

# Data Models

# Entity-Relationship Model

An ER Model is a high level description of the data and the relationships among the data, rather how data is stored. Some features of E-R Model are:

1. The E-R diagram used for representing E-R Model can be easily converted into relations(tables) in relational model.
2. The E-R Model is used for the purpose of good database design by the database developer so as to use that data model in various DBMA.
3. It is a top-down approach to database design.
4. It gives the precise understanding of the nature of data and how it is used by an enterprise.
5. It is very simple, non technical, free of ambiguities and easy to understand by various types of users, programmers and designers because a specific standards are used for their representation.

# E-R Diagram

- Entity relationship model defines the conceptual view of database.
- It works around real world entity and association among them.

# E-R Model Terminology

## ENTITY

- A real-world thing either animate or inanimate that can be easily identifiable and distinguishable.
- Objects about which information can be stored.
  - For example, in a school database, student, teachers, class and course offered can be considered as entities.
- **Properties of Entity :**
  1. Entity is represented by set of properties called attributes.
  2. Entity is atomic and cannot be broken down into smaller pieces.
  3. Entity is an instance of entity type so it is represented in E-R diagrams a rectangular box enclosing its name.

## ENTITY TYPE

- An entity type is a collection of entities that have the same attributes but different values.
- **Properties of Entity Type:**
  1. An entity is an instance of entity type.
  2. An entity type is identified by its name and properties.
  3. An entity type is represented by ER diagram as a rectangular box enclosing the entity type.

## **ENTITY SET**

An entity set is a collection of all instances of a particular entity type at any point of time in the database. Each entity set is referred by its name and attribute values. Its name is same as that of entity type.

## **ATTRIBUTES**

Entities are represented by means of their properties, called attributes.

- For example, a student entity may have name, class, age as attributes.

### **Properties of Attributes**

1. The attributes of an entity should be unique.
2. The attributes should uniquely identify the entity.
3. The set of permitted values for each attribute is known as its domain from where its values are chosen.
4. An attribute of an entity is represented in ER diagram as an ellipse attached to a relevant entity by a line and labelled with entity name.

# Types of attributes:

- **Simple attribute:**

Simple attributes are atomic values, which cannot be divided further.

- For example, student's phone-number is an atomic value of 10 digits.

- **Composite attribute:**

Composite attributes are made of more than one simple attribute.

- For example, a student's complete name may have first\_name and last\_name.

- **Derived attribute:**

Derived attributes are attributes, which do not exist physical in the database, but there values are derived from other attributes presented in the database.

- For example, average\_salary in a department should be saved in database instead it can be derived.
- For another example, age can be derived from data\_of\_birth.

- **Single-valued attribute:**


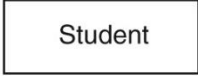
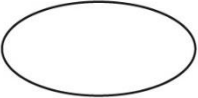
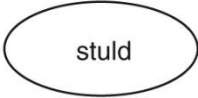
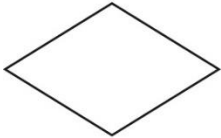
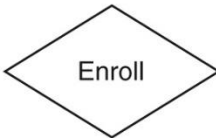

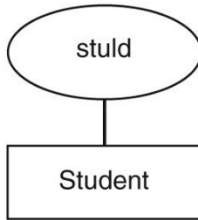
Single valued attributes is the one that has only a single value for a particular entity.

- For example: RollNo.

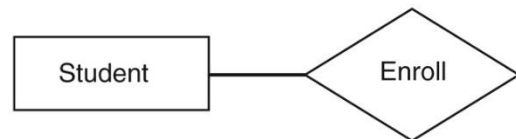
- **Multi-value attribute:**

Multi-value attribute is an attribute that holds multiple values for a particular single entity.

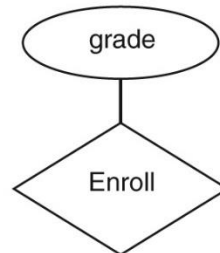
- For example, a person can have more than one phone numbers, email\_addresses etc.









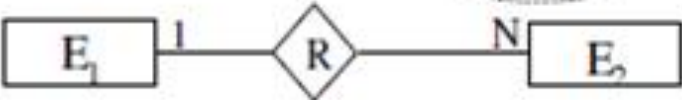
SYMBOL	NAME	MEANING	EXAMPLE
	Rectangle	Entity Set	
	Oval	Attribute	
	Diamond	Relationship	
	Line	Links: Attribute to Entity	

Entity Set to  
Relationship



Attribute to  
Relationship



Symbol	Meaning
	ENTITY TYPE
	WEAK ENTITY TYPE
	RELATIONSHIP TYPE
	ATTRIBUTE
	KEY ATTRIBUTE
	MULTIVALUED ATTRIBUTE
	COMPOSITE ATTRIBUTE
	DERIVED ATTRIBUTE
	CARDINALITY RATIO 1:N



# Relationship

- The association between two or more entities is called relationship.
  - For example, employee entity has relation works\_at with department
- **Relationship Set:**
- Relationship of similar type is called relationship set.
  
- **DEGREE OF RELATIONSHIP**
- The number of participating entities in a relationship defines the degree of the relationship **or** Total number of attributes or columns of a relation is known as the degree of the relation.
- **Unary** – When association exists within a single entity type. The degree of unary relationship is 1. It is sometimes called as Recursive Relationship.
- **Binary** - When association exists between two entity types. The degree of binary relationship is 2.
- **Ternary** – When association exists among three entity types. The degree of ternary relationship is 3. It is rarely used in real world. If it exists, it should be decomposed into one or more binary relationships.
- **Quaternary** - When association exists among four entity types. The degree of Quaternary relationship is 4.

# Connectivity of Relationship

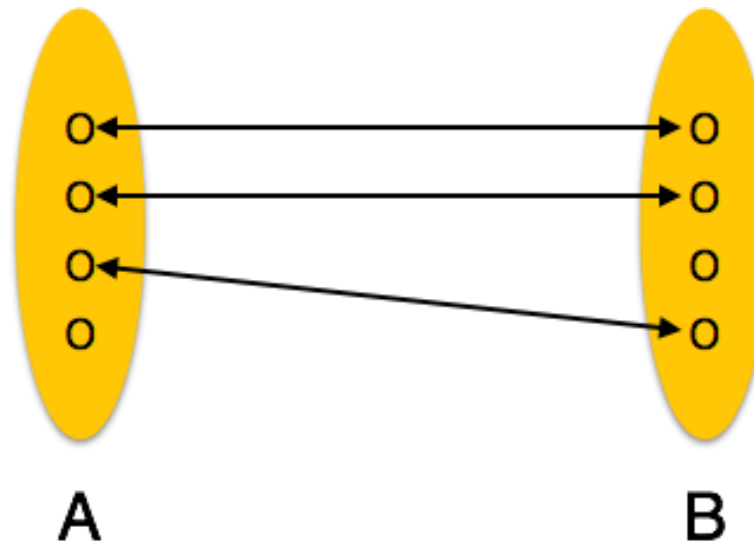
- Connectivity of relationship means how many instances of one entity are associated with how many instances of other entity in a relationship. It describes the mapping of associated entity instances in the relationship.

# Mapping Cardinalities:

- **Cardinality** defines the number of entities in one entity set which can be associated to the number of entities of other set via relationship set.

