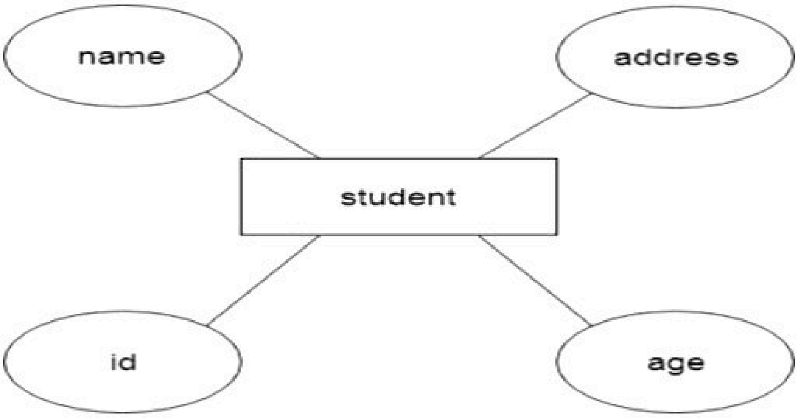
ENTITY-RELATIONSHIP MODEL

part B 1)

- ER model stands for an Entity-Relationship model. It is a high-level data model. This modelis 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 todesign view of data.
- In ER modeling, the database structure is portrayed as a diagram called an entity-relationshipdiagram.

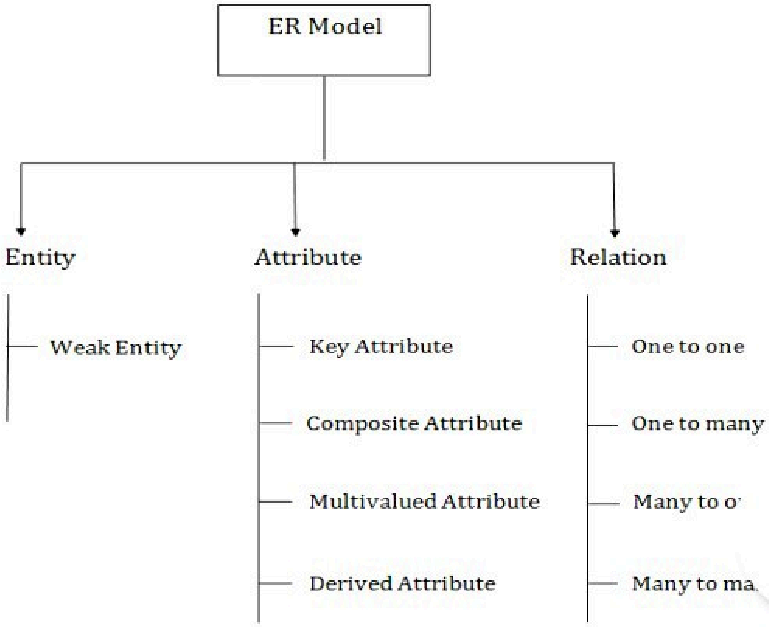
Example:



E-R DIAGRAMS

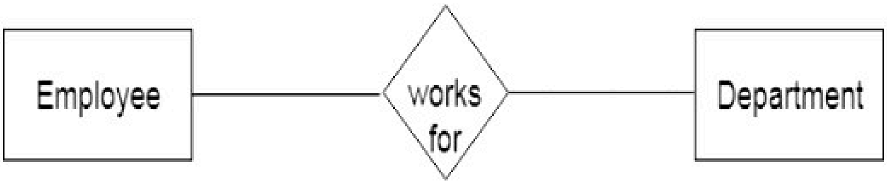
An entity relationship diagram (ERD) **shows the relationships of entity sets stored in a database**. ... By defining the entities, their attributes, and showing the relationships between them, an ER diagram illustrates the logical structure of databases. ER diagrams are used to sketch out the designof a database.

Component of ER Diagram



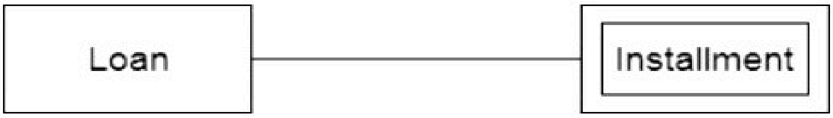
Entity

- 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.



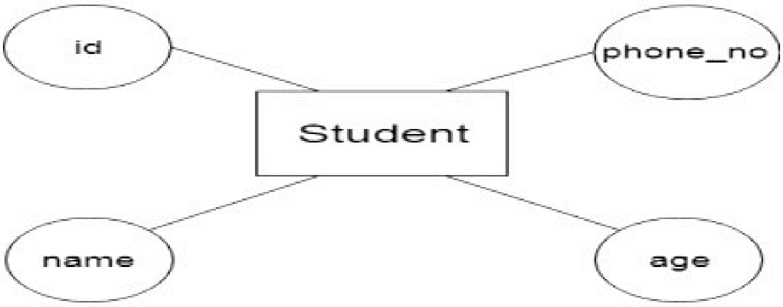
WEAK ENTITY

- An entity that depends on another entity called a weak entity. The weak entity doesn't contain any key attribute of its own. The weak entity is represented by a double rectangle.

