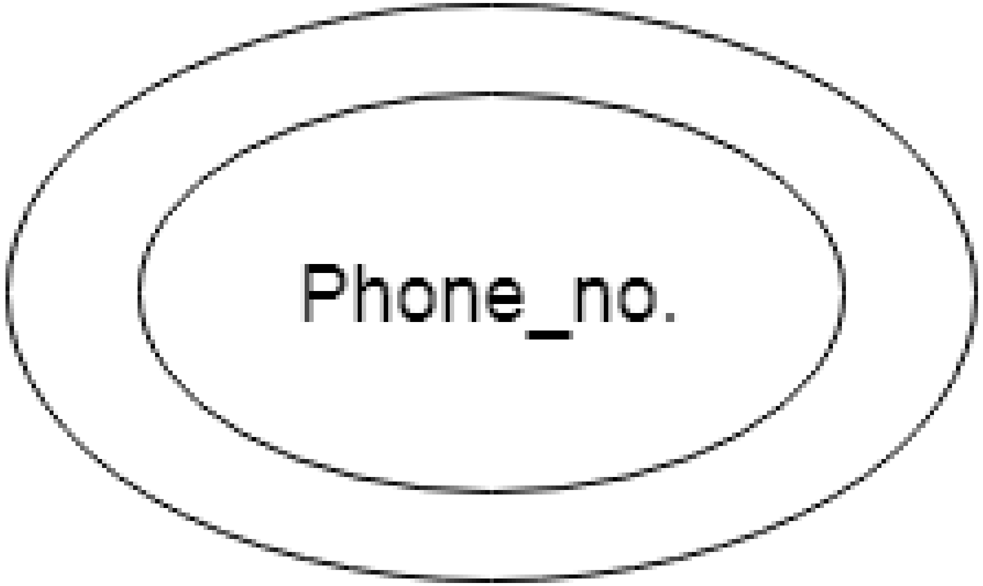
An attribute that composed of many other attributes is known as a composite attribute. The composite attribute is represented by an ellipse, and those ellipses are connected with an ellipse.



MULTIVALUED ATTRIBUTE

An attribute can have more than one value. These attributes are known as a multivalued attribute. The double oval is used to represent multivalued attribute.

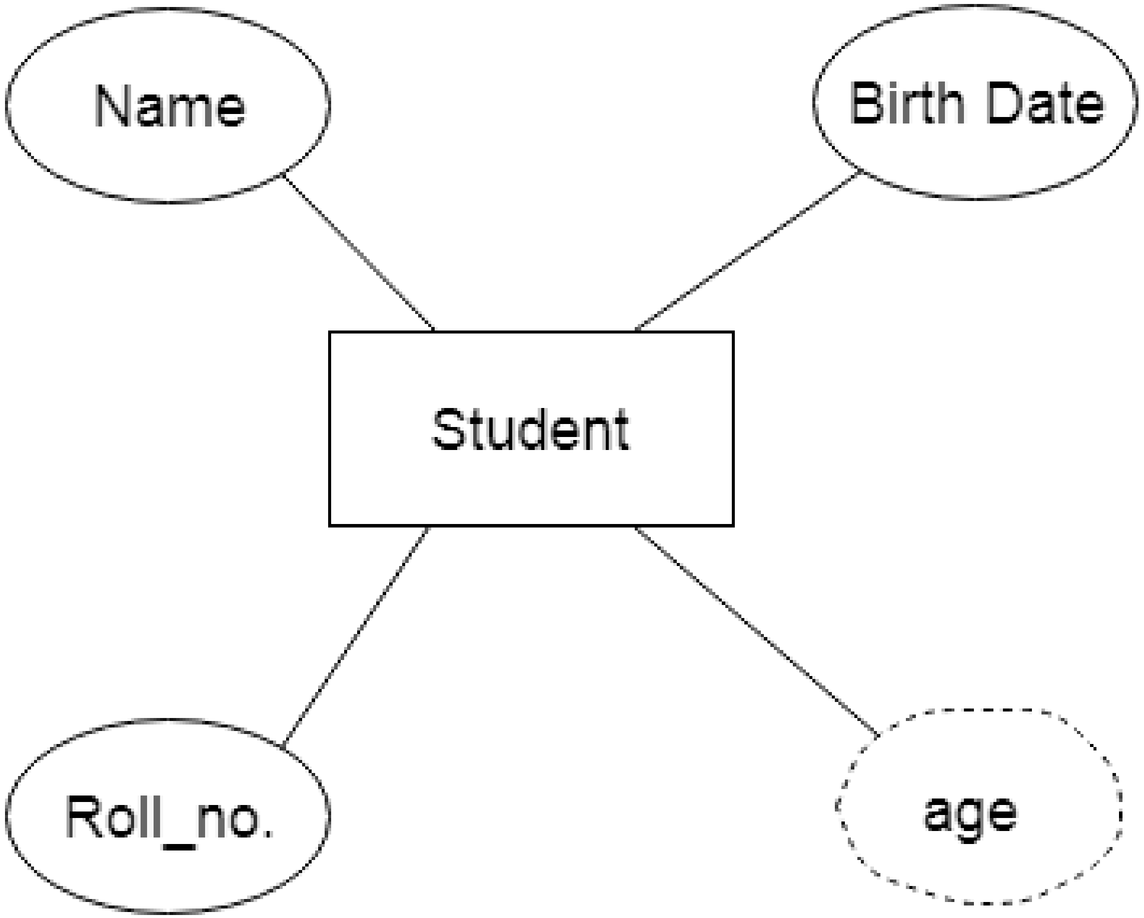
For example, a student can have more than one phone number.



