Data Modeling

Introduction

- Process of creating a data model for the data to be stored in a database.
- This data model is a conceptual representation of Data objects, the associations between different data objects, and the rules.
- Data modeling helps in the visual representation of data and enforces business rules, regulatory compliances, and government policies on the data.
- Data Models ensure consistency in naming conventions, default values, semantics, security while ensuring quality of the data.

Data Model

- It is defined as an abstract model that organizes data description, data semantics, and consistency constraints of data.
- It emphasizes on what data is needed and how it should be organized instead of what operations will be performed on data.
- Data Model is like an architect's building plan, which helps to build conceptual models and set a relationship between data items.

Types Of Data Modeling Techniques

The 2 types of Data Modeling Techniques are:

I. Entity Relationship (E-R) Model
II. UML (Unified Modeling Language)

We will discuss them in detail later.

Why use Data Model?

The primary goal of using data model are:

a) Ensures that all data objects required by the database are accurately represented. Omission of data will lead to creation of faulty reports and produce incorrect results.

b) Helps in designing the database at the conceptual, physical and logical levels.

c) Helps to define the relational tables, primary and foreign keys and stored procedures.

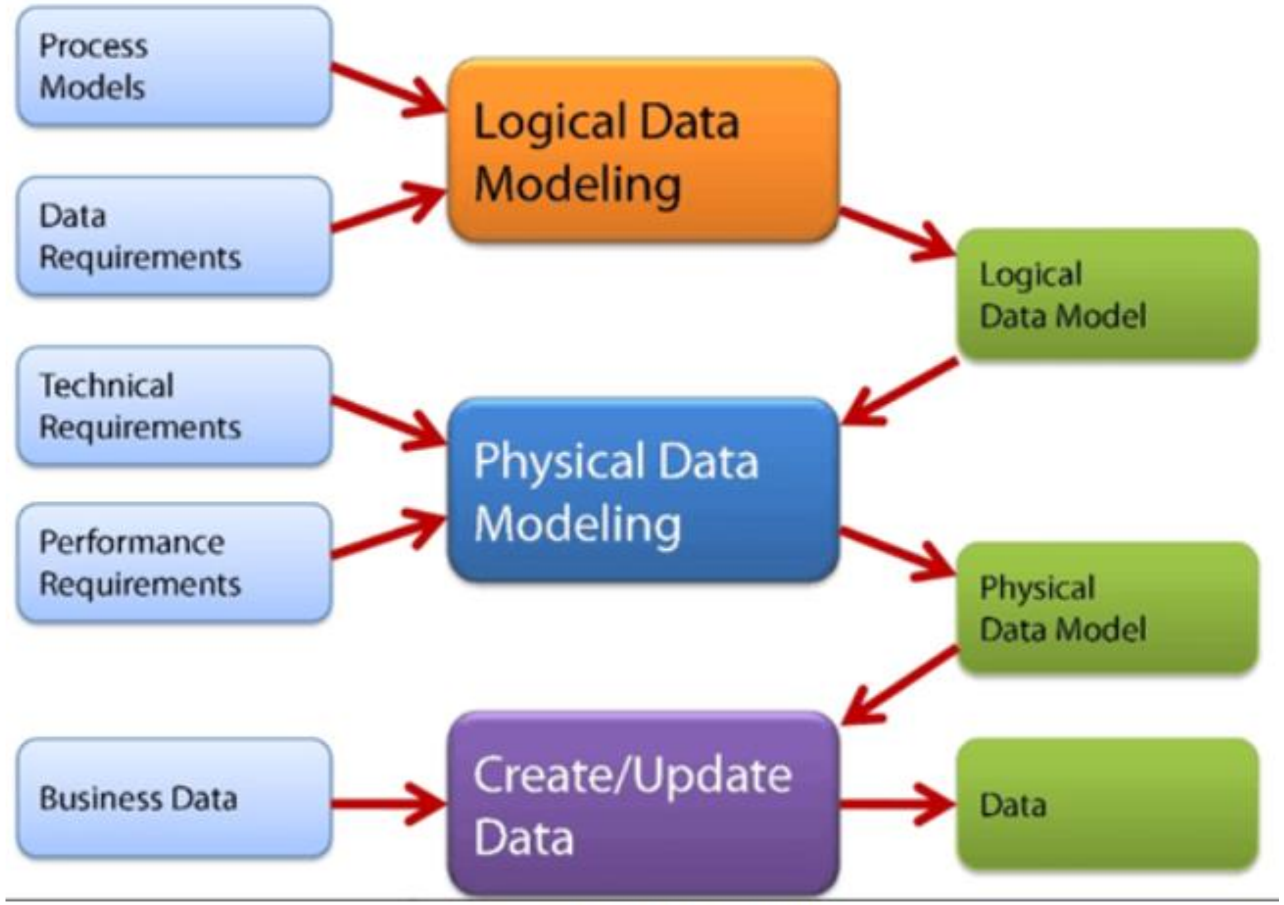
d) Provides a clear picture of the base data and can be used by database developers to create a physical database.

e) Also helpful to identify missing and redundant data.

f) Though the initial creation of data model is labor and time consuming, in the long run, it makes your IT infrastructure upgrade and maintenance cheaper and faster.

Types of Data Models

- There are mainly 3 different types of data models: ***CONCEPTUAL DATA MODELS***, ***LOGICAL DATA MODELS***, and ***PHYSICAL DATA MODELS***, and each one has a specific purpose.



Types of Data Model

I. ***Conceptual Data Model:*** Defines **WHAT** the system contains. This model is typically created by Business stakeholders and Data Architects. The purpose is to **organize, scope** and **define business concepts and rules**.

II. ***Logical Data Model:*** Defines **HOW** the system should be implemented **regardless of the DBMS**. This model is typically created by Data Architects and Business Analysts. The purpose is to **developed technical map of rules and data structures**.

III. ***Physical Data Model:*** Describes **HOW** the system will be implemented **using a specific DBMS system**. This model is typically created by DBA and developers. The purpose is **actual implementation of the database**.

Conceptual Data Model

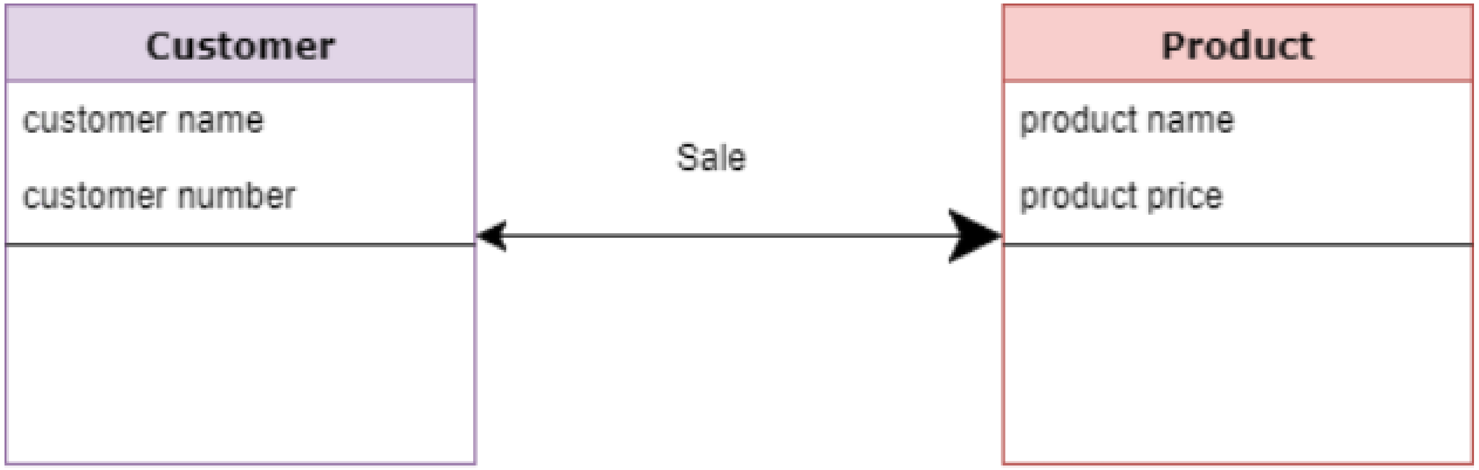
- A **Conceptual Data Model** is an organized view of database concepts and their relationships.
- **Purpose**: to establish entities, their attributes, and relationships.
- In this data modeling level, there is hardly any detail available on the actual database structure.

- The 3 basic tenants of Conceptual Data Model are:

1. **Entity**: A real-world thing
2. **Attribute**: Characteristics or properties of an entity
3. **Relationship**: Dependency or association between two entities

Example

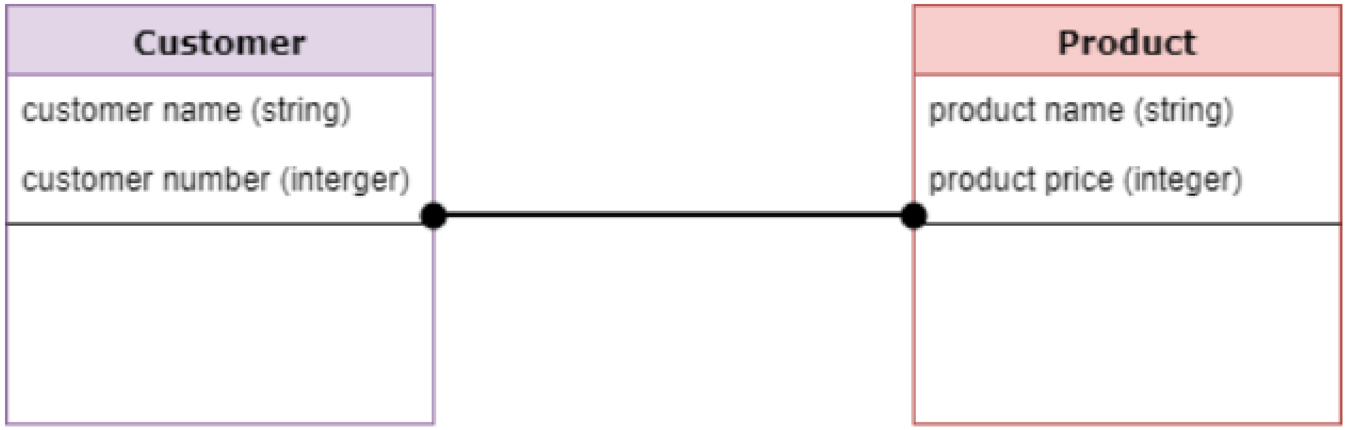
- Consider 2 entities: Customer and Product
- Attributes of the Customer entity: Customer number and name.
- Attributes of Product entity: Product name and price
- Relationship between the customer and product: *Sale*



Conceptual Data Model

Logical Data Model

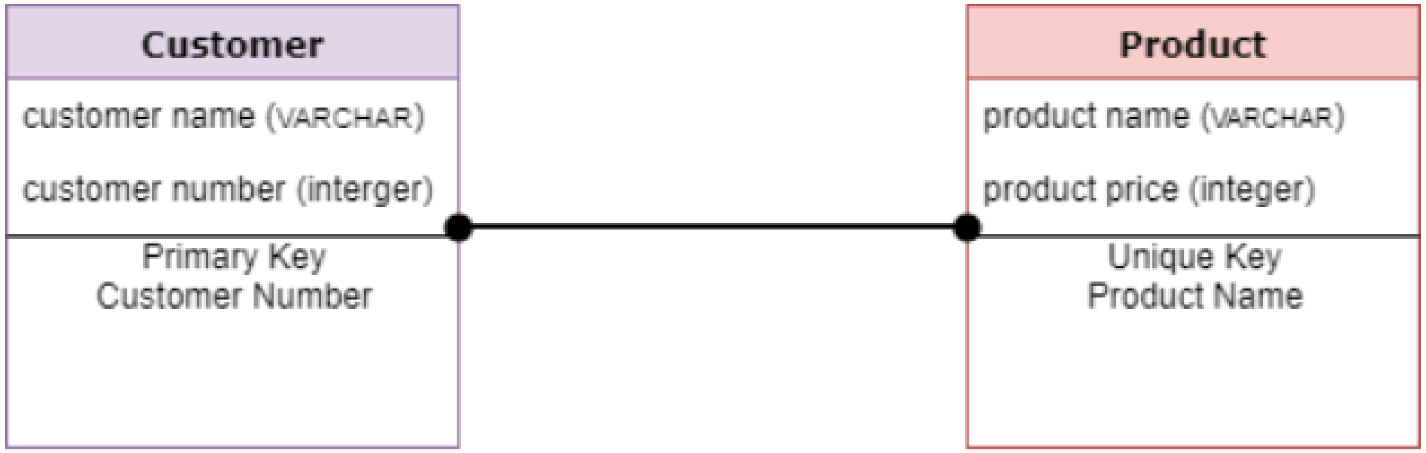
- The **Logical Data Model** is used to define the structure of data elements and to set relationships between them.
- It adds further information to the conceptual data model elements.
- Advantage: provide a foundation to form the base for the Physical model. However, the modeling structure remains generic.