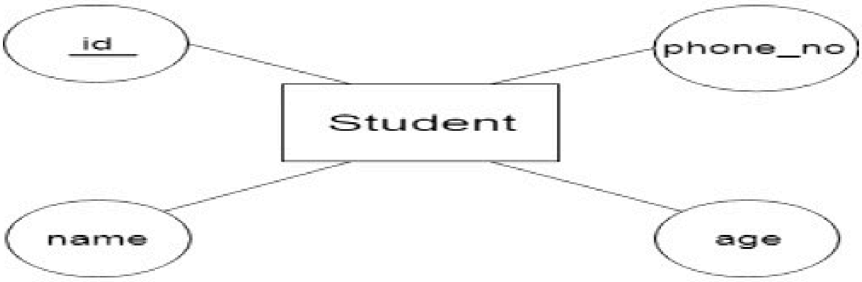


ATTRIBUTE

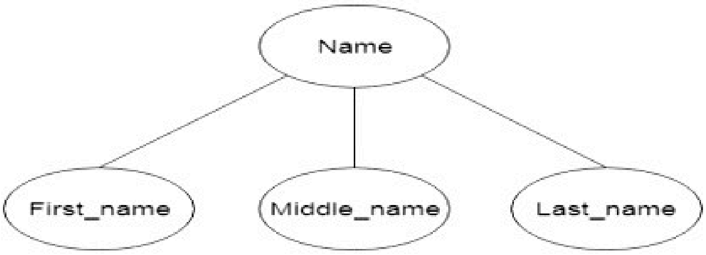
- The attribute is used to describe the property of an entity. Eclipse is used to represent an attribute.
- **For example,** id, age, contact number, name, etc. can be attributes of a student.



a. **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.

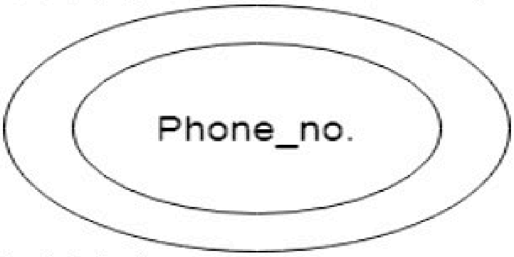


b. **Composite Attribute**
An attribute that composed of many other attributes is known as a composite attr
composite attribute is represented by an ellipse, and those ellipses are connected
ellipse.



c. **Multivalued Attribute**

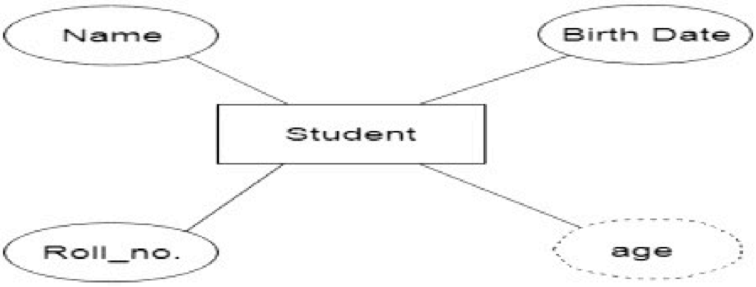
- An attribute can have more than one value. These attributes are known as a multivalued attribute. The double oval is used to represent multivalued attribute.
  For example, a student can have more than one phone number.



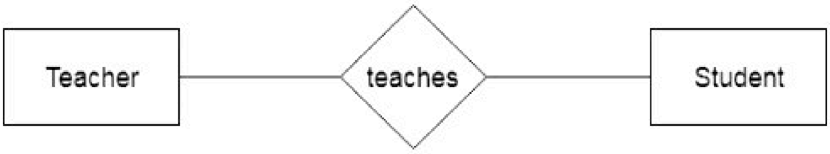
d. **Derived Attribute**

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.

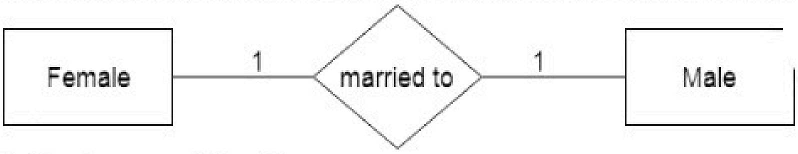


RELATIONSHIP A relationship is used to describe the relation between entities. Diamond or rhombus is used to represent the relationship.



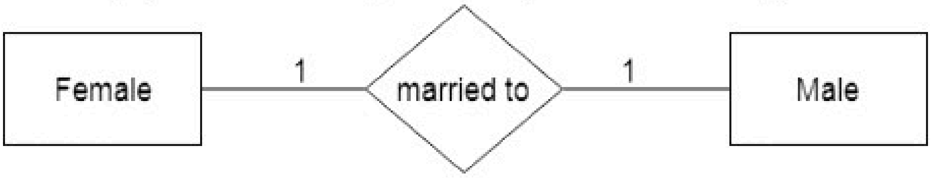
a. One-to-One Relationship

- 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 fema'



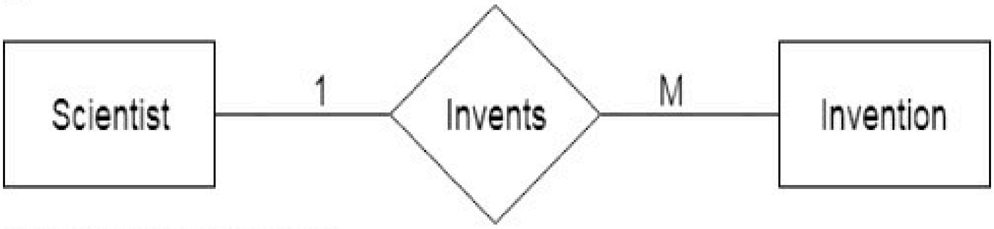
a. One-to-One Relationship

- 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.



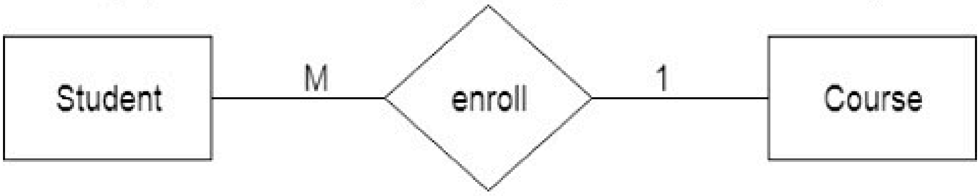
b. One-to-many relationship

- 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.