DERIVED ATTRIBUTE

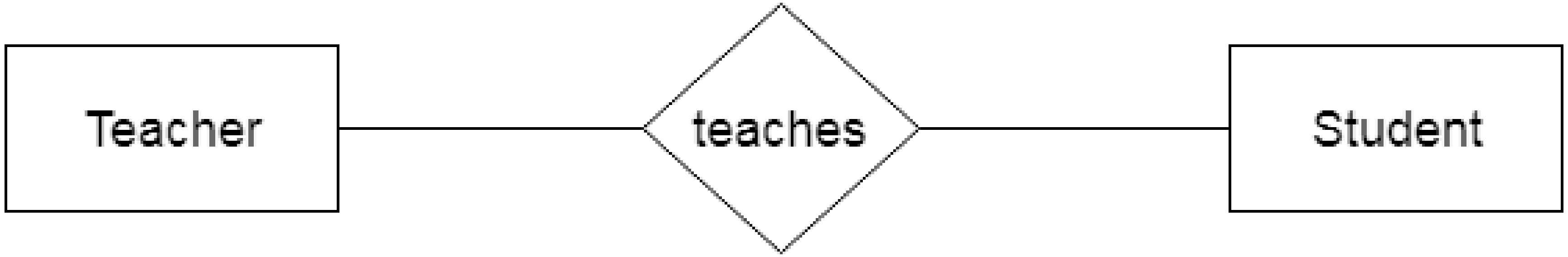
An attribute that can be derived from other attribute is known as a derived attribute. It can be represented by a dashed ellipse.

For example, A person's age changes over time and can be derived from another attribute like Date of birth.



RELATIONSHIP

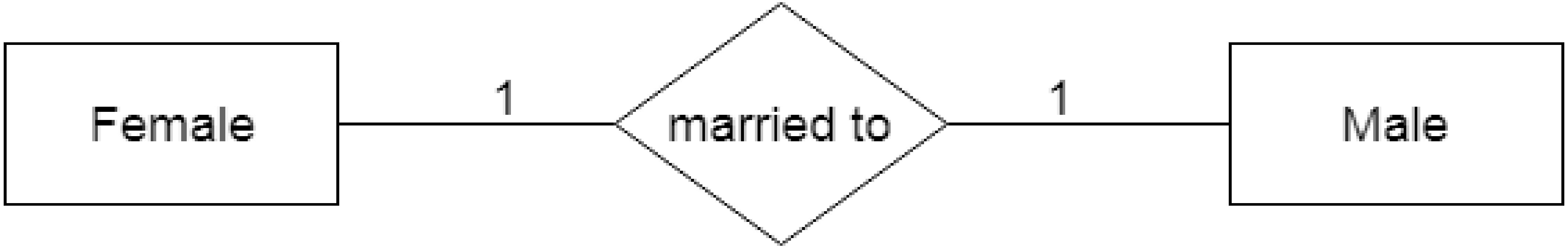
- A relationship is used to describe the relation between entities.
- Diamond or rhombus is used to represent the relationship.



Types of relationship are as follows:
1.One-to-One Relationship
2.One-to-Many Relationship
3.Many-toOne Relationship
4.Many-to-Many Relationship