Logical Data Model

Physical Data Model

- A **Physical Data Model** describes a database-specific implementation of the data model.
- It offers database abstraction and helps generate the schema. This is because of the *richness of meta-data offered by a Physical Data Model*.
- The physical data model also helps in visualizing database structure by replicating database column keys, constraints, indexes, triggers, and other RDBMS features.



Physical Data Model

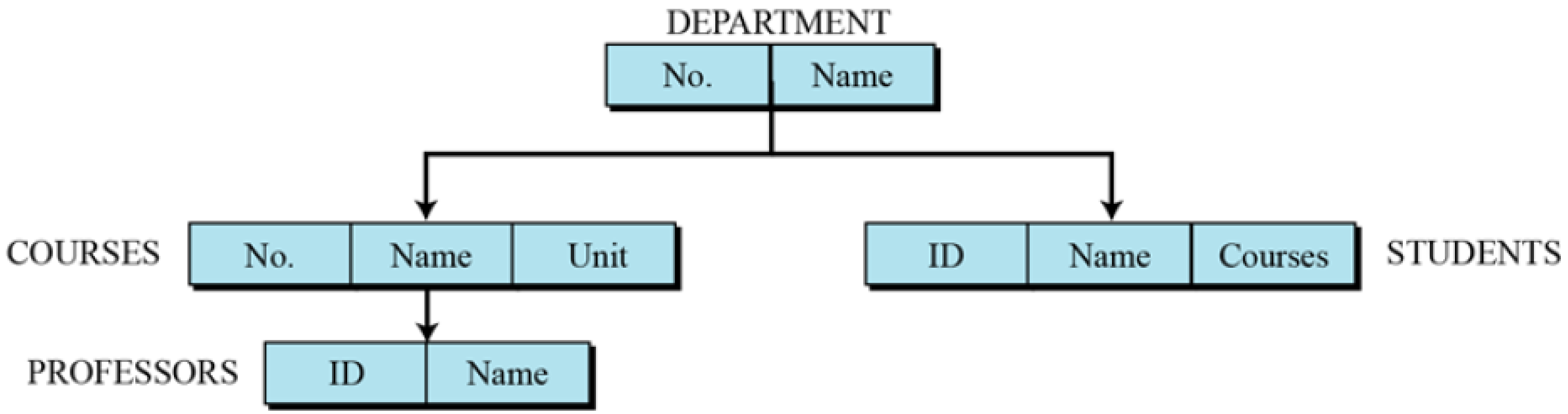
Types Of Data Models On The Basis Of Data Relationships

- The data model also describes the relationships between different parts of the data.
- Various types of data models are:

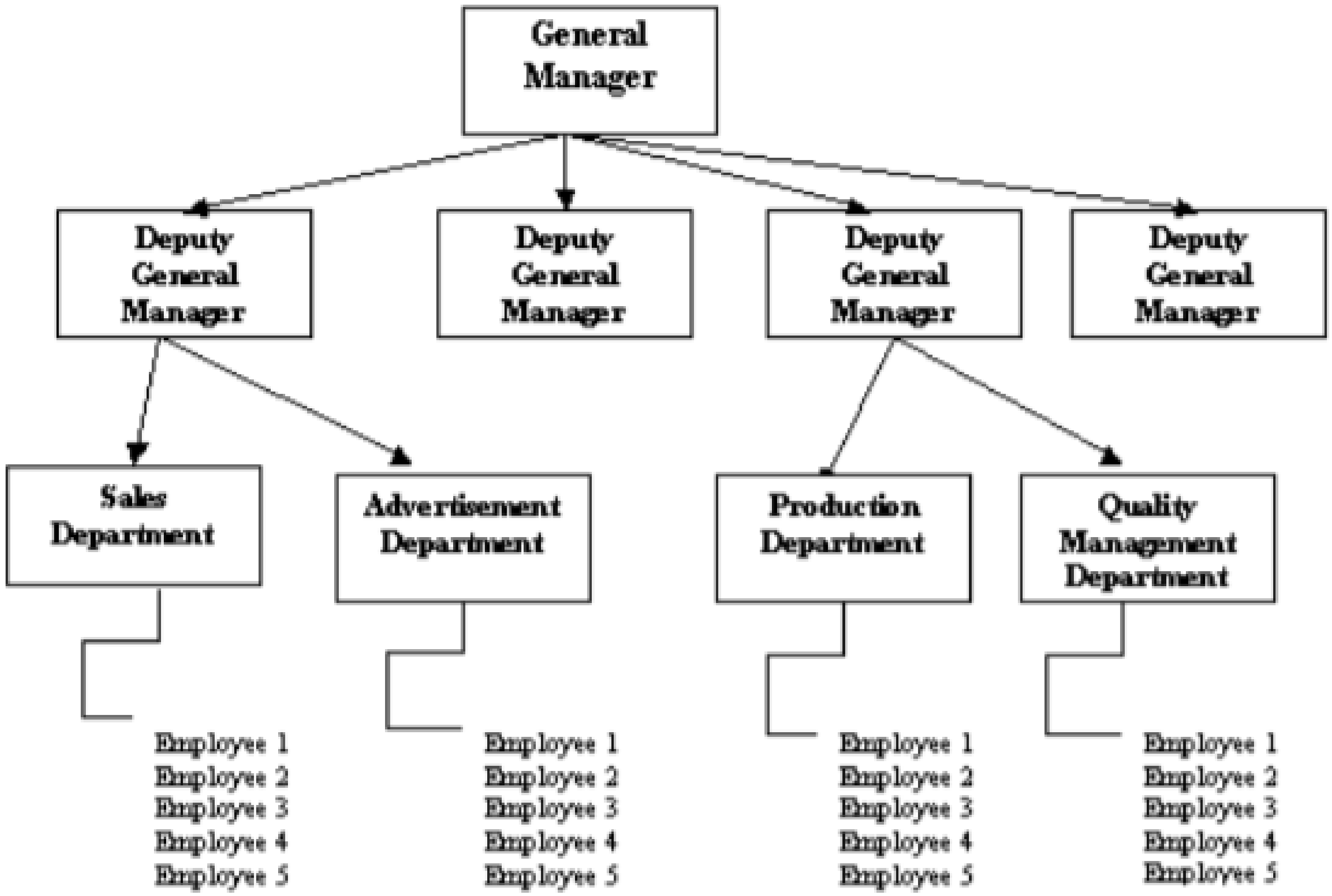
1. Object oriented model
2. Hierarchical data model
3. Relational data model
4. Network model

Hierarchical Data Model

- ✓ This is called a ***PARENT-CHILD*** relationship.
- ✓ In this model, each entity has only one parent but several children.
- ✓ This model is based on *tree* data structure.
- ✓ At the top of the hierarchy, there is only one entity which is called ***ROOT***.

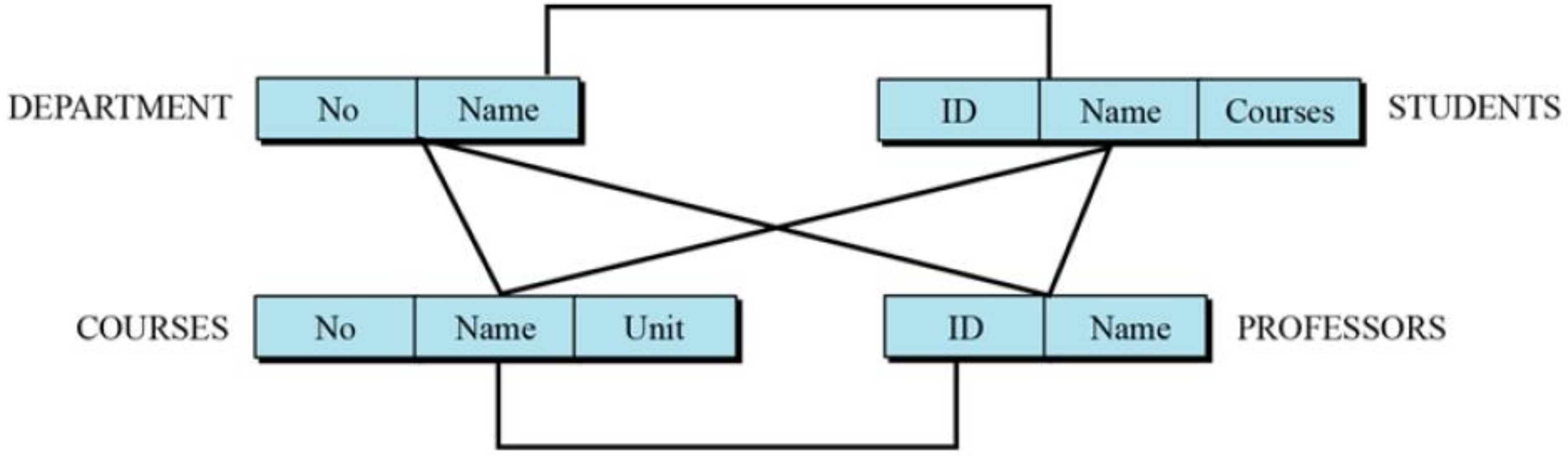


Example


General Manager
Deputy General Manager
Deputy General Manager
Deputy General Manager
Deputy General Manager
Sales Department
Advertisement Department
Production Department
Quality Management Department
Employee 1
Employee 2
Employee 3
Employee 4
Employee 5
Employee 1
Employee 2
Employee 3
Employee 4
Employee 5
Employee 1
Employee 2
Employee 3
Employee 4
Employee 5
Employee 1
Employee 2
Employee 3
Employee 4
Employee 5

Network Model

- ✓ In this model, *every data item can be related to many others ones.*
- ✓ The database structure is like a graph.
- ✓ This is similar to the hierarchical model and also provides a tree-like structure.
- ✓ However, a *child* is allowed to have *more than one parent.*



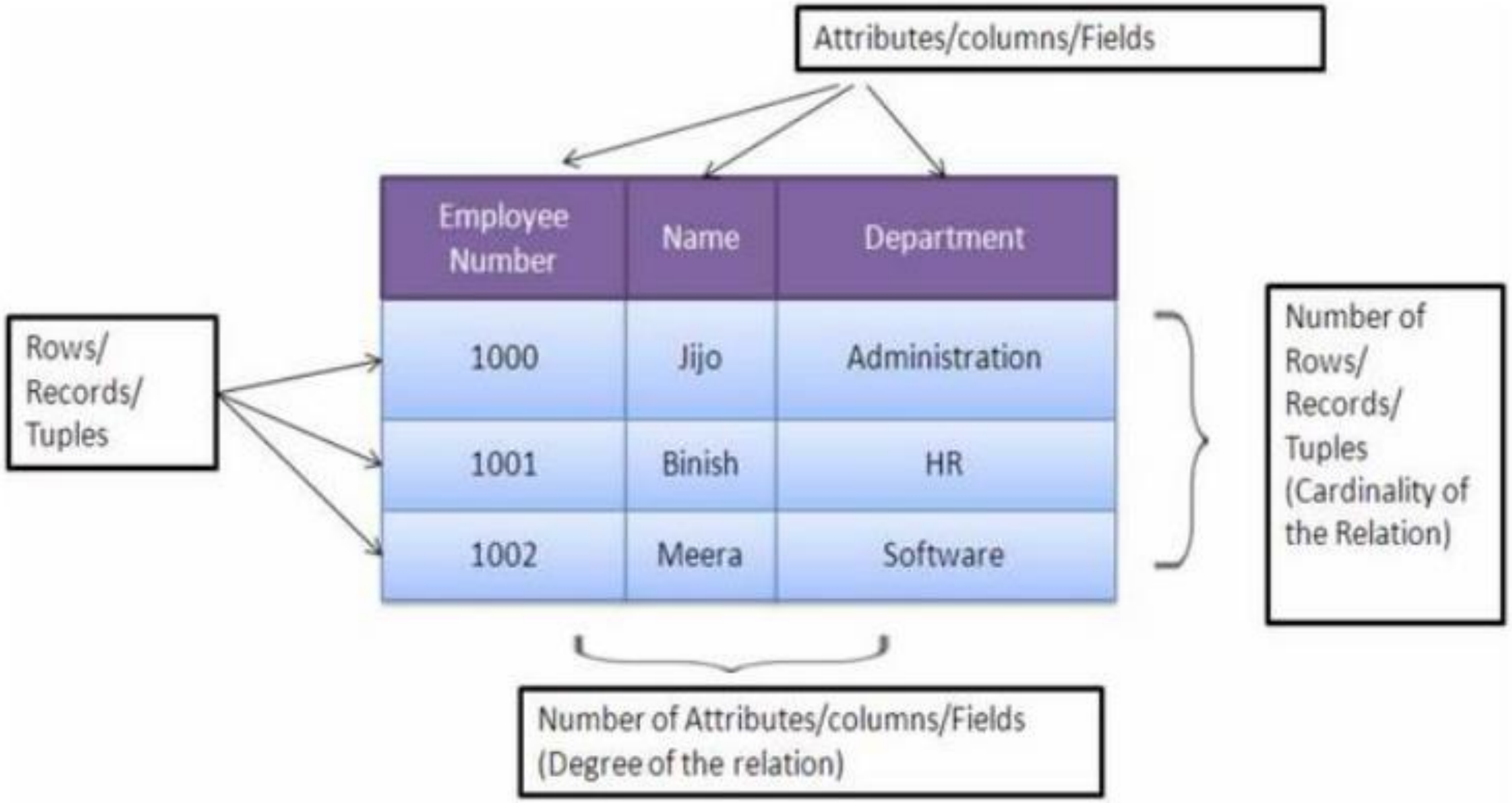
- The *main difference* of the network model from the hierarchical model, is its ability to handle many to many *(M:M)* relations.

