Entity-Relationship Model

Dr. Geeta Kasana
Assistant Professor
CSED ,Thapar Institute of Engineering & Technology

How to design the database?

There are two approaches

- E-R Modeling: Identifying entity and relations
- Normalization: Refinement of database designing

Entity-Relation Model

E-R Model

- The Entity-Relationship (ER) model was originally proposed by Peter in 1976
- The ER model is a conceptual data model that views the real world as entities and relationships.
- A basic component of the model is the Entity-Relationship diagram, which is used to visually represent data objects.

Basic Constructs of E-R Modeling

- **Entities**
- Entities are the principal data object about which information is to be collected. Entities are usually recognizable concepts, either concrete or abstract, such as person, places things events, which have relevance to the database. Some specific examples of entities are EMPLOYEES, PROJECTS, and INVOICES. An entity is analogous to a table in the relational model.

Basic Constructs of E-R Modeling

Tangible Entity: Tangible Entities are those entities which exist in the real world physically. ***Example:*** Person, car, etc.

Intangible Entity: Intangible Entities are those entities which exist only logically and have no physical existence. ***Example:*** Bank Account, etc.

Basic Constructs of E-R Modeling

Strong Entity

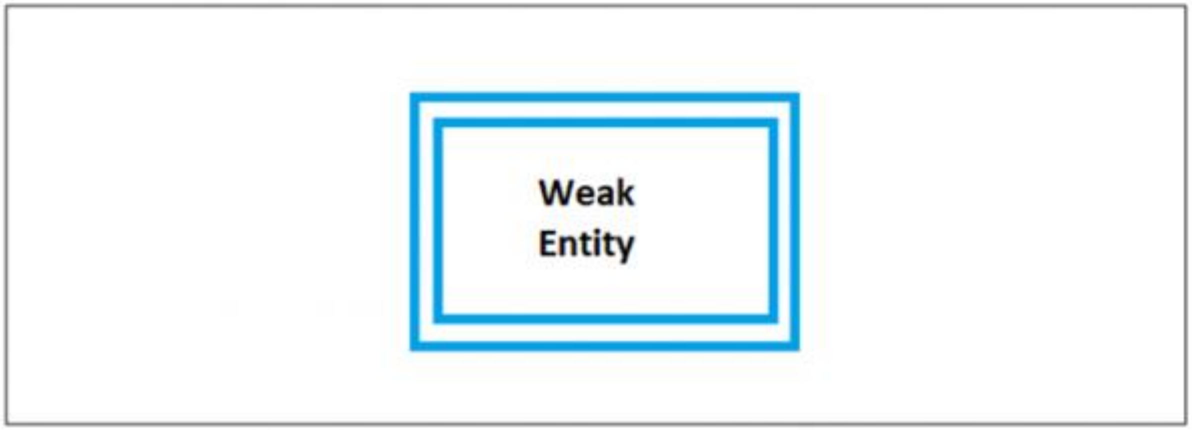
The strong entity has a primary key. Weak entities are dependent on strong entity. Its existence is not dependent on any other entity.

Basic Constructs of E-R Modeling

Weak Entity

The weak entity in DBMS do not have a primary key and are dependent on the parent entity. It mainly depends on other entities.