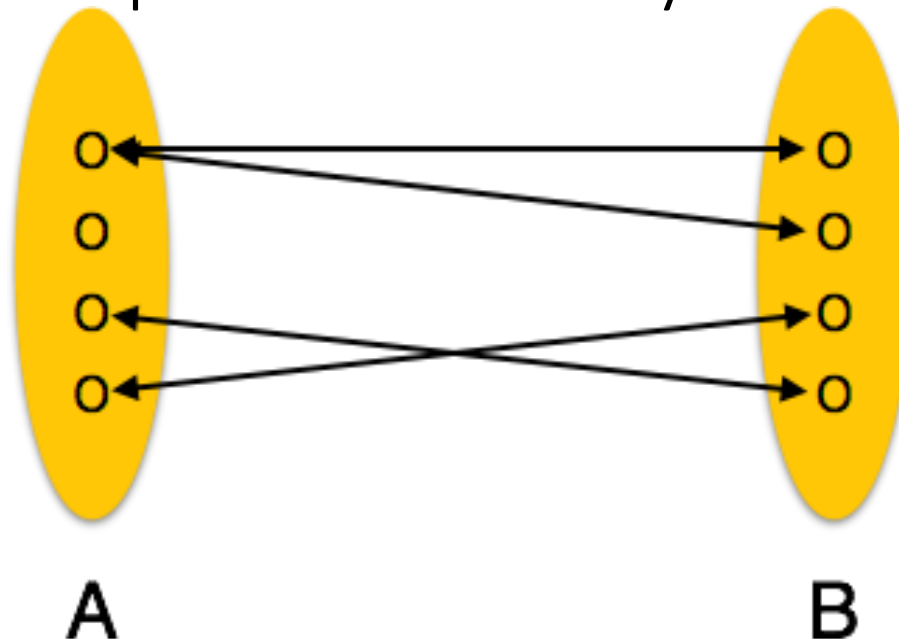
- **One-to-one(1:1):** one entity from entity set A can be associated with at most one entity of entity set B and vice versa. 1:1 relationship occurs when an instance of entity type A is associated with exactly one instance of entity type B.

Eg- No student can do the same project and no student can do more than 1 project.



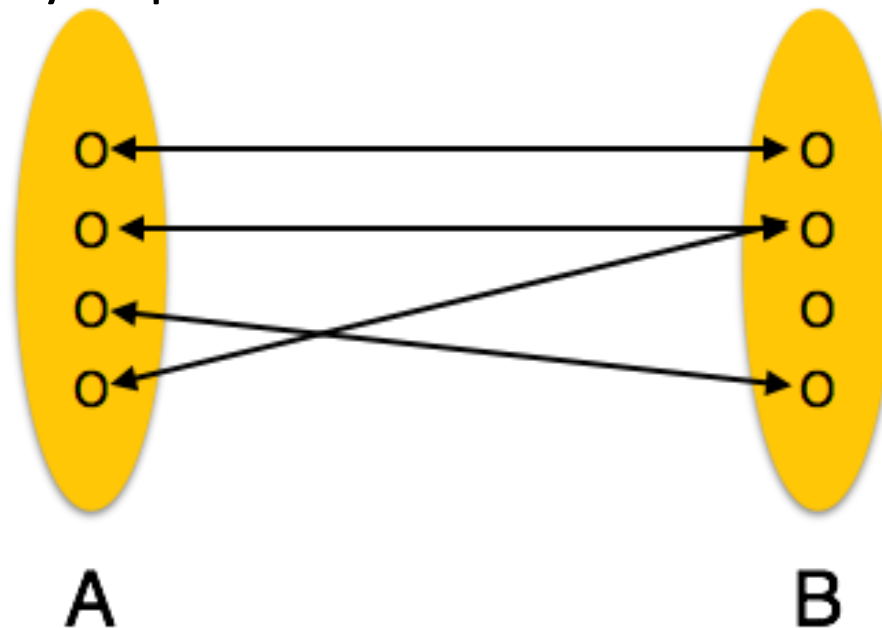
- **One-to-many(1:N):** One entity from entity set A can be associated with more than one entities of entity set B but from entity set B one entity can be associated with at most one entity. 1:N relationship occurs when any one instance of entity type A is associated with N number of instance of another entity type B.

Eg- Relationship between Department and Faculty.



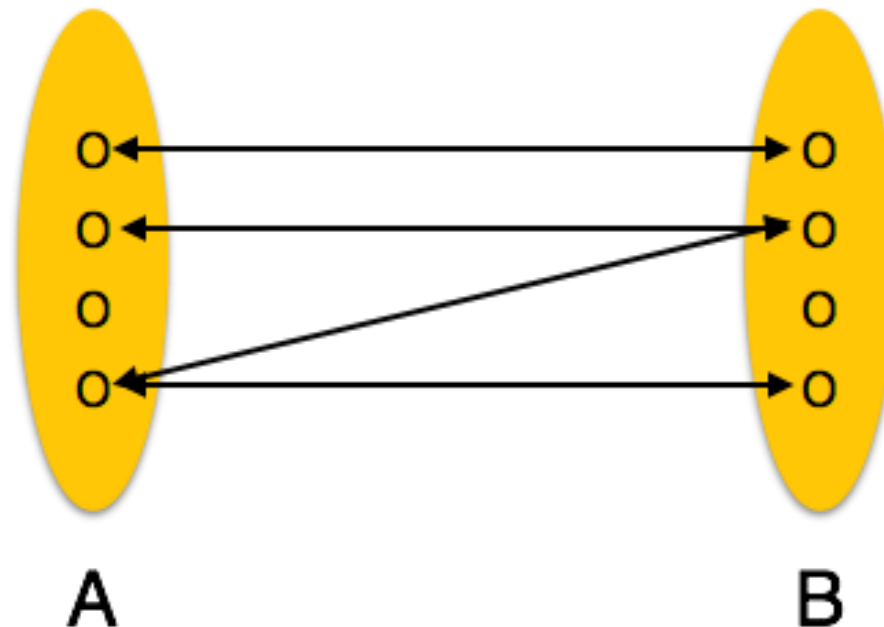
- **Many-to-one:** More than one entities from entity set A can be associated with at most one entity of entity set B but one entity from entity set B can be associated with more than one entity from entity set A. N:1 relationship occurs when any N number of instances of entity type A is associated with atmost one instance of entity type B.

Eg- faculty are employed by department.



- **Many-to-many:** one entity from A can be associated with more than one entity from B and vice versa. M:N relationship occurs when an instance of entity type A is associated with any number of instances of entity type B and instance of entity type B is associated with any number of instances of entity type A.

Eg- Author writes books.



# ER Diagram Representation

- **Entity**
- Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.