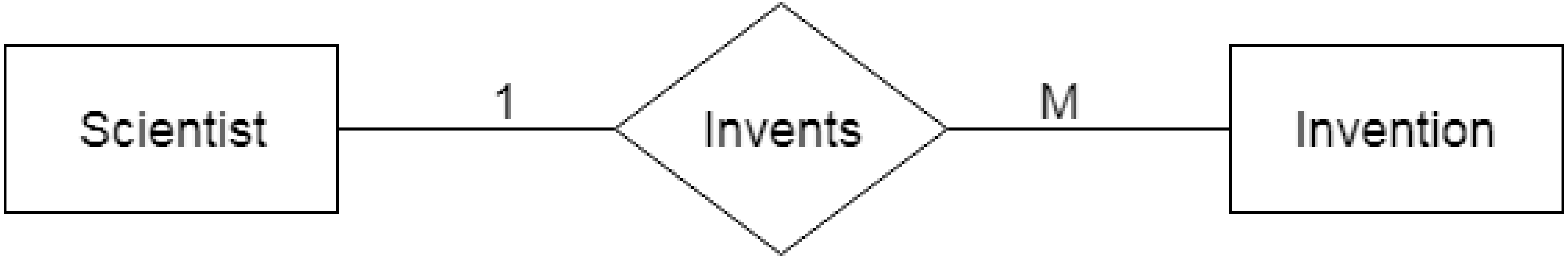
ONE TO ONE RELATIONSHIP

- When only one instance of an entity is associated with the relationship, then it is known as one to one relationship.
- **For example,** A female can marry to one male, and a male can marry to one female.



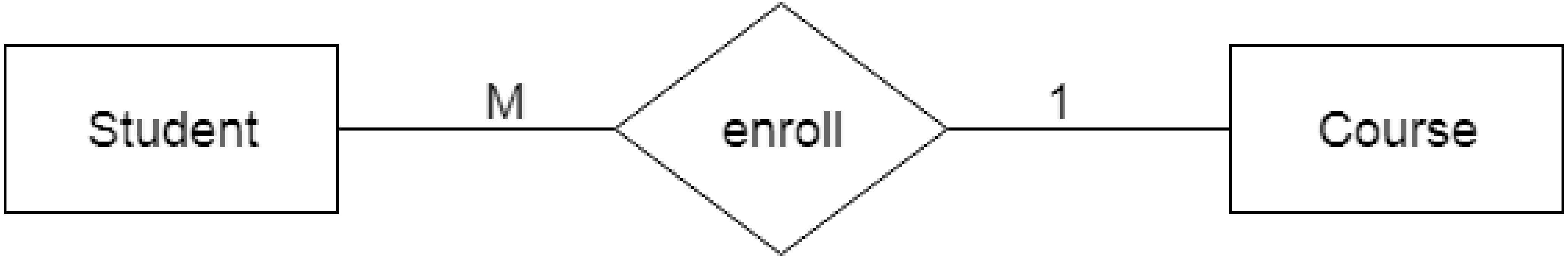
ONE TO MANY RELATIONSHIP

- When only one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then this is known as a one-to-many relationship.
- **For example,** Scientist can invent many inventions, but the invention is done by the only specific scientist.



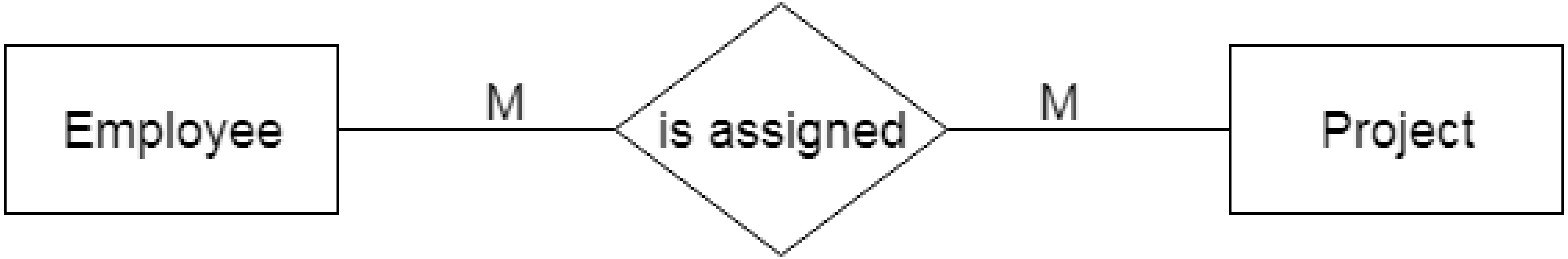
MANY TO ONE RELATIONSHIP

- ➢When more than one instance of the entity on the left, and only one instance of an entity on the right associates with the relationship then it is known as a many-to-one relationship.
- ➢**For example,** Student enrolls for only one course, but a course can have many students.