Weak Entity is represented by double rectangle



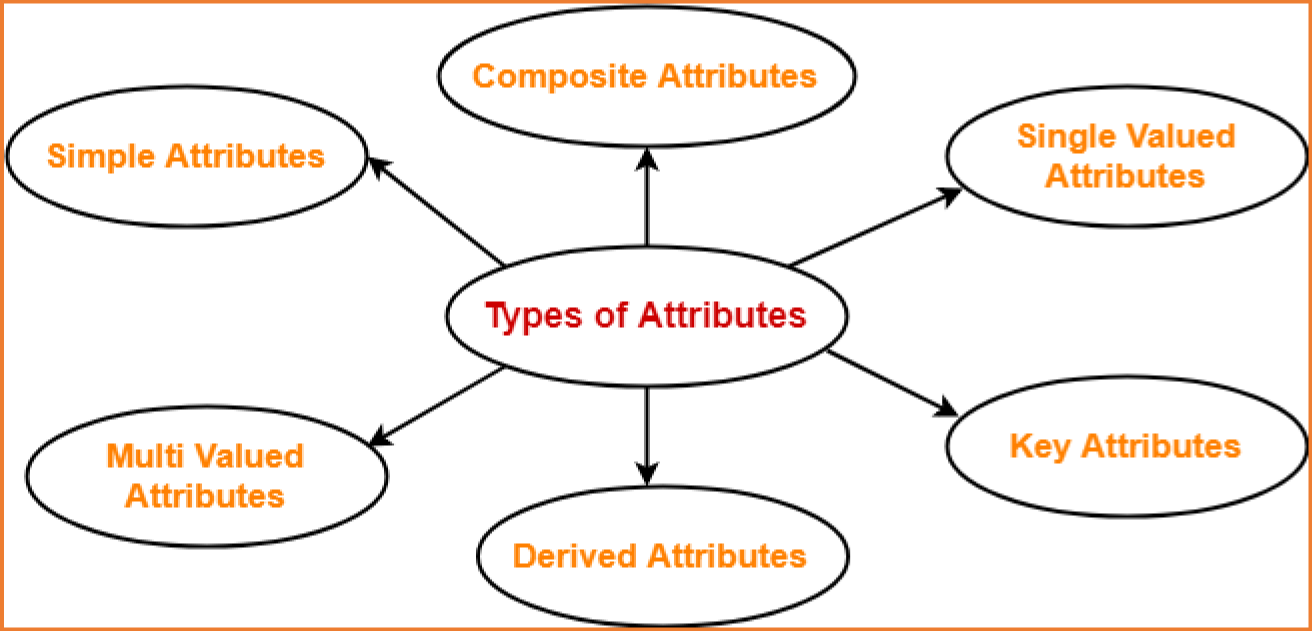
Attributes

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. The domain of an attribute is the collection of all possible values an attribute can have.

We can classify attributes as following:

- Simple
- Composite
- Single-values
- Multi-values
- Derived

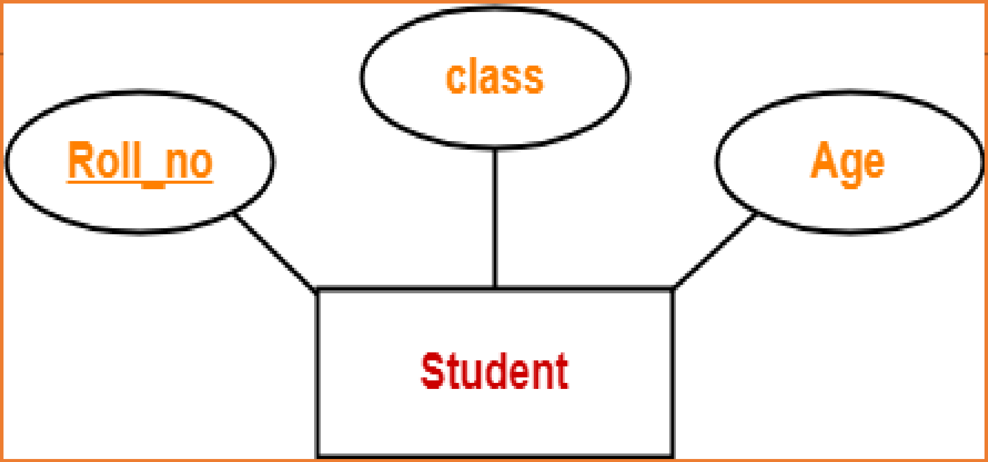
Attributes


Composite Attributes
Simple Attributes
Single Valued Attributes
Types of Attributes
Multi Valued Attributes
Key Attributes
Derived Attributes