- Limitation of Network Model

The only limitation of network model is its ***complexity***.

Relational Data Model

- ✓ In relational data model, data exists in two dimensional tables known as *RELATIONS*.
- ✓ A relation (table) consists of *unique attributes (columns)* and *tuples (rows).*



Employee Number	Name	Department
1000	Jijo	Administration
1001	Binish	HR
1002	Meera	Software

Attributes/columns/Fields
Rows/ Records/ Tuples
Number of Rows/ Records/ Tuples (Cardinality of the Relation)
Number of Attributes/columns/Fields (Degree of the relation)

Object Oriented Model

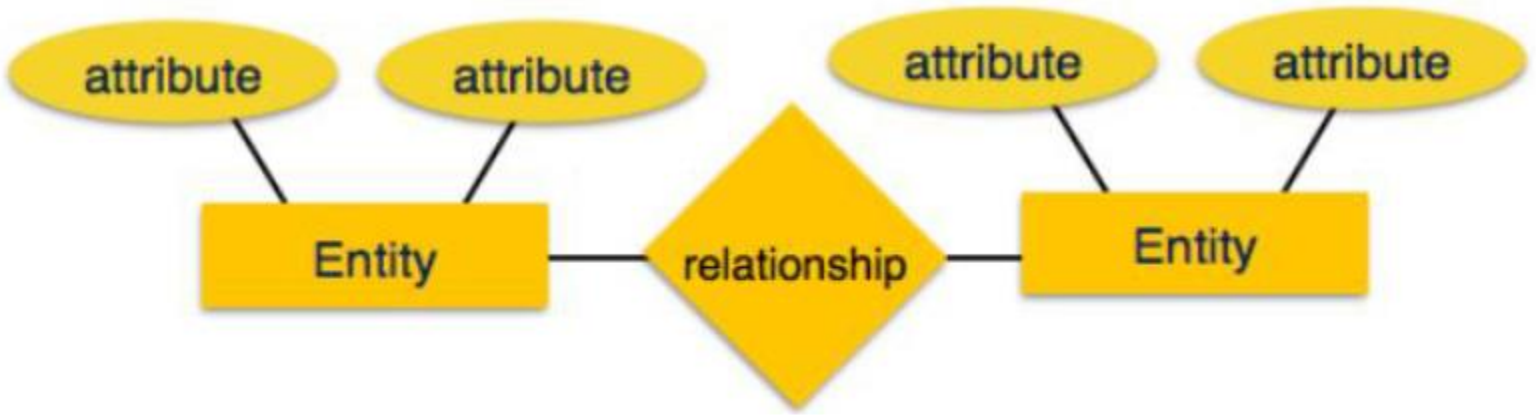
- ✓ Object based data models use concepts such as ***entities***, ***attributes***, and ***relationships***
- ✓ The **entity relational model(E-R model)** has emerged as one of the main techniques for modeling database design.
- ✓ **ER model** is a logical representation of an enterprise data. ER model is a diagrammatic representation of logical structure of database.

E-R Model

- ER model describes relationship among entities and attributes.
- ER diagram is firstly developed by Peter Chen in 1976.
- From the name its clear that, ER Model is based on:

❑ **Entities** and their **attributes**

❑ **Relationships** among entities



E-R Model Components

1) **Entity**: An entity in ER Model is real world entity, which has some properties called ***attributes***. Every attribute is defined by its set of values, called ***domain***.

For example: in a school database, a student is considered as an entity. Student has various attributes like name, age and class etc.

2) **Entity Set** is a collection of similar types of entities.

For example: Students set may contain all the student of a school.

Attributes

- Attributes describe the properties of the entity of which they are associated.
- A particular instance of an attribute is a value. For example, "Ram" is one value of the attribute Name.
- We can classify attributes as following:

i. Simple

ii. Composite

iii. Single-valued

iv. Multi-valued

v. Derived

Types of Attributes

1. **Simple attribute:** It cannot be further subdivided into components.

Example: The roll number of a student, the id number of an employee.

2. **Composite attribute:** It can be splitted into components.

Example: The address can be further splitted into house number, street number, city, state, country and pincode, the name can also be splitted into first name middle name and last name.

3. **Single-valued attribute:** It takes up only a single value for each entity instance.

Example: The age of a student.

4. **Multi-valued attribute:** It takes up more than a single value for each entity.

Example: Phone number of a student: Landline and mobile.

5. **Derived attribute:** It can be derived from other attributes.

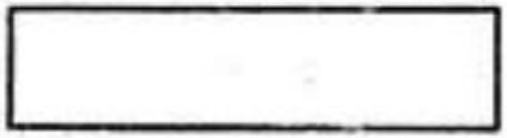
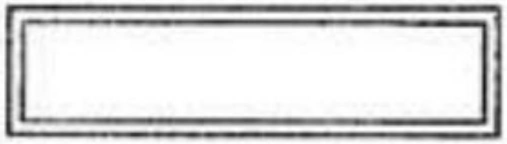
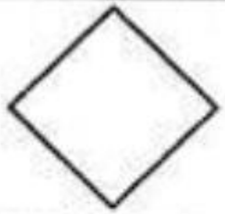
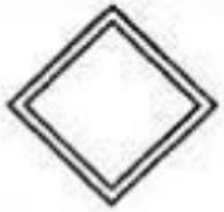
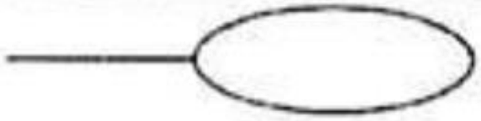
Example: Total and average marks of a student.

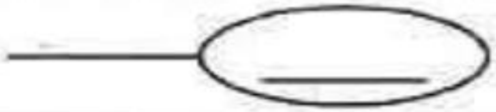
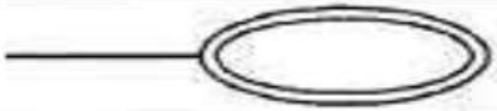
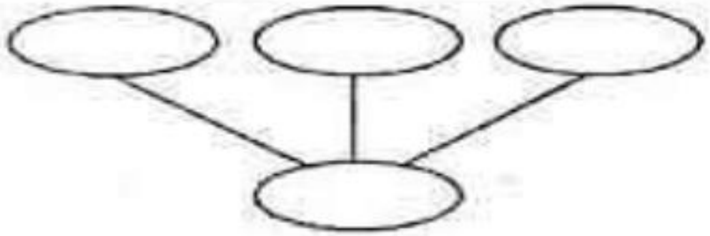
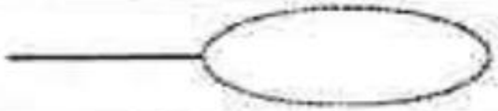
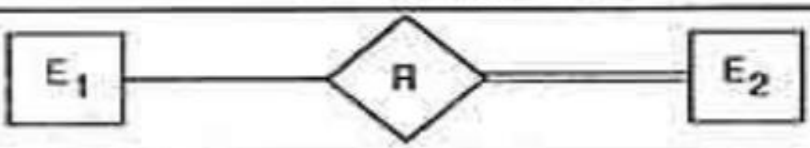
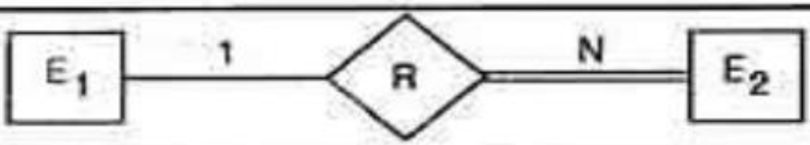
Age (can be derived from DOB).

Key attribute: The attribute which uniquely identifies each entity in the entity set is called key attribute.

For example, Roll_No will be unique for each student.

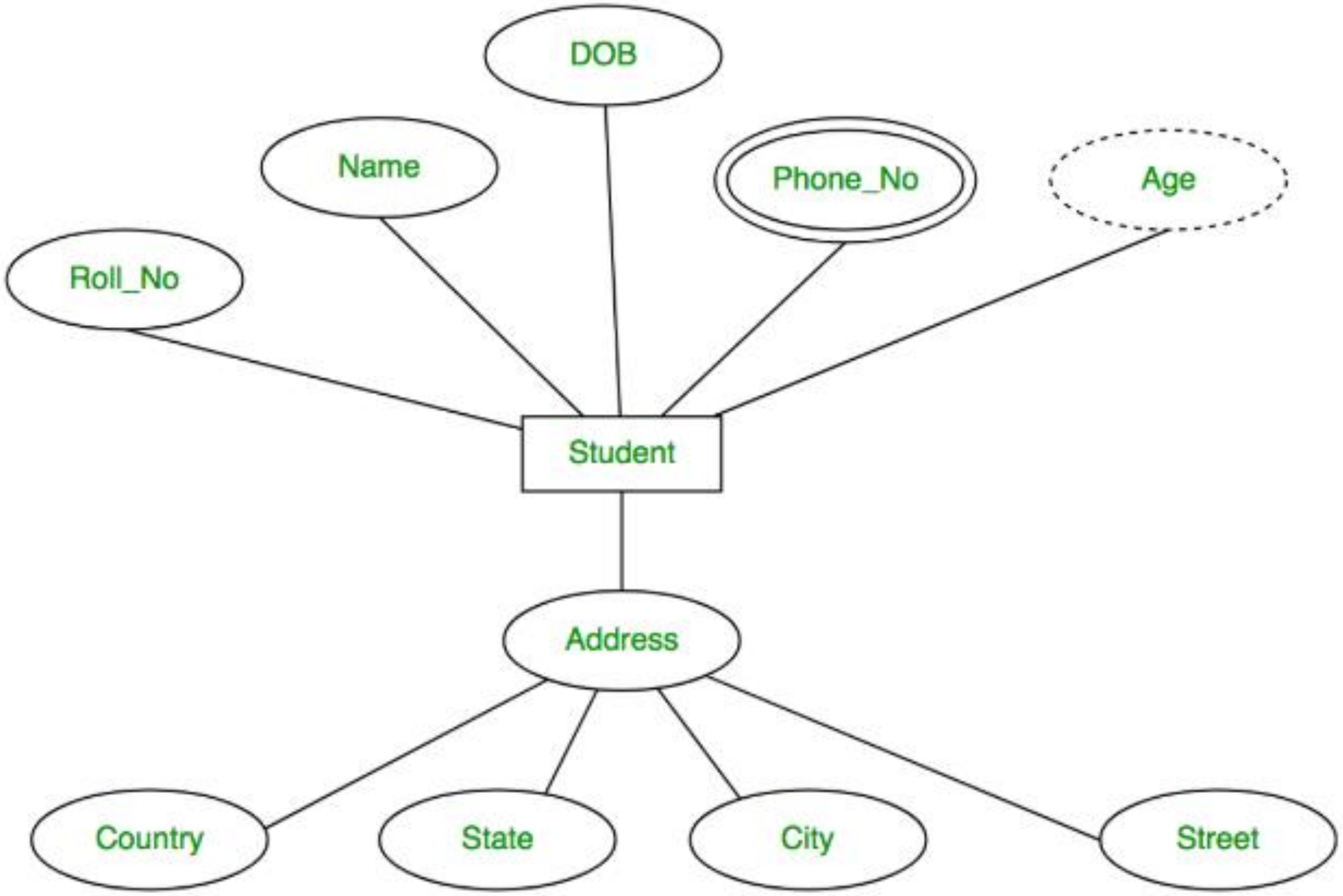
E-R Model:
Symbols and Notations

Symbol	Meaning
	Entity Type
	Weak Entity Type
	Relationship Type
	Identifying Relationship Type
	Attribute

Symbol	Meaning
	Key Attribute
	Multivalued Attribute
	Composite Attribute
	Derived Attribute
	Total Participation of E2 in R
	Cardinality Relation 1 : N for E1 : E2 in R

Example of ER diagram with Entity and Attributes

The complete entity type **Student** with its attributes can be represented as:

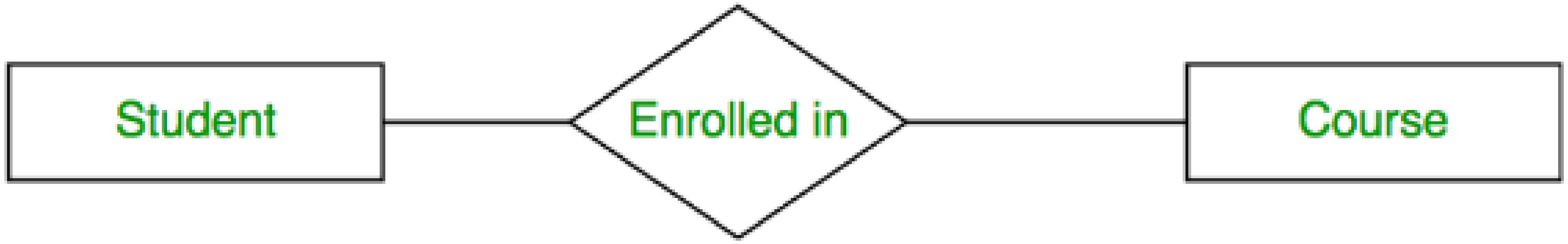


Relationship

- The association among entities is called relationship.
- For example:

a) employee ENTITY has relation ***works_at*** with department.