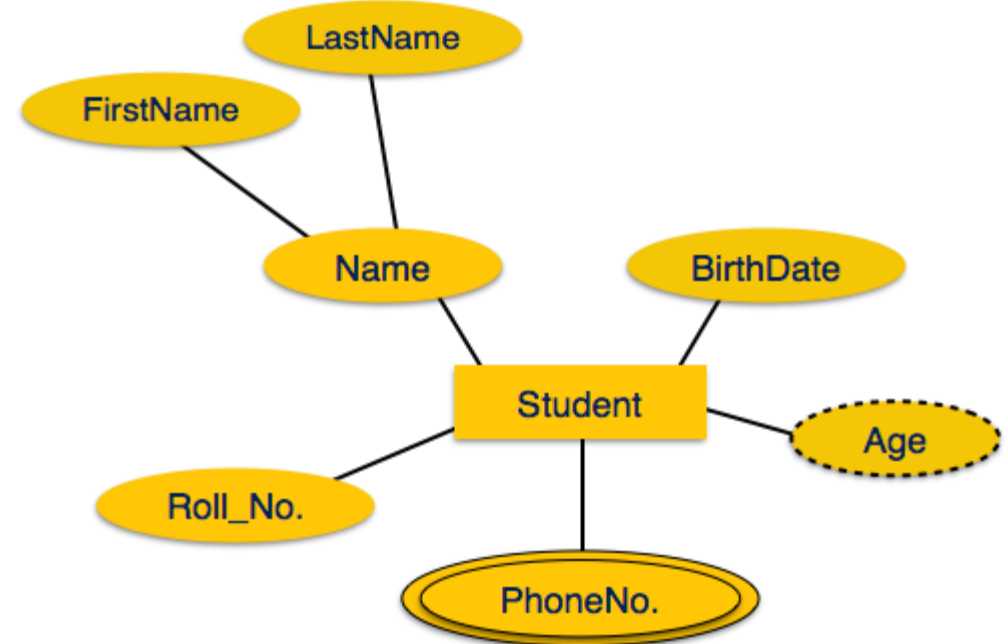
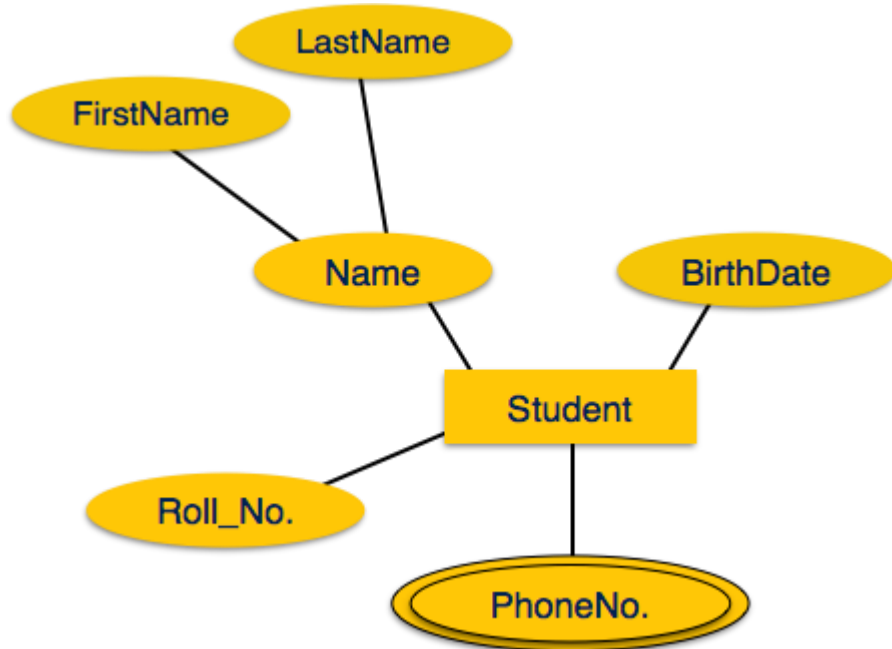
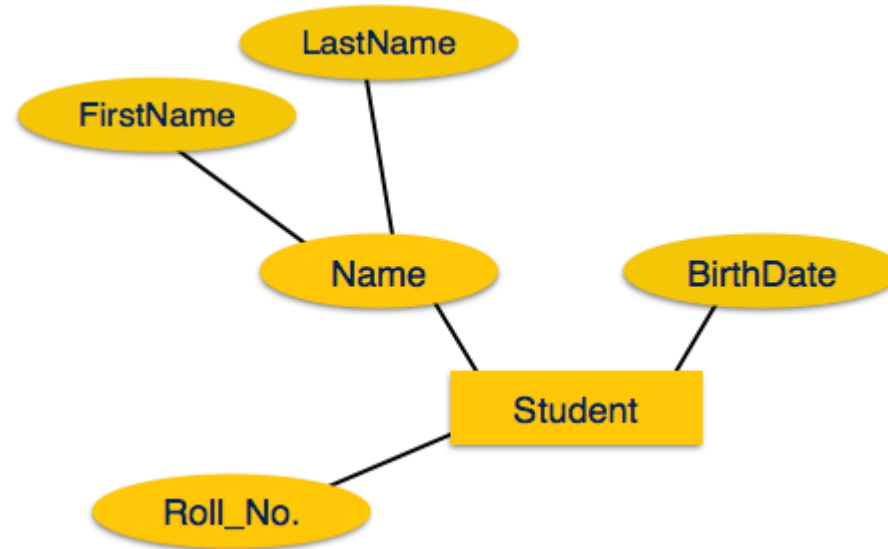
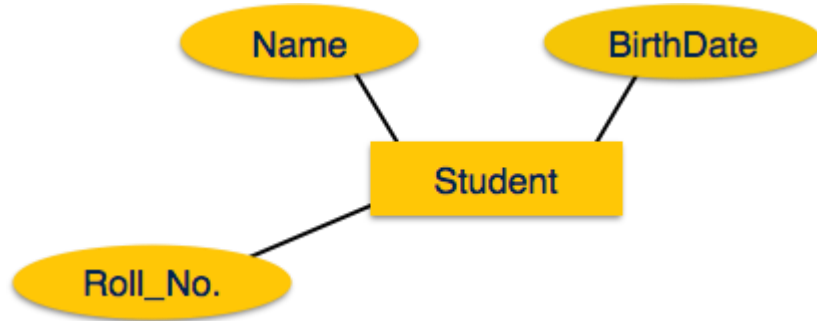
Student

Teacher

Projects



- Attributes



# Cardinality of Relationship

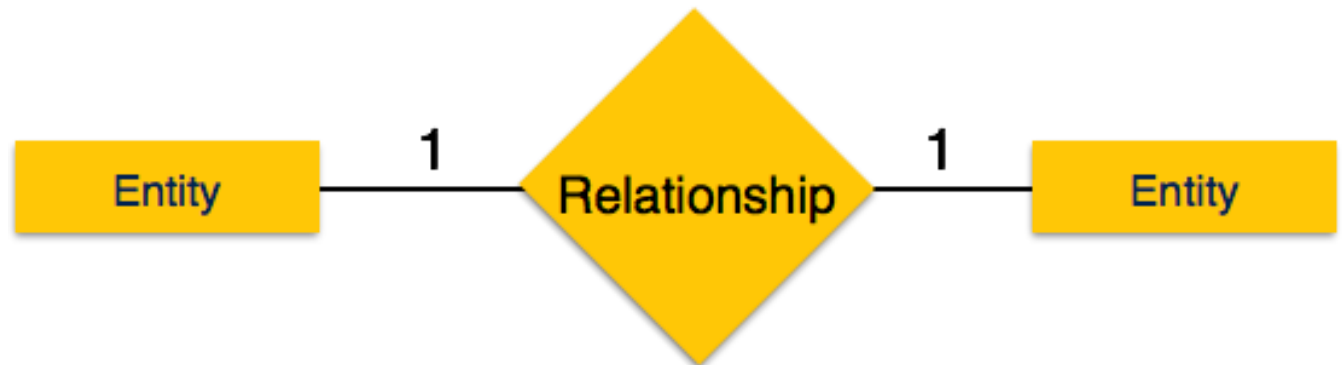
- The total number of rows or tuples in a table is referred as Cardinality of a relation.
- Cardinality of a relationship quantifies the relationship between entities by measuring how many instances of one entity type are related to a single instance of another.
- The actual count of elements associated with the connectivity is called Cardinality.

# Direction of Relationship

- The direction of the relationship is the line connecting the two entities related to each other by a relationship.
- The entity from where the relationship starts is the parent entity and the entity where the relationship ends is called the child entity.
- The type of relation is determined by the line connecting the entities and the relationship components.

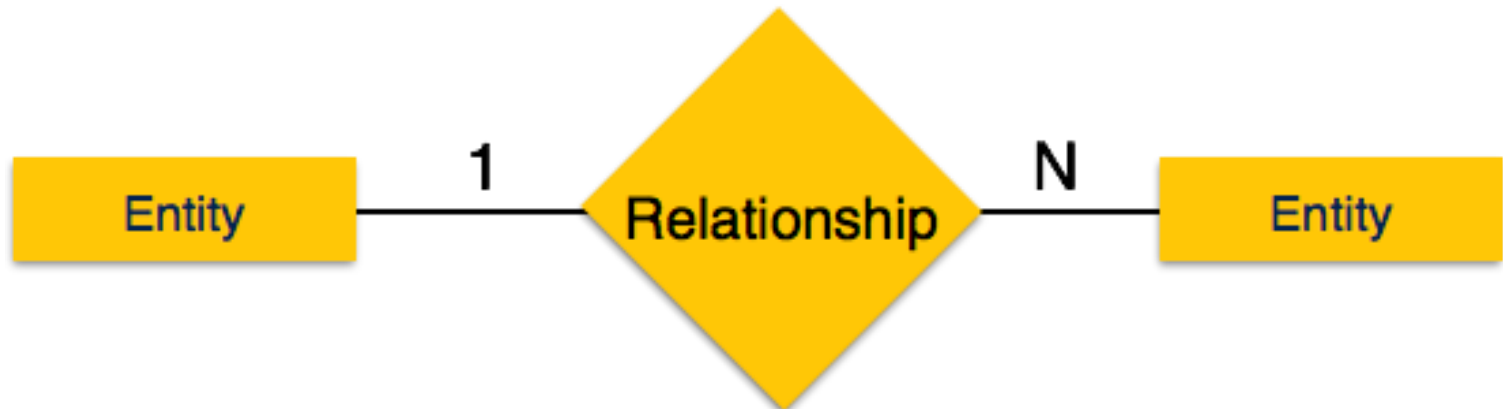
# relationship and cardinality

- **One-to-one**
- When only one instance of entity is associated with the relationship, it is marked as '1'. This image below reflects that only 1 instance of each entity should be associated with the relationship. It depicts one-to-one relationship.