1. **Simple** Attributes

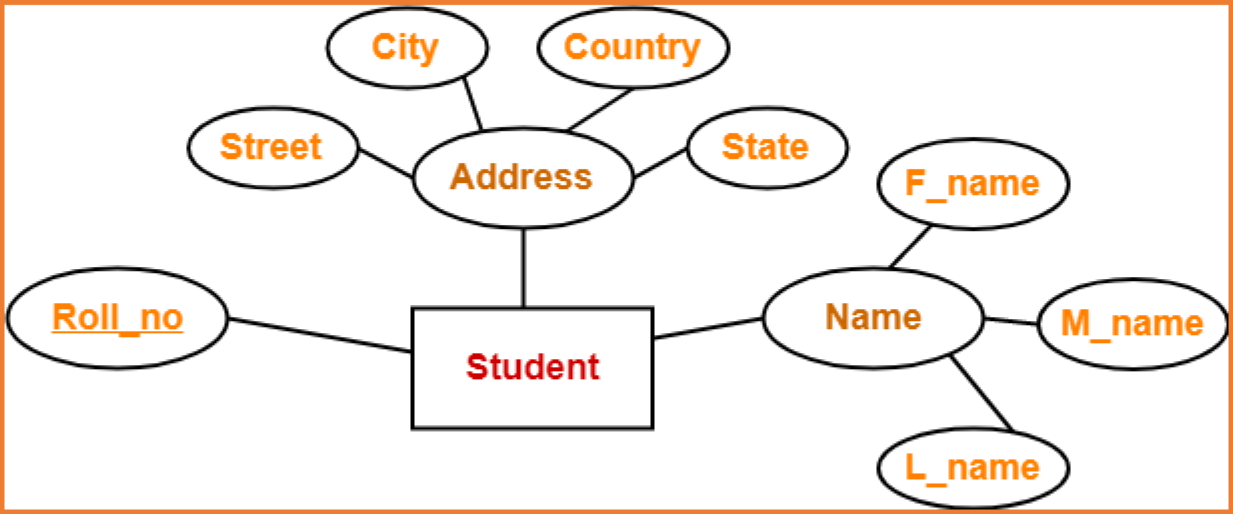
–
Simple attributes are those attributes which can not be divided further.

