DATABASE CONCEPTS & ER MODEL

Author: TrungDVQ (Fsoft-Academy)

Lesson Objectives

01

The Relational Database

- ✓ Understand an overview of the basic RDBMS Concepts
- ✓ Understand an insight into the architecture and components of a Database System.
- ✓ Describe how entities, attributes and relationships are used to model data;

02

Entity-relationship Model

- ✓ Basic E-R Notation
- ✓ How to design Database

03

Tip

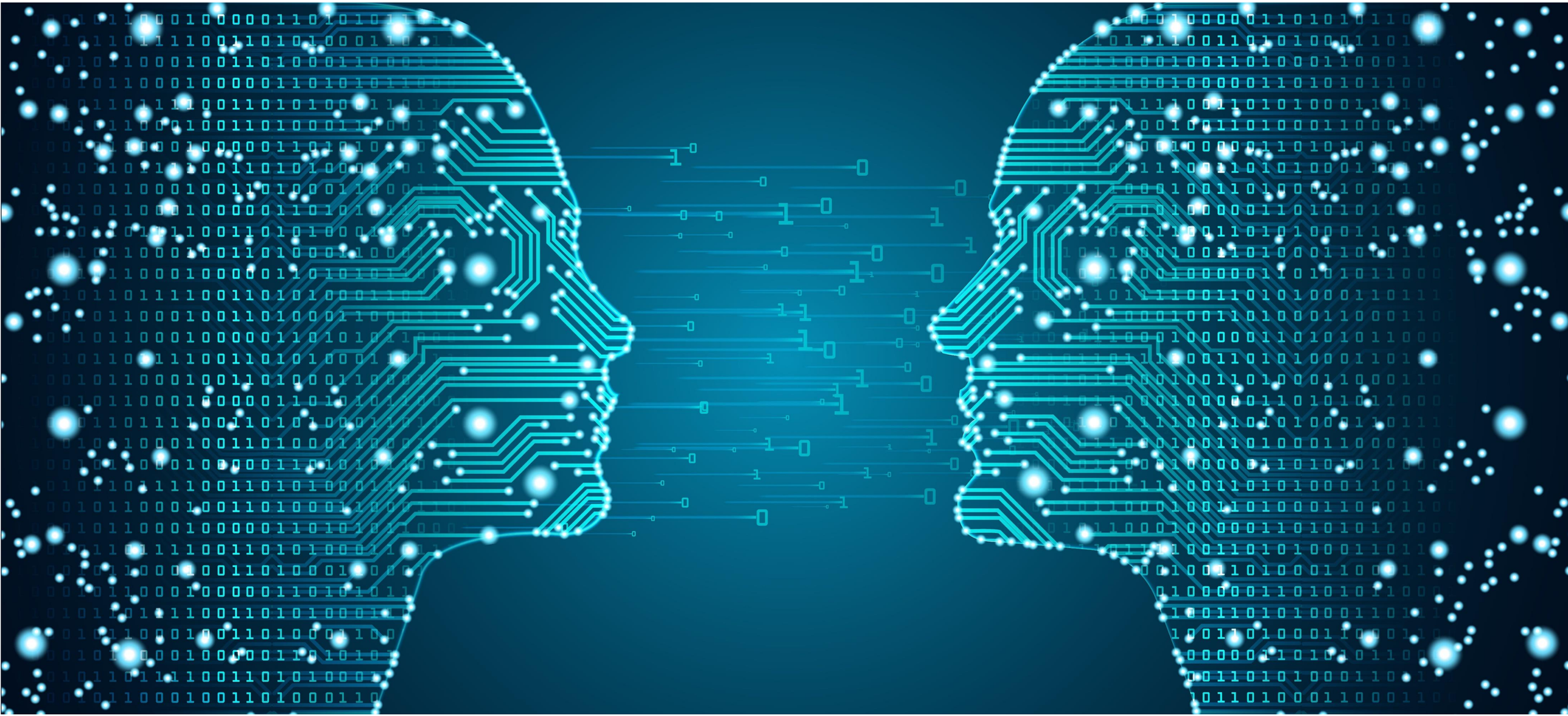
- ✓ Converting ER Model to relational schema

FPT Software | FRESHER ACADEMY

1

Introduction SQL

What is DATA?



What is a Database

A **database** is an **organized** collection of data, typically stored and accessed electronically from a computer system or electronic device.

What is a Database Management System (DBMS)?

Database Management System (DBMS) is a collection of programs which enables its users to **access database**, **manipulate data**, **reporting** and **representation** of data.
It also helps to **control access** to the database.

Types of DBMS

Hierarchical

Network DBMS

Relational DBMS

Object-Oriented Relation DBMS

Relational Database Management System (RDBMS)?

A RDBMS is a DBMS that manages data as collection of **tables** in which all data **relationships** are represented by common values in related tables

A DBMS that follows all the ***twelve rules*** of ***CODD*** is called **RDBMS**

0: Foundation Rule
1: Information Rule
2: Guaranteed Access Rule
3: Systematic Treatment of Null Values
4: Dynamic On-line Catalog Based on the Relational Model
5: Comprehensive Data Sublanguage Rule
6: View Updating Rule
7: High-level Insert, Update, and Delete
8: Physical Data Independence
9: Logical Data Independence
10: Integrity Independence
11: Distribution Independence
12: Non subversion Rule

https://www.studytonight.com/dbms/codd-rule.php

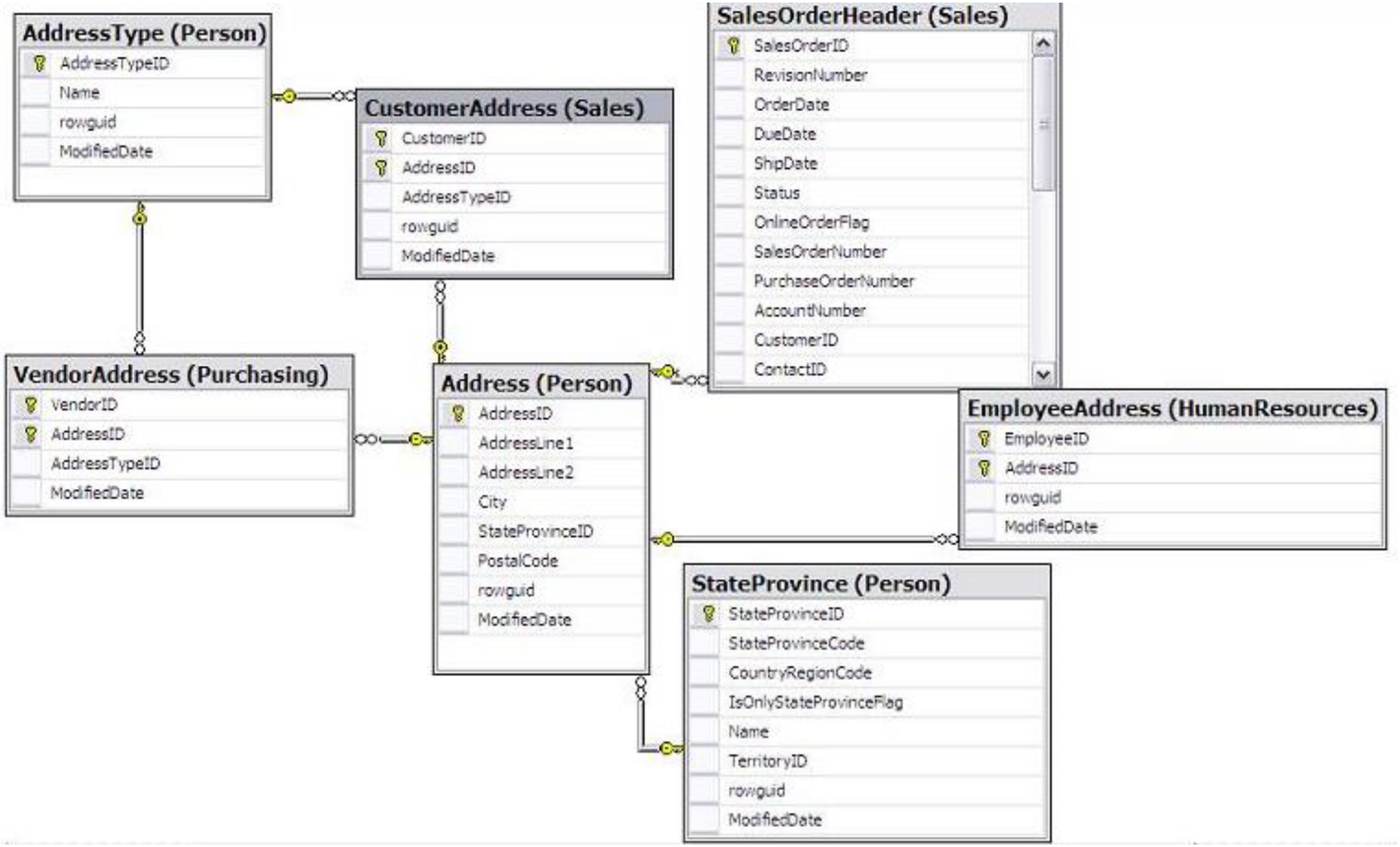
Relational Database Concepts

Table
(Think of an Excel sheet)

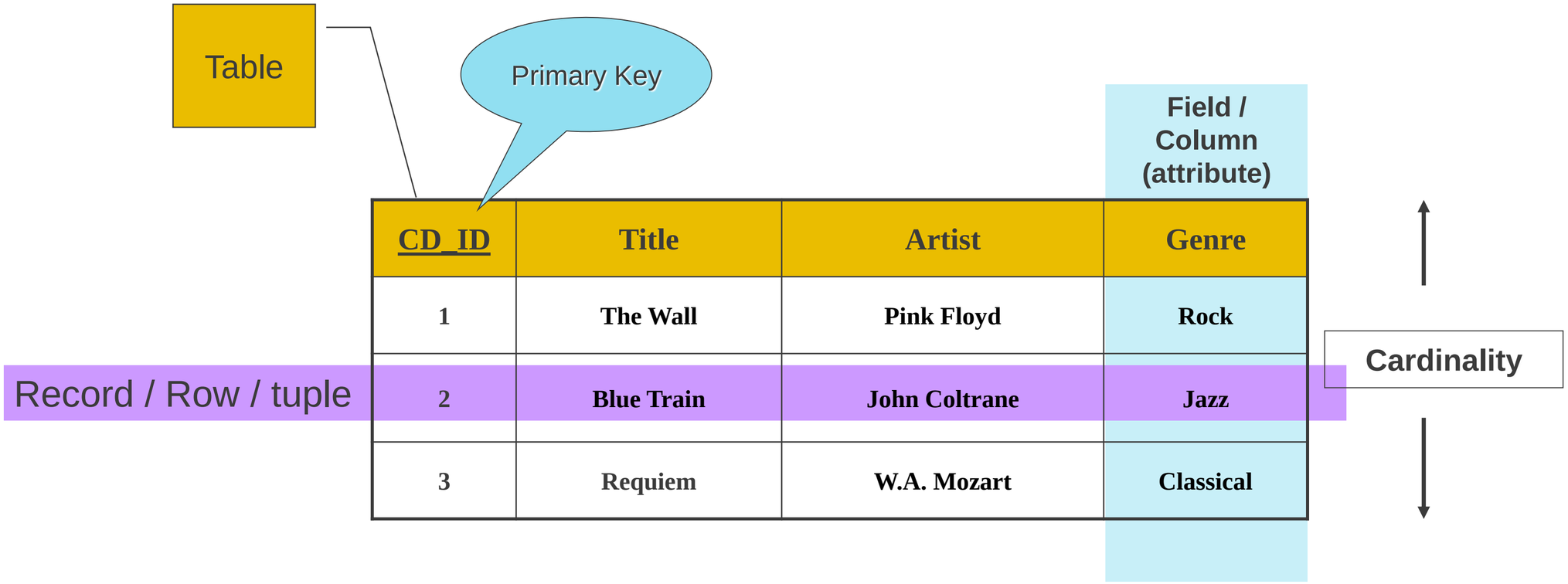
Column
(Think of a column in an Excel sheet)

Row
(Think of a row in an Excel sheet)

Relational Database



Relational Database



DBMS Schema and DBMS Instance

Definition of schema: Design of a database is called the schema.

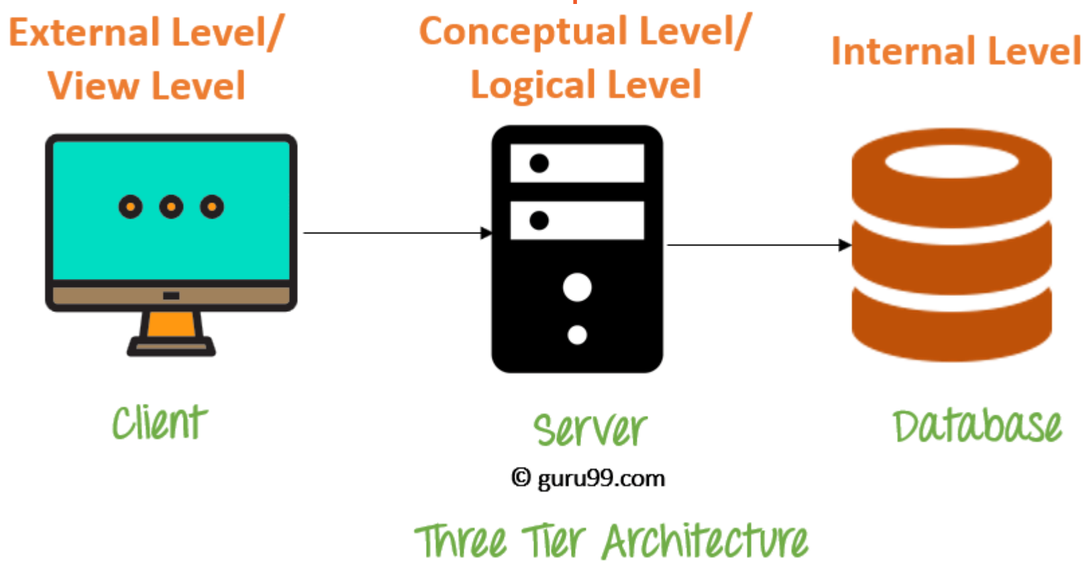
Student(studno,name,address)

Course(courseno,lecturer)

Definition of instance: The data stored in database at a particular moment of time is called instance of database.