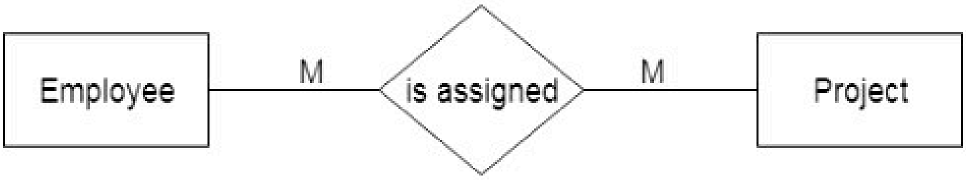
c. Many-to-one relationship

- 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.



d. Many-to-many relationship

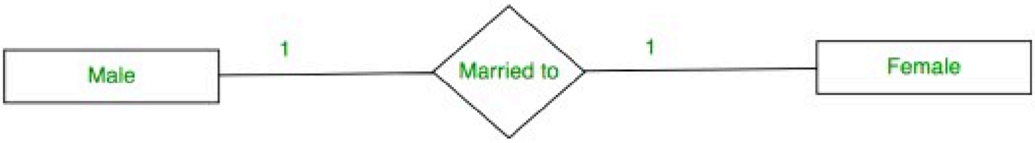
When more than one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then it is known as a many-to-many relationship. For example, Employee can assign by many projects and project can have many employees.



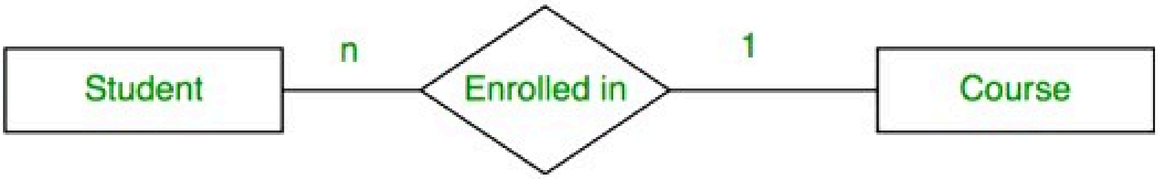
CARDINALITY

The number of times an entity of an entity set participates in a relationship set is known as cardinality. Cardinality can be of different types:

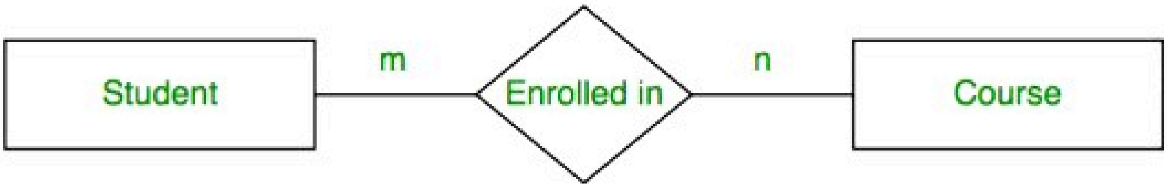
1. One to one – When each entity in each entity set can take part only once in the relationship, the cardinality is one to one.



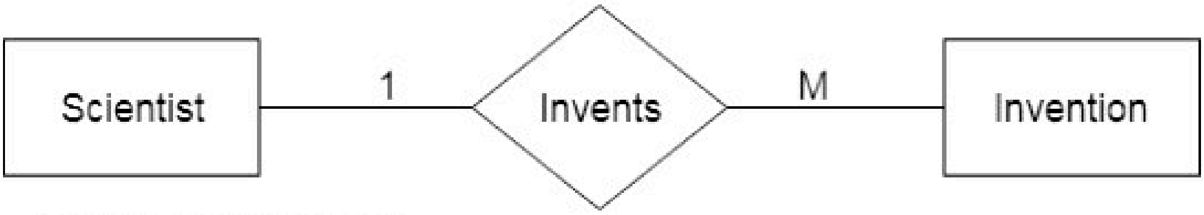
2. Many to one – When entities in one entity set can take part only once in the relationship set and entities in other entity set can take part more than once in the relationship set



3. Many to many – When entities in all entity sets can take part more than once in the relationship cardinality is many to many.



4. One to Many – When each entity in entity set can take part more than once in the relationship, the cardinality is one to many.



ENHANCED-ER MODEL

- Enhanced entity-relationship diagrams are advanced database diagrams very similar to regular ER diagrams which represent requirements and complexities of complex databases.
- It is a diagrammatic technique for displaying the Sub Class and Super Class
- EER is a high-level data model that incorporates the extensions to the original ER model. **It is a diagrammatic technique for displaying the following concepts**