MANY TO MANY RELATIONSHIP

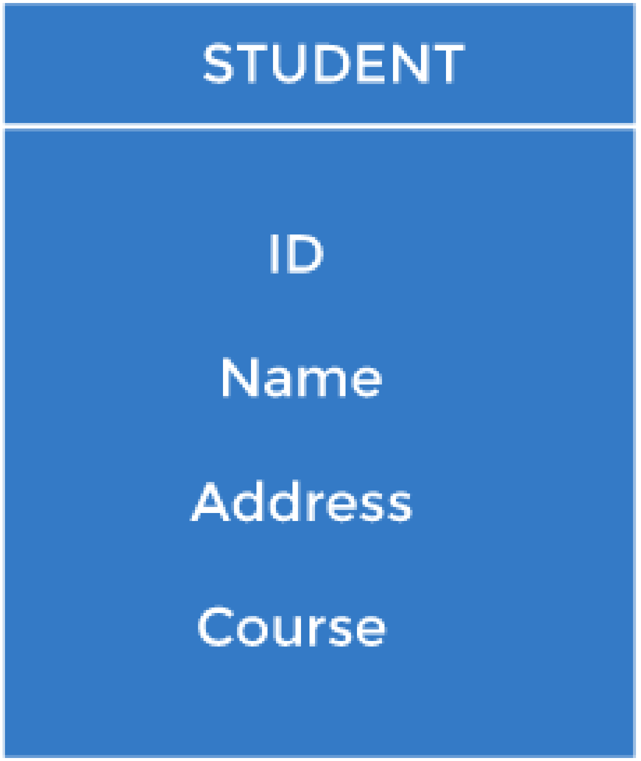
- When more than one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then it is known as a many-to-many relationship.
- **For example,** Employee can assign by many projects and project can have many employees.



KEYS IN DATA MODELING

- Keys play an important role in the relational database.
- It is used to uniquely identify any record or row of data from the table. It is also used to establish and identify relationships between tables.

- **For example,** ID is used as a key in the Student table because it is unique for each student. In the PERSON table, passport_number, license_number, SSN are keys since they are unique for each person



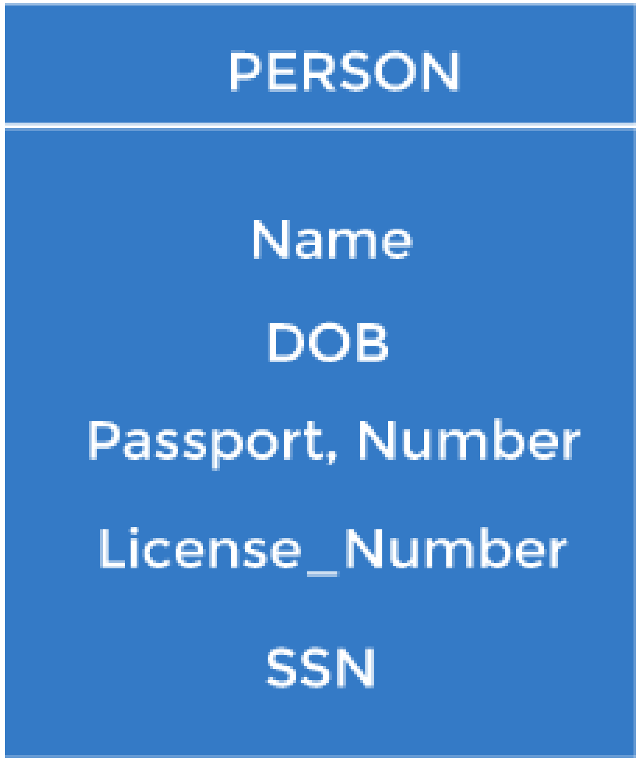
STUDENT

ID

Name

Address

Course



PERSON

Name

DOB

Passport, Number

License_Number

SSN

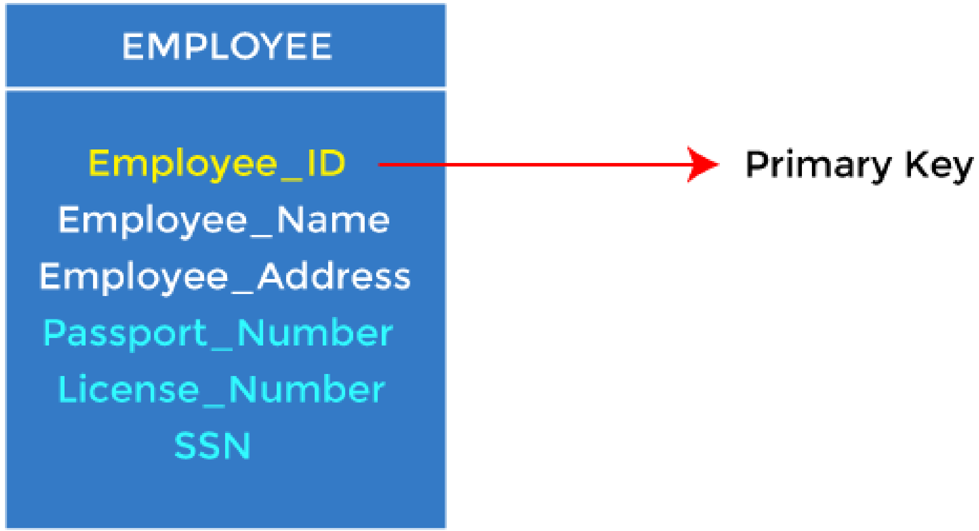
TYPES OF KEYS

1.Primary Keys:

It is the first key used to identify one and only one instance of an entity uniquely. An entity can contain multiple keys, as we saw in the PERSON table. The key which is most suitable from those lists becomes a primary key.