Student(123,Bloggs,Woolton)

(321,Jones,Owens)

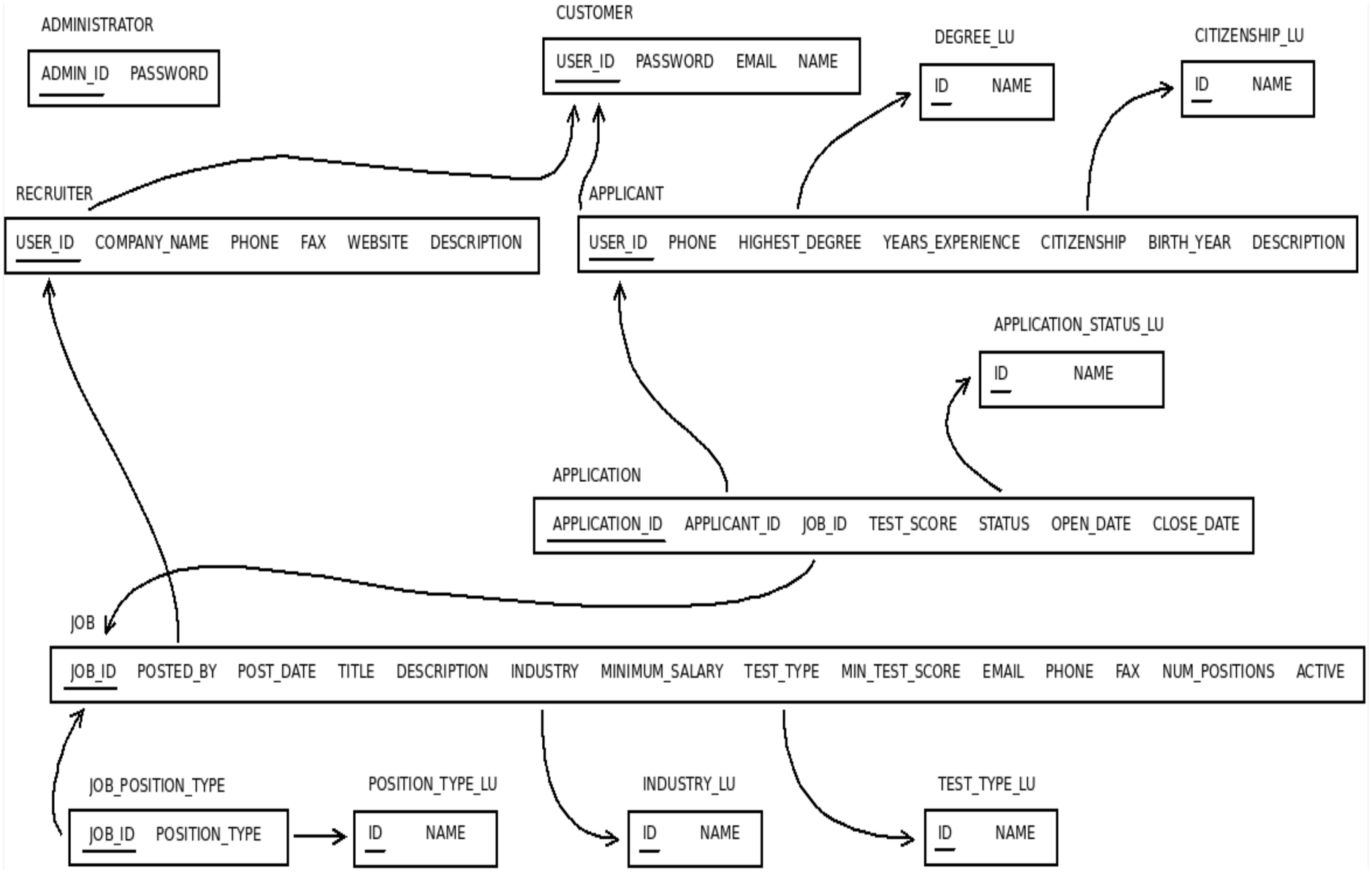


Schema is of three types: **Physical** schema, **logical** schema and **view** schema.

DBMS Schema and DBMS Instance

- The name of a relation and the set of attributes for a relation is called a **schema**.
  - **Example**: **Supplier** (SCode, SName, Quantity, City)
- **Relation schema** = name(attributes) + other structure info., e.g., keys, other constraints.
- Order of attributes is arbitrary, but in practice we need to assume the (*standard*) order given in the relation schema.
- **Relational database schema** = collection of relation schemas.

DBMS Schema and DBMS Instance



2

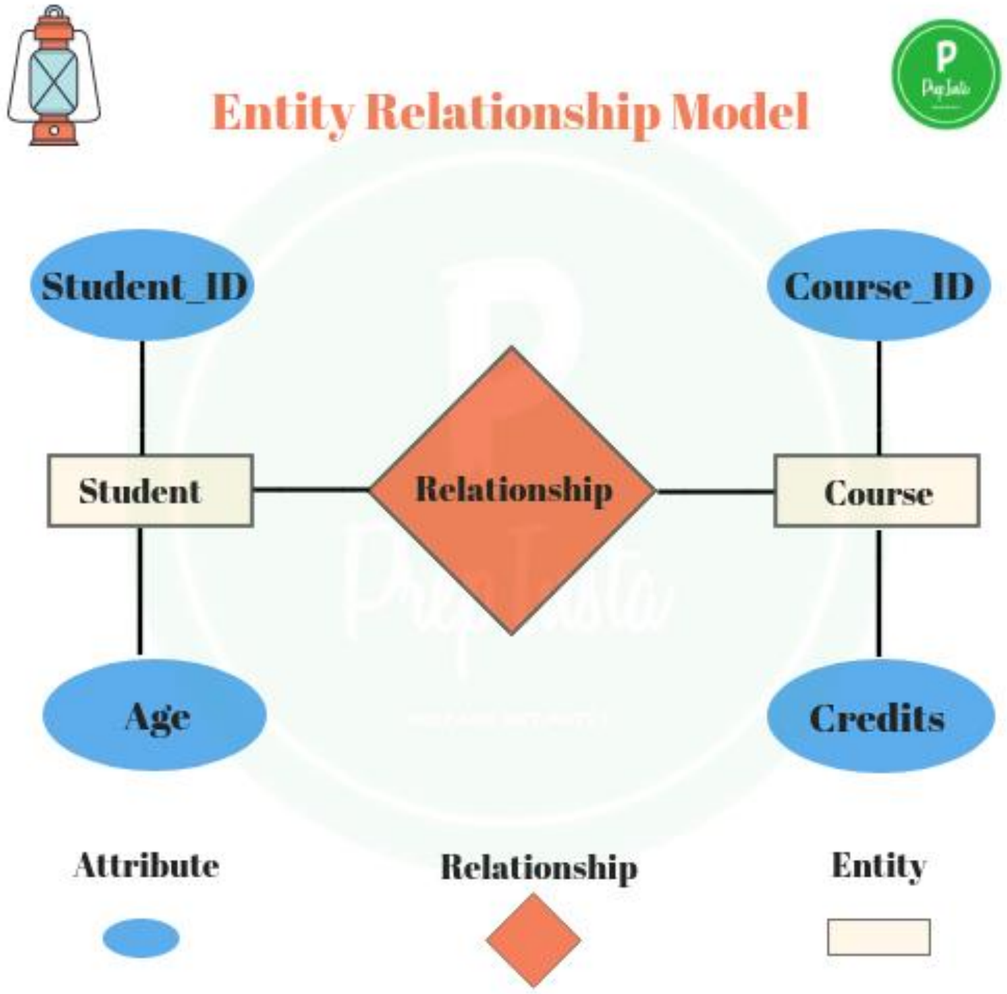
Entity-relationship Model

Entity relationship model (ER model)

Entity-relationship model is a model used for design and representation of relationships between data.

Key components of the ER model:

- Entity
- Attributes
- Relationships

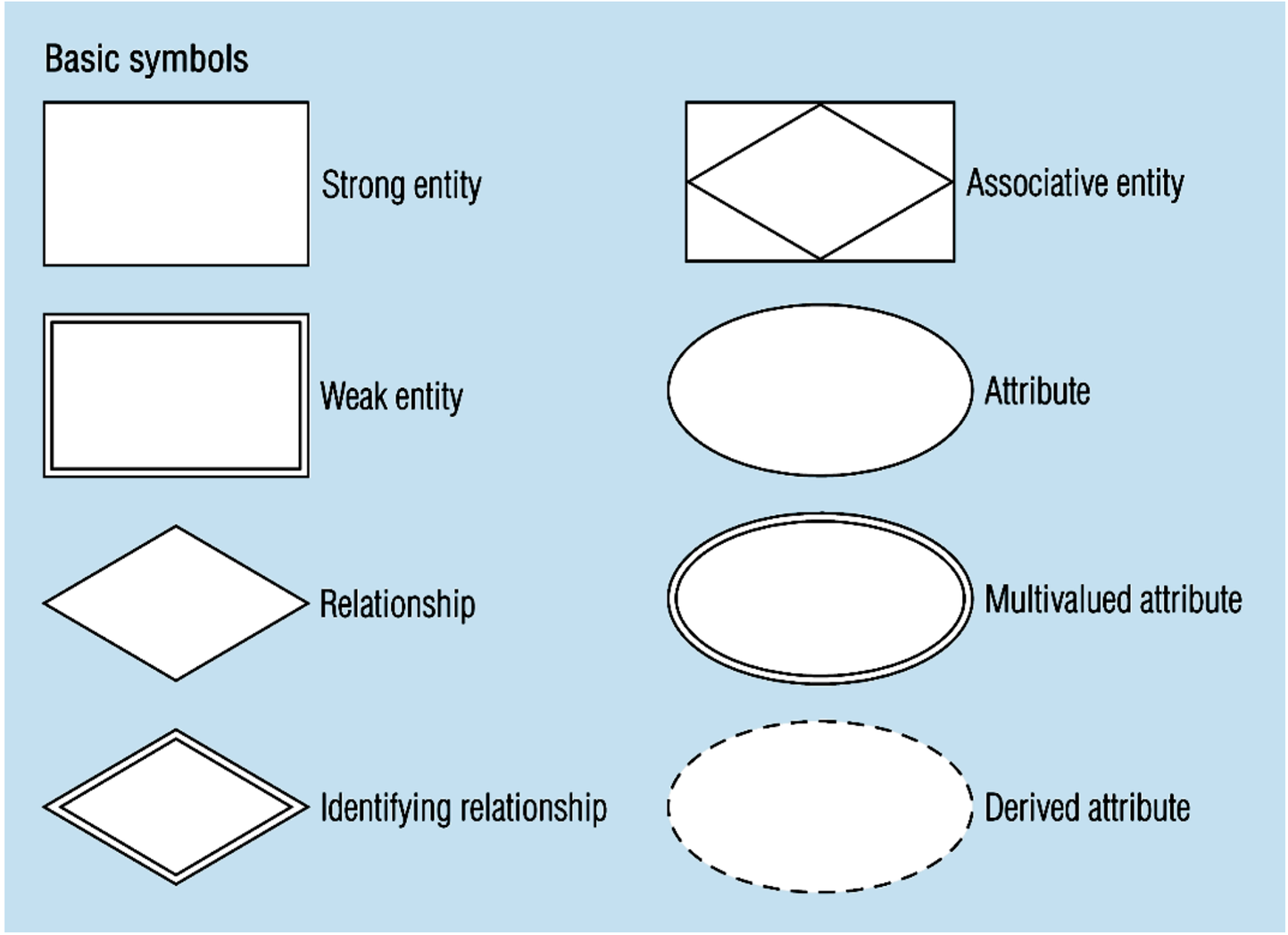


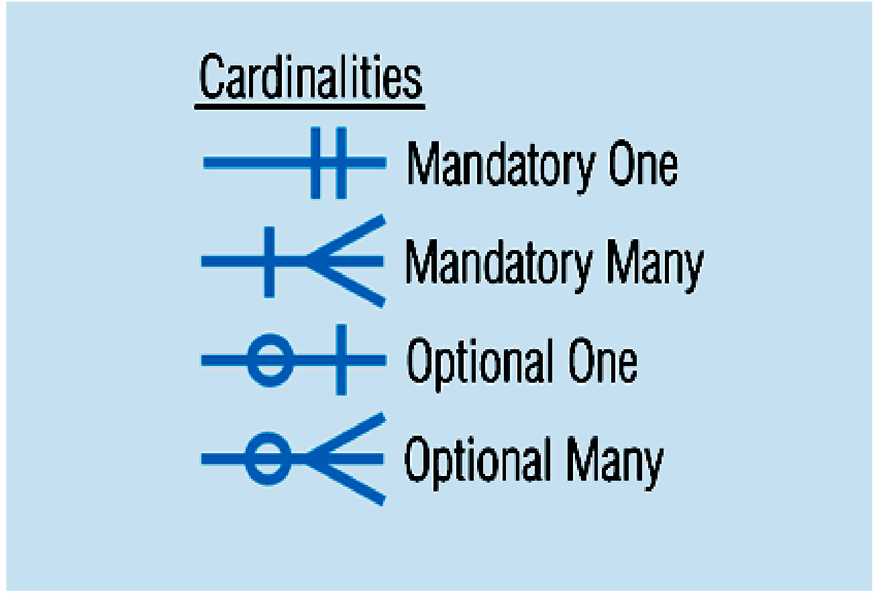
The ER model separates the information required in the business i.e it establishes a clear distinction between data that is useful and not useful for business operations

Entity relationship model (ER model)

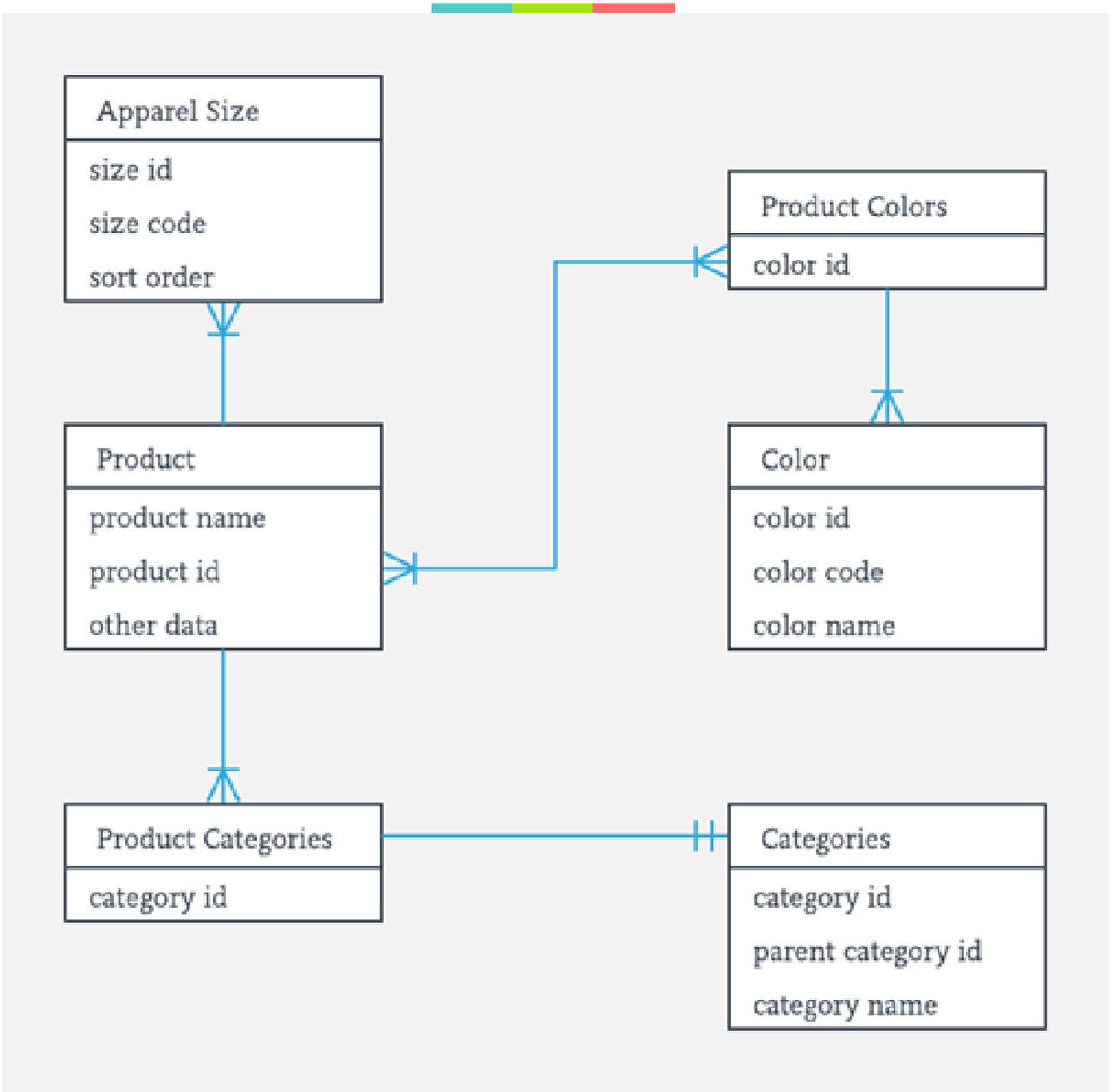
- **Entity**
  - Weak
  - strong
- **attributes**
  - Simple
  - Composite
  - Single valued
  - Multi valued
  - Key
  - Derived
- **Relationship**
  - Unary
  - Binary
  - N-ary
  - Ternary
  - Recursive
- **Cardinality**
  - One-to-One
  - One-to-Many
  - Many-to-one
  - Many-to-Many