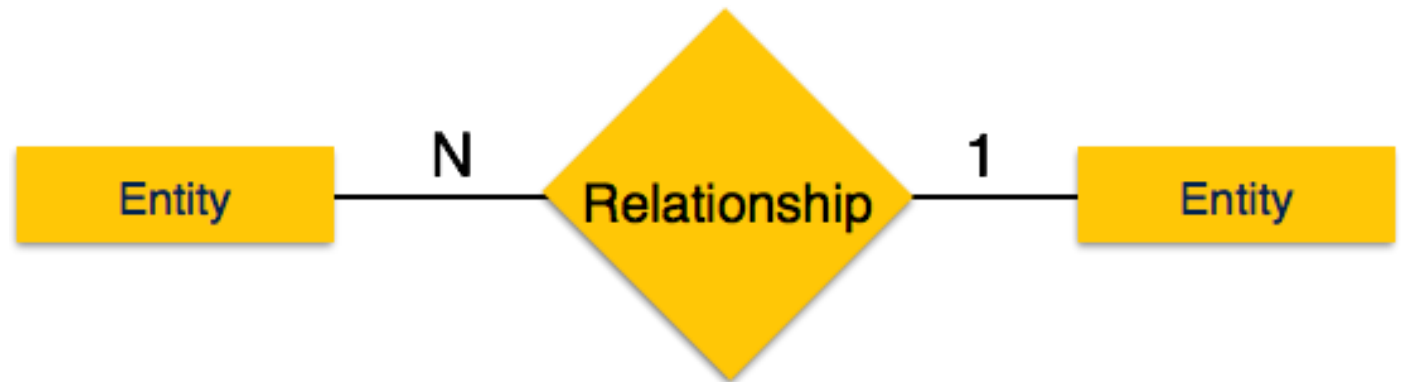
- **One-to-many**

- When more than one instance of entity is associated with the relationship, it is marked as 'N'. This image below reflects that only 1 instance of entity on the left and more than one instance of entity on the right can be associated with the relationship. It depicts one-to-many relationship.

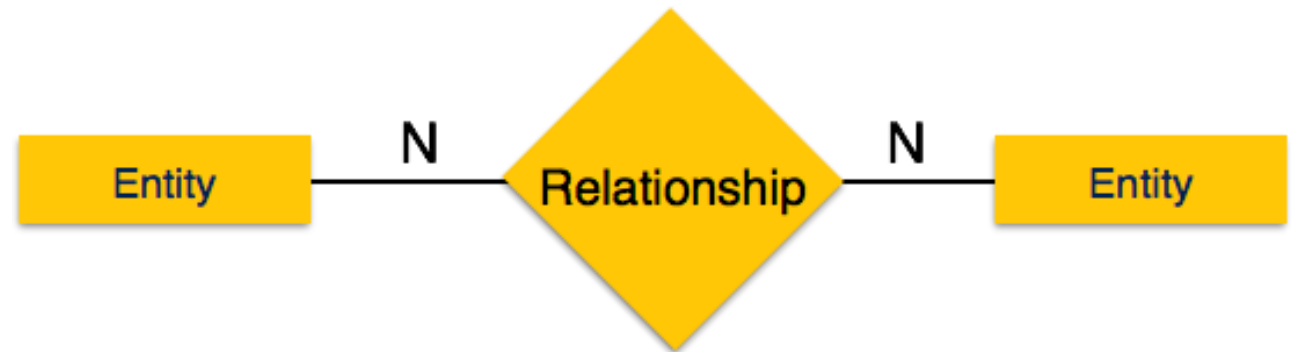


- **Many-to-one**

- When more than one instance of entity is associated with the relationship, it is marked as 'N'. This image below reflects that more than one instance of entity on the left and only one instance of entity on the right can be associated with the relationship. It depicts many-to-one relationship

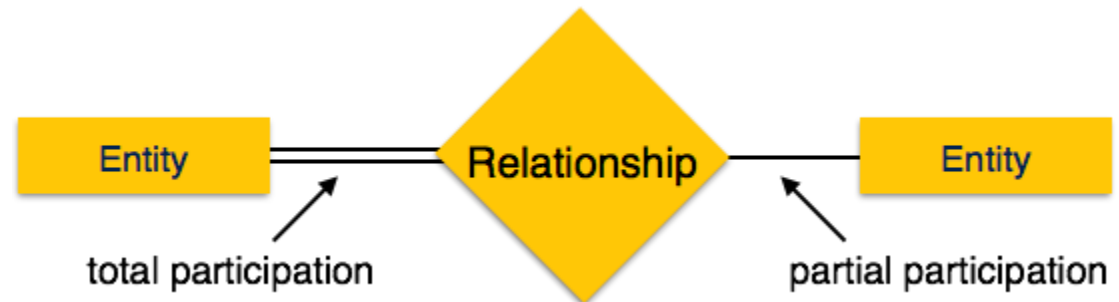


- **Many-to-many**
- This image below reflects that more than one instance of entity on the left and more than one instance of entity on the right can be associated with the relationship. It depicts many-to-many relationship.



# Participation Constraints

- **Total Participation:** Each entity in the entity is involved in the relationship. Total participation is represented by double lines.
- **Partial participation:** Not all entities are involved in the relation ship. Partial participation is represented by single line.





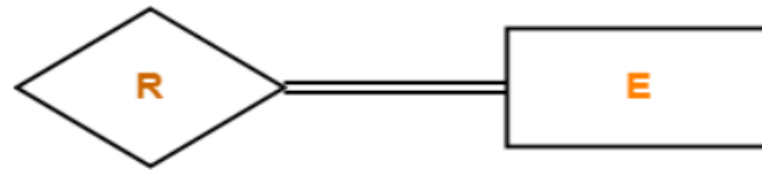
# EXTENDED E-R FEATURES(Enhanced ER Model)

- With the rapid increase in the developments of new database applications like engineering, design and manufacturing, telecommunications, images and graphics, multimedia, data mining, data warehousing, etc..new semantic data modelling concepts were needed because the basic ER Modelling concepts were no longer sufficient to represent the requirements of these complex applications. So the Enhanced ER Model(EER Model) was developed to include the concepts that should represent these complex applications. These concepts include:
  1. Specialization
  2. Generalization
  3. Aggregation

**SUBCLASS :** A subclass or higher level entity sets includes distinct subclasses also known as low level entity sets that has a distinct role on the basis of some distinguishing characteristics of the entities in the superclass. Eg- EMPLOYEE entity can have subclass such as Manager, Secretary, Technician, Engineer and so on. The EMPLOYEE entity acts as a superclass and other entities acts as a subclass. The Relationship between superclass and one of its subclasses is known as superclass/subclass relationship.

## 1. Total Participation-

- It specifies that each entity in the entity set must compulsorily participate in at least one relationship instance in that relationship set.
- That is why, it is also called as **mandatory participation**.
- Total participation is represented using a double line between the entity set and relationship set.



**Total Participation**

### Example-

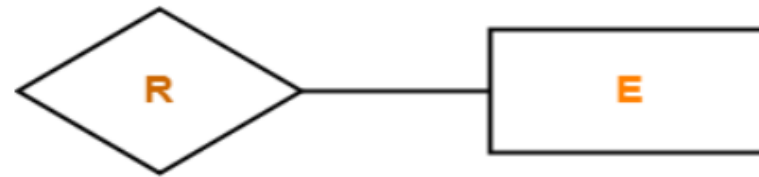


Here,

- Double line between the entity set "Student" and relationship set "Enrolled in" signifies total participation.
- It specifies that each student must be enrolled in at least one course.

## 2. Partial Participation-

- It specifies that each entity in the entity set may or may not participate in the relationship instance in that relationship set.
- That is why, it is also called as **optional participation**.
- Partial participation is represented using a single line between the entity set and relationship set.