a) **Sub Class and Super Class**

b) **Specialization and Generalization**

c) **Union or Category**

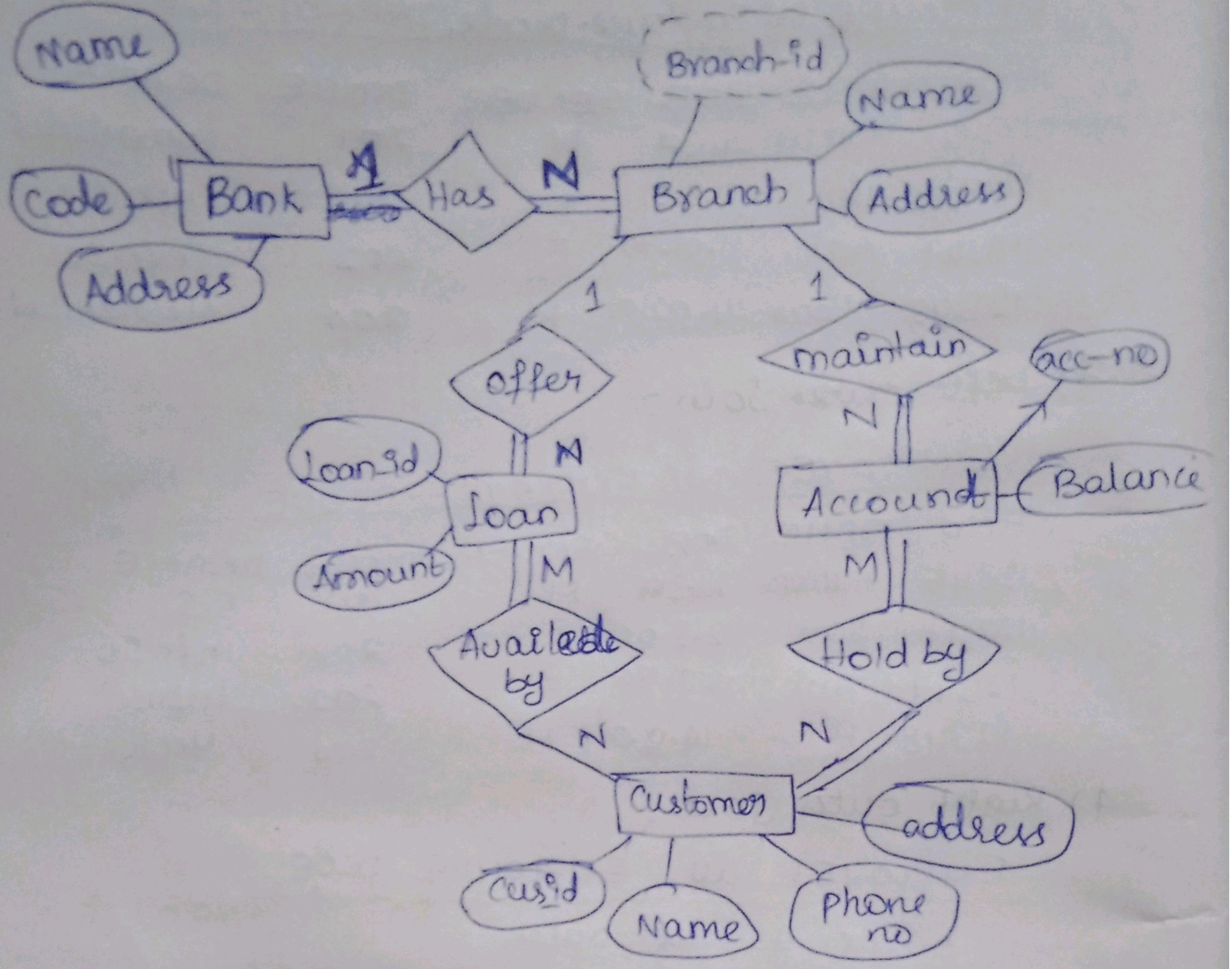
d) **Aggregation**

These concepts are used when the comes in EER schema and the resulting schema diagrams called as EER Diagrams.

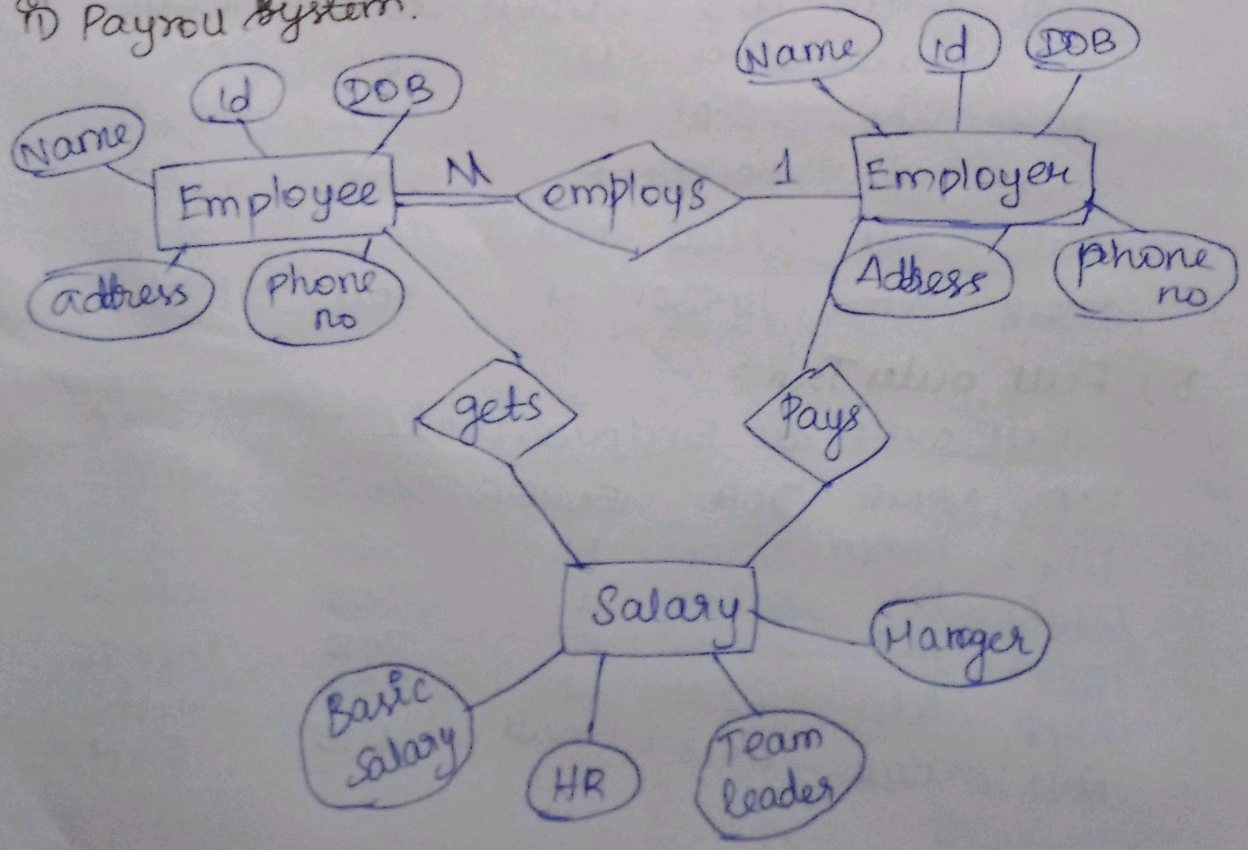
part B 2)