Basic E-R Notation





Basic E-R Notation



Entity

Entity Name

Entity

Person,place,object,event or concept about which data is to be maintained

Example: Car, Student

Jack

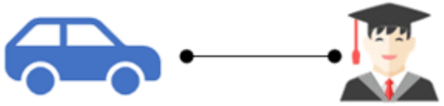
Attribute Name

Attribute

Property or characteristic of an entity

Example: Color of car Entity

Name of Student Entity



Relation

Verb Phrase

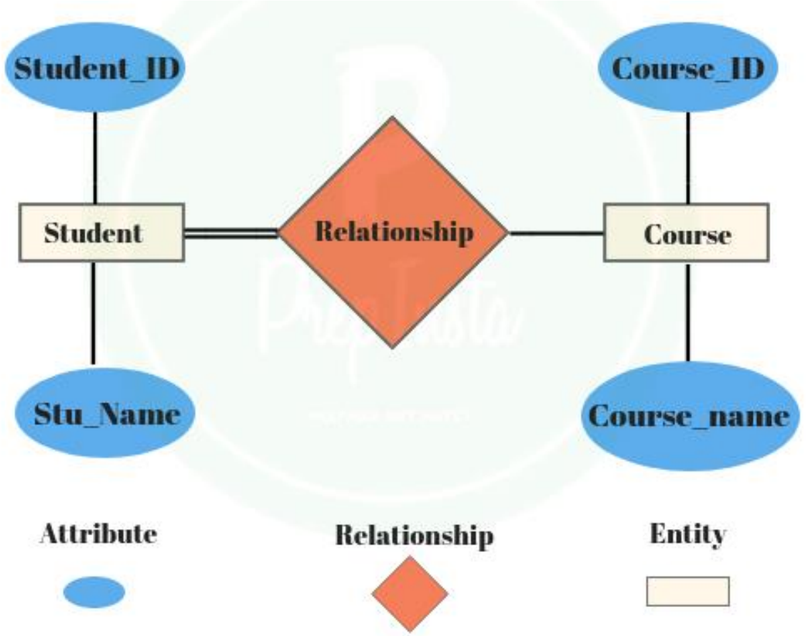
Association between the instances of one or more entity types

Example: Blue Car Belongs to Student Jack

Strong and Weak entity

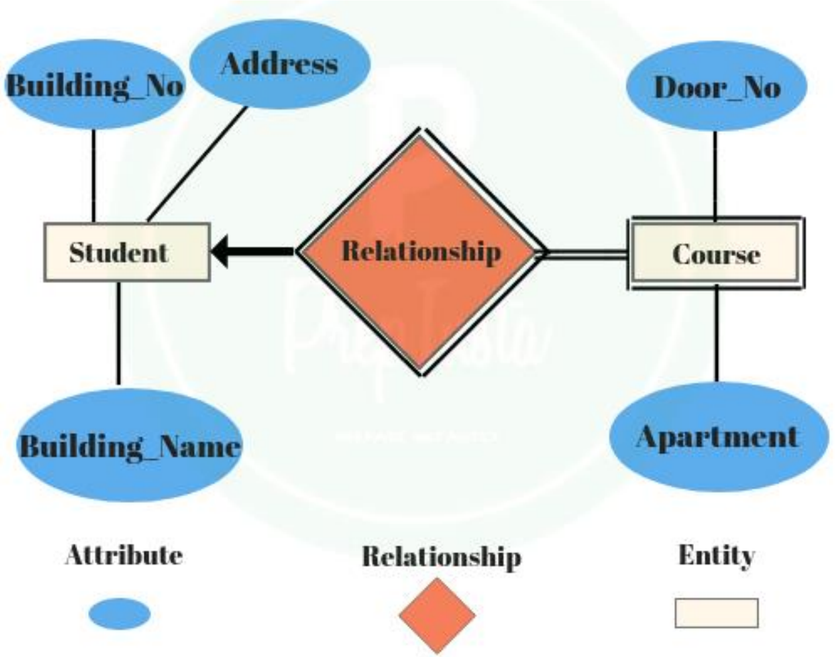
Strong entity

1. Strong entity always ***has a primary key***
2. It is ***not dependent*** on any other entity
3. Represented by a ***single rectangle***
4. Relationship between two strong entities is represented by a ***single diamond***
5. A strong entity has ***may or may not have total participation***



Weak entity

1. Will ***not have a primary key but it has partial discriminator key***
2. Which entity is ***dependent*** on the strong entity
3. Represented by ***double rectangle*** relationship between a strong entity and the weak entity is represented by ***double Diamond***
4. It has always ***total participation***

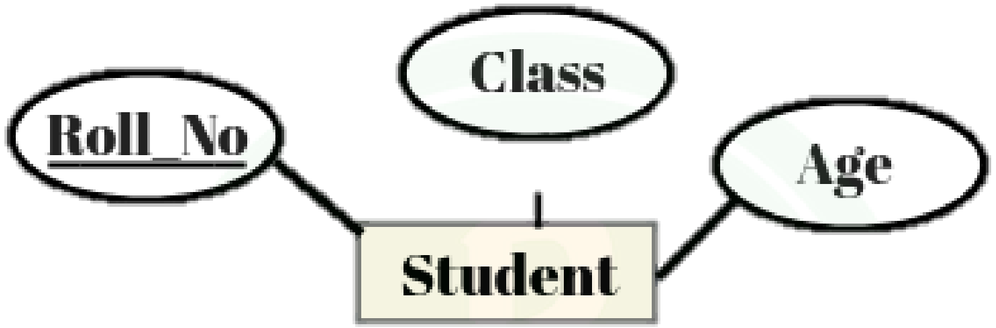


attributes

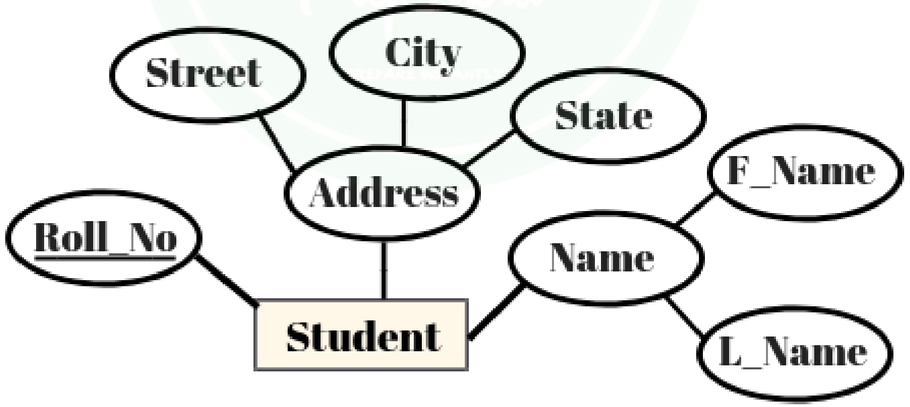
- Simple
- Composite
- Single valued
- Multi valued
- Key
- Derived

Attributes

Simple attributes are those **attributes which can not be divided further.**



Composite attributes are those **attributes which are composed of many other simple attributes.**



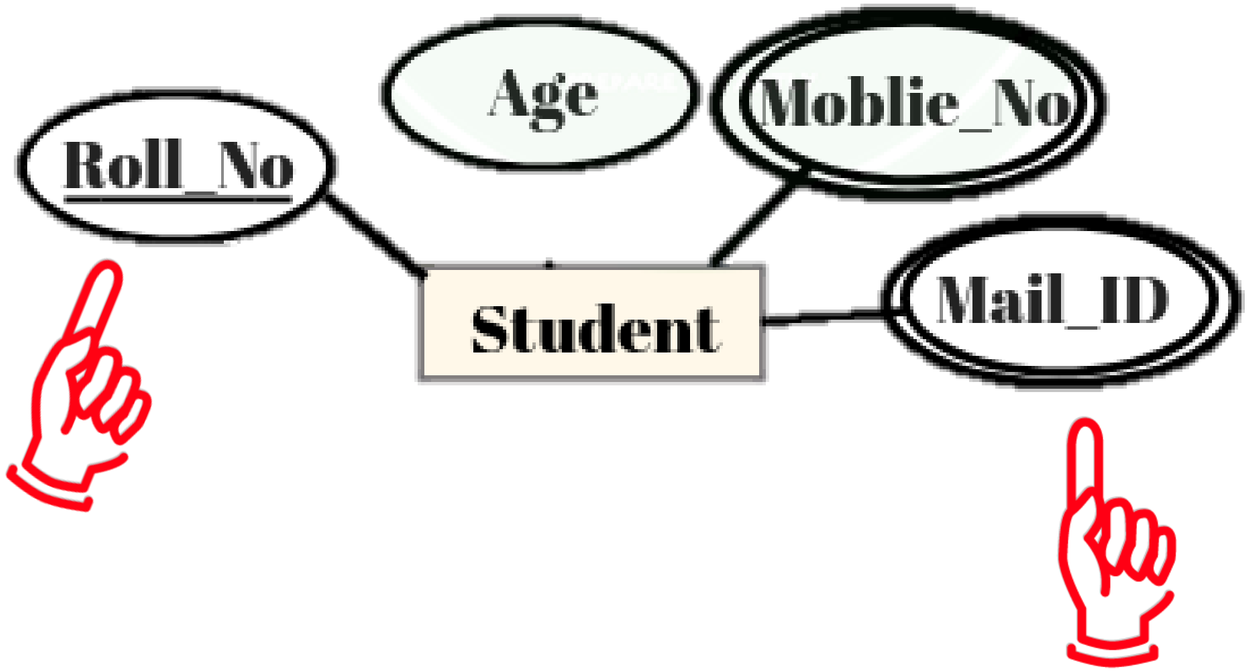
attributes

- Simple
- Composite
- Single valued
- Multi valued
- Key
- Derived

Attributes

Single valued attributes are those **attributes which can take only one value for a given entity from an entity set.**

Multi valued attributes are those **attributes which can take more than one value for a given entity from an entity set.**

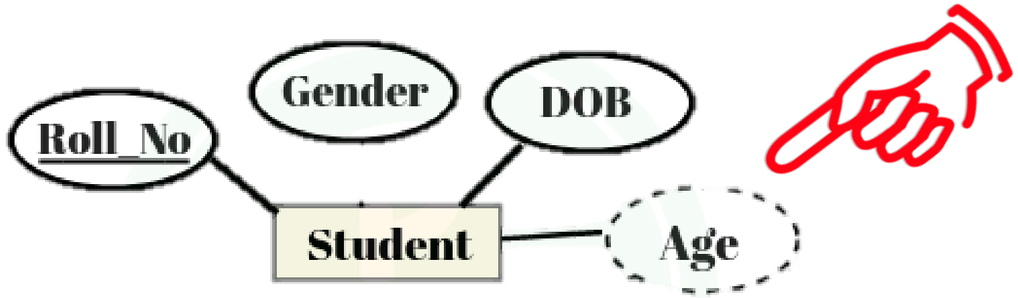


attributes

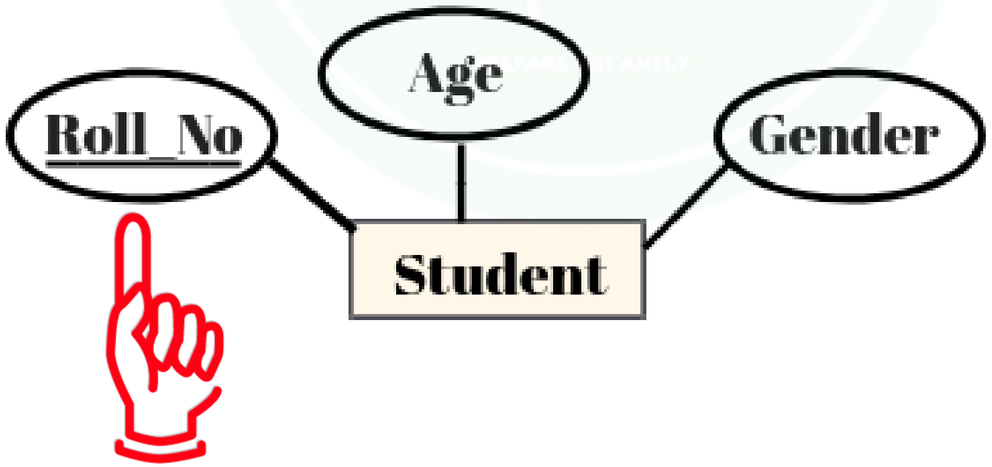
- Simple
- Composite
- Single valued
- Multi valued
- Key
- Derived

Attributes

Derived attributes are those **attributes which can be derived from other attribute(s).**



Key attributes are those **attributes which can identify an entity uniquely in an entity set.**