In the EMPLOYEE table, ID can be the primary key since it is unique for each employee. In the EMPLOYEE table, we can even select License_Number and Passport_Number as primary keys since they are also unique.

For each entity, the primary key selection is based on requirements and developers.



EMPLOYEE

Employee_ID

Employee_Name

Employee_Address

Passport_Number

License_Number

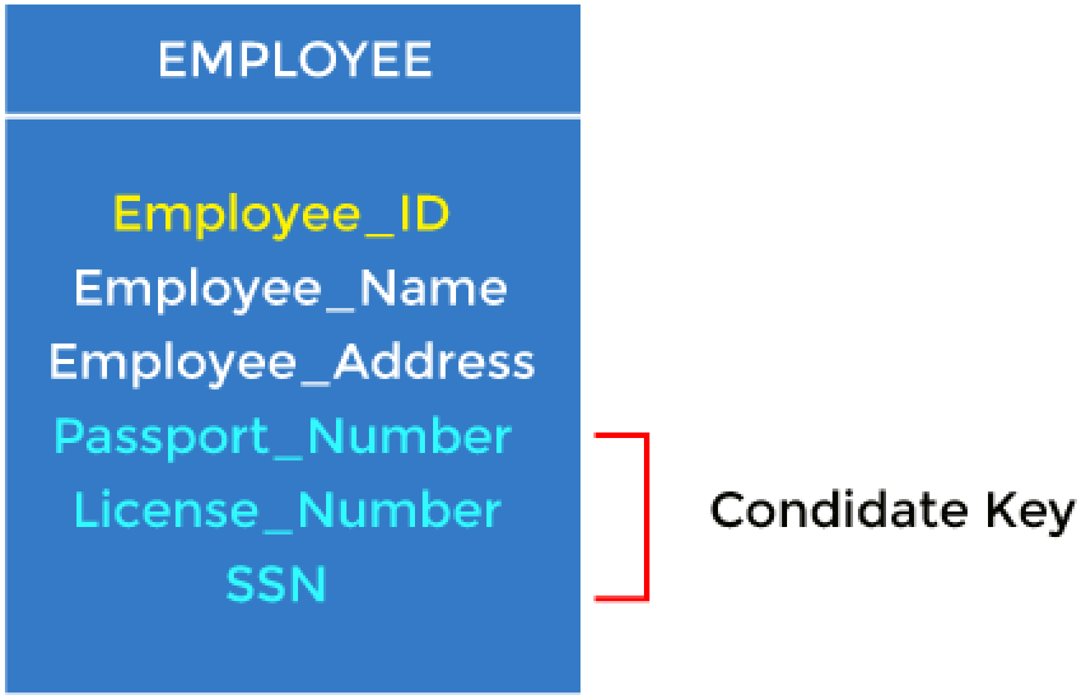
SSN

Primary Key

CANDIDATE KEY

A candidate key is an attribute or set of attributes that can uniquely identify a tuple.

Except for the primary key, the remaining attributes are considered a candidate key. The candidate keys are as strong as the primary key.
For example: In the EMPLOYEE table, id is best suited for the primary key. The rest of the attributes, like SSN, Passport_Number, License_Number, etc., are considered a candidate key.a