**Partial Participation**

### Example-

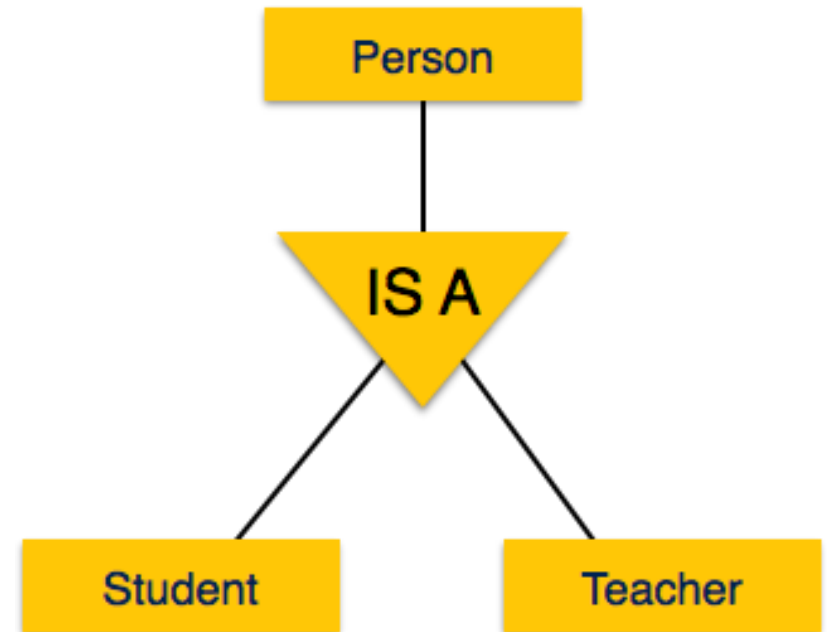


Here,

- Single line between the entity set "Course" and relationship set "Enrolled in" signifies partial participation.
- It specifies that there might exist some courses for which no enrollments are made.

# Specialization

- Specialization is a process of defining one or more subclass entity sets from the superclass entity sets based on some unique attribute specific to the subclass entity set.
- Specialization is a process, which is opposite to generalization. In specialization, a group of entities is divided into sub-groups based on their characteristics.
- In a company, a person can be identified as employee, employer, customer or vendor based on what role do they play in company.

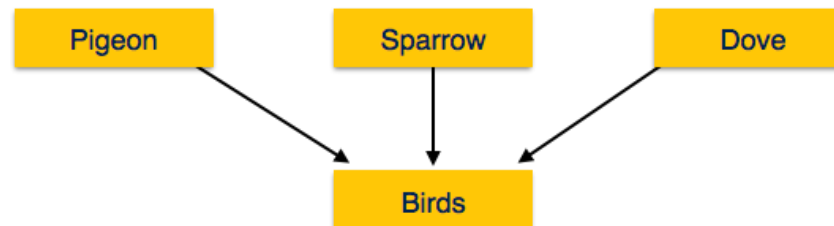


## Properties of Specialization

1. Specialization exists only when there exists specific or unique attributes of the subclass entity set. If there is no unique attributes of subclass entity set then there is no need of specialization.
2. It is needed when some relationship types may be participated only by the entity sets which are members of that subclass. Eg-MANAGER relationship exists between Manager subclass and relating it to Project entity type.
3. In the EER diagram, specialization is represented by a triangle component labelled "IS A".
4. It is top down process.
5. It maximizes the difference between the entity sets by identifying unique attributes of each subclass.

# Generalization

- It is reverse of Specialization. There may exist a number of lower level entity sets that share the same attributes and participate in the same relationship sets. So we generalize these low level entity sets into a single higher level entity set based on these common attributes and relationships.
- Generalization is a process of defining a single higher level entity set from a set of specialized low level entity sets or identification of generalized superclass entity set from original subclass entity sets.
- In generalization, a number of entities are brought together into one generalized entity based on their similar characteristics. For an example, pigeon, house sparrow, crow and dove all can be generalized as Birds.



## Properties of Generalization

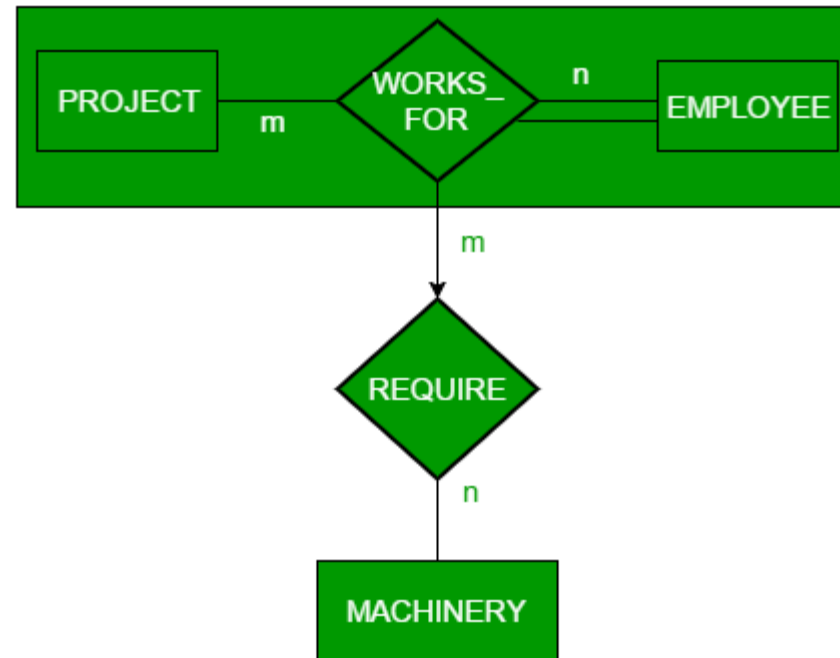
1. Generalization exists only when the distinct low level entity sets have some common attributes and same relationship types.
2. It is a bottom up approach.
3. It minimizes the difference between the low level entity sets by creating a single high level entity set using common attributes.
4. Shared attributes are not repeated.
5. In the EER diagram, generalization is represented by a triangle component labelled "IS A".

# Aggregation

- One of the main limitations of ER diagram is that it cannot express Relationship among Relationships.
- So in order to represent these relationship among relationships, we combine the entity sets and there relationships to form a higher level entity set. This process of combining entity sets and their relationship to form a higher level entity set so as to represent relationship among relationship is known as Aggregation.
- Aggregation is an abstraction in which relationship sets are 'treated as higher level entity sets used for the purpose of participating in other relationship.