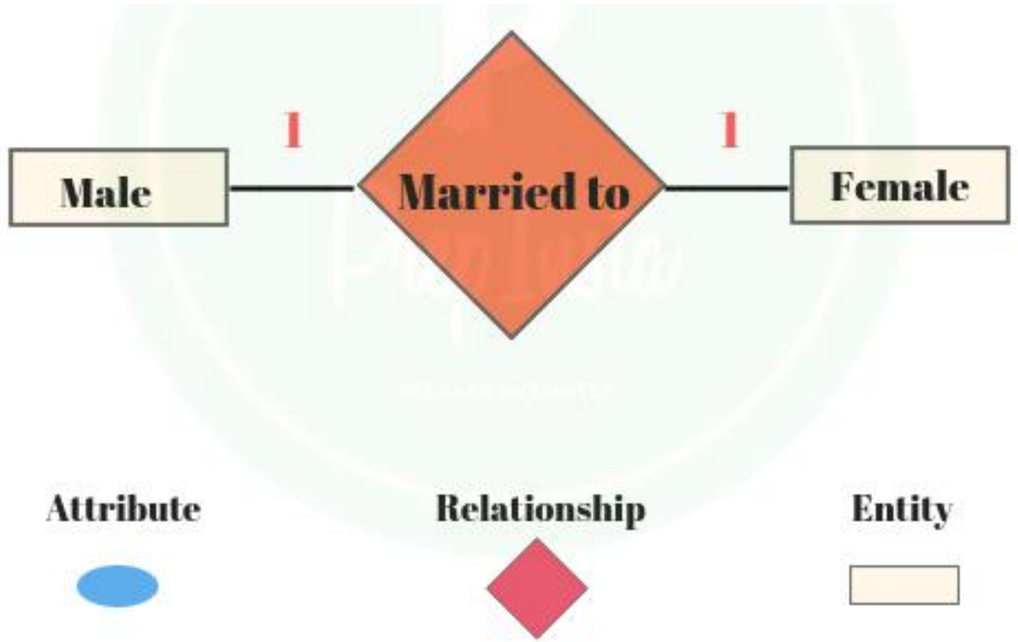


Cardinality Notation

Cardinality

- One-to-One
- One-to-Many
- Many-to-one
- Many-to-Many

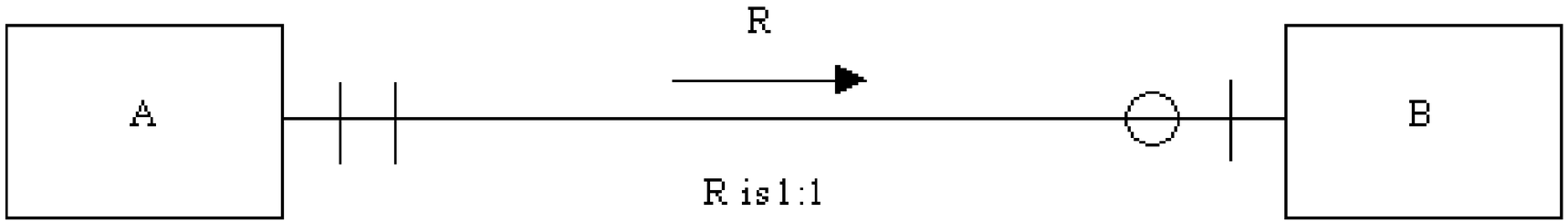
- When a single instance of an entity is associated with a single instance of another entity, then it is called as ***one to one cardinality***
- Here each entity of the entity set participate only once in the relationship



Cardinality Notation

Cardinality

- One-to-One
- One-to-Many
- Many-to-one
- Many-to-Many

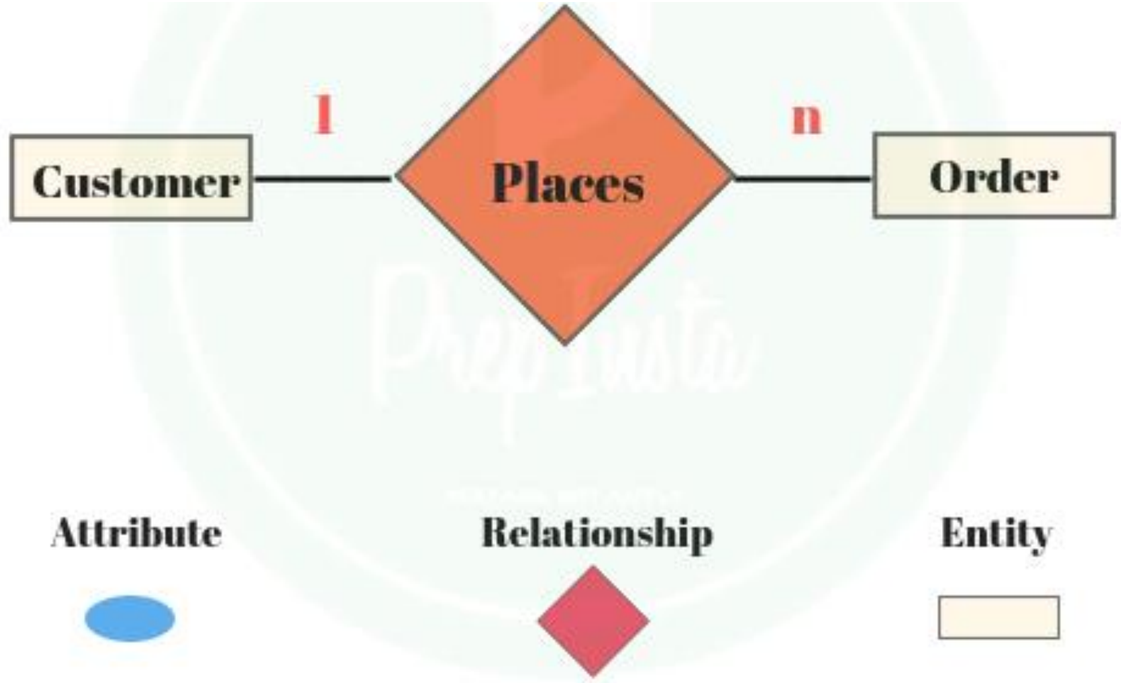


Cardinality Notation

Cardinality

- One-to-One
- One-to-Many
- Many-to-one
- Many-to-Many

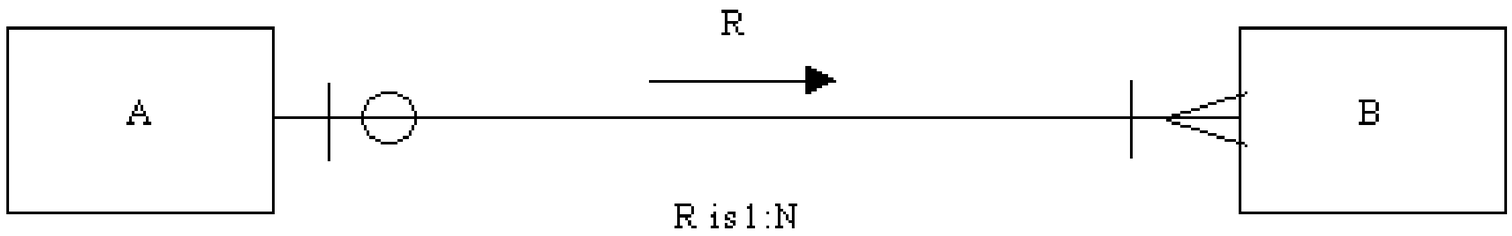
- When is a single instance of an entity is associated with more than one instance of another entity then this type of relationship is called one to many relationships
- Here entities in one entity set can take participation in any number of times in relationships set and entities in another entity set can take participation only once in a relationship set



Cardinality Notation

Cardinality

- One-to-One
- One-to-Many
- Many-to-one
- Many-to-Many

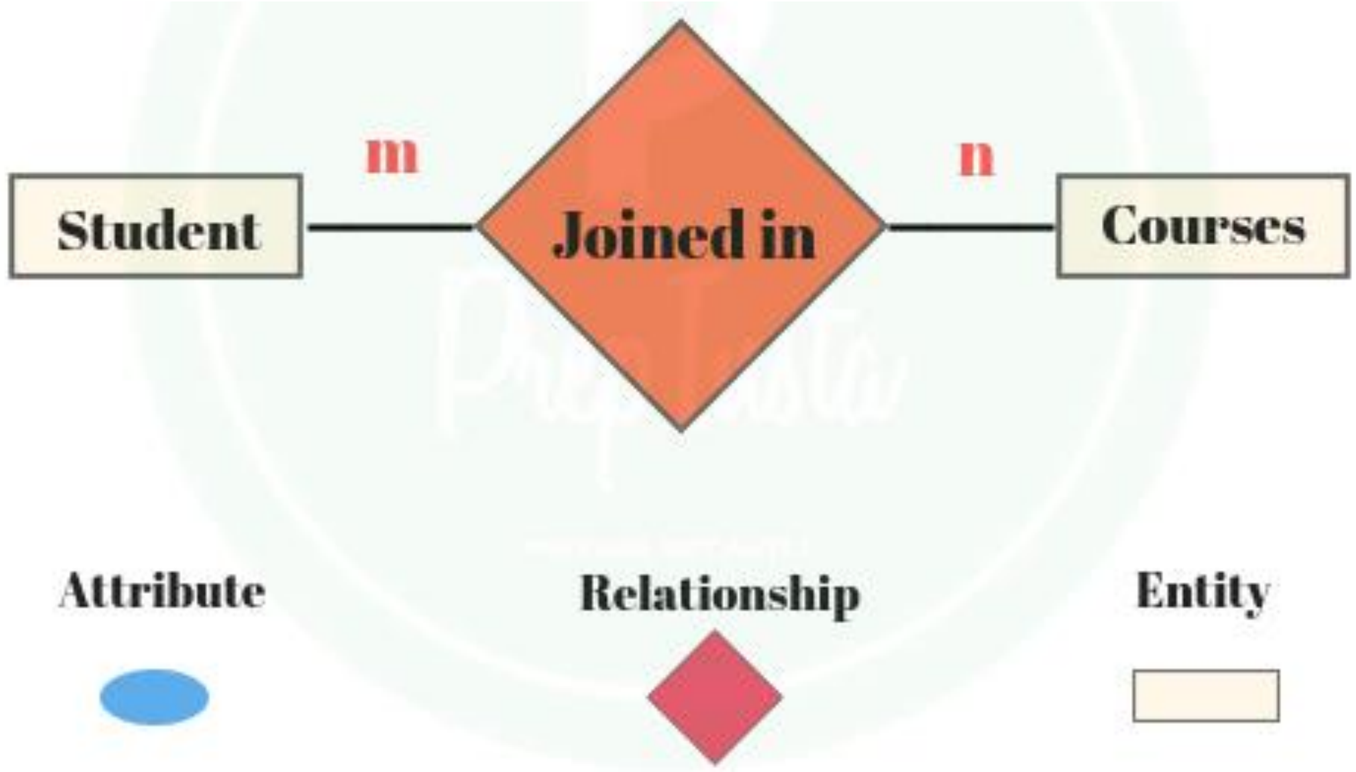


Cardinality Notation

Cardinality

- One-to-One
- One-to-Many
- Many-to-one
- Many-to-Many

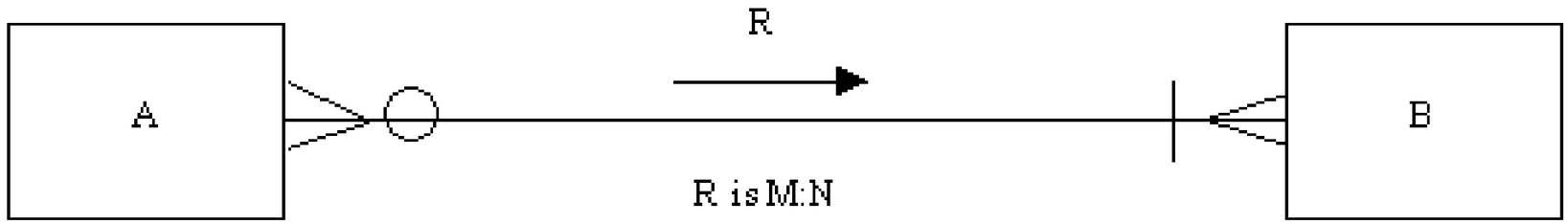
When entities in one entity set can participate only once in a relationship set and entities in another entity can participate more than once in the relationship set, then such type of cardinality is called many-to-one



Cardinality Notation

Cardinality

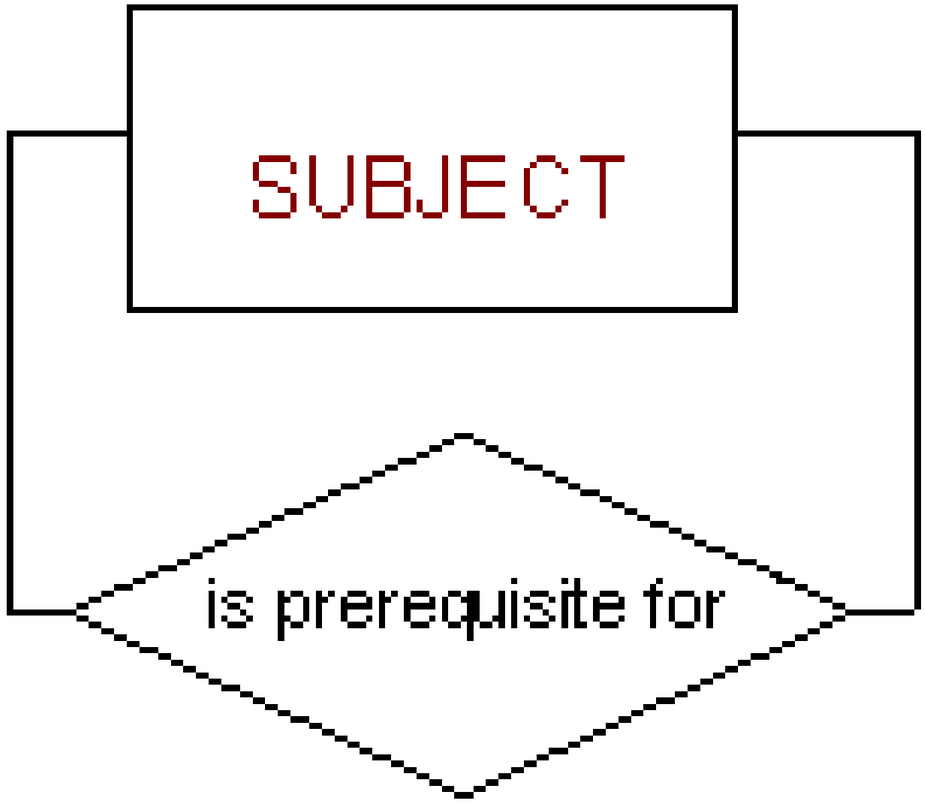
- One-to-One
- One-to-Many
- Many-to-one
- Many-to-Many



Relationship

- Relationship
  - Unary
  - Recursive
  - Binary
  - Ternary
  - N-ary

A **unary relationship** is when both participants in the relationship are the same entity.