EMPLOYEE

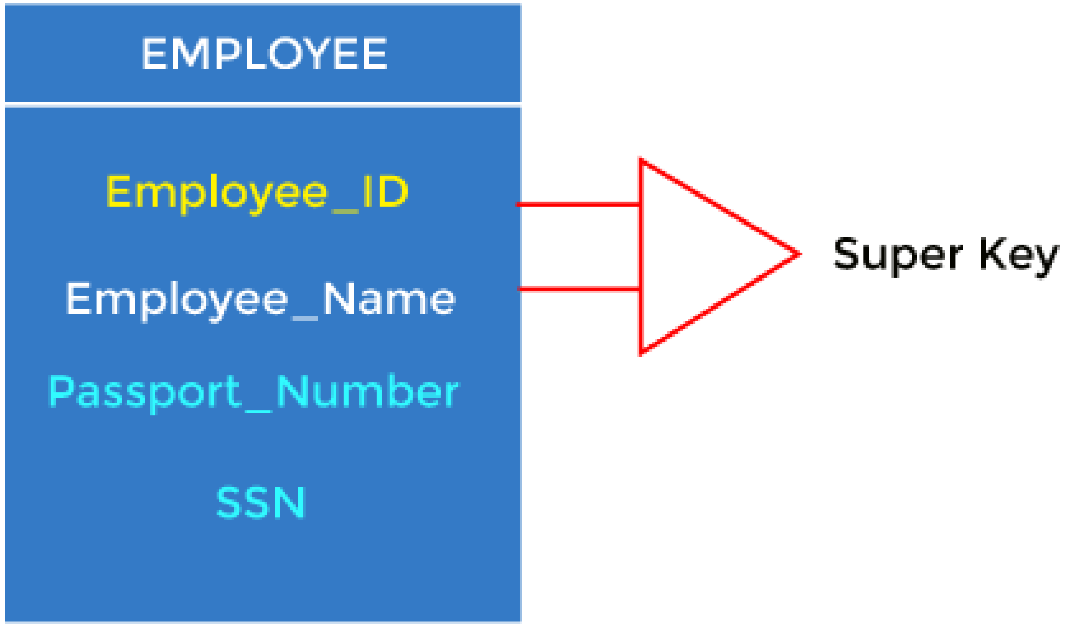
Employee_ID
Employee_Name
Employee_Address
Passport_Number
License_Number
SSN

Condidate Key

SUPER KEY

Super key is an attribute set that can uniquely identify a tuple. A super key is a superset of a candidate key.
For example: In the above EMPLOYEE table, for(EMPLOEE_ID, EMPLOYEE_NAME), the name of two employees can be the same, but their EMPLYEE_ID can't be the same. Hence, this combination can also be a key.

The super key would be EMPLOYEE-ID (EMPLOYEE_ID, EMPLOYEE-NAME), etc



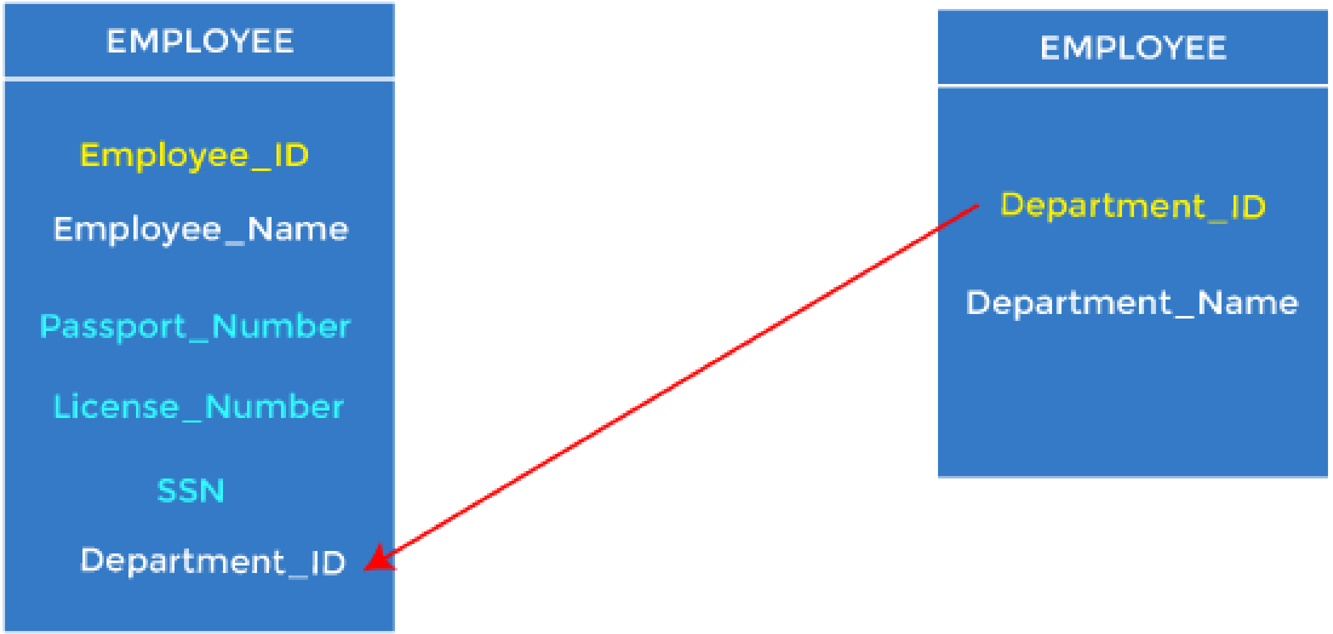
FOREIGN KEY

Foreign keys are the column of the table used to point to the primary key of another table.

Every employee works in a specific department in a company, and employee and department are two different entities. So we can't store the department's information in the employee table. That's why we link these two tables through the primary key of one table.