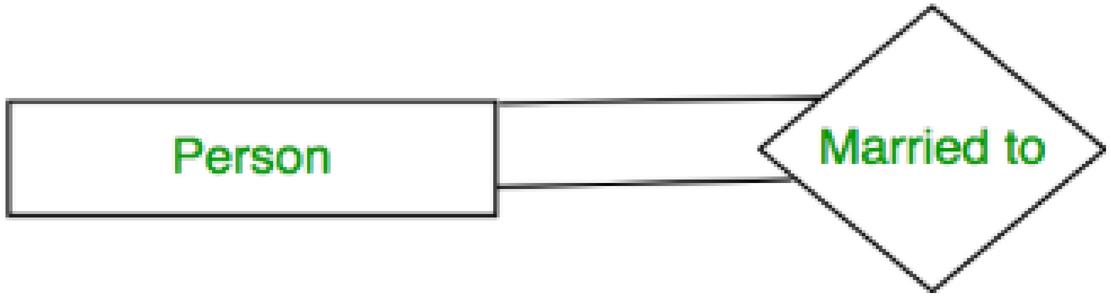
b) ***Enrolled in*** is a relationship type that exists between entity type Student and Course.



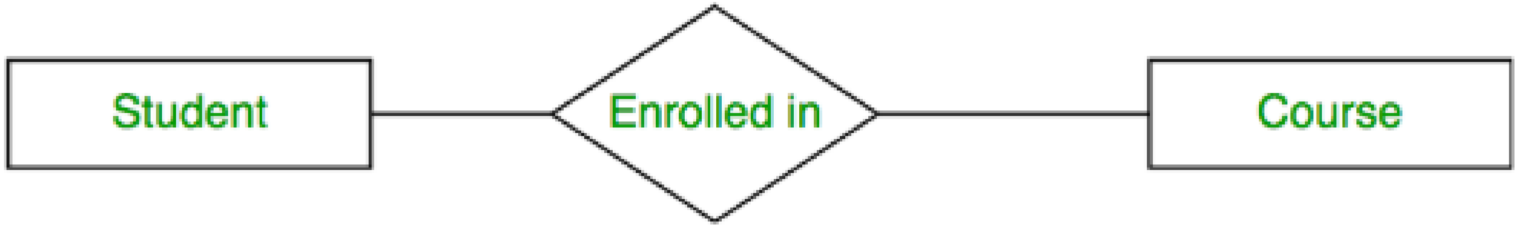
Degree Of A Relationship Set

- The number of different entity sets participating in a relationship set is called as degree of a relationship set.
- ***Types:***

1. **Unary Relationship**: When there is only **ONE** entity set participating in a relation, the relationship is called as unary relationship. For example, one person is married to only one person.



2. **Binary Relationship:** When there are **TWO** entities set participating in a relation, the relationship is called as binary relationship. For example, Student is enrolled in Course.



3. **n-ary Relationship:** When there are **N** entities set participating in a relation, the relationship is called as n-ary relationship.

Constraints

- Relational constraints are the restrictions imposed on the database contents and operations.
- They ensure the correctness of data in the database.
- Various types of constraints are:

i. Mapping cardinalities

ii. Participation constraints

iii. Keys

Participation or Existence Constraint

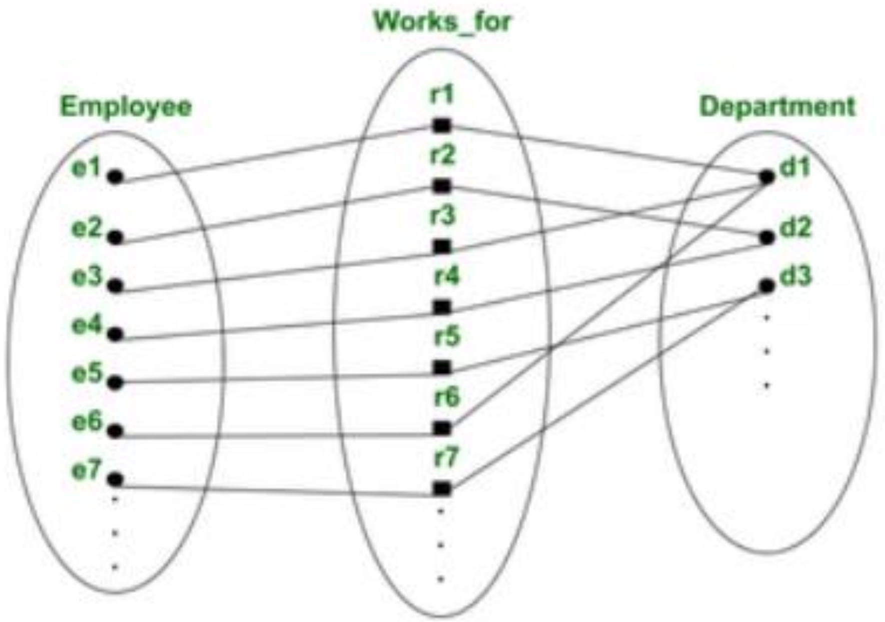
- It represents the minimum number of relationship instances that each entity can participate in and it is also called the *MINIMUM CARDINALITY CONSTRAINT*.

- Types of participation constraints:

a. Total: The participation of an entity set E in a relationship set R is said to be **total** if every entity in E participates in at least one relationship in R.

b. Partial: If only some entities in E participate in relationships in R, the participation is said to be **partial**.

- **Example –**

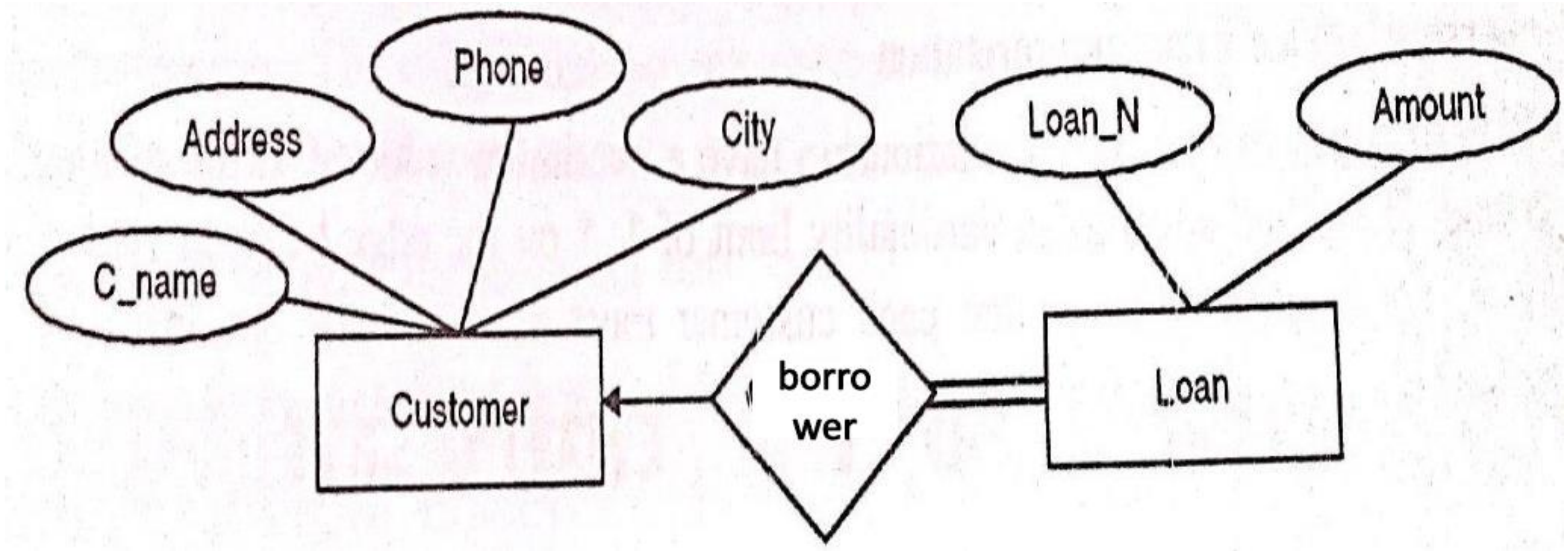


Here (e1, e2,e3…) are instances of entity set Employee and (d1,d2, d3 ….) are the instances of entity type department and (r1, r2,,r3 …..) and (r1, r2, r3 …) are relationship instances of relationship type.

- If the company policy is that every employee should work for a department. Then all the employees in the employee entity set must be related to the department by a works_for relationship. Therefore, the participation of the employee entity type is ***total*** in the relationship type. The total participation is also called ***EXISTENCE DEPENDENCY***.

- And if there is a constraint that a new department need not have employees, then some entity in the employee entity set is not related to the department entity by **works_for** relationship. Therefore, the participation of employee entity in this relationship(works_for) is ***partial***.
- The total participation is represented using a ***double line*** connecting the participating entity type to the relationship and a ***single line*** is used for partial participation.

- Example:



Total participation of Loan entity

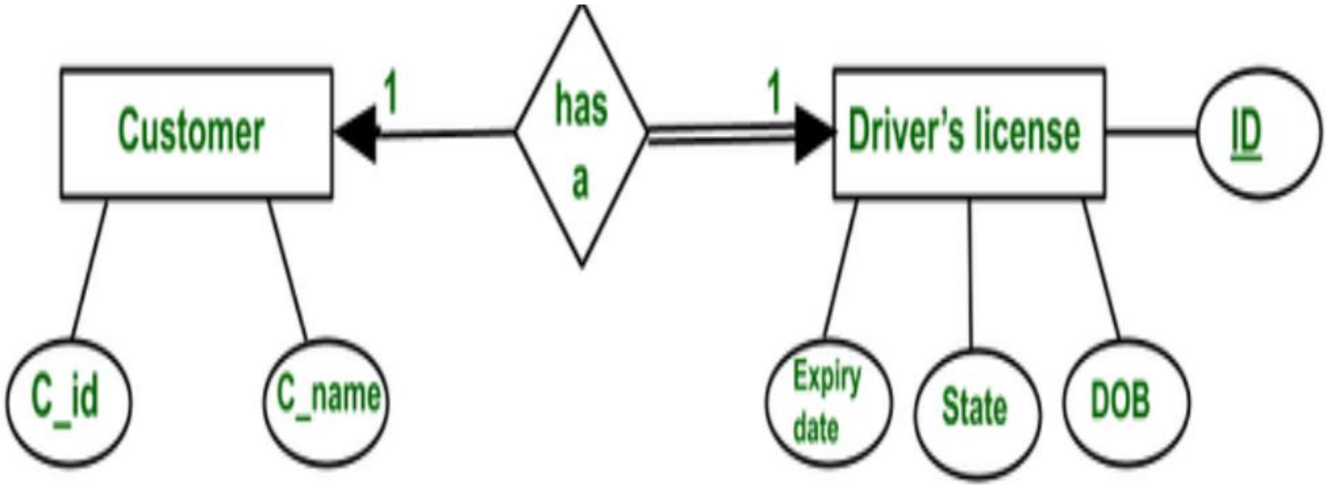
Mapping Cardinalities

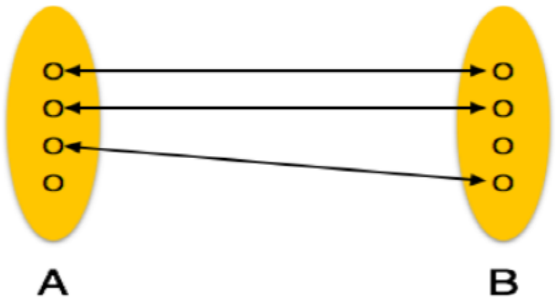
- MAPPING CARDINALITY or CARDINALITY RATIOS , express the number of entities to which another entity can be associated via a relationship set.