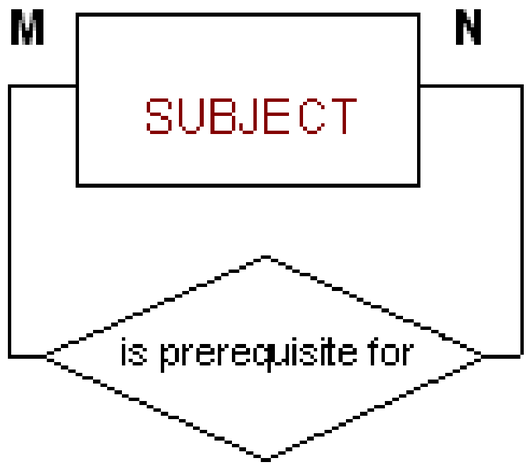


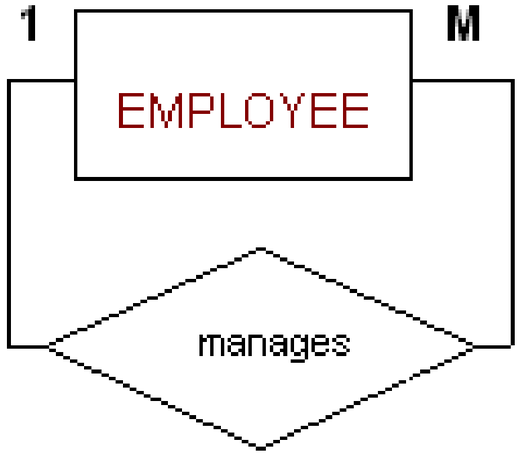
Relationship

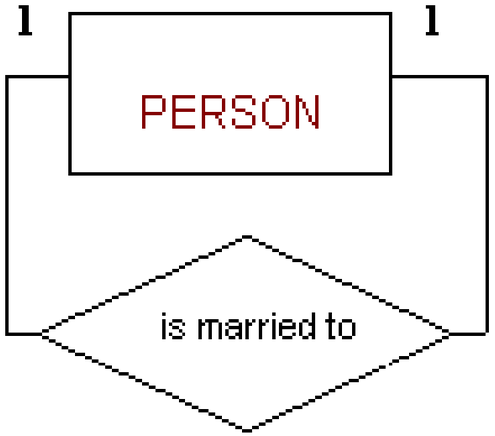
- Relationship
  - Unary
  - Recursive
  - Binary
  - Ternary
  - N-ary

Relationship

Recursive relationships occur within unary relationships. The relationship may be one to one, one to many or many to many. That is the cardinality of the relationship is unary. The connectivity may be **1:1**, **1:M**, or **M:N**.



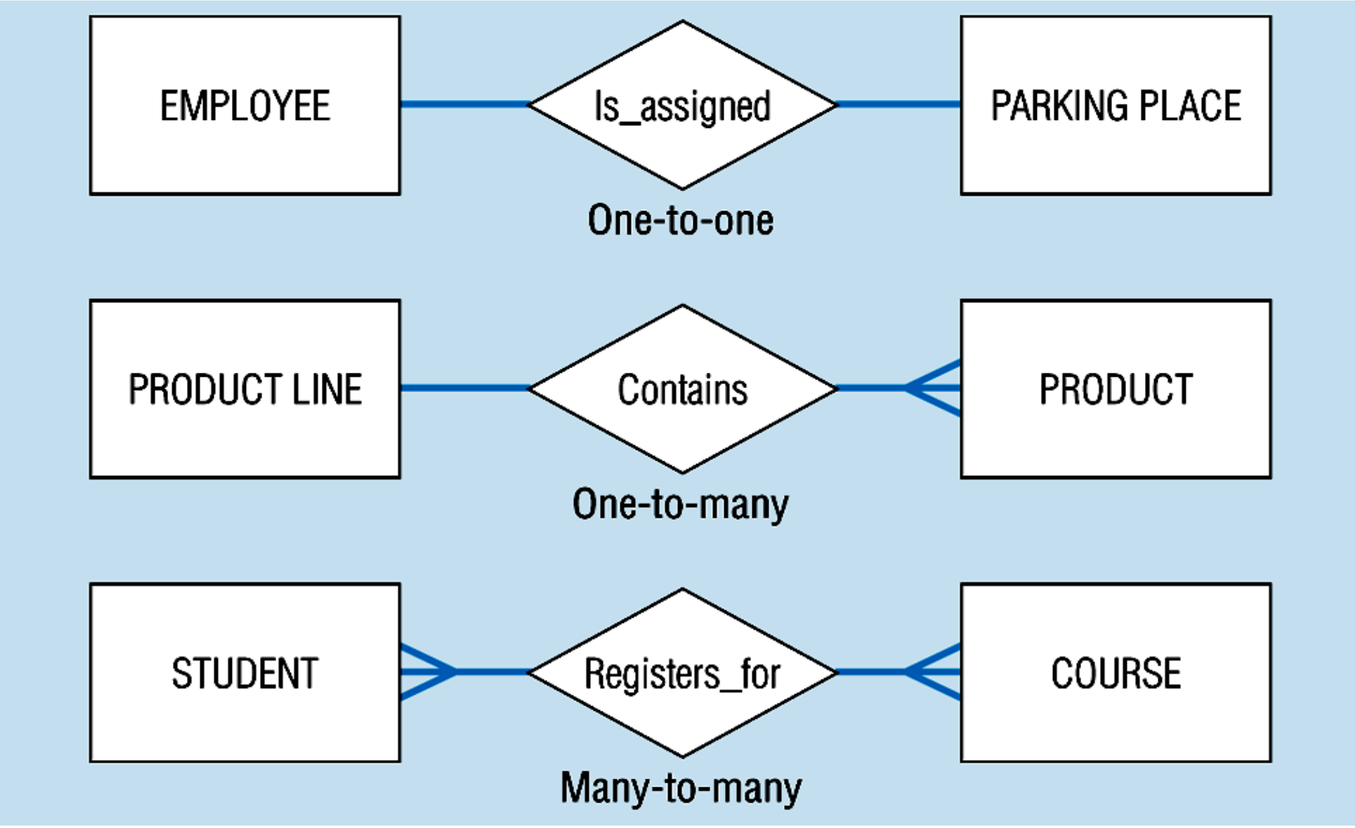




Cardinality Notation

- Relationship
  - Unary
  - Recursive
  - Binary
  - Ternary
  - N-ary

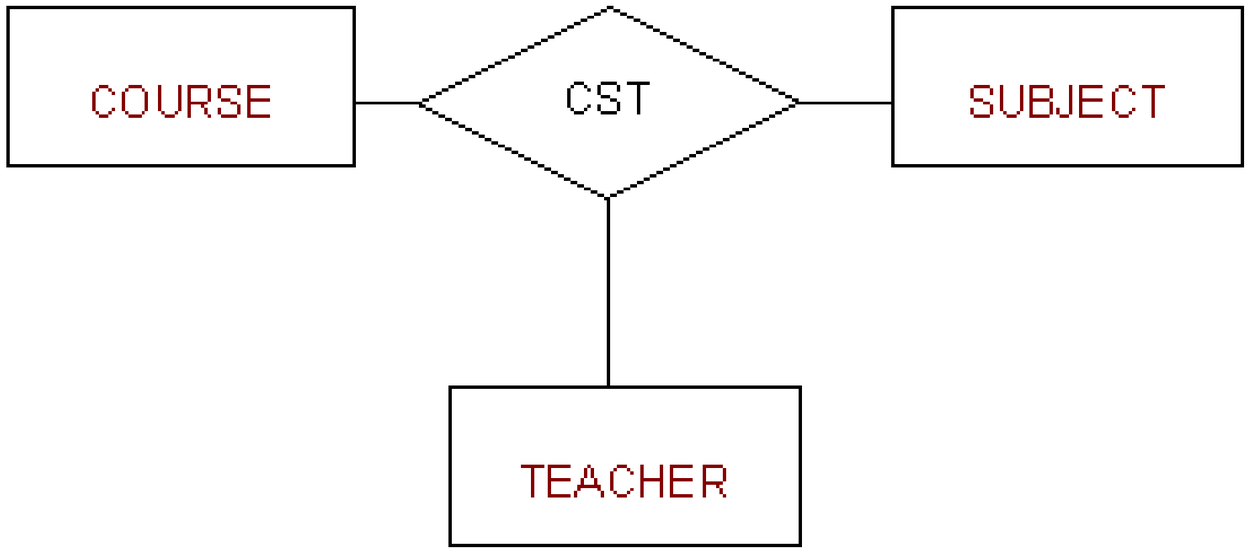
A **binary relationship** is when two entities participate and is the most common relationship degree.



Cardinality Notation

- Relationship
  - Unary
  - Recursive
  - Binary
  - Ternary
  - N-ary

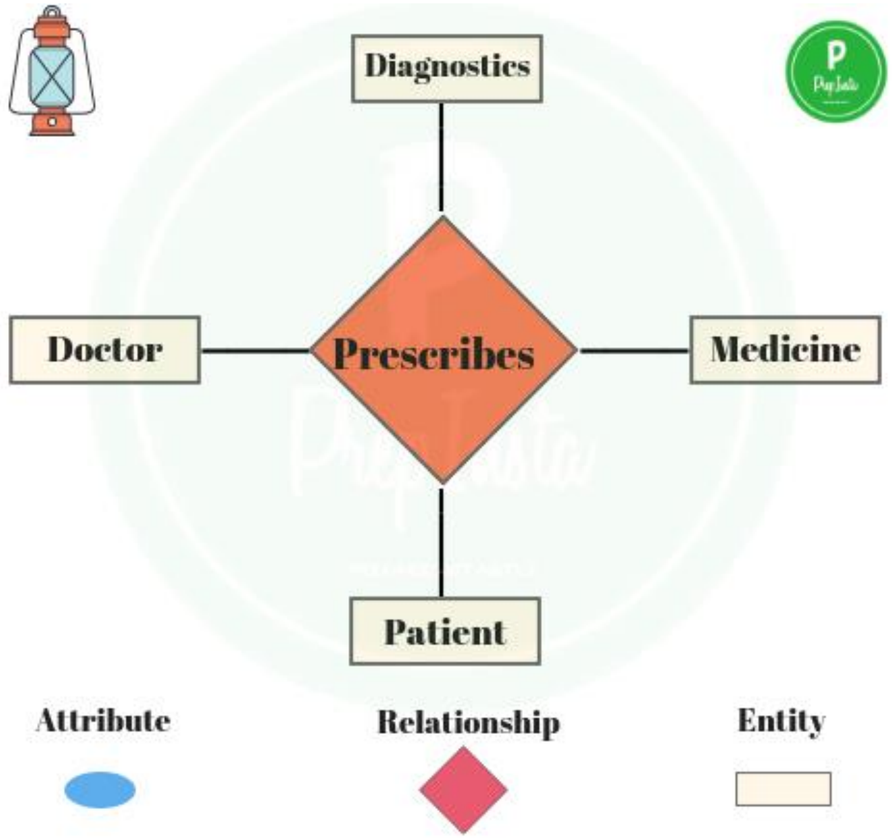
A **ternary relationship** is when three entities participate in the relationship.



Cardinality Notation

- Relationship
  - Unary
  - Recursive
  - Binary
  - Ternary
  - N-ary

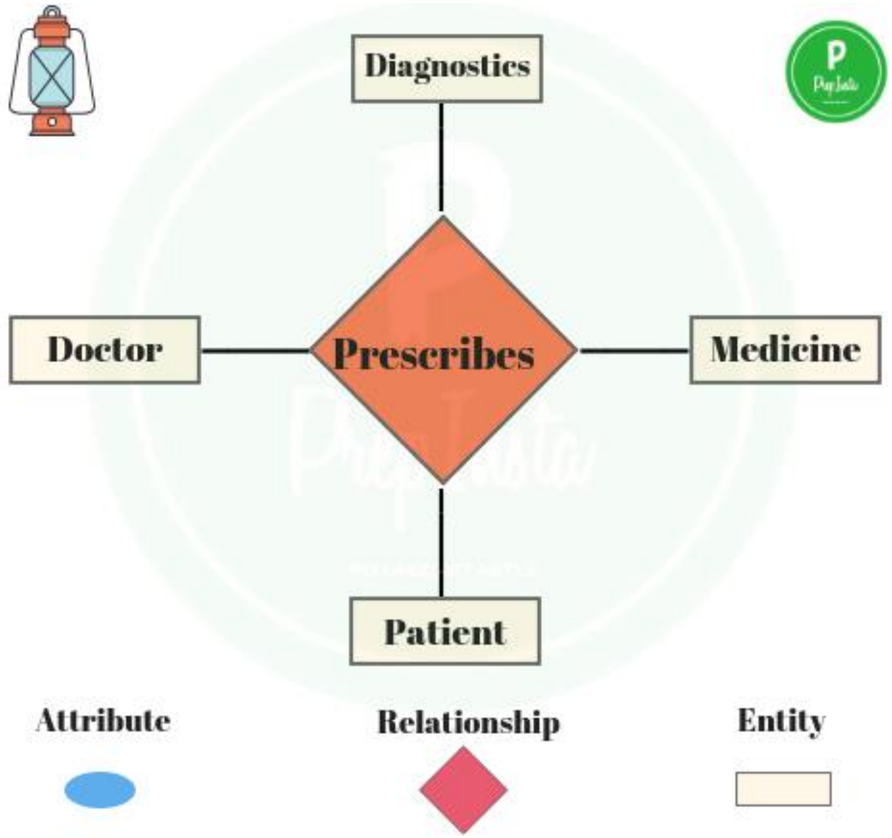
When a large number of entity sets are participating in a relationship, then such type of relationship is called an n-ary relationship



Cardinality Notation

- Relationship
  - Unary
  - Recursive
  - Binary
  - Ternary
  - N-ary

When a large number of entity sets are participating in a relationship, then such type of relationship is called an n-ary relationship



Sample 1

Design a ERD “The Music”