We add the primary key of the DEPARTMENT table, Department_Id, as a new attribute in the EMPLOYEE table.

In the EMPLOYEE table, Department_Id is the foreign key, and both the tables are related.



EXTENDED ER MODEL

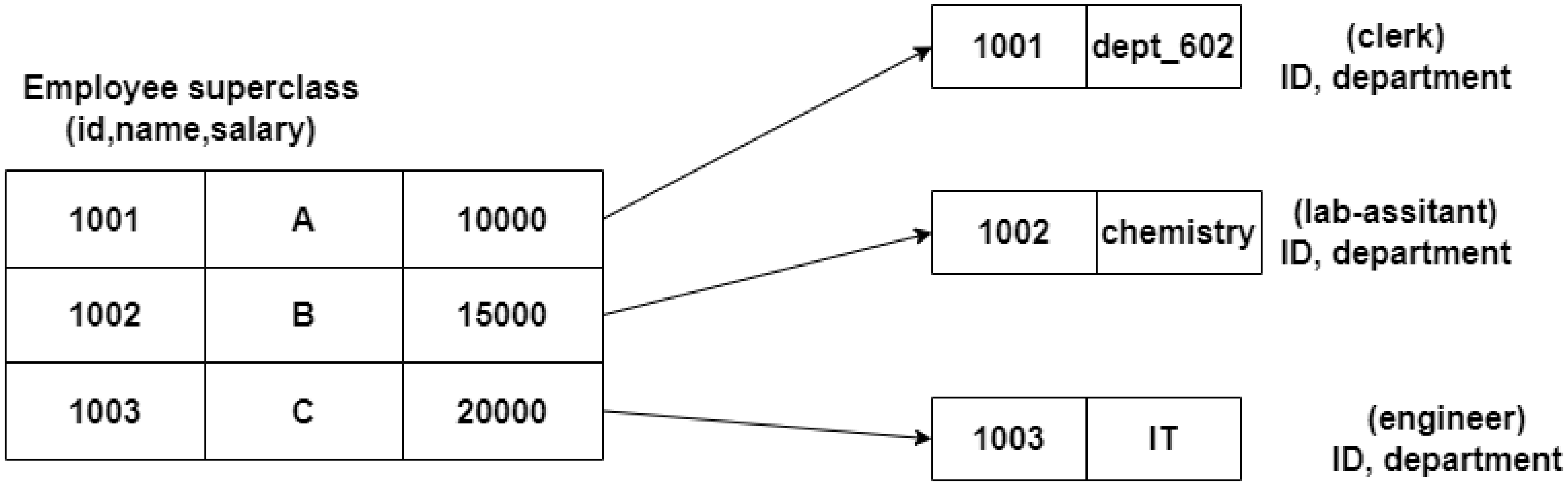
- The Extended ER (EER) Model: The Extended ER Model is an extension of the ER Model, enhancing its capabilities to represent complex relationships and additional features.

In addition to ER model concepts EE-R includes –

- Subclasses and Super classes.
- Specialization and Generalization.
- Category or union type.
- Aggregation.

ER MODEL

- For example, in a university, a faculty member or clerk is a specialized class of employees. So an employee is a generalized class, and all others are its subclass.
- We can draw the ER diagram for these relationships. Let's suppose we have a superclass Employee and subclasses as a clerk, engineer, and lab assistant.



Employee superclass
(id,name,salary)