UNIT-2

2. i) ER diagram for Banking system:



ii) Payroll ~~System~~:



FUNCTIONAL DEPENDENCIES part B 3)

The whole database is described by a single **universal** relation schema $R = \{A1, 2, ..., An\}$.

Definition:

- A **functional dependency**, denoted by $X \rightarrow Y$, between two sets of attributes X and Y that are subsets of R specifies a **constraint** on the possible tuples that can form a relation state r of R.
- The constraint is that, for any two tuples $t1$ and $t2$ in r that have $t1[X] = t2[X]$, they must also have $t1[Y] = t2[Y]$.
- The values of the Y component of a tuple in r depend on, or are *determined by*, the values of the X component.
- The values of the X component of a tuple uniquely (or **functionally**) *determine* the values of the Y component.
- There is a functional dependency (FD or f.d) from X to Y, or that Y is **functionally dependent** on X.
- X functionally determines Y in a relation schema R if, and only if, whenever two tuples of $r(R)$ agree on their X-value, they must necessarily agree on their Y value.

Note the following:

- If a constraint on R states that there cannot be more than one tuple with a given X-value in any relation instance $r(R)$
- That is, X is a **candidate key** of R—this implies that $X \rightarrow Y$ for any subset of attributes Y of R.
- If $X \rightarrow Y$ in R, this does not say whether or not $Y \rightarrow X$ in R.

- A functional dependency is a property of the **semantics** or **meaning of the attributes**.
- Whenever the semantics of two sets of attributes in R indicate that a functional dependency should hold, specify the dependency as a constraint.
- Relation extensions $r(R)$ that satisfy the functional dependency constraints are called **legal relation states** (or **legal extensions**) of R.

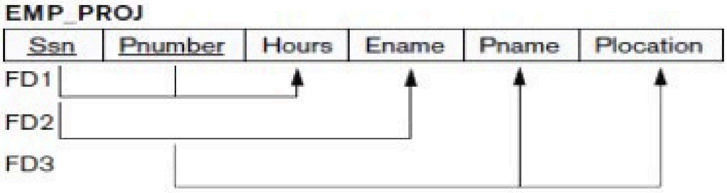


Fig. Relation schemas EMP_PROJ.

Consider the relation schema EMP_PROJ in Figure; from the semantics of the attributes and the relation, the following functional dependencies should hold:

Ssn $\rightarrow$ Ename

Pnumber $\rightarrow$ {Pname, Plocation}

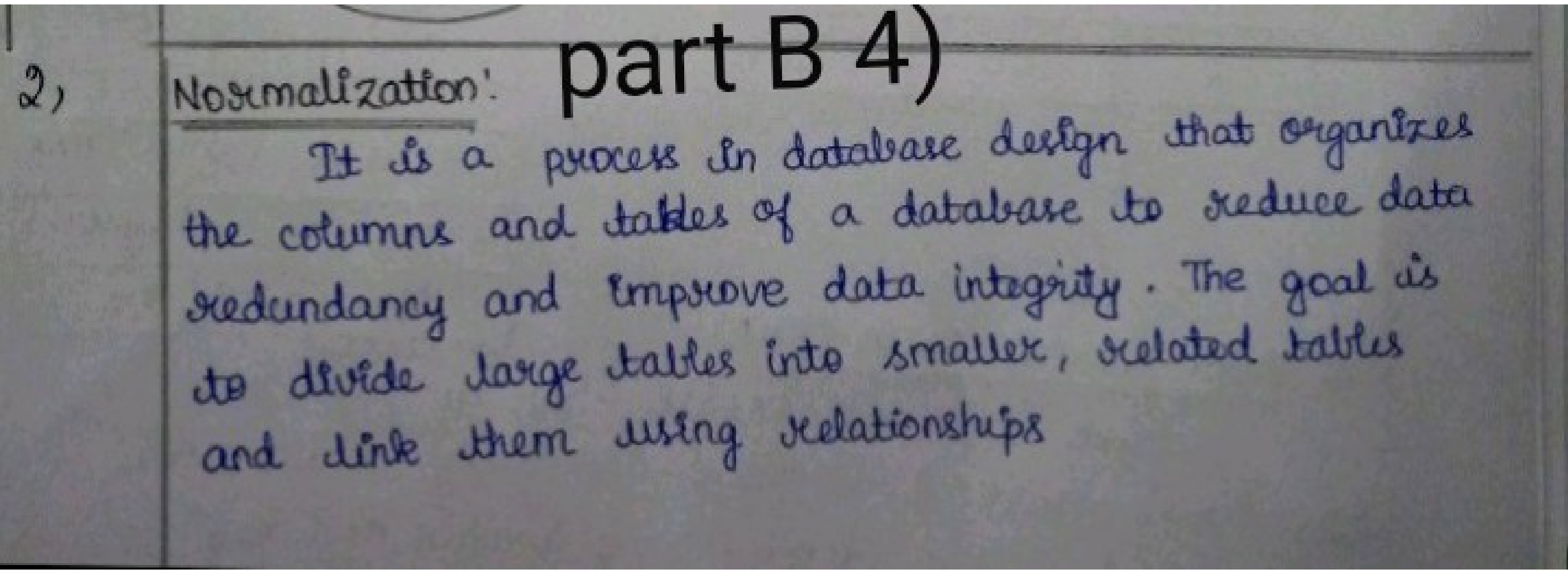
{Ssn, Pnumber} $\rightarrow$ Hours

These functional dependencies specify that

- The value of an employee's Social Security number (Ssn) uniquely determines the employee name (Ename),
- The value of a project's number (Pnumber) uniquely determines the project name (Pname) and location (Plocation),
- A combination of Ssn and Pnumber values uniquely determines the number of hours the employee currently works on the project per week (Hours).
- Alternatively, Ename is functionally determined by (or functionally dependent on) Ssn.

DEPENDENCY PRESERVING

- It is an important constraint of the database.
- In the dependency preservation, at least one decomposed table must satisfy every dependency.
- If a relation R is decomposed into relation R1 and R2, then the dependencies of R either must be a part of R1 or R2 or must be derivable from the combination of functional dependencies of R1 and R2.
- For example, suppose there is a relation R (A, B, C, D) with functional dependency set (A->BC). The relational R is decomposed into R1(ABC) and R2(AD) which is dependency preserving because FD A->BC is a part of relation R1(ABC).



2) Normalization: part B 4)

It is a process in database design that organizes the columns and tables of a database to reduce data redundancy and improve data integrity. The goal is to divide large tables into smaller, related tables and link them using relationships

⇒ First Normal Form (1NF):

Ensure that each column contains atomic valu and that each column contains values of a single type. There should be no repeating groups or arrays

EmpId	Name	Ph. no
1.	John	123456, 789101
2.	Sara	112-233

Emp. Id	Name	Ph. no
1.	John	123456
1.	John	789101
2.	Sara	112233

⇒ Second Normal Form (2NF):

Achieve 1NF and ensure that all non-key attribute are fully functionally dependent on the primary key. In other words, remove partial dependencies where a non-key attribute is dependent on only a part of the composite primary key.

OrderId	ProductId	Product name	quantity
1	101	widget	10
2	105	Gizmo	5

Order Id	Product Id	Quant
1	101	10
2	105	5

ProductId	Product na
101	widget
105	Gizmo

⇒ Third Normal Form (3NF):

Achieve 2NF and ensure that no cransitive dependencies exist. A transitive dependency occurs when a non-key attribute depends on another non-key attribute, which in turn depends on the primary key

Student Id	Name	Dept	Hod
1.	Alice	CSE	DSmith
2.	Bob	EEE	Dr John

Stu.Id	Name	Dept
1	Alice	CSE
2	Bob	EEE

Dept	Hod
CSE	Dr.Smith
EEE	Dr.John

⇒ Boyce-Codd Normal Form (BCNF)

Achieve 3NF, but with stricter requirement every determinant must be a candidate key. BCNF is a Stronger version of 3NF

course Id	Instructor	Room
CS101	Prof.A	101
CS102	Prof.B	102
CS101	Prof.C	103

Course Id	Room
CS101	101
CS102	102

Instructor	Course Id
Prof.A	CS101
Prof.B	CS102
Prof.C	CS101

⇒ Fourth Normal Form: (4NF)

Achieve BCNF and ensure that no multi-valued dependencies exist. A multi-valued dependency occurs when one attribute in a table uniquely determines another attribute independently of all other attributes