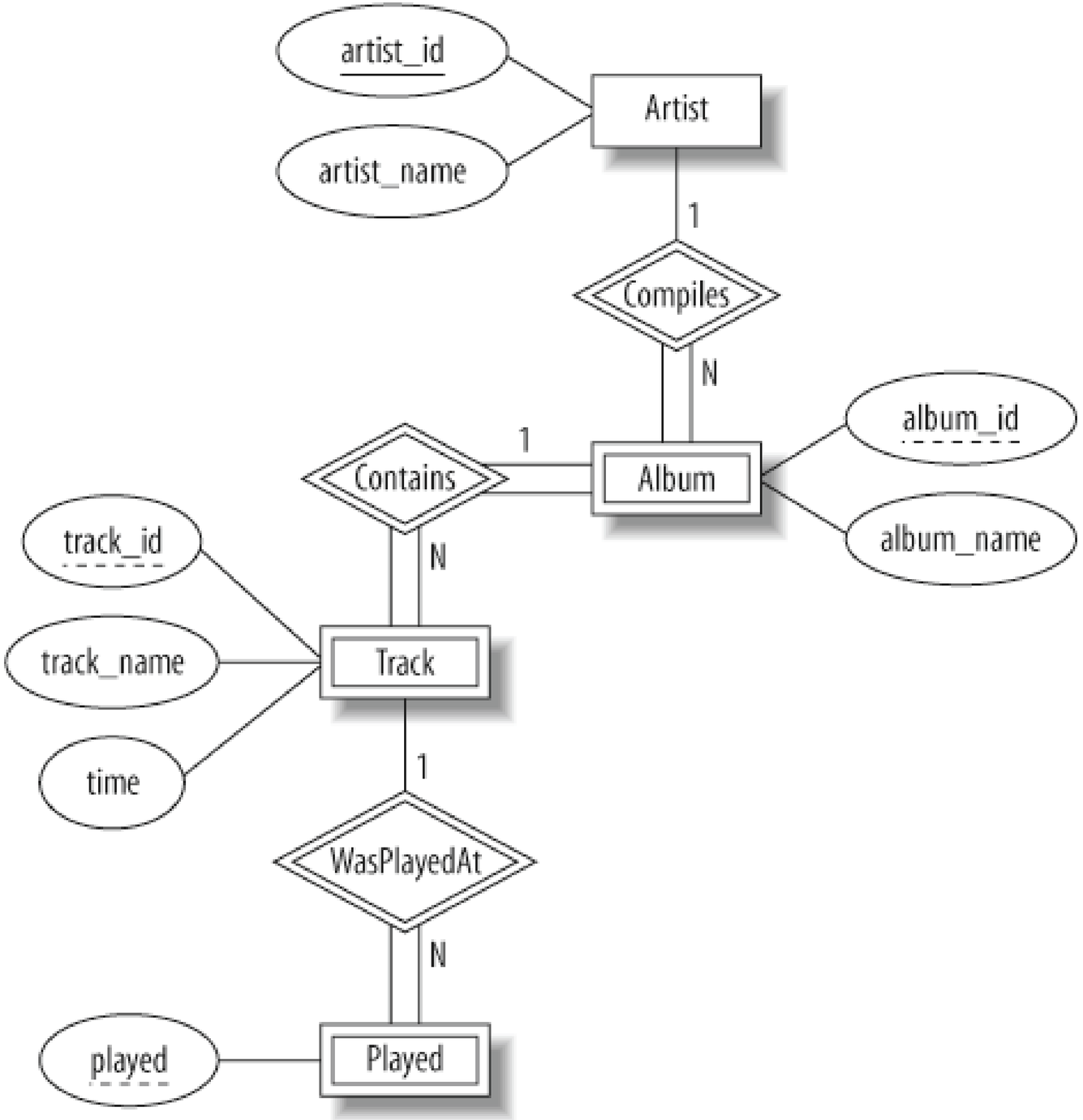
Description

Cơ sở dữ liệu lưu trữ thông tin về bộ sưu tập âm nhạc cá nhân. Bộ sưu tập gồm các mp3, CD, các bộ sưu tập.

Vì đây là cơ sở dữ liệu cho bộ sưu tập âm nhạc cá nhân nên chỉ cần quản lý về nghệ sĩ, album và các bản nhạc. Bỏ qua các thông tin về thể loại âm nhạc, tác giả ….

Xác định các yêu cầu về quản lý trong database:

- Dữ liệu của bộ sưu tập là các album.
- Một album được thực hiện bởi một nghệ sĩ.
- Một nghệ sĩ có một hoặc nhiều album.
- Một album chứa một hoặc nhiều bản nhạc
- Mỗi nghệ sĩ, album và bản nhạc đều có một cái tên.
- Mỗi bản nhạc nằm trong đúng một album.
- Mỗi bản nhạc có độ dài thời gian, tính bằng giây.
- Khi một bản nhạc được phát, ngày và giờ bắt đầu phát lại (chính xác đến giây gần nhất) phải được ghi lại; điều này được sử dụng để báo cáo khi một bản nhạc được phát lần cuối cũng như số lần nhạc của một nghệ sĩ, từ một album hoặc một bản nhạc đã được phát.

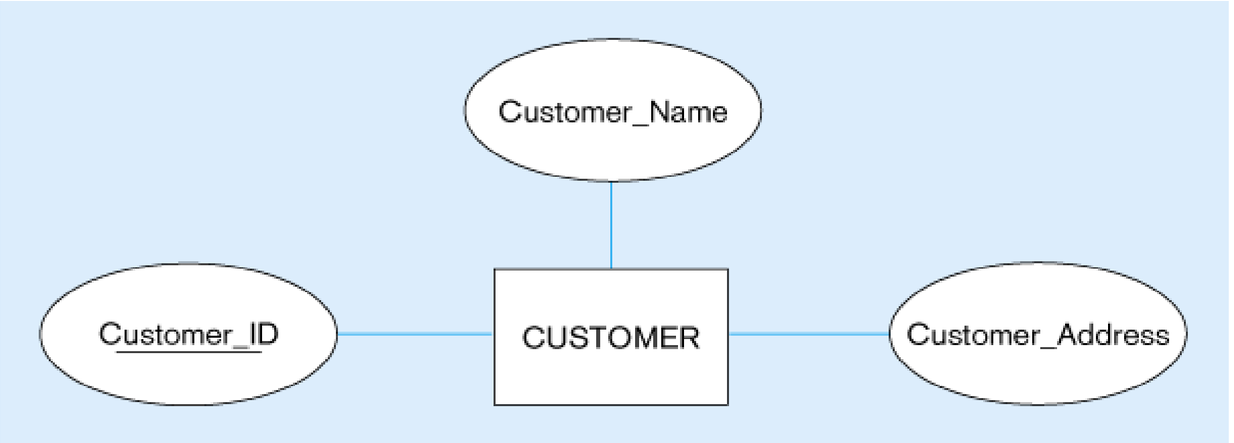

artist_id
artist_name
Artist
1
Compiles
N
album_id
1
Contains
Album
album_name
track_id
N
track_name
Track
time
1
WasPlayedAt
N
played
Played

3

Converting ER Model to relational schema Rule

Convert ER Model

Rule 1 - Convert entity type with simple attributes



CUSTOMER entity type with simple attributes

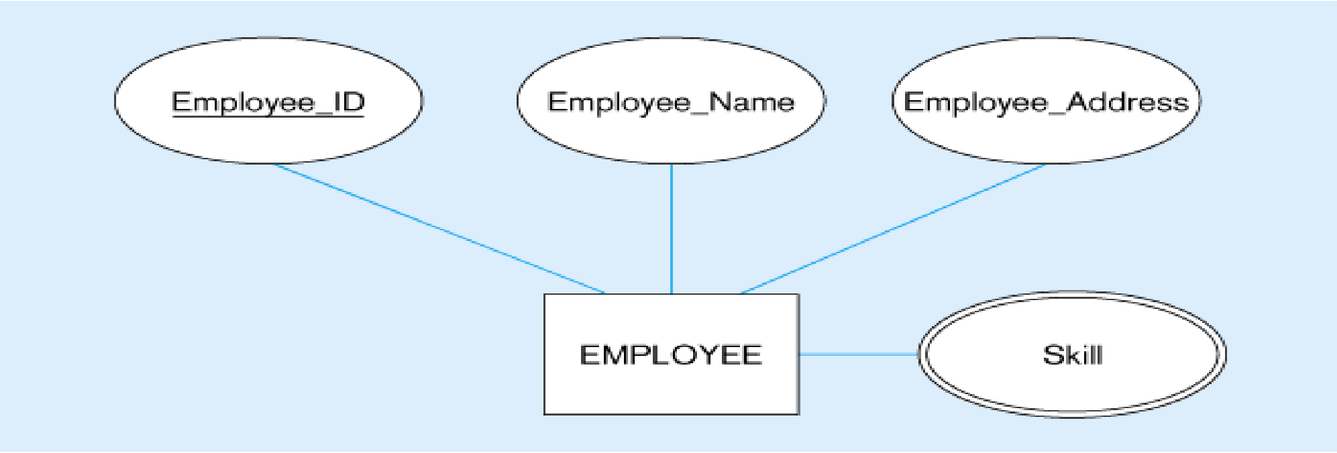
CUSTOMER relation

CUSTOMER

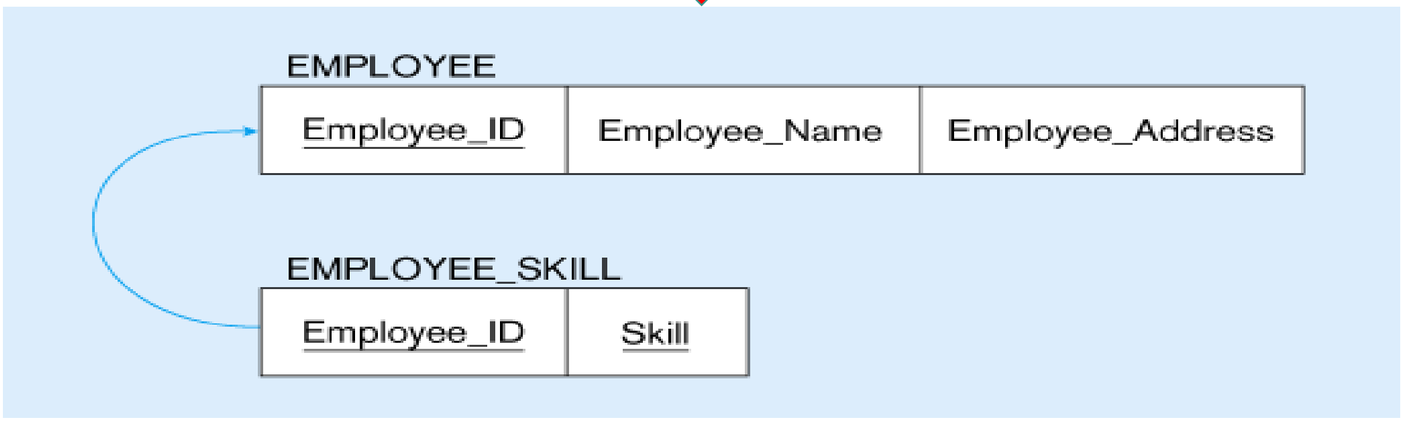
Customer_ID	Customer_Name	Customer_Address

Convert ER Model

Rule 2 - Convert Multi value attribute



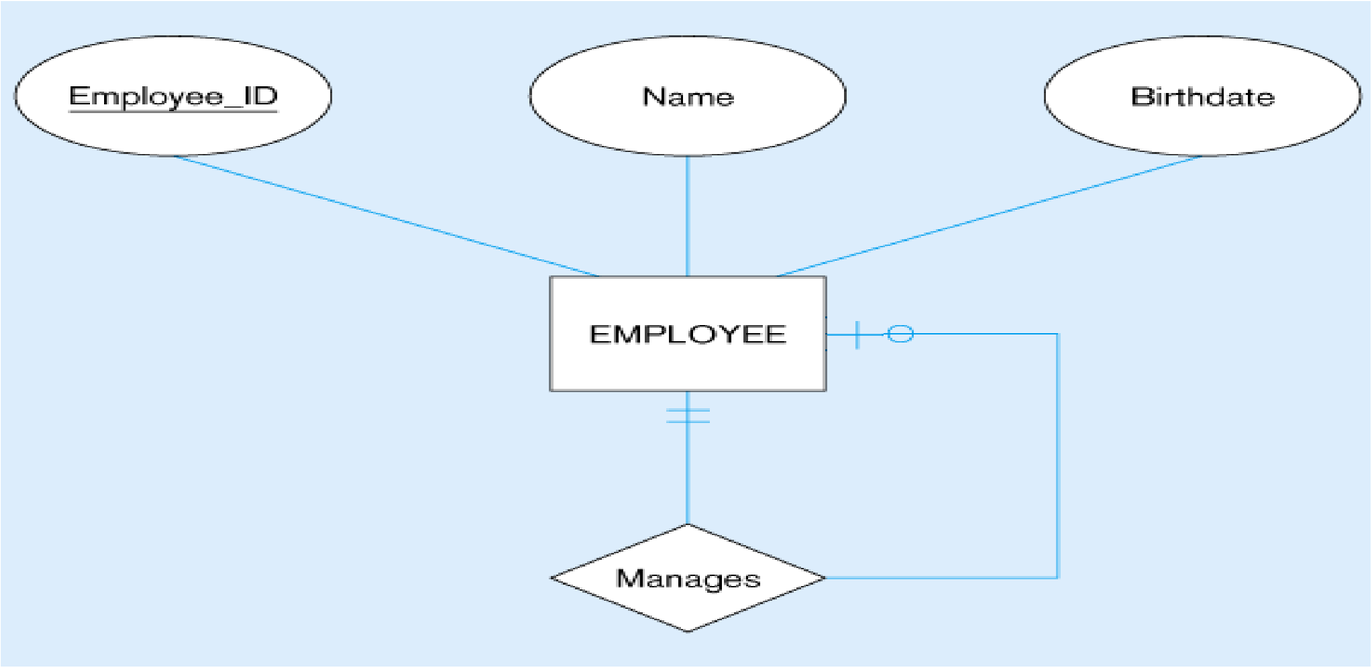
Multivalued attribute becomes a separate relation with foreign key



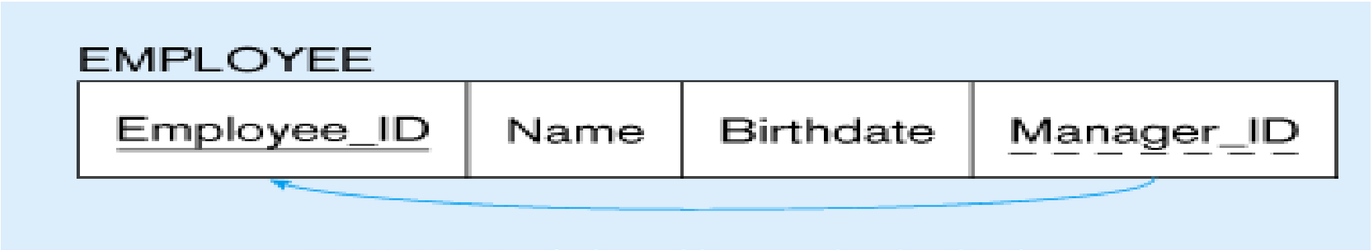
1–to–many relationship between original entity and new relation