1001	A	10000
1002	B	15000
1003	C	20000

(clerk)
ID, department

1001	dept_602

(lab-assitant)
ID, department

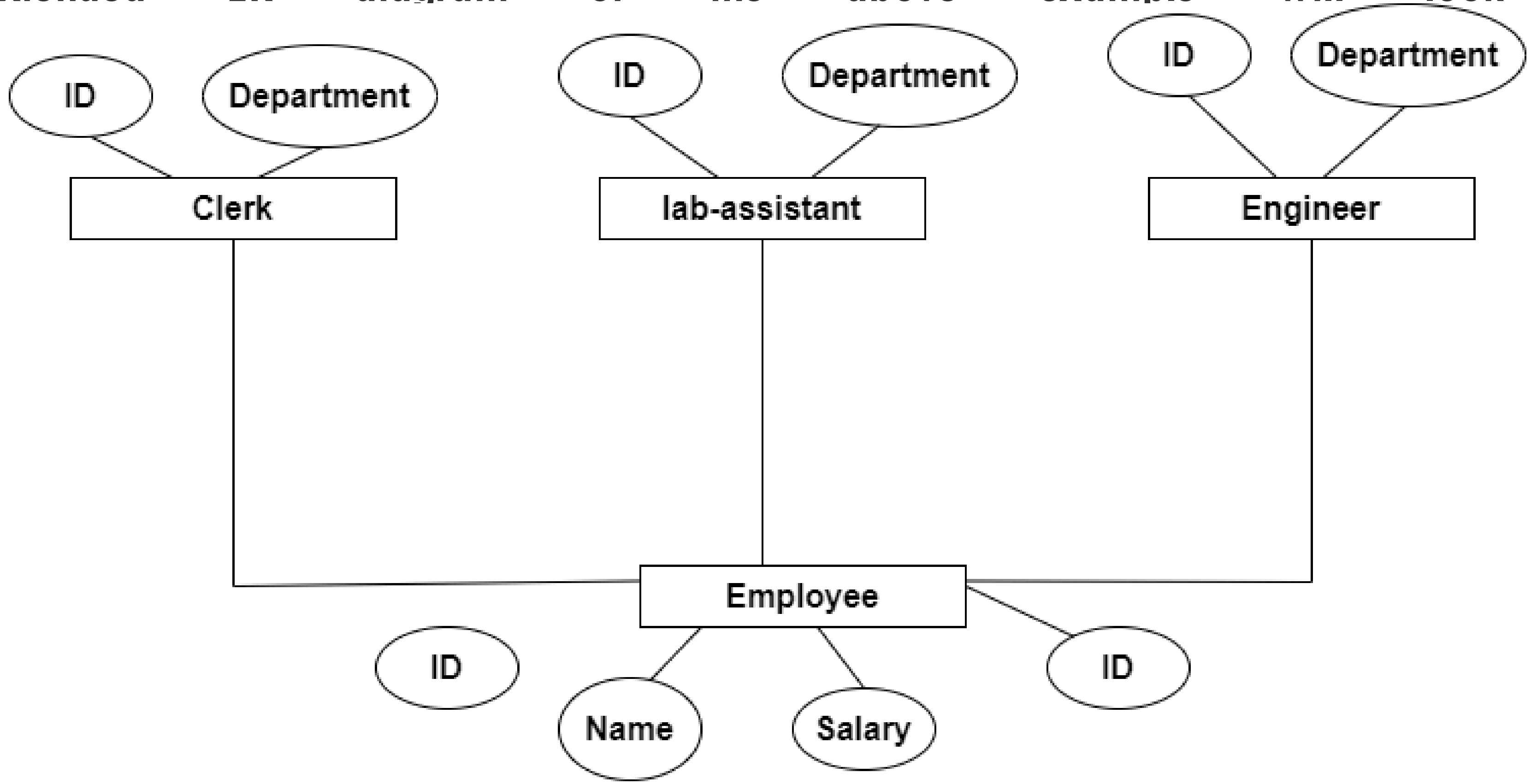
1002	chemistry

(engineer)
ID, department

1003	IT

EER MODEL

- **The Extended ER diagram of the above example will look like this:**



In the above example, we have one superclass and three subclasses. Each subclass inherits all the attributes from its superclass so that a lab assistant will have all its attributes, like its name, salary, etc.

RELATIONAL MODEL

- The relational model represents how data is stored in Relational Databases.
- A relational database consists of a collection of tables, each of which is assigned a unique name.
- Consider a relation STUDENT with attributes NAME,ROLL_NO, PHONE,ADDRESS and AGE shown in the table.

NAME	ROLL_NO	PHONE_NO	ADDRESS	AGE
Ram	14795	7305758992	Noida	24
Shyam	12839	9026288936	Delhi	35
Laxman	33289	8583287182	Gurugram	20
Mahesh	27857	7086819134	Ghaziabad	27
Ganesh	17282	9028 983988	Delhi	40

- In the given table, NAME, ROLL_NO, PHONE_NO, ADDRESS, and AGE are the attributes.
- The instance of schema STUDENT has 5 tuples

CODD'S RULES

- Every database has tables, and constraints cannot be referred to as a rational database system. And if any database has only relational data model, it cannot be a **Relational Database System (RDBMS)**. So, some rules define a database to be the correct RDBMS.
- These rules were developed by **Dr. Edgar F. Codd (E.F. Codd)** in **1985**, who has vast research knowledge on the Relational Model of database Systems.
- Codd presents his 13 rules for a database to test the concept of DBMS against his relational model, and if a database follows the rule, it is called a **true relational database (RDBMS)**.
- These 13 rules are popular in RDBMS, known as **Codd's 12 rules**

RULES

0.The Foundation Rule

1.The Information Rule

2.The Guaranteed Access Rule

3.Systematic Treatment Of NULL Values

4.Active/Dynamic Online Catalog Based On The Relational Model

5.The Comprehensive Data Sublanguage Rule

6.The View Updating Rule

7.High-Level Insert, Update & Delete Rule

8.Physical Data Independence

9.Logical Data Independence

10.Integrity Independence

11.Distribution Independence

12.The Non-subversion Rule