–
Example-



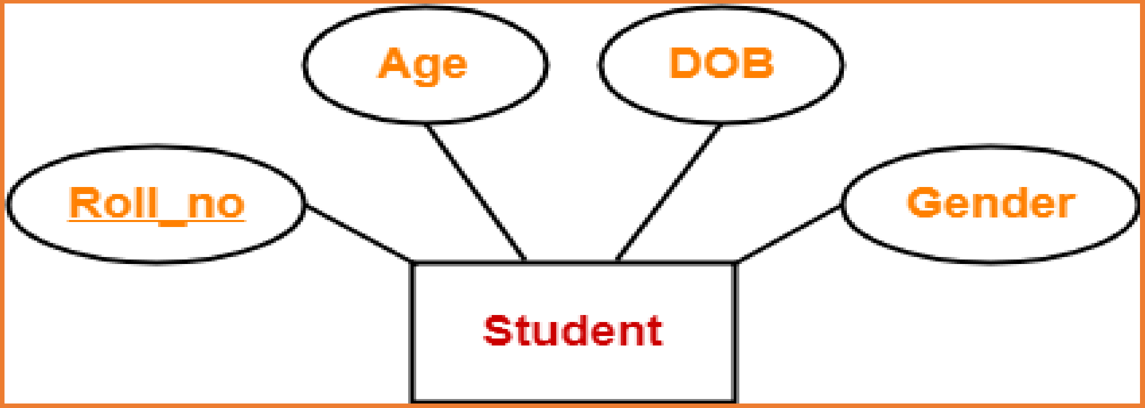
2. Composite Attributes

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



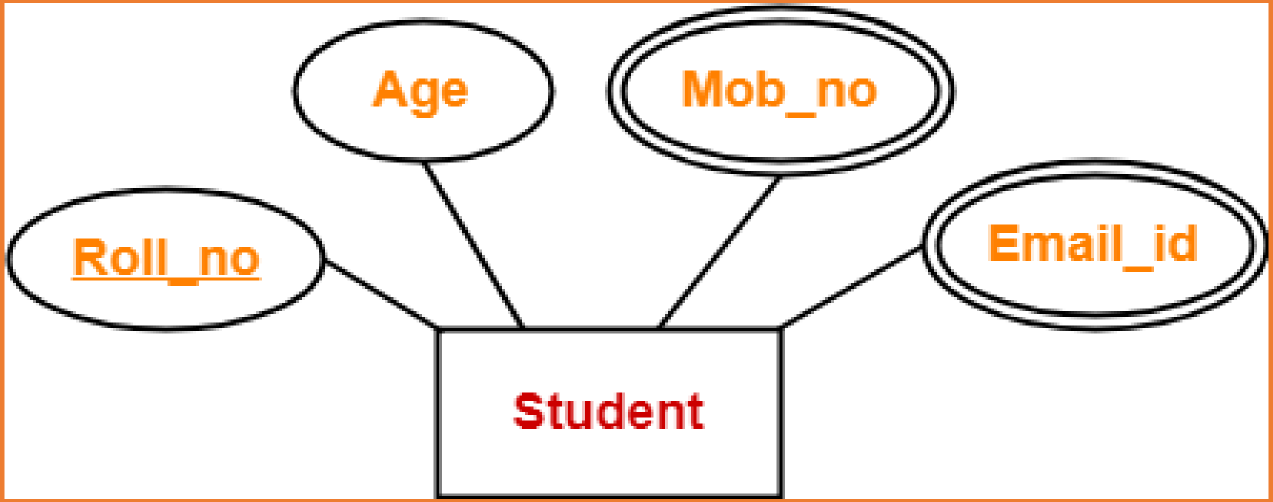
3. Single Valued **Attributes**

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



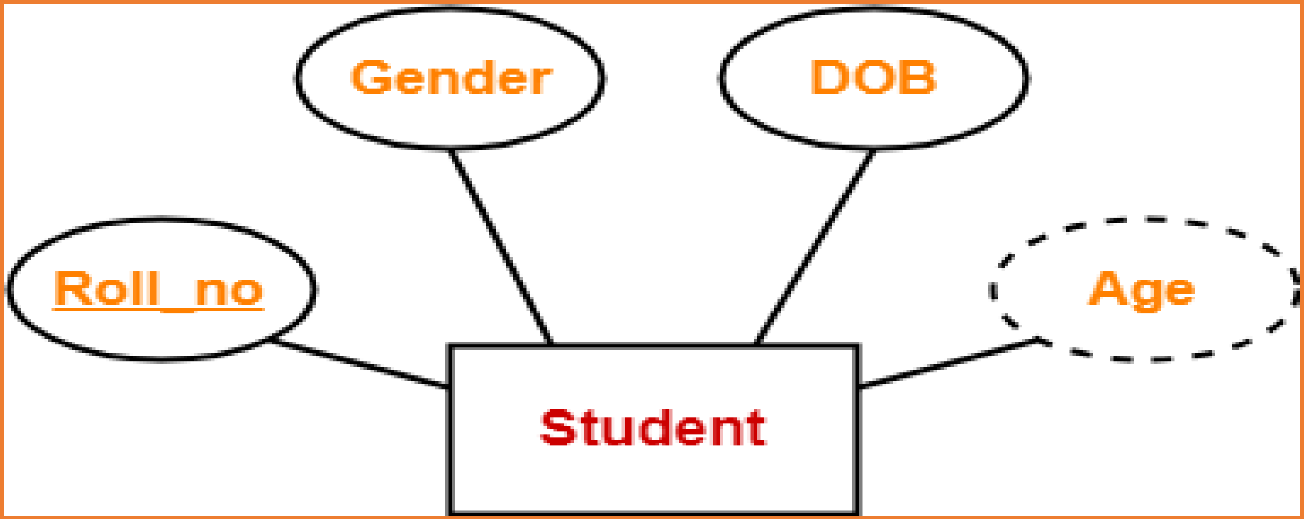
4. Multi Valued **Attributes**