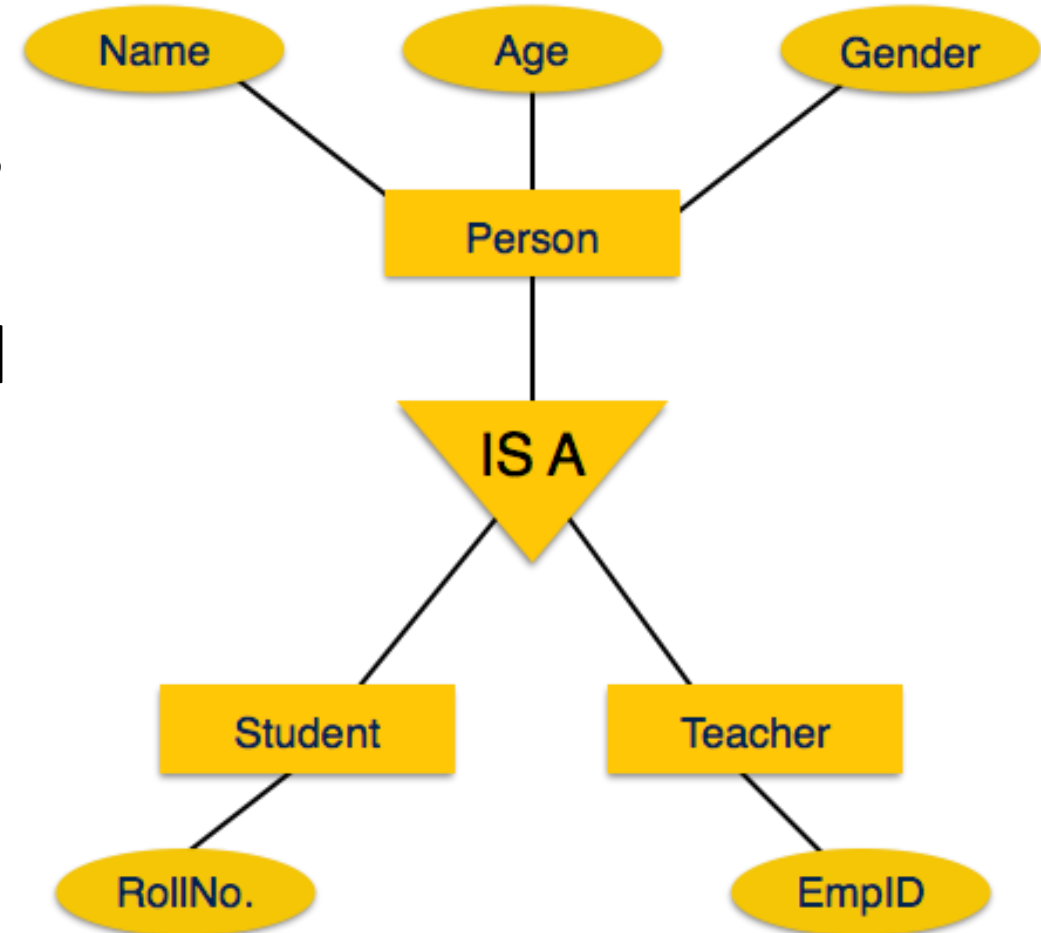
For Example, Employee working for a project may require some machinery. So, REQUIRE relationship is needed between relationship WORKS\_FOR and entity MACHINERY. Using aggregation, WORKS\_FOR relationship with its entities EMPLOYEE and PROJECT is aggregated into single entity and relationship REQUIRE is created between aggregated entity and MACHINERY.



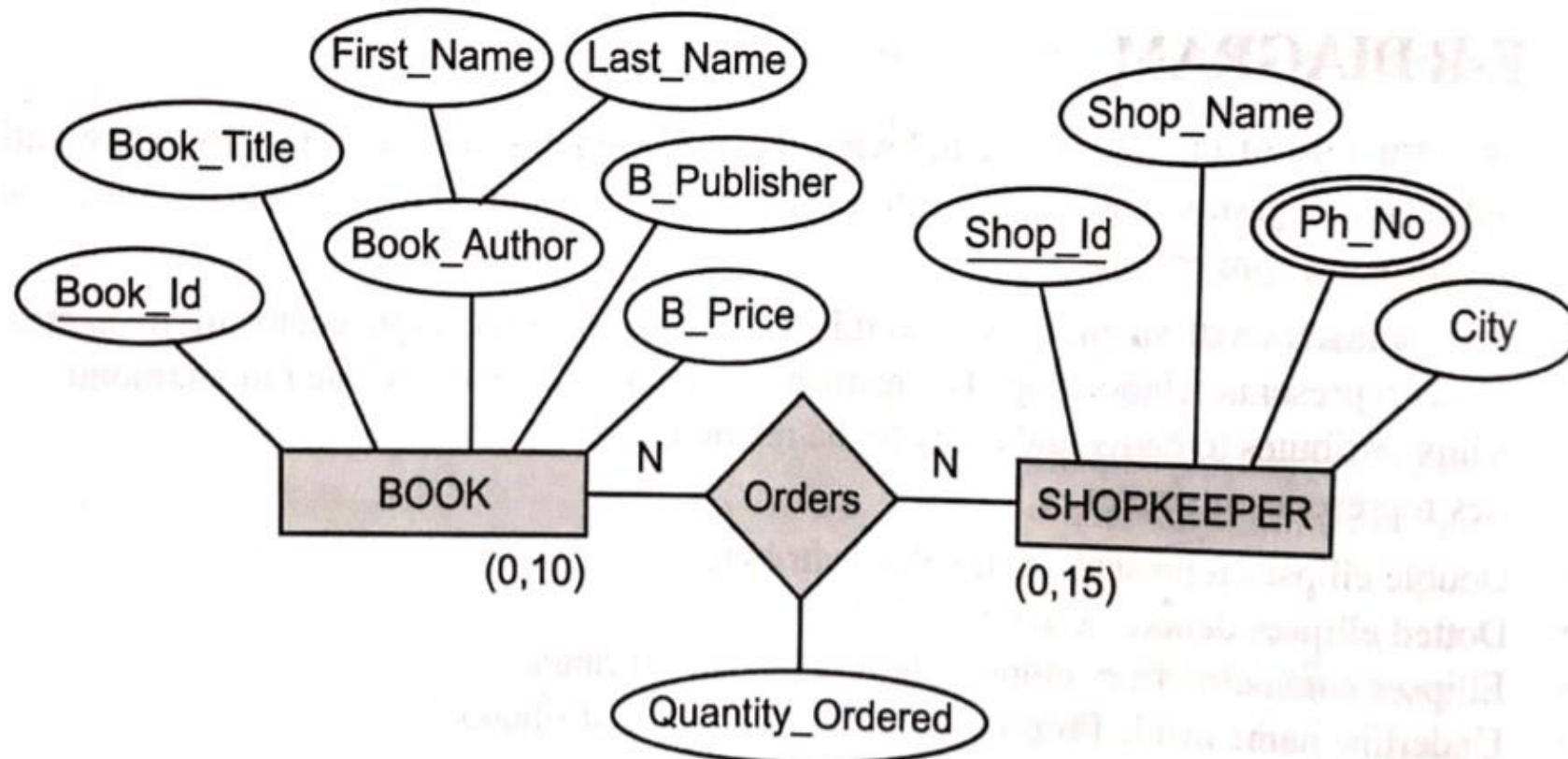
Aggregation

# Inheritance

- The important features of Generalization and Specialization, is inheritance, that is, the attributes of higher-level entities are inherited by the lower level entities.



Consider a case where a shopkeeper can order for atmost 15 books and a book can be ordered by 10 shopkeepers. The two entity types BOOK and SHOPKEEPER are connected by a relationship “ORDERS”. Draw the ER diagram showing their relationship.



*Fig. 3.12*

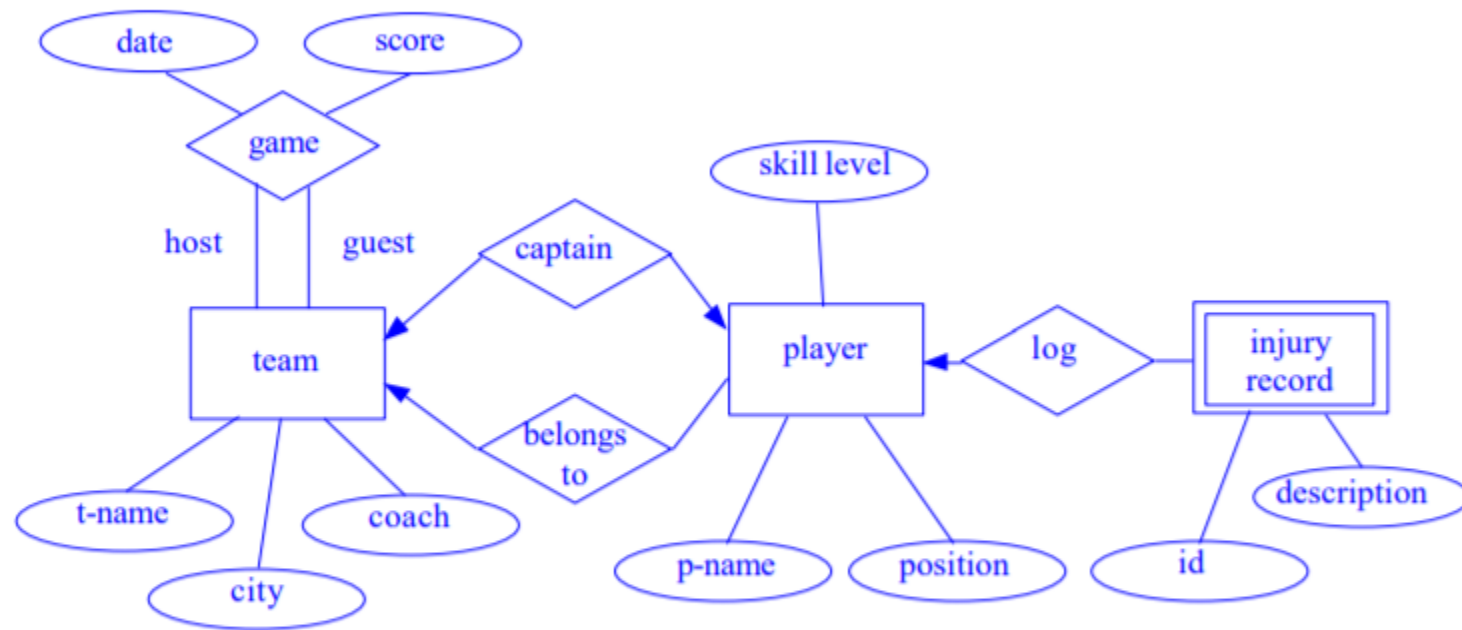
BOOK (**Book\_Id**, Book\_Title, Book\_Author (First\_Name, Last\_Name), B\_Publisher, B\_Price)  
 SHOPKEEPER (**Shop\_Id**, Shop\_Name, Ph\_No, City)  
 ORDERS (Quantity\_Ordered)