Converting ER Model to relational schema

Rule 3 - Convert Unary relationship one to one



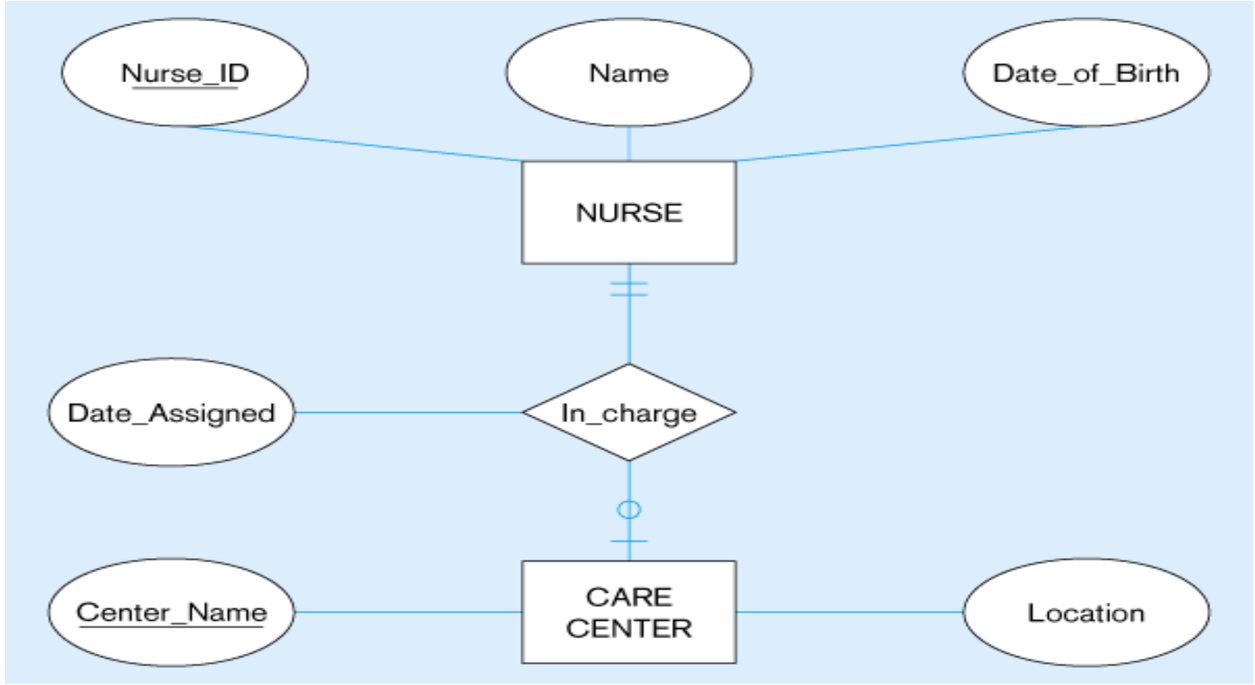
EMPLOYEE entity with Manages relationship

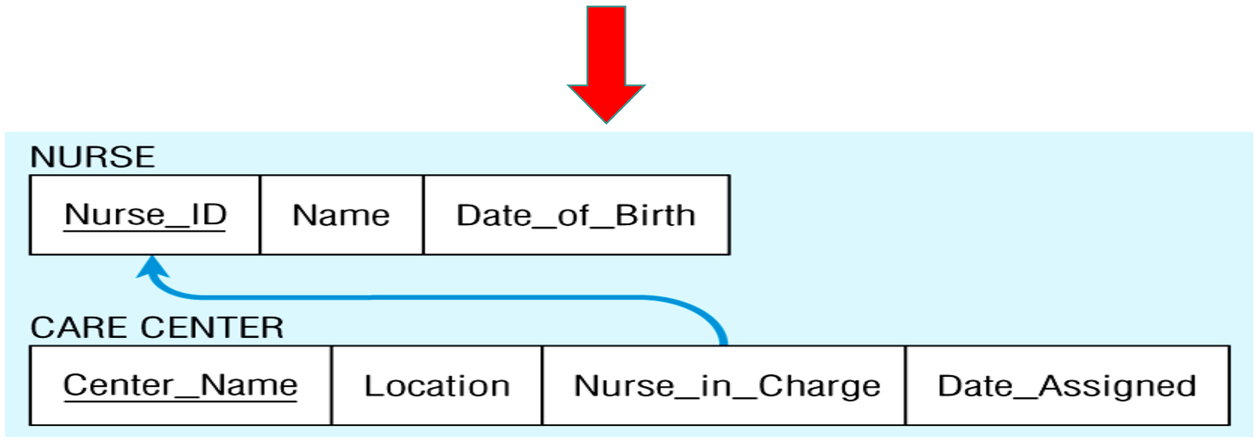


EMPLOYEE relation with recursive foreign key

Converting ER Model to relational schema

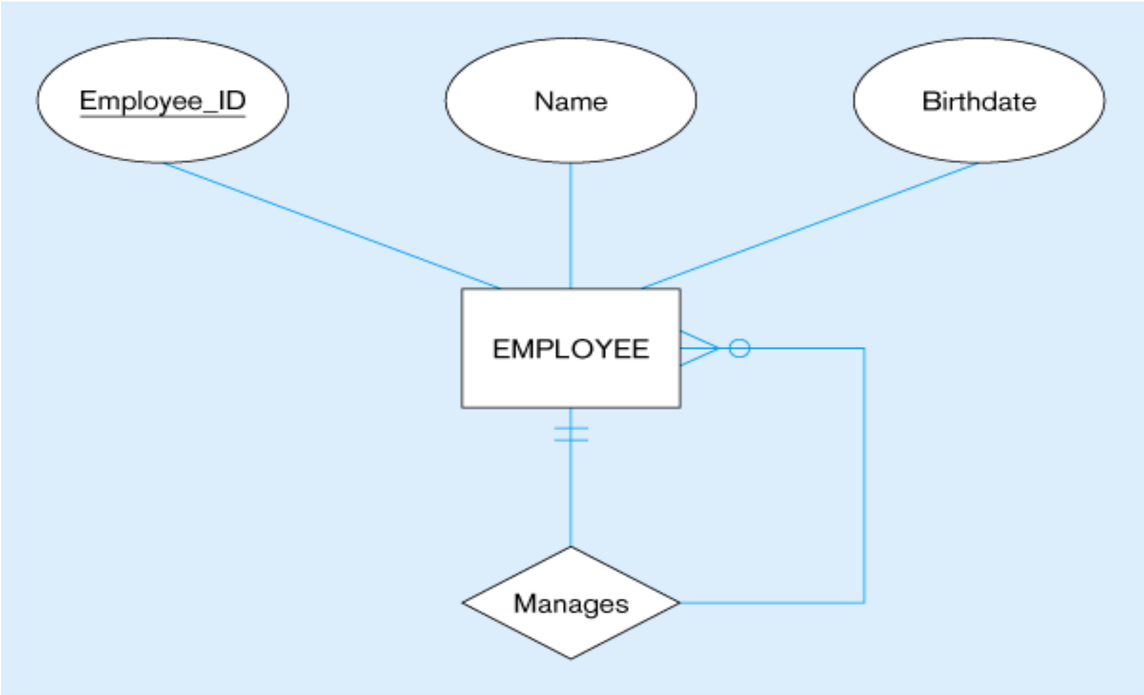
Rule 4 – Convert binary relationship one to one



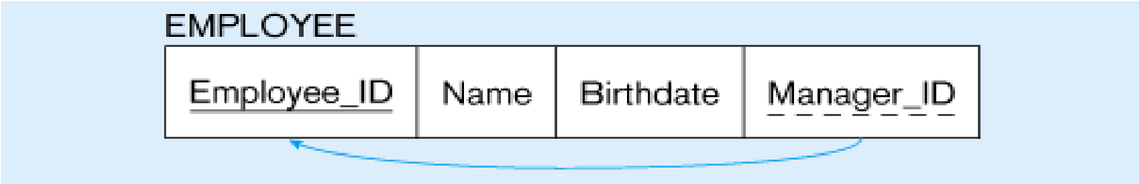


Converting ER Model to relational schema

Rule 5 – Convert Unary relationship one to many



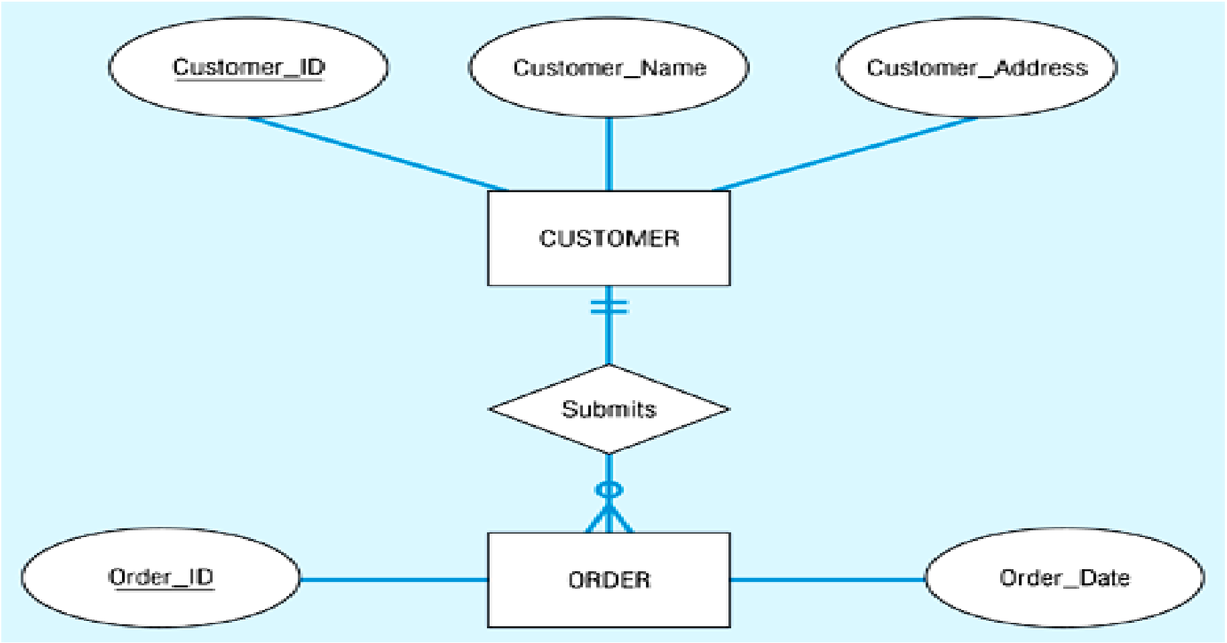
EMPLOYEE entity with Manages relationship



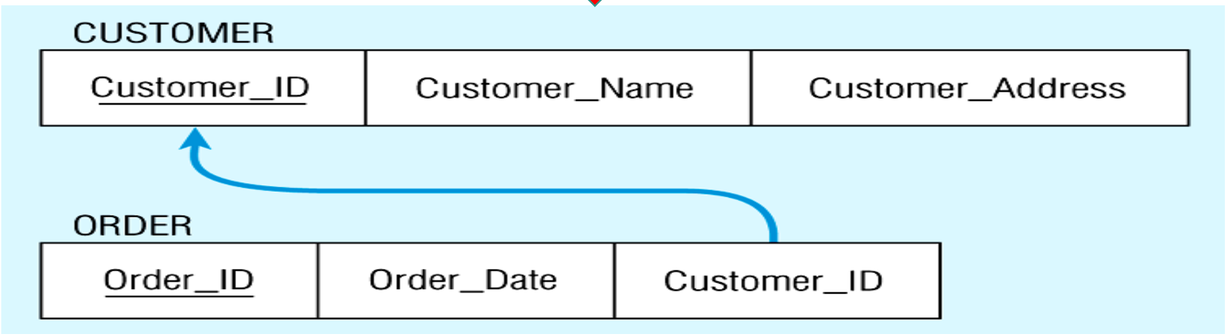
EMPLOYEE relation with recursive foreign key

Converting ER Model to relational schema

Rule 6 – Convert Binary relationship one to many



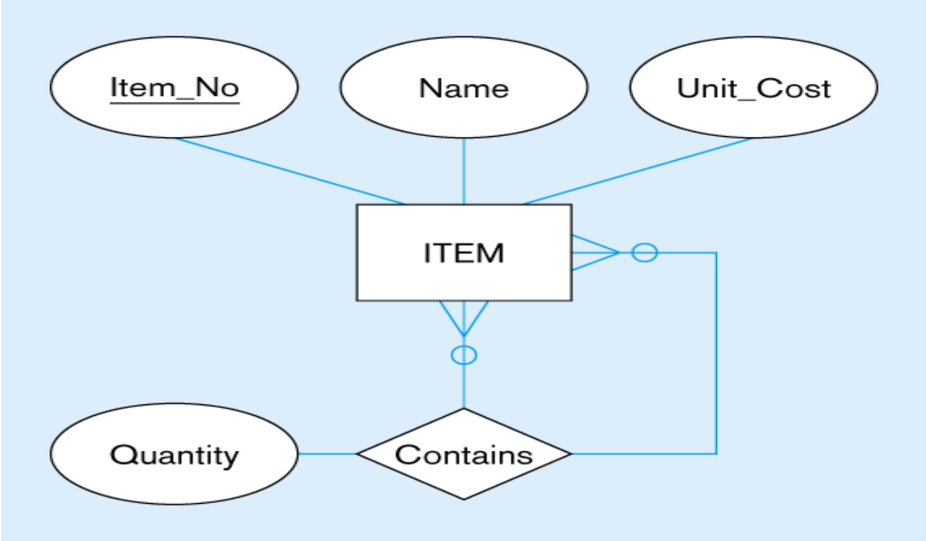
Note the mandatory one



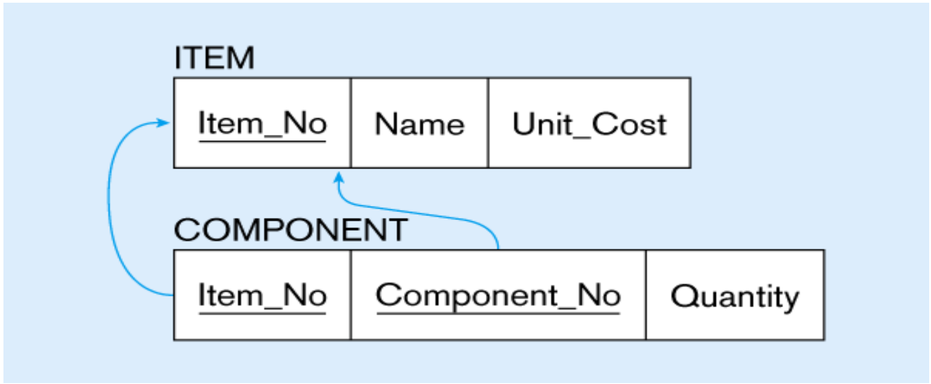
Again, no null value in the foreign key…this is because of the mandatory minimum cardinality

Converting ER Model to relational schema

Rule 7 – Convert Unary relationship many to many



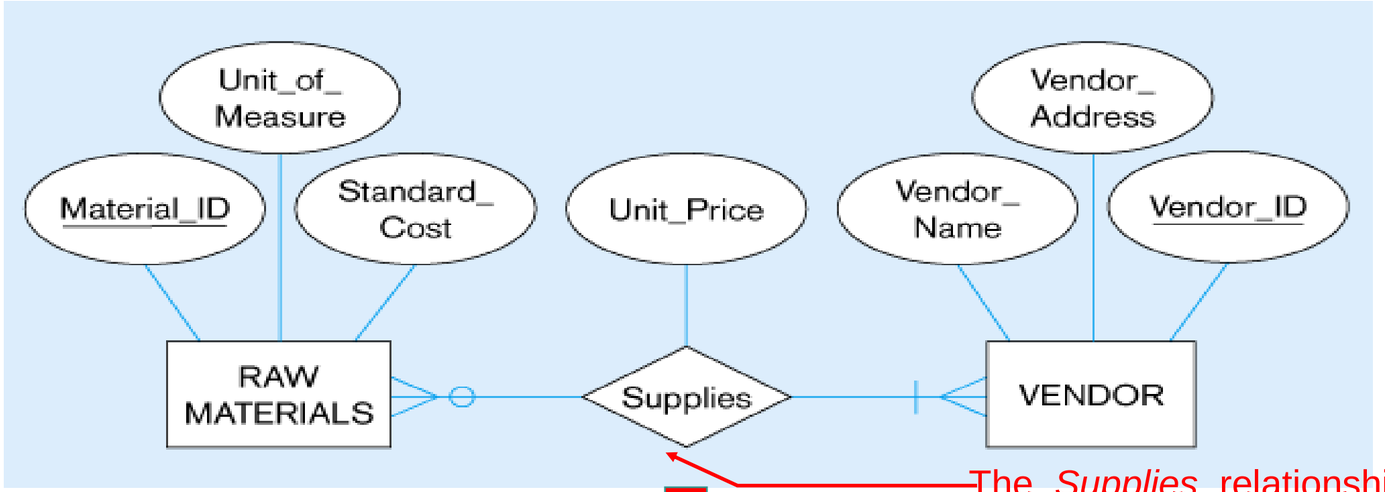
Bill-of-materials relationships (M:N)