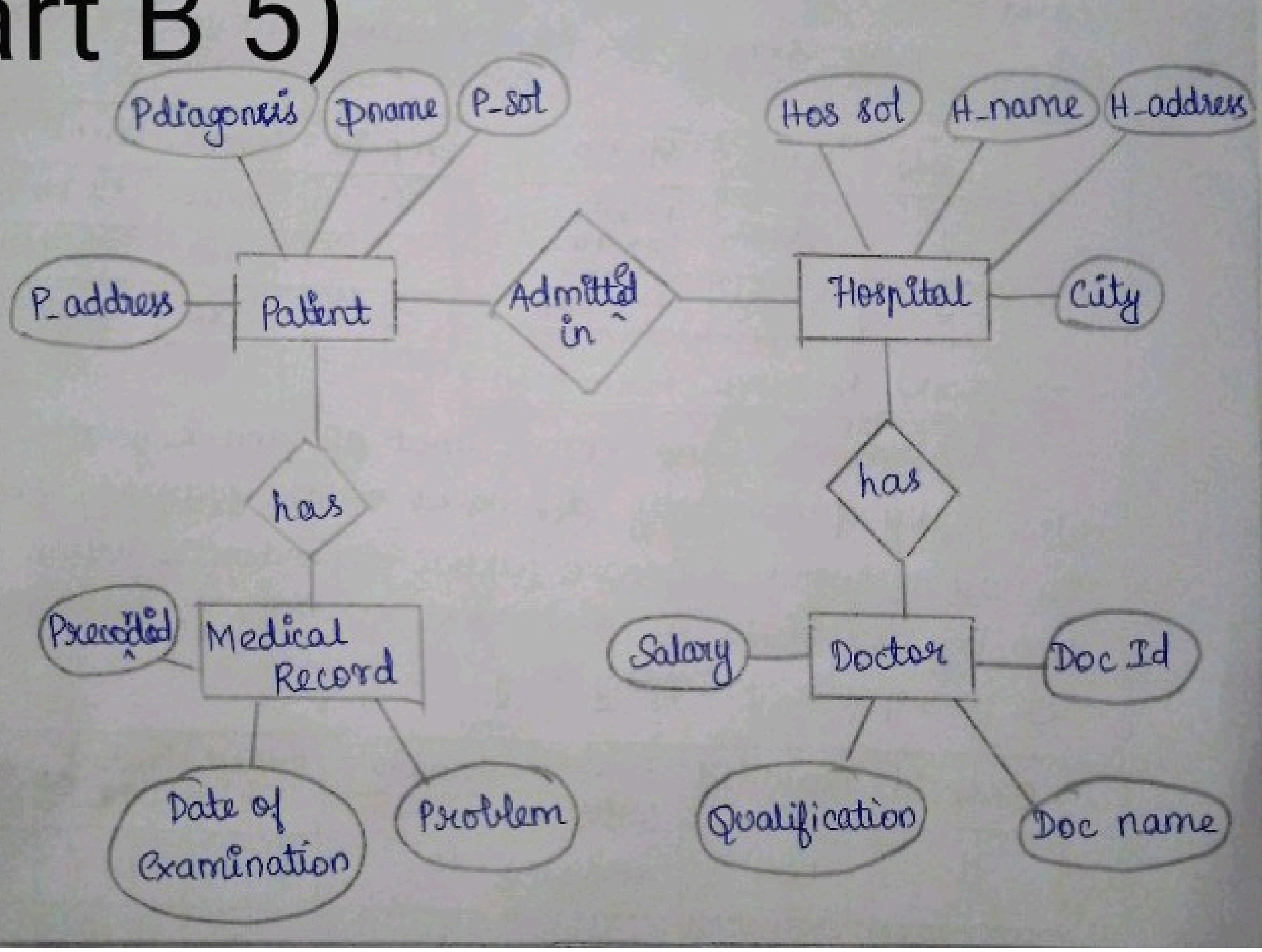
Emp_Id	Project	Skills
1	A	Java
1	A	SQL
1	B	Java
1	B	SQL

Emp Id	Project
1	A
1	B

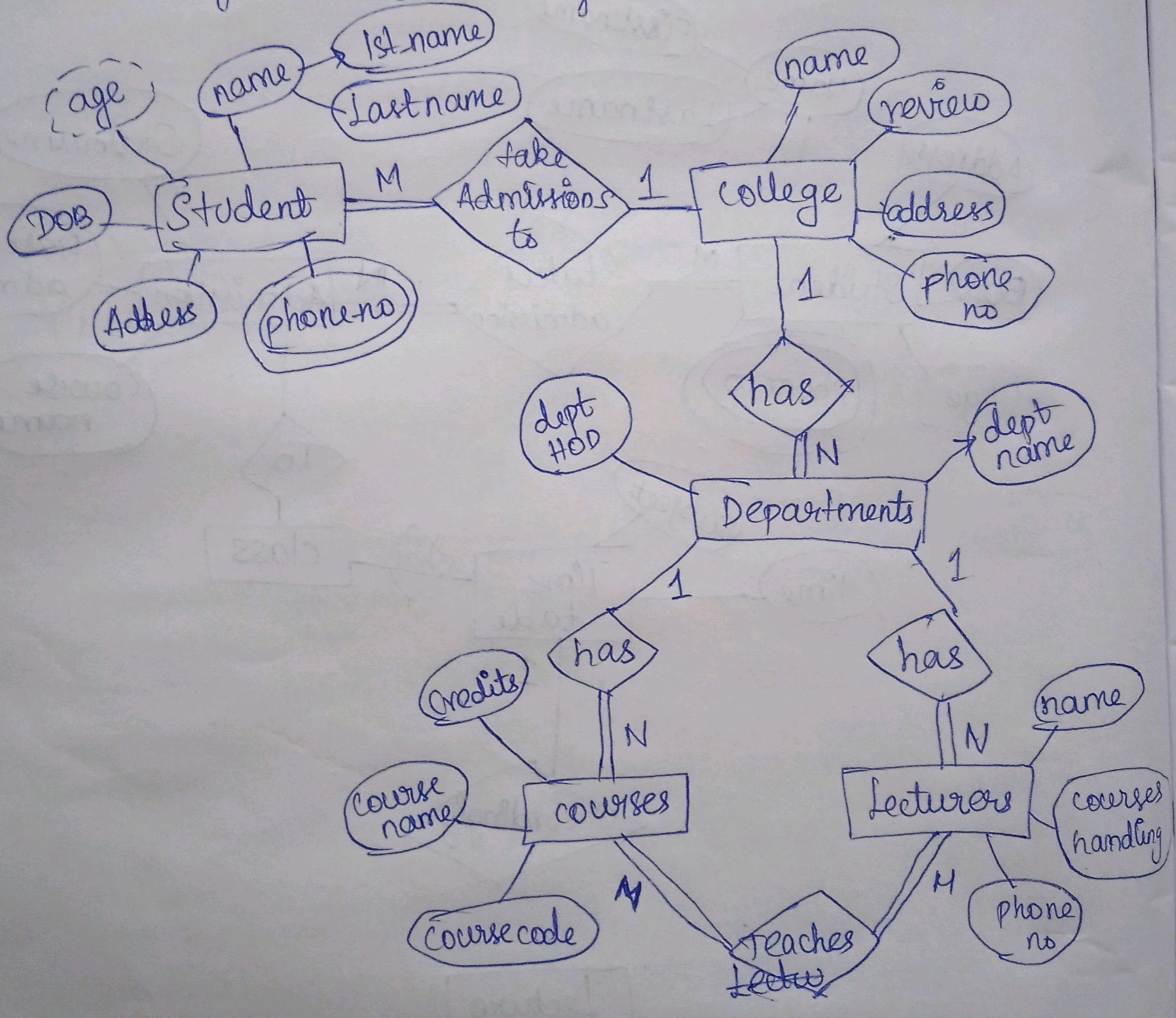
Emp_Id	Skills
1	Java
1	SQL

part B 5)