### ***Practice ER Diagram Question – A Sample Solution***

Suppose you are given the following requirements for a simple database for the National Hockey League (NHL):

- the NHL has many teams,
- each team has a name, a city, a coach, a captain, and a set of players,
- each player belongs to only one team,
- each player has a name, a position (such as *left wing* or *goalie*), a skill level, and a set of injury records,
- a team captain is also a player,
- a game is played between two teams (referred to as *host\_team* and *guest\_team*) and has a date (such as *May 11<sup>th</sup>, 1999*) and a score (such as *4 to 2*).

Construct a clean and concise ER diagram for the NHL database using the Chen notation as in your textbook. List your assumptions and clearly indicate the cardinality mappings as well as any role indicators in your ER diagram.



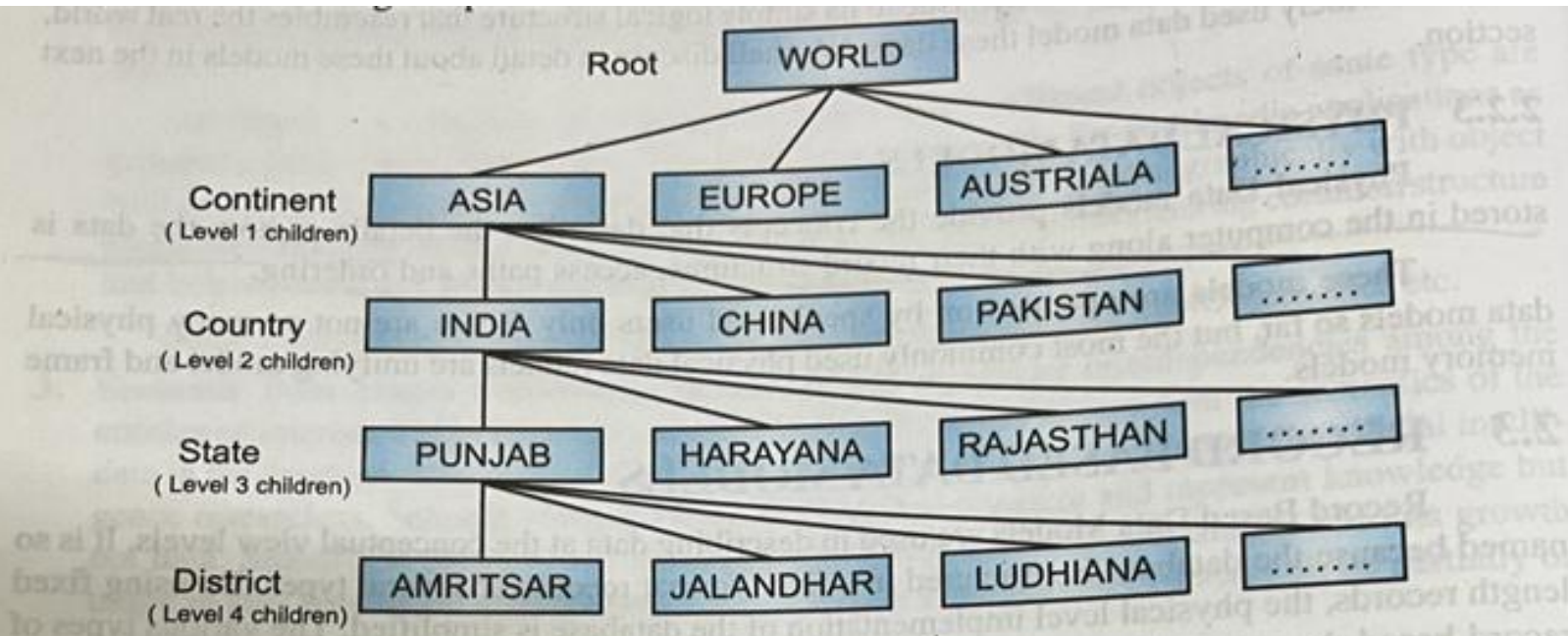
# Hierarchical Model

- First hierarchical data model was designed by IBM and North American Rockwell in 1950.
- Records are arranged in tree like structure, and are linked with their superior records on which they are dependent and to them which are dependent on them.
- One to many relationship.

# Operations on Hierarchical Model

- Insert
- Update
- Delete
- Retrieval of information





**Fig. 2.1 - A Hierarchical Model**

In the Fig. 2.1, the 'WORLD' acts as a root of the tree structure which has many children's like Asia, Europe, Australia etc. These children are at level 1 and as a parent for different countries such as ASIA continent acts as a parent for countries like India, China, Pakistan etc. Similarly, these children are at level 2 and act as a parent for different states such as INDIA country acts as a parent for states Punjab, Haryana, Rajasthan etc. Further the same follows.

A child 'Amritsar' has a parent 'PUNJAB' which further has a parent 'INDIA' and so on. Now 'India' will act as a grandparent for the child 'Amritsar'.

on. Now 'India' will act as a grandparent for the child 'Amritsar'.

Consider a sample database consisting of three records types : BOOKS, SHOPKEEPER and BOOKS\_ORDER. The BOOKS record type contains for each book, a Book\_Id, book\_title, Author, Publisher and Price fields. The SHOPKEEPER record type contains for each shopkeeper a shopkeeper number i.e. its Id, shopkeeper's name, phone number, city. The BOOKS\_ORDER record type contains for each order a Book\_Id, Shop\_Id and quantity ordered by each shopkeeper to a wholesaler.

its chil  
tween  
The re  
BOOI

Book  
make



Book_Id	Book_title	Author	Publisher	Price	Shop_Id	Shop_Name	Ph_No	City
B1	C	Anshuman	LPB	175	S1	Sanju	221023	Asr
B2	C++	Yashwant	BPB	200	S2	Ravi	509984	Ldh
B3	UNIX	Robert	Tata	250	S3	Mukesh	317076	Jal
B4	JAVA	Herbt	PHI	195				