- Types of Mapping cardinalities:

a. one to one
b. one to many
c. many to one
d. many to many

I. **One-to-one:** One entity from entity set A can be associated with at most one entity of entity set B and vice versa.

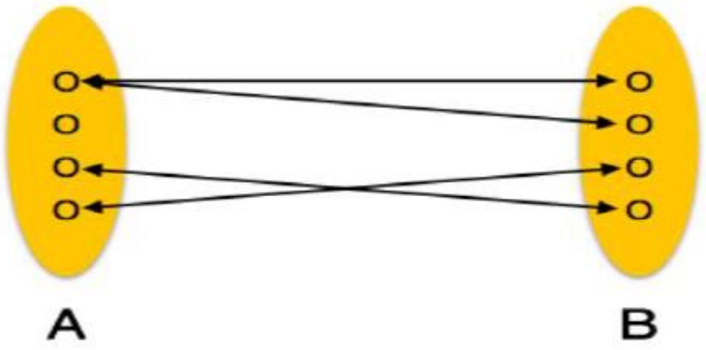


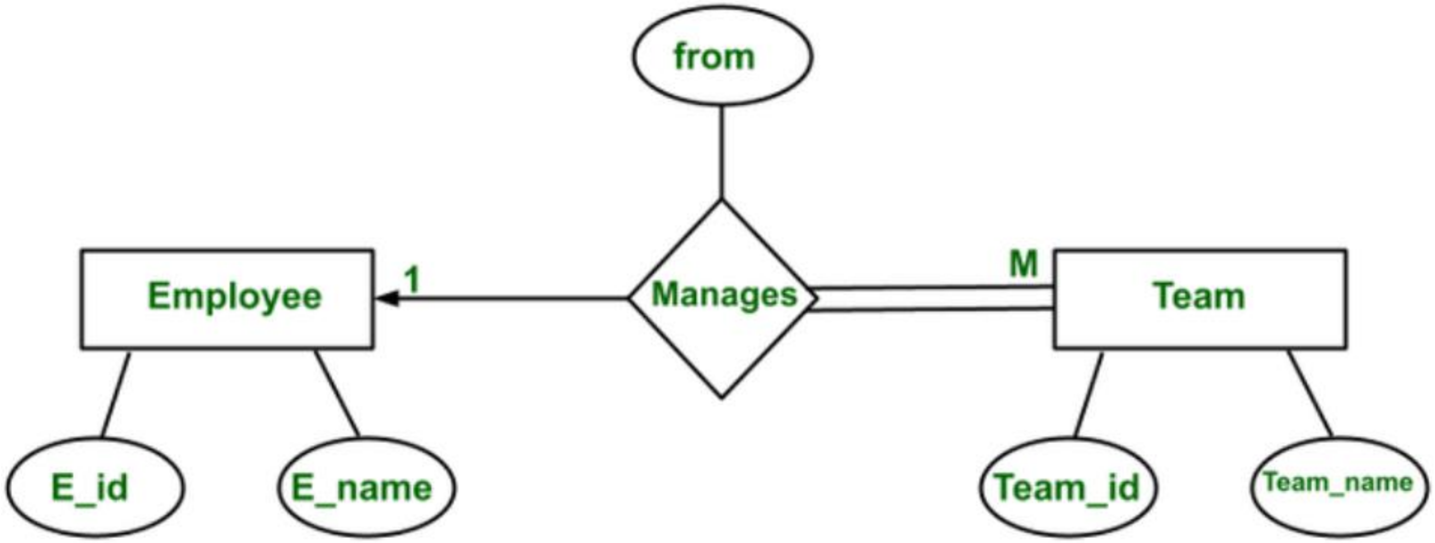


In above example, both entities customer and driving license having an arrow which means the entity Customer is participating in the relation "has a" in a one-to-one fashion. It could be read as '*Each customer has exactly one driving license and every driving license is associated with exactly one customer*'.

There may be customers who do not have a credit card, but every credit card is associated with exactly one customer. Therefore, ***the entity customer has a TOTAL participation in a relation***.

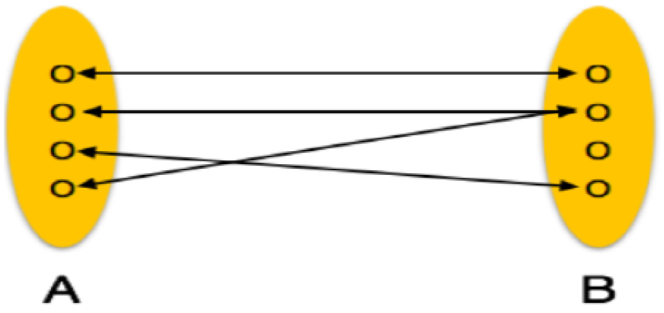
II. **One-to-many(1:M)**: One entity from entity set A can be associated more than one entities of entity set B but from entity set B one entity can be associated with at most one entity.

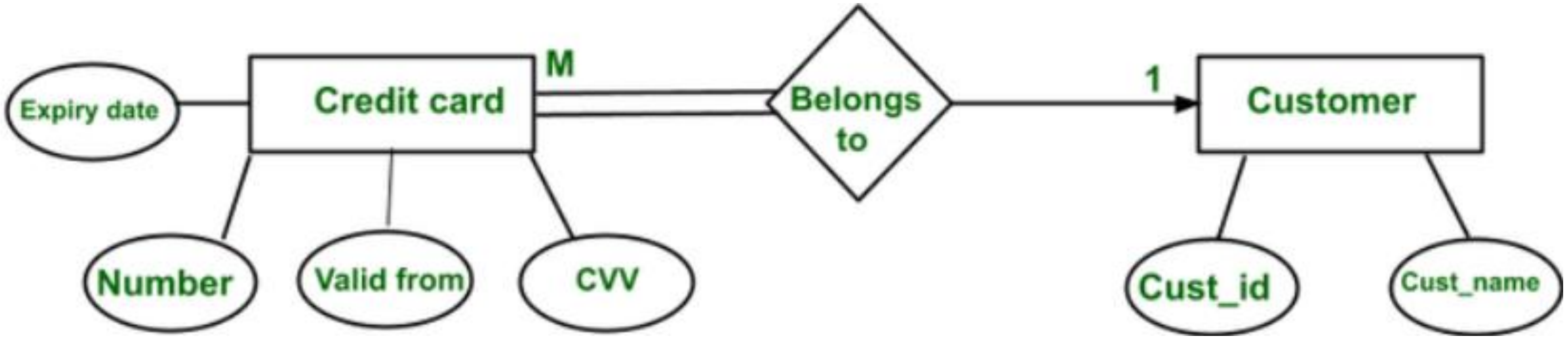




There are some employees who manage more than one team while there is only one manager to manage a team.

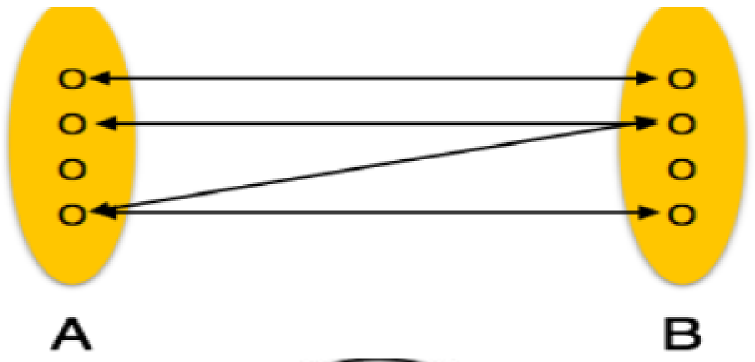
III. Many to one relationship (M:1) : More than one entities from entity set A can be associated with at most one entity of entity set B but one entity from entity set B can be associated with more than one entity from entity set A.

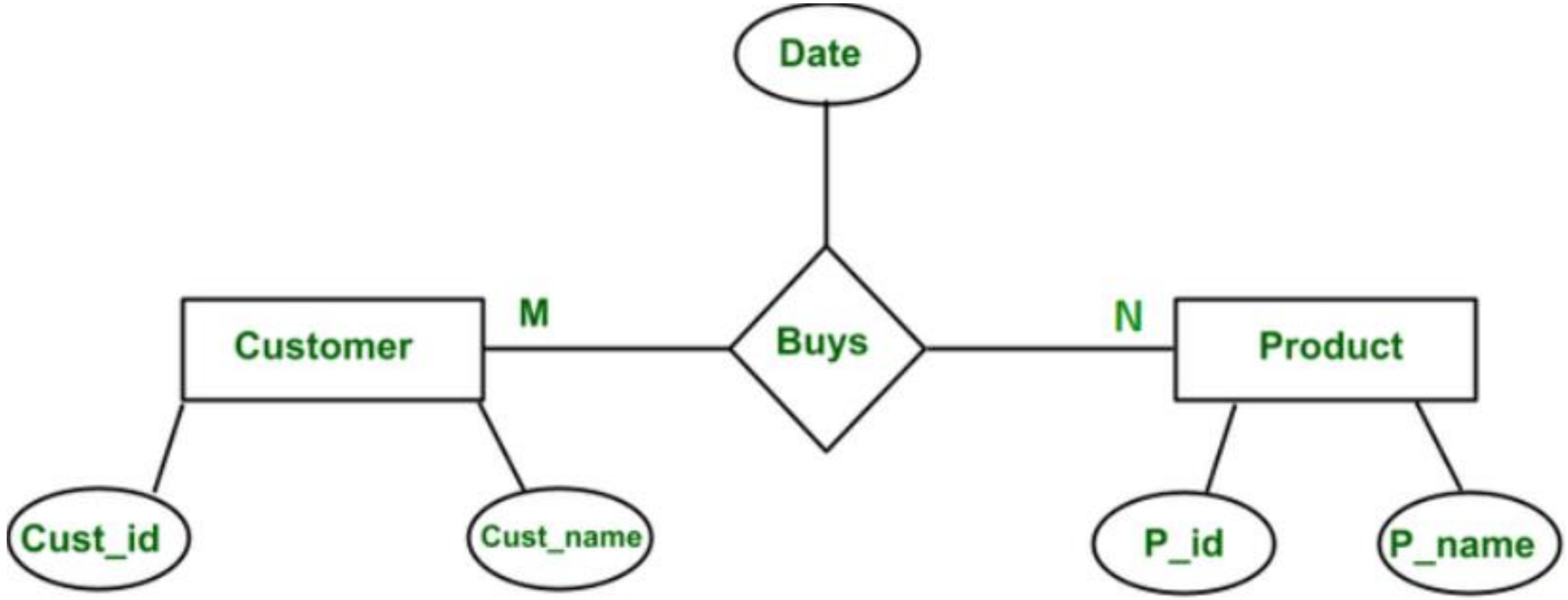




It is related to a one-to-many relationship but the difference is due to perspective . *Any number of credit cards can belong to a customer and there might be some customers who do not have any credit card, but every credit card in a system has to be associated with an employee(i.e. total participation). While a single credit card can not belong to multiple customers.*

IV. **Many to many relationship (M:N) :** One entity from A can be associated with more than one entity from B and vice versa.