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



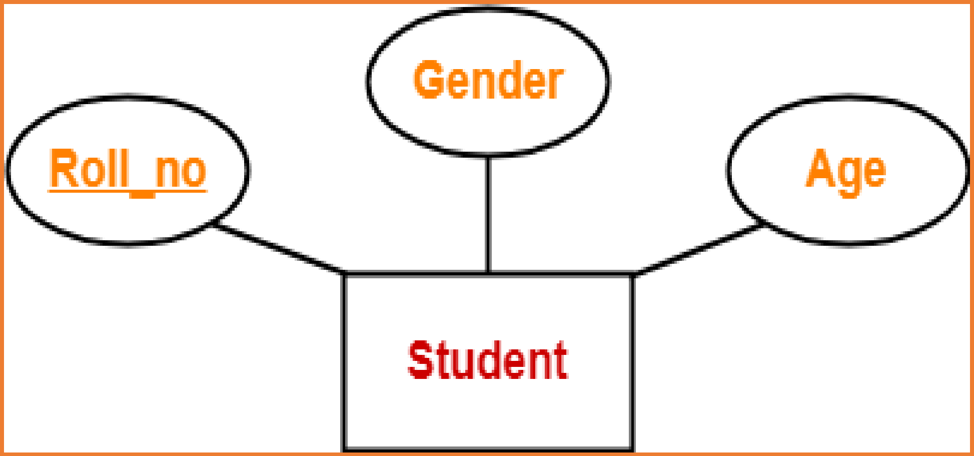
5. Derived **Attributes**

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

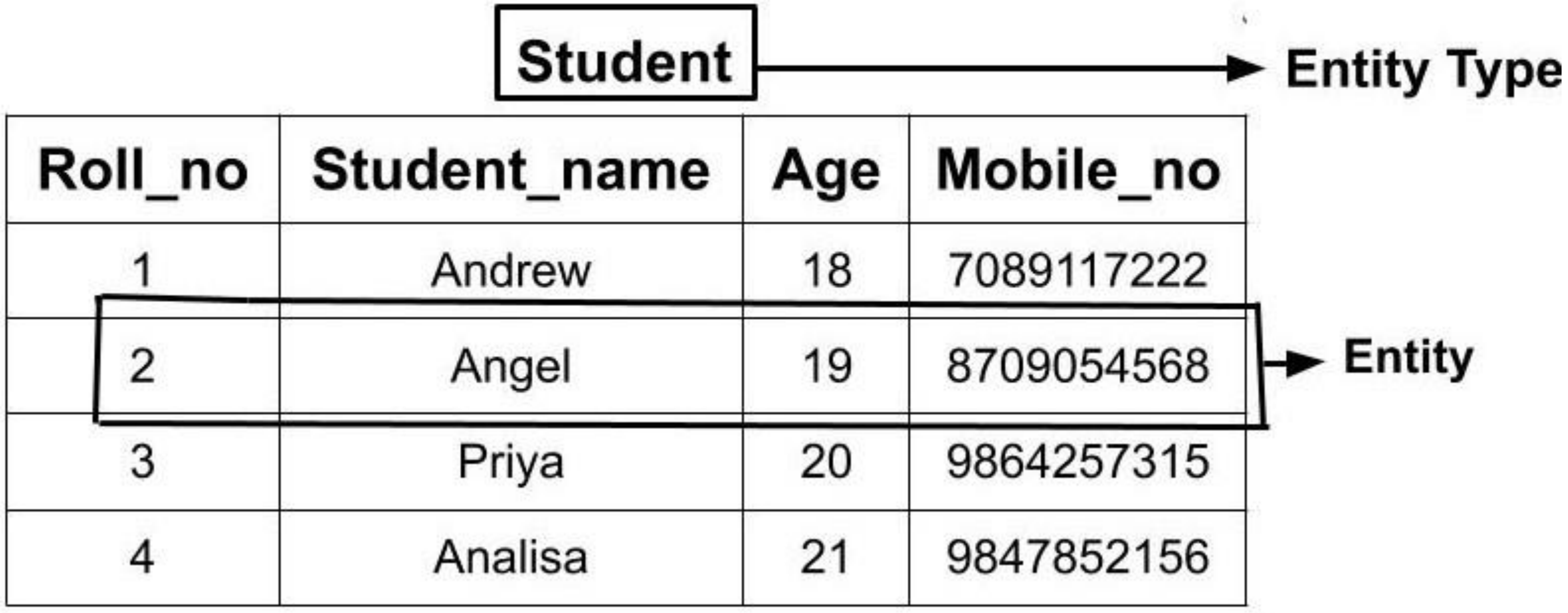


6. Key **Attributes**

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



Entity