**BOOKS record type**

Shop_Id	Book_Id	Qty_order
S1	B1	50
S1	B2	10
S1	B3	20
S2	B1	30
S2	B2	20
S3	B2	15

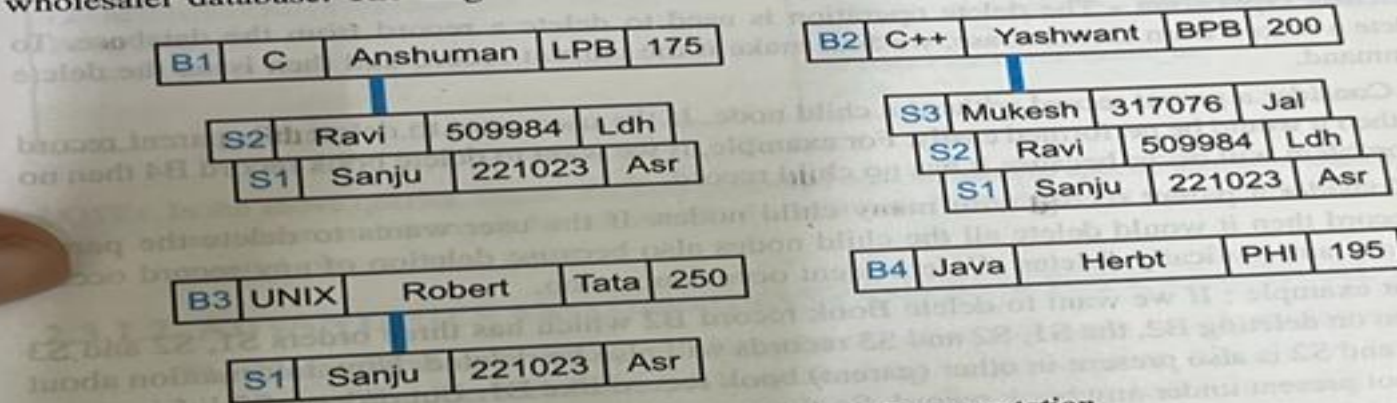
**SHOPKEEPER record type**

**BOOKS\_ORDER record type**

**Fig. 2.2**

We assume that a particular shopkeeper can order many books. Also each book is having a unique Book\_Id in BOOKS record type and each shopkeeper is having unique Shop\_Id in SHOPKEEPER record type.

Now consider the hierarchical diagram as shown in Fig. 2.3 which represents the part of the wholesaler database. The diagrams depicts the tree structure.

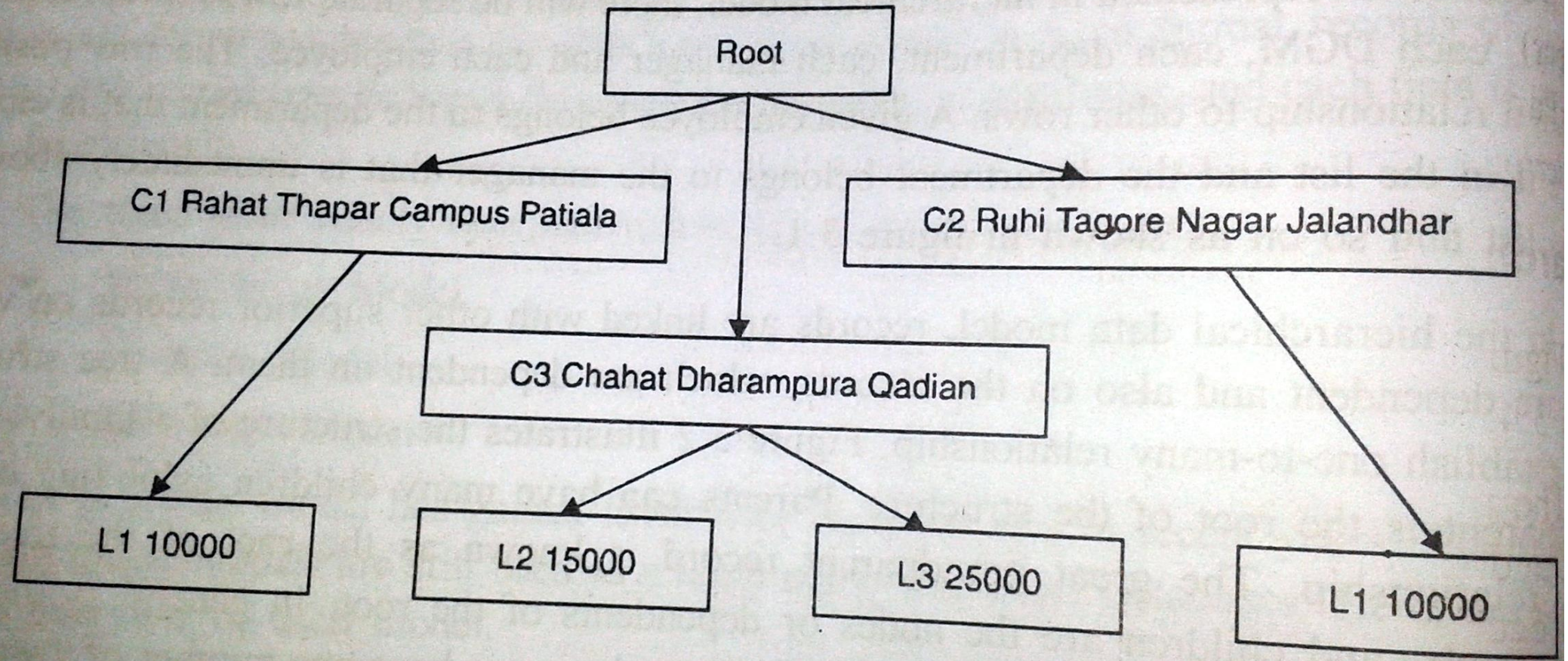


**Fig. 2.3 - Hierarchical Representation**

In the Fig. 2.3, each tree takes book's record to be the parent and Shopkeeper's record to be its child. There is one Shopkeeper record for each order of a particular book. There are links between Shopkeeper's occurrences and book's occurrences representing the associated book's order. The relationship or link depicts the quantity ordered for a particular book which is shown in the BOOKS\_ORDER record type.

Now from these tree diagrams, it is clear that book B1 is ordered by S1 and S2 shopkeeper. Book B2 is ordered by S1, S2 and S3 shopkeeper and book B3 is ordered by S1 shopkeeper. No one makes an order for book B4.





# Advantages/ Disadvantages

- Advantages:
- Simplicity
- Data security
- Data integrity
- Efficiency
- Disadvantages
- Implementation complexity
- Database management problem
- Lack of structural independence

# Network Model

- Network model replace the tree structure with graph.
- Advantages:
  - Conceptual simplicity
  - Capable to handle more relationship typed: 1:N,N:N
  - Data integrity
  - Data independence
- Database standards:
  - ANSI in 1970
- Disadvantages :
  - System complexity
  - Operational anomalies
  - Absence of structural independence



Consider a sample database, one similar to SHOPKEEPER\_BOOKS database which we have discussed earlier in the hierarchical model. So the network model is represented as

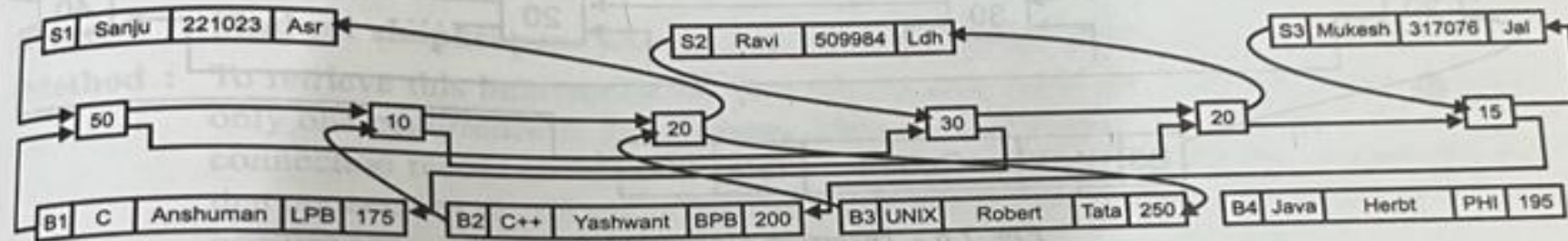


Fig. 2.5 - M:N Relationship in Network model

The figure 2.5 shows a method of implementing the many to many relationship using the new type of record called *Connection Record*. The connection record contains common data (Quantity Ordered in our case) that describes the association between BOOKS and SHOPKEEPER records and two pointers, one for each of the sets it is involved in. All the connection record's occurrences for a given shopkeeper are placed on a chain, starting and returning to that chain. Similarly, connection record occurrences for a given book are placed on a chain, starting and returning to that chain. This means that each owner record (S1, S2 or S3) must have pointers to all its member records {(50, 10, 20), (30, 20), (15)} and in order to enable owner records to be found from member records then each member record (B1, B2, B3 or B4) needs to have pointers back to all of its owners.

For example : In the above relationships, shopkeeper S1 orders 50 quantity of B1 and 10