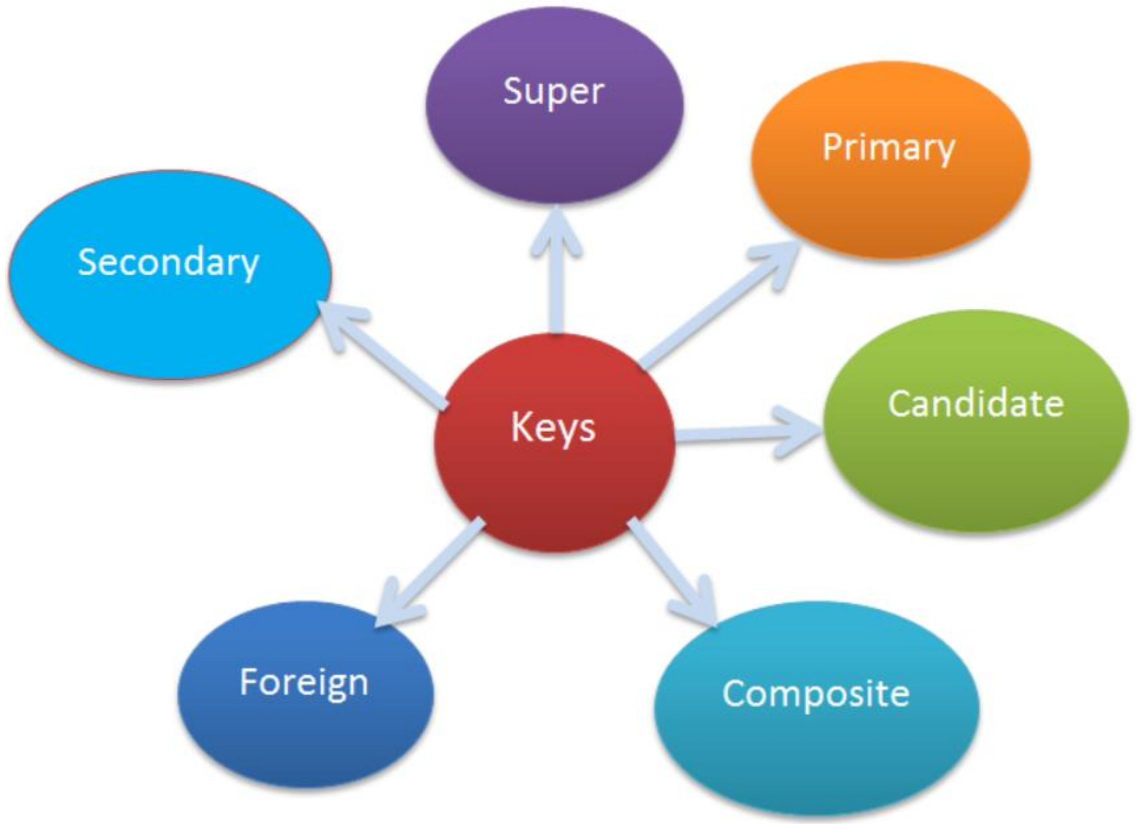




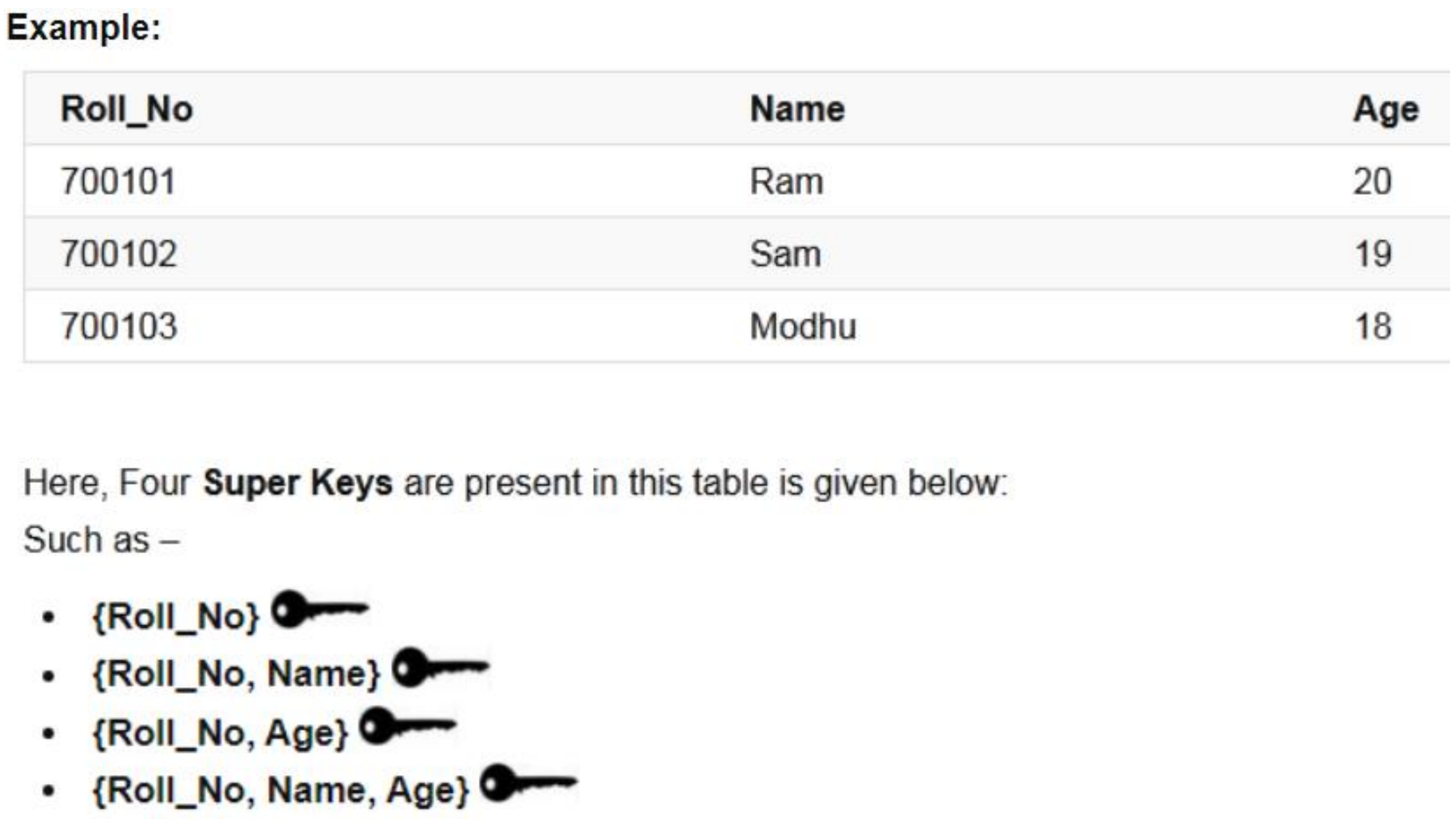
A customer can buy any number of products and a product can be bought by many customers.

Keys

- It is defined as the column or attribute of the database table.
- They are used to establish and identify relation between tables.
- They also ensure that each record within a table can be uniquely identified by combination of one or more fields within a table.
- *Types of keys*:



1. Super key: It is a group of single or multiple keys which identifies rows in a table. A Super key may have additional attributes that are not needed for unique identification.



Example:

Roll_No	Name	Age
700101	Ram	20
700102	Sam	19
700103	Modhu	18

Here, Four **Super Keys** are present in this table is given below:

Such as –

- **{Roll_No}**
- **{Roll_No, Name}**
- **{Roll_No, Age}**
- **{Roll_No, Name, Age}**

2. Primary Key: It is a column or group of columns in a table that uniquely identify every row in that table. The Primary Key ***can't be a duplicate,*** meaning the same value can't appear more than once in the table. A table cannot have more than one primary key.

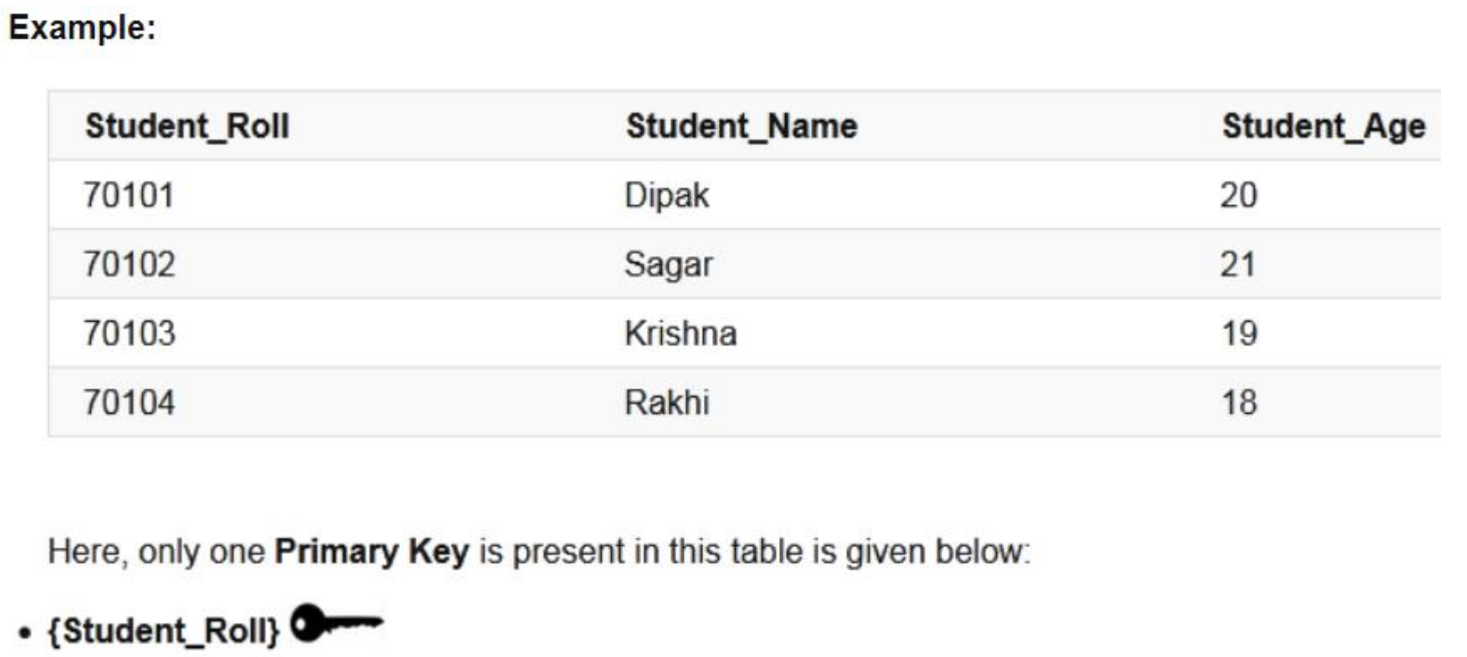
❖ **Rules for defining Primary key:**

✓ *Two rows can't have the same primary key value*

✓ *It must for every row to have a primary key value.*

✓ *The primary key field cannot be null.*

✓ *The value in a primary key column can never be modified or updated if any foreign key refers to that primary key.*



Example:

Student_Roll	Student_Name	Student_Age
70101	Dipak	20
70102	Sagar	21
70103	Krishna	19
70104	Rakhi	18

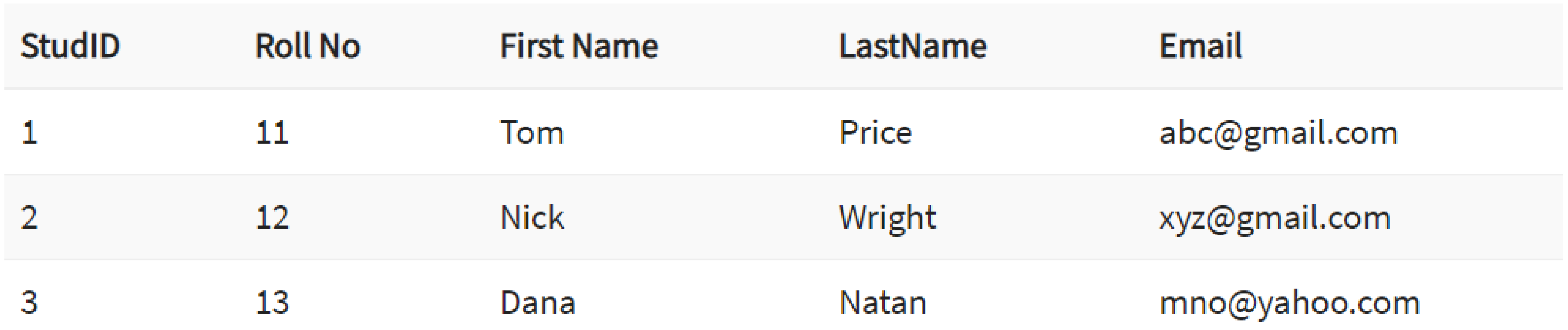
Here, only one **Primary Key** is present in this table is given below:

- **{Student_Roll}**

3. Candidate Key: It is a set of attributes that uniquely identify tuples(row) in a table. Candidate Key is a super key with no repeated attributes. The Primary key should be selected from the candidate keys. Every table must have at least a single candidate key. A table can have multiple candidate keys but only a single primary key.