1) Entity relationship model for hospital management system



part B 6) College management System



b) SPECIALIZATION AND GENERALIZATION part B 7)

GENERALIZATION

- Generalization is the process of generalizing the entities which contain the properties of all the generalized entities.
- It is a bottom up approach, in which two lower level entities combine to form a higher level entity.
- Generalization is the reverse process of Specialization.

- It defines a general entity type from a set of specialized entity type.
- It minimizes the difference between the entities by identifying the common features.

In the below example, Tiger, Lion, Elephant can all be generalized as Animals.

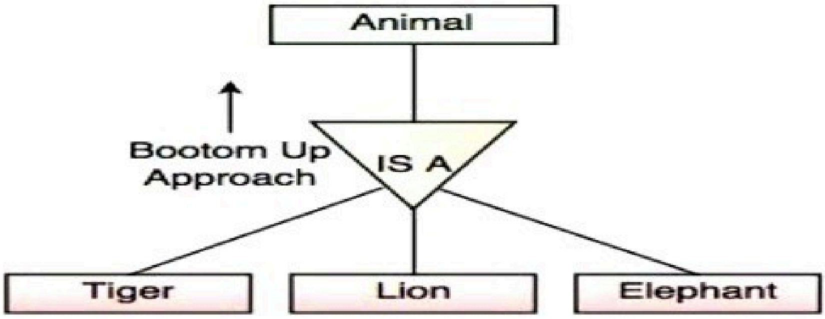


Fig. Generalization

SPECIALIZATION

- Specialization is a process that defines a group entities which is divided into sub groups based on their characteristic.
- It is a top down approach, in which one higher entity can be broken down into two lower level entity.
- It maximizes the difference between the members of an entity by identifying the unique characteristic or attributes of each member.
- It defines one or more sub class for the super class and also forms the superclass/subclass relationship.

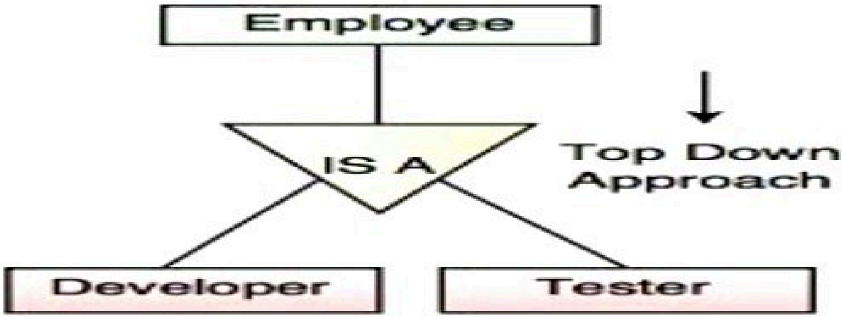


Fig. Specialization

d) AGGREGATION

- Aggregation is a process that represent a relationship between a whole object and its component parts.
- It abstracts a relationship between objects and viewing the relationship as an object.
- It is a process when two entities are treated as a single entity.

  In the below example, the relation between College and Course is acting as an Entity in Relation with Student.

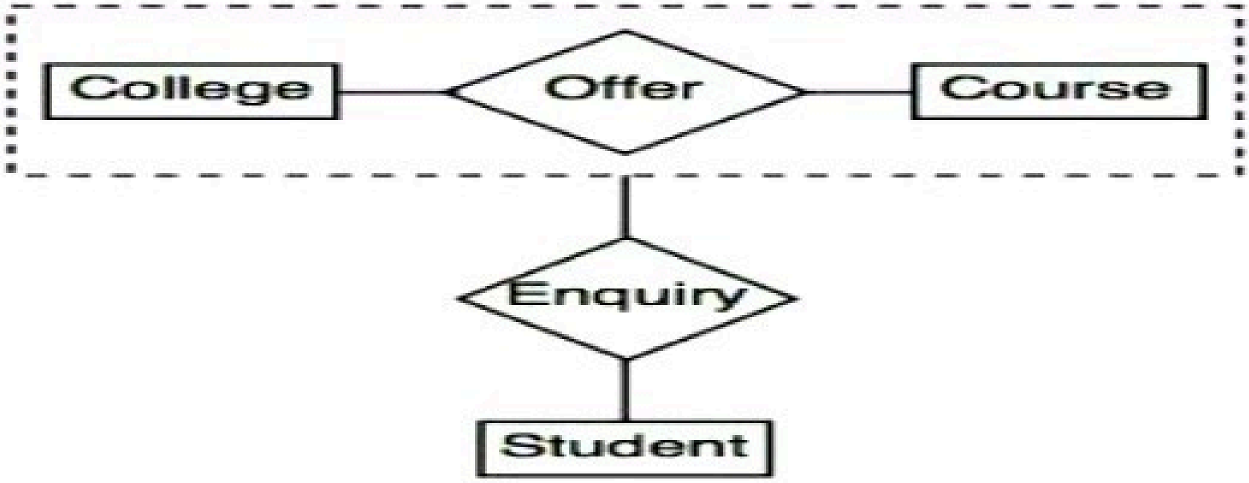


Fig. Aggregation

part B 8) same answer as 4)