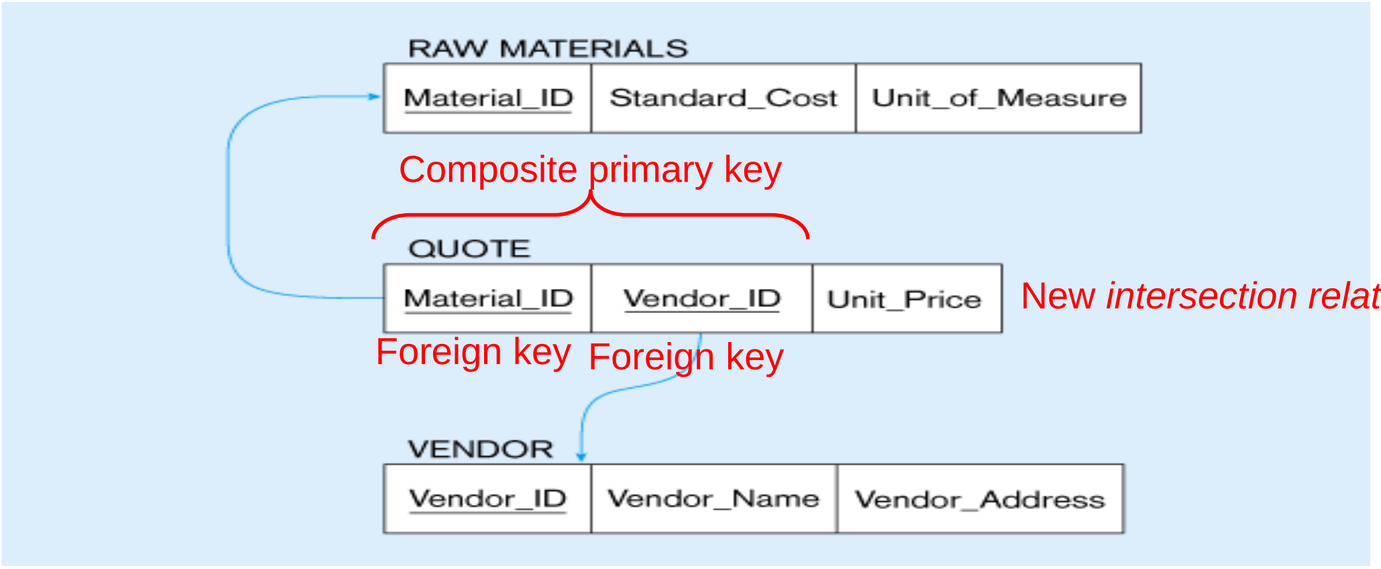
ITEM and COMPONENT relations

Converting ER Model to relational schema

Rule 8 – Convert Binary relationship many to many

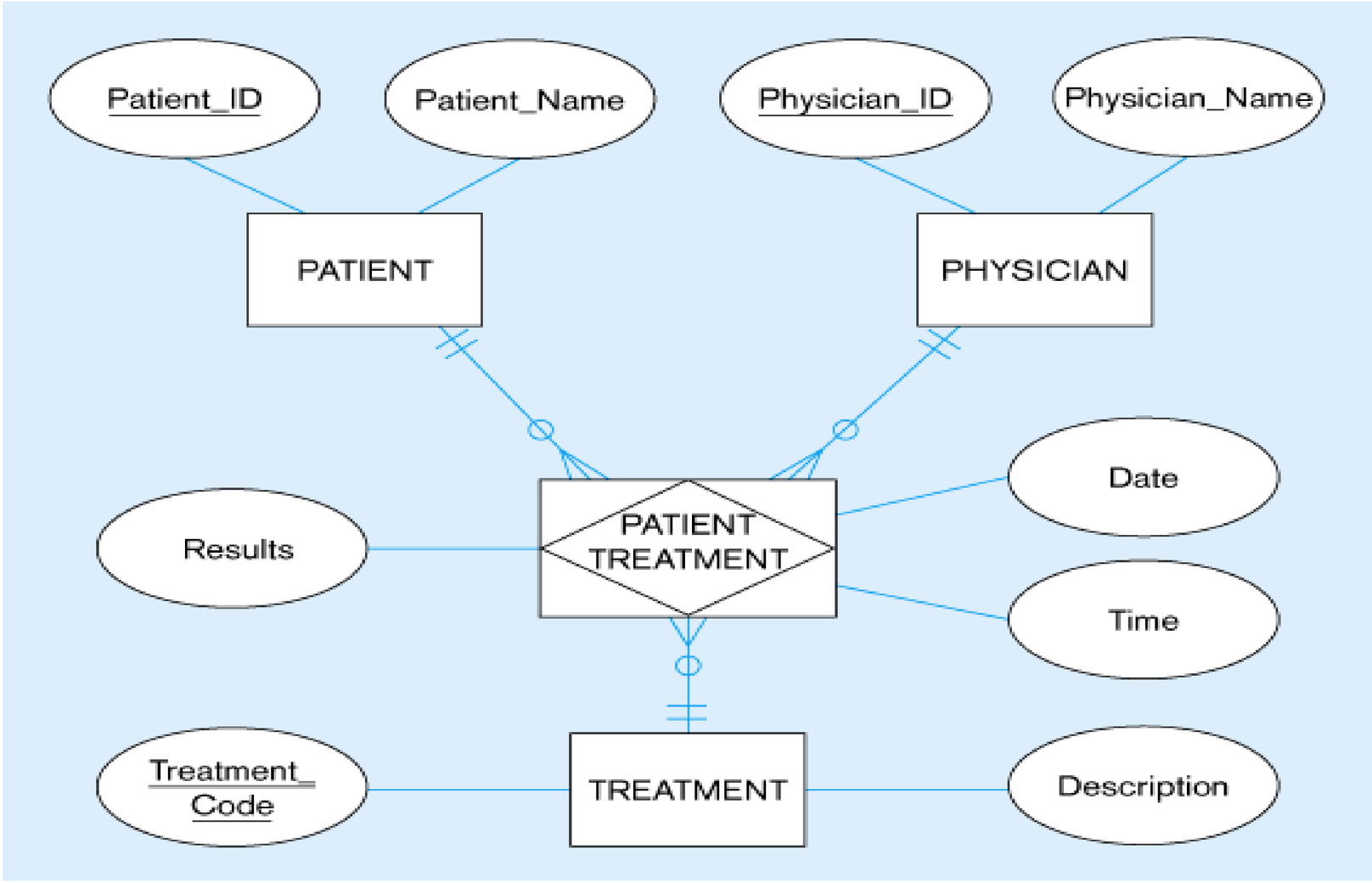


The *Supplies* relationship will need to become a separate relation



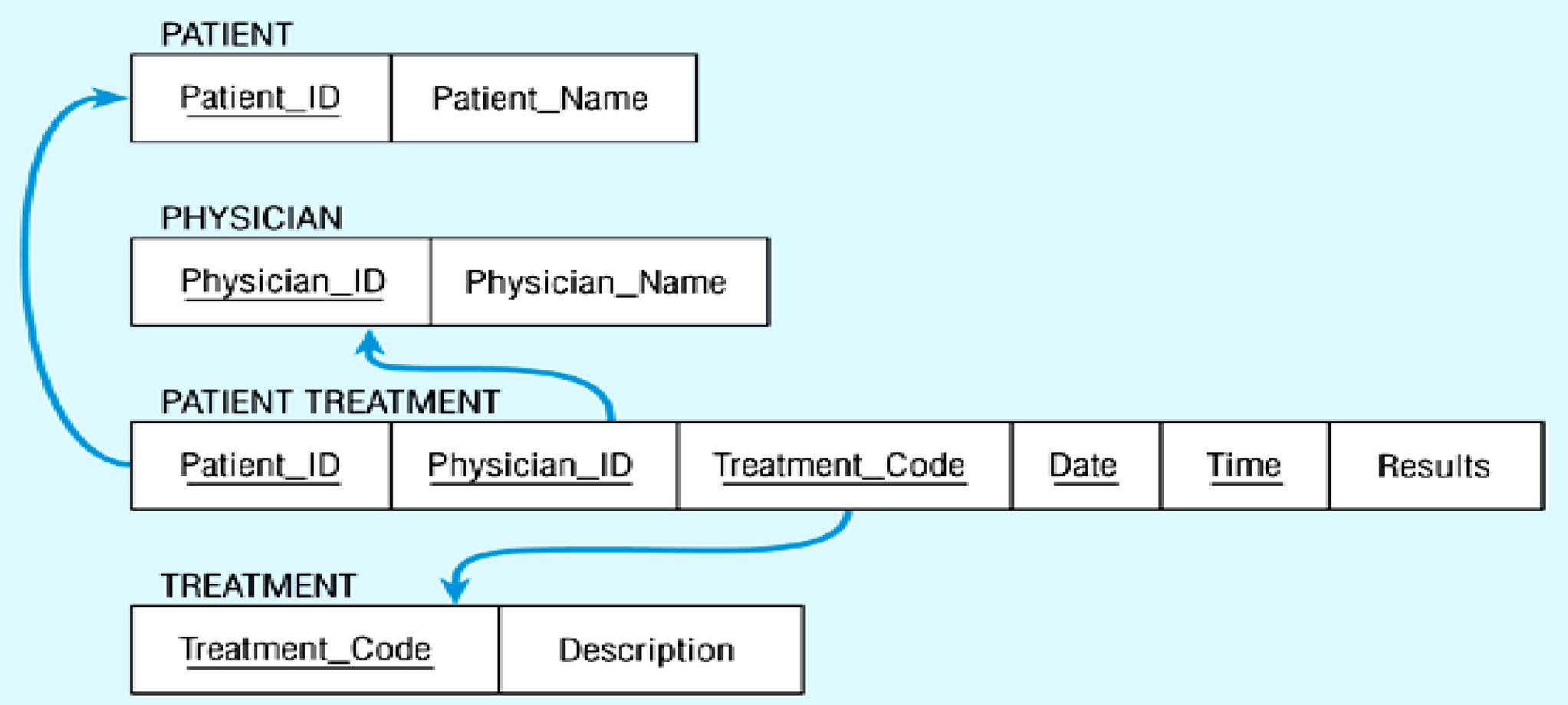
Converting ER Model to relational schema

Another - Convert Ternary relationship

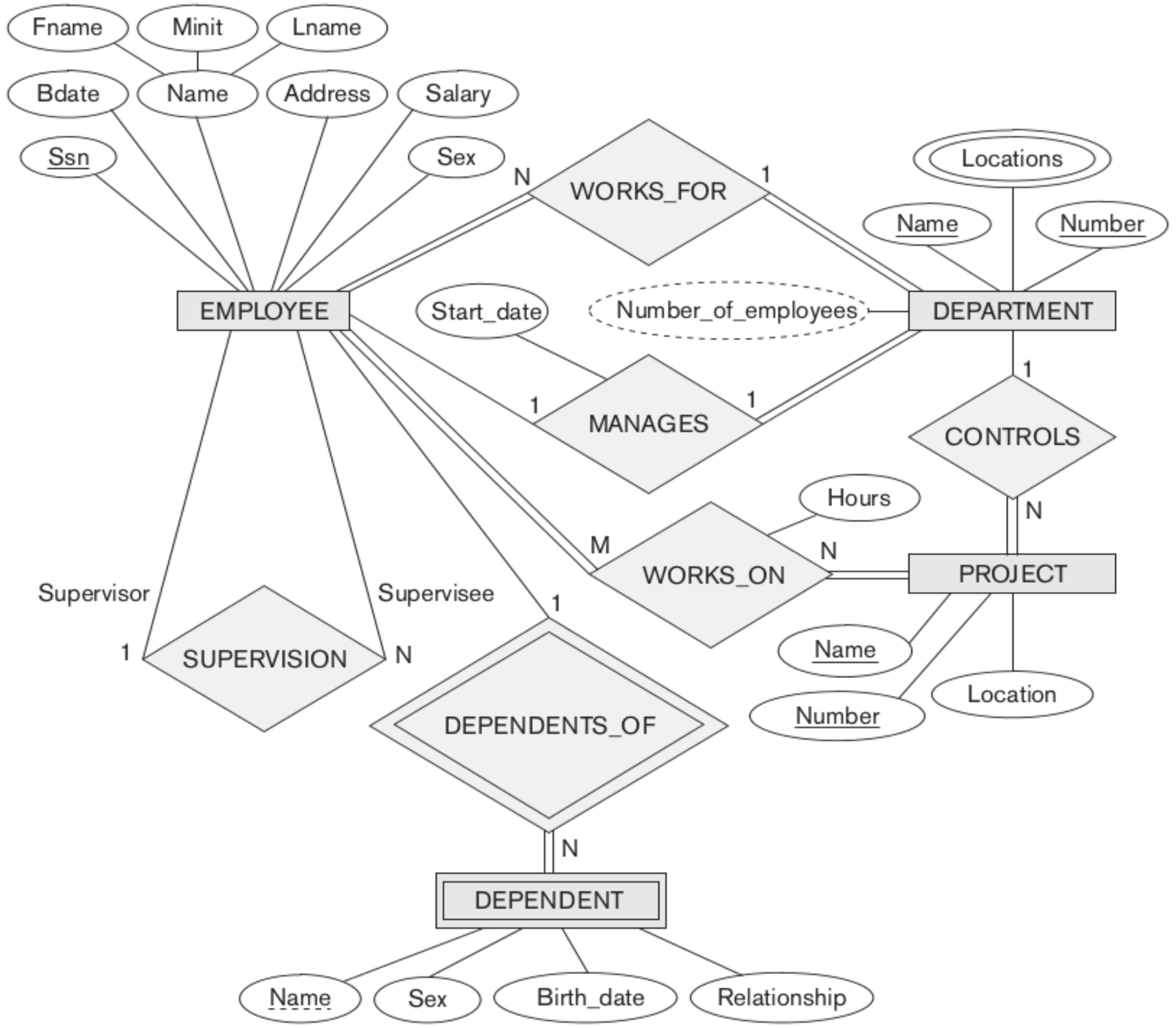


Converting ER Model to relational schema

Another- Convert Ternary relationship (2)



Converting ER Model to relational schema



Thank you!

Any questions?


Until You Get It Right
Keep Training and Keep Learning
START