❖ **Properties of Candidate key:**

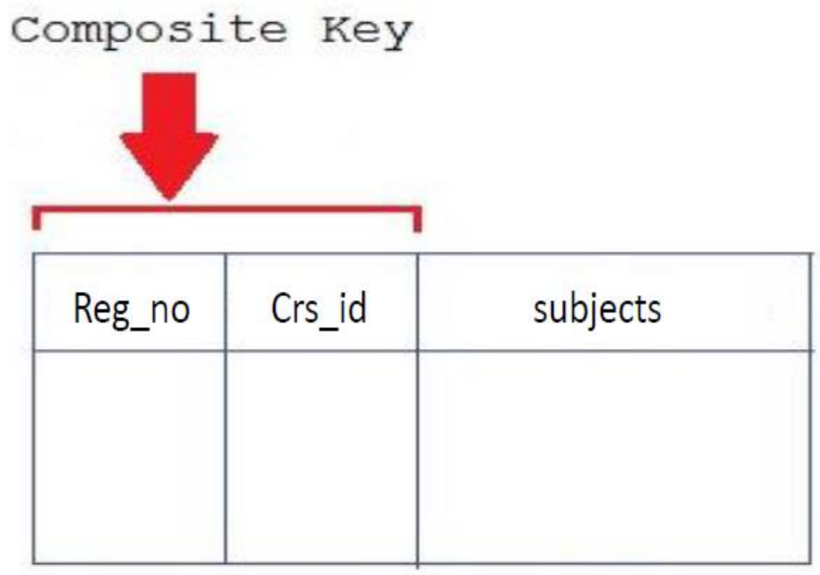
- ✓ *It must contain unique values*
- ✓ *Candidate key in SQL may have multiple attributes*
- ✓ *Must not contain null values*
- ✓ *It should contain minimum fields to ensure uniqueness*
- ✓ *Uniquely identify each record in a table*



StudID	Roll No	First Name	LastName	Email
1	11	Tom	Price	abc@gmail.com
2	12	Nick	Wright	xyz@gmail.com
3	13	Dana	Natan	mno@yahoo.com

In the given table Stud ID, Roll No, and email are candidate keys which help us to uniquely identify the student record in the table.

4. Composite Key: It is a group of fields(rows) that it combines to uniquely identify a record. When we design a database, we will have tables that will use more than one column as a part of the primary key. So, it is called a *composite key* or *concatenated key*.



5. Foreign key: It is a column that creates a relationship between two tables. The purpose of Foreign keys is to maintain data integrity and allow navigation between two different instances of an entity. It acts as a cross-reference between two tables as it references the primary key of another table.

Example:

DeptCode	DeptName
001	Science
002	English
005	Computer

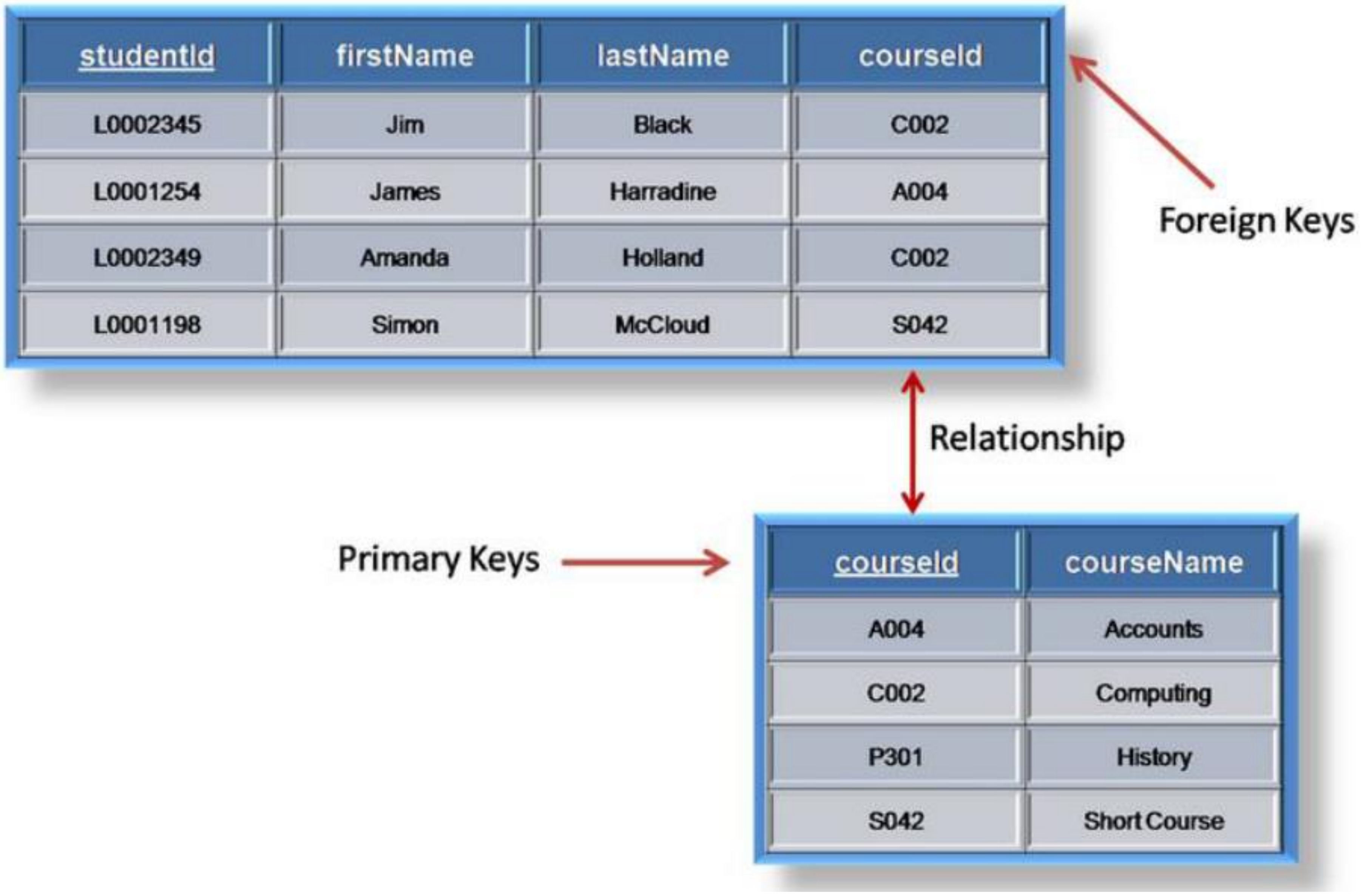
Teacher ID	Fname	Lname
B002	David	Warner
B017	Sara	Joseph
B009	Mike	Brunton

We have two table, **teacher** and **department** in a school. However, there is no way to see which search work in which department. In this table, adding the foreign key in Deptcode to the Teacher name, we can create a relationship between the two tables.

Teacher ID	DeptCode	Fname	Lname
B002	002	David	Warner
B017	002	Sara	Joseph
B009	001	Mike	Brunton

This concept is also known as ***Referential Integrity.***

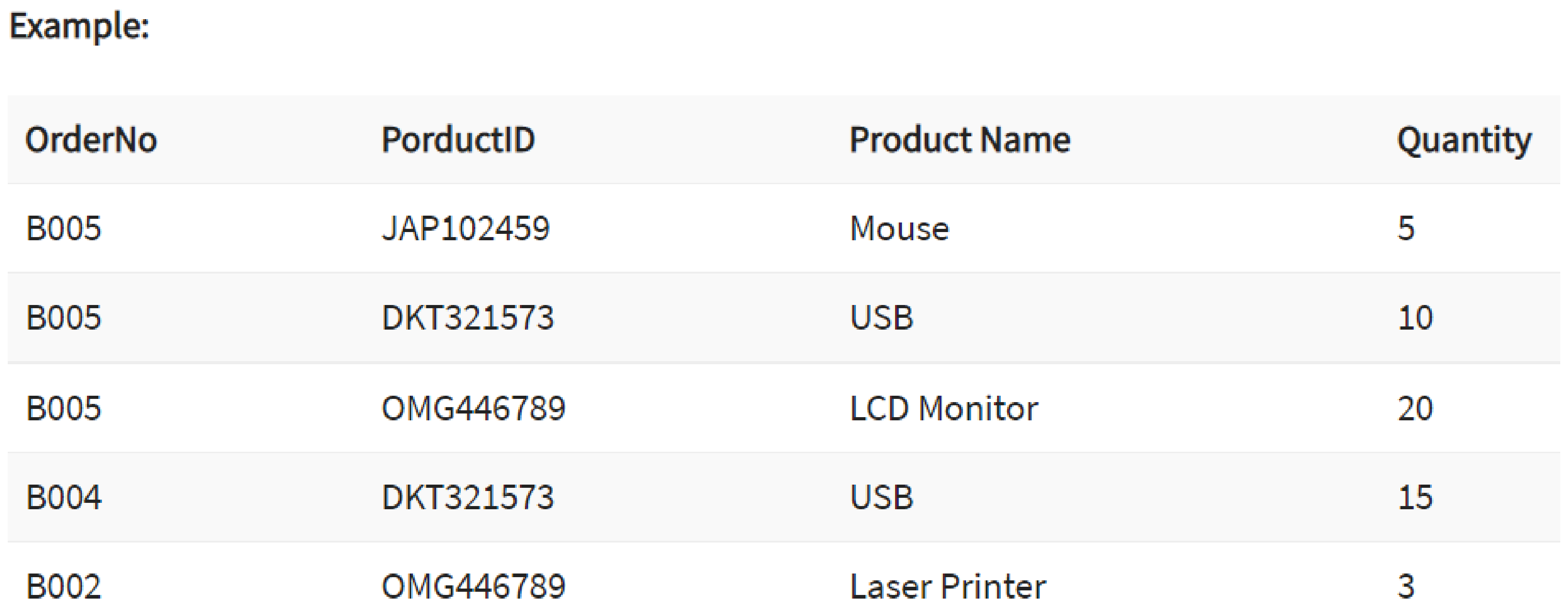
- Example:



studentId	firstName	lastName	courseId
L0002345	Jim	Black	C002
L0001254	James	Harradine	A004
L0002349	Amanda	Holland	C002
L0001198	Simon	McCloud	S042

courseId	courseName
A004	Accounts
C002	Computing
P301	History
S042	Short Course

6. Compound key: It has two or more attributes that allow you to uniquely recognize a specific record. It is possible that each column may not be unique by itself within the database. However, when combined with the other column or columns the combination of composite keys become unique. The purpose of the compound key in database is to uniquely identify each record in the table.



Example:

OrderNo	PorductID	Product Name	Quantity
B005	JAP102459	Mouse	5
B005	DKT321573	USB	10
B005	OMG446789	LCD Monitor	20
B004	DKT321573	USB	15
B002	OMG446789	Laser Printer	3

OrderNo and ProductID *can't be a primary key* as it does not uniquely identify a record. However, *a compound key of* OrderNo *and* ProductID could be used as it uniquely identified each record.

7. Secondary Key: It is an attribute or combination of attributes that may not be a candidate key but classifies the entity set on a particular characteristic. A table can have multiple choices for a primary key but the only one can be set as the primary key. All the keys which are not the primary key that is called ***Secondary Key***.

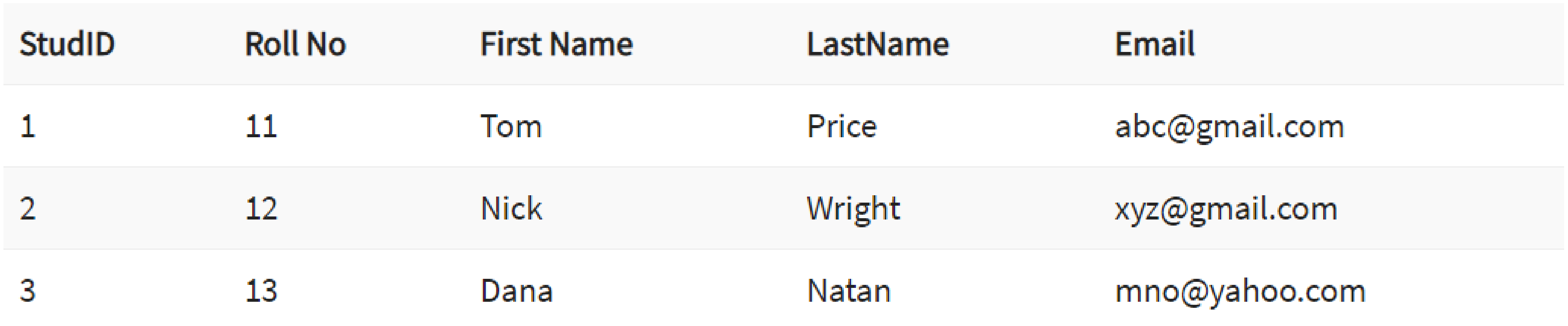
Example:

Student_id	Roll_no	Name	Stream	email_id
025	71875	Jack	EE	jack4u@gmail.com
050	71874	Rohit	CSE	rohitsk@outlook.com
055	71834	Nataly	ETCE	natasa20@hotmail.com

In this table, Student_id, Roll_no, email_id are qualified to become a primary key. But since Student_id is only the primary key. Here **Two Secondary Keys** are present in this table is given below:

- Roll_no
- email_id

8. Alternate key: It is a column or group of columns in a table that uniquely identify every row in that table. A table can have multiple choices for a primary key but only one can be set as the primary key. *All the keys which are not primary key are called an **Alternate Key***.



StudID	Roll No	First Name	LastName	Email
1	11	Tom	Price	abc@gmail.com
2	12	Nick	Wright	xyz@gmail.com
3	13	Dana	Natan	mno@yahoo.com

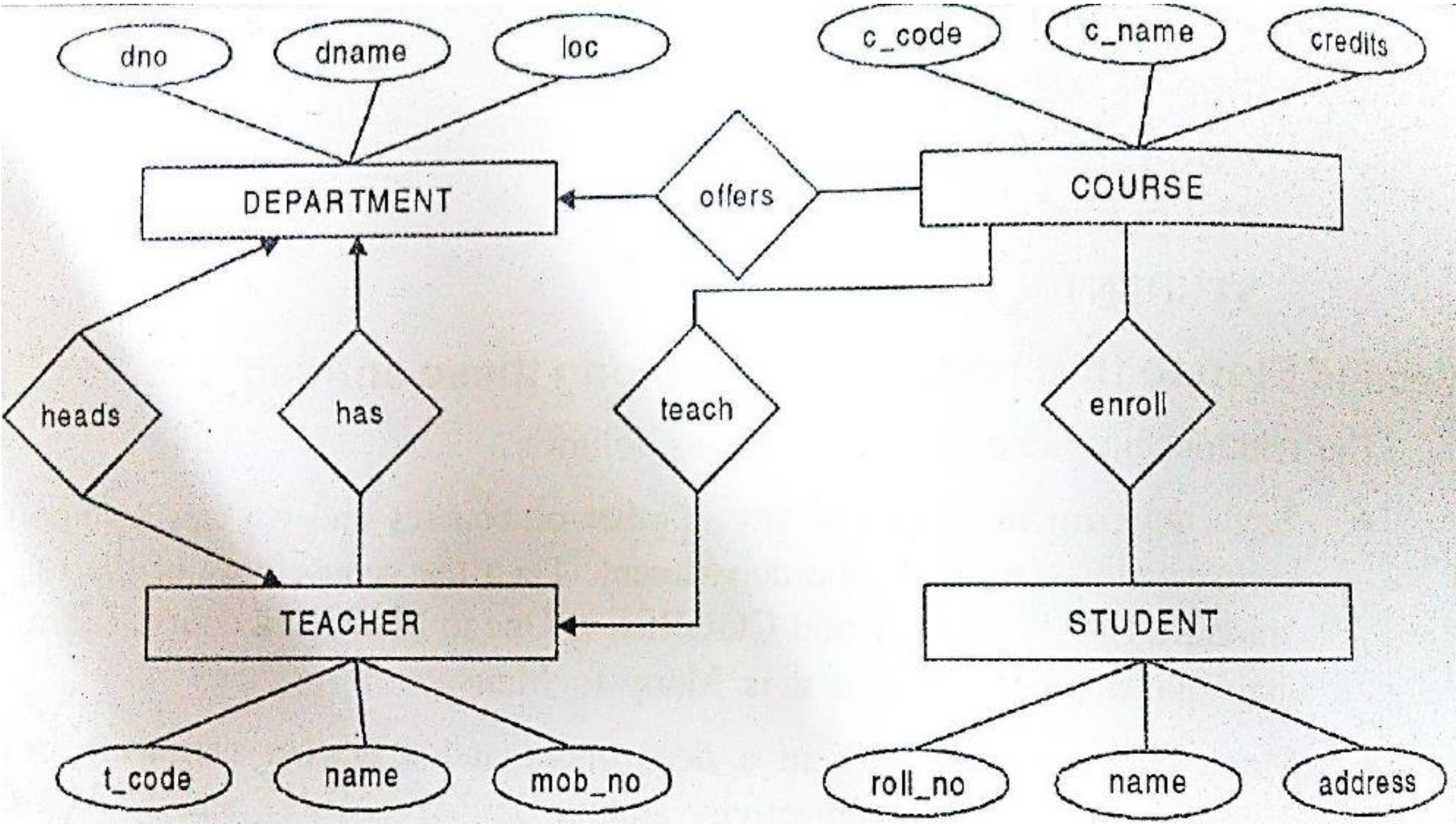
StudID, RollNo, Email are qualified to become a primary key. But since StudID is the primary key, Roll No, Email becomes the alternative key.

Steps in designing E-R Diagram

There are following steps:

i. Identify the entities from the requirement sheets
ii. Find relationships among those entities
iii. Identify the key attributes for every entity
iv. Identify other relevant attributes
v. Draw complete E-R diagram with all attributes including Primary key
vi. Review your results with your Business users

Example: University Management System



Strong and Weak entity sets

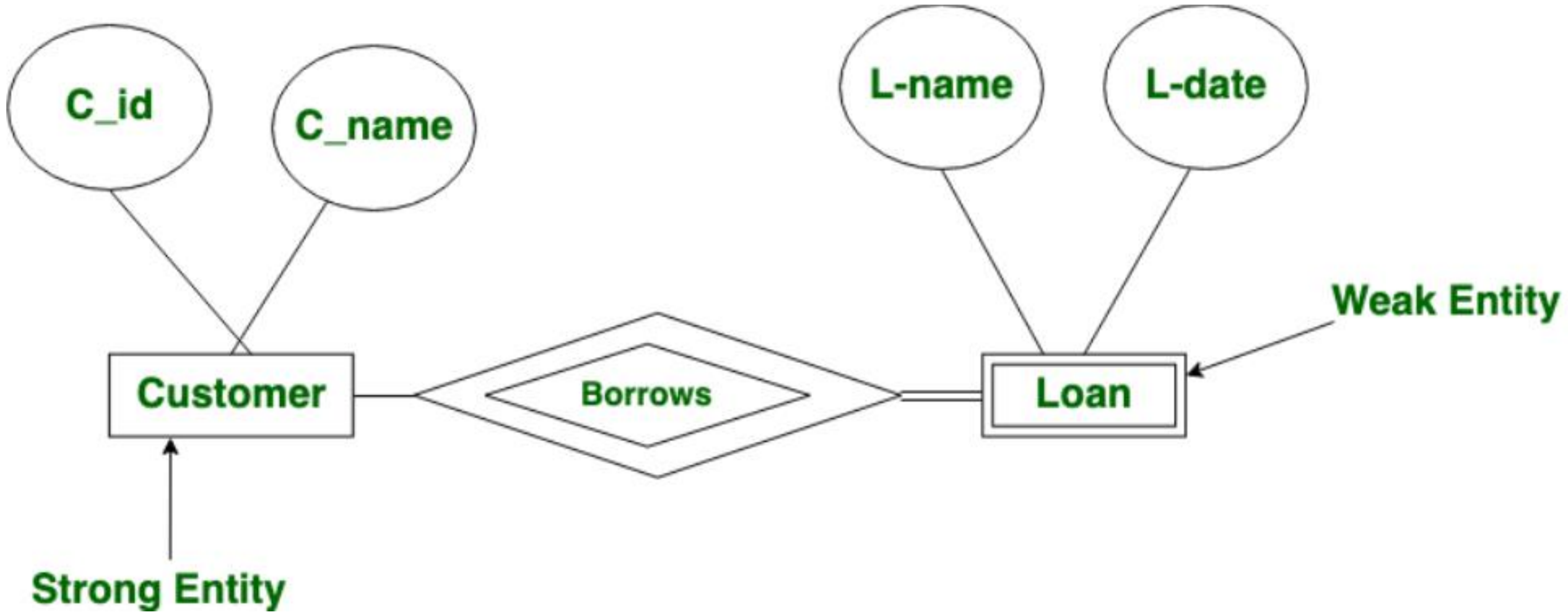
STRONG ENTITY

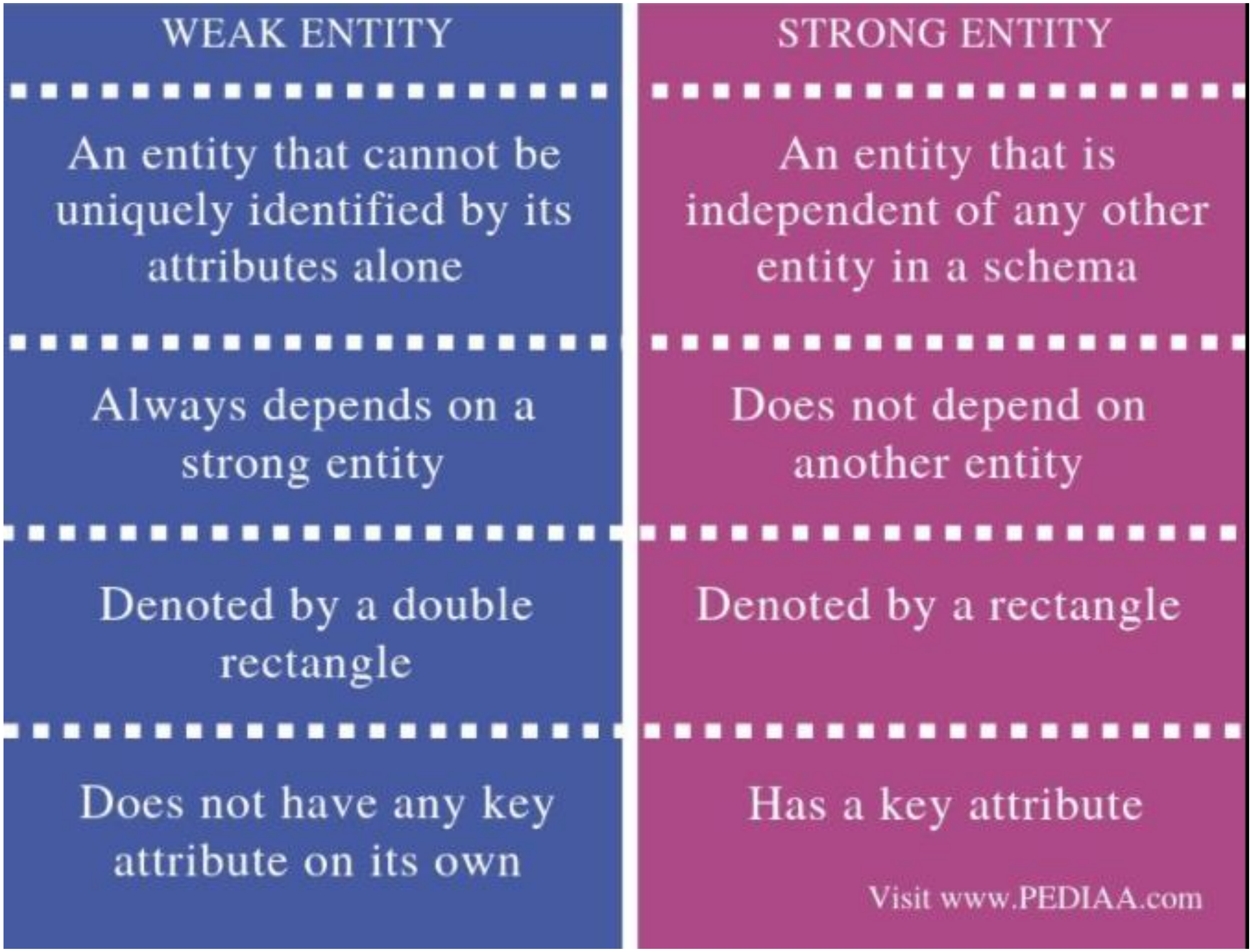
- It is not dependent on any other entity in the schema. A strong entity will **always have** a primary key. Strong entities are represented by a *single rectangle*. The **relationship** of two strong entities is represented by a *single diamond*.
- Various strong entities, when combined together, create a strong entity set.

WEAK ENTITY

- It is dependent on a strong entity to ensure its existence. Unlike a strong entity, a weak entity **does not have** any primary key. It instead has a partial discriminator key. A weak entity is represented by a *double rectangle.*
- The **relation** between one strong and one weak entity is represented by a *double diamond.*

- Example:


C_id
C_name
L-name
L-date
Weak Entity
Customer
Borrows
Loan
Strong Entity



WEAK ENTITY	STRONG ENTITY
An entity that cannot be uniquely identified by its attributes alone	An entity that is independent of any other entity in a schema
Always depends on a strong entity	Does not depend on another entity
Denoted by a double rectangle	Denoted by a rectangle
Does not have any key attribute on its own	Has a key attribute

Visit www.PEDIAA.com