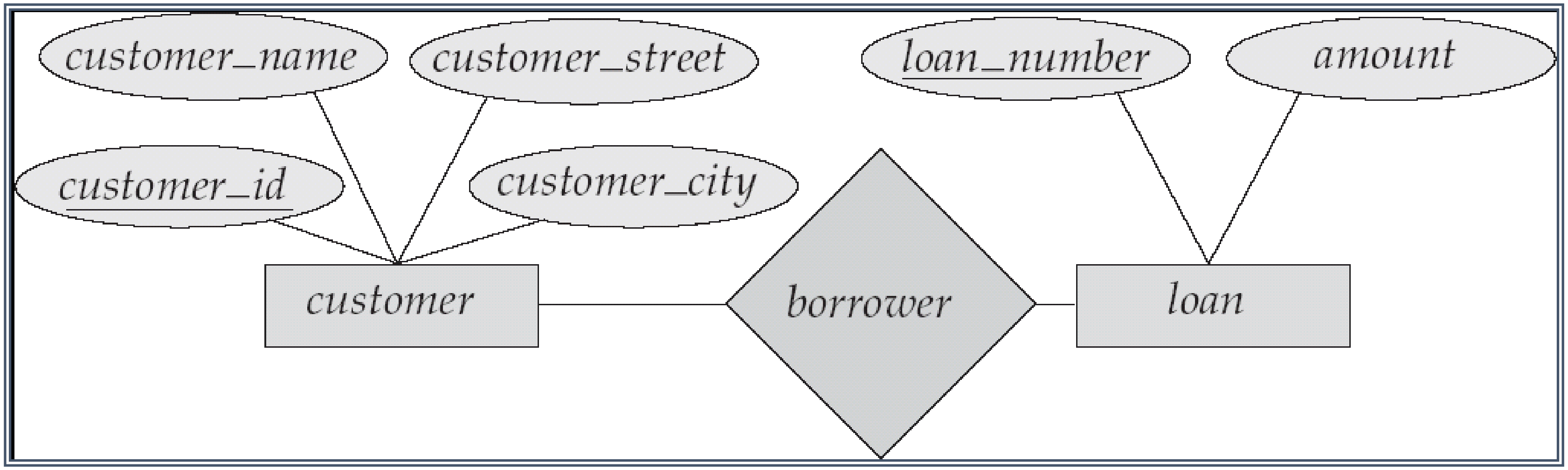


Student → Entity Type

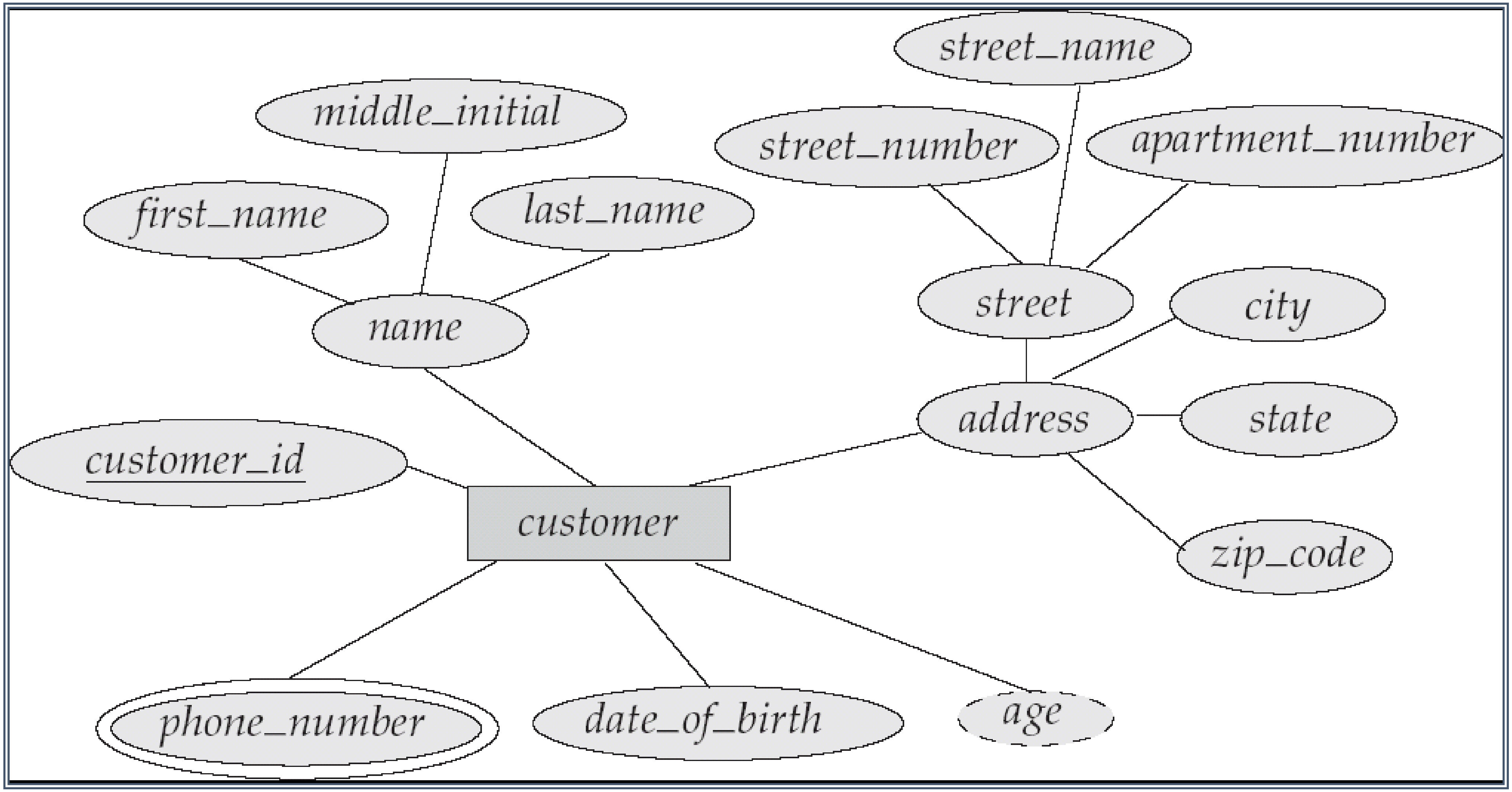
Roll_no	Student_name	Age	Mobile_no
1	Andrew	18	7089117222
2	Angel	19	8709054568
3	Priya	20	9864257315
4	Analisa	21	9847852156

→ Entity

E-R Diagrams


customer_name
customer_street
loan_number
amount
customer_id
customer_city
customer
borrower
loan

E-R Diagram With Composite, Multivalued, and Derived Attributes



Relationships

A Relationship represents an association between two or more entities. Relationships are classified in terms of degree, connectivity, cardinality, and existence. An example of a relationship would be:

¨ Employees are assigned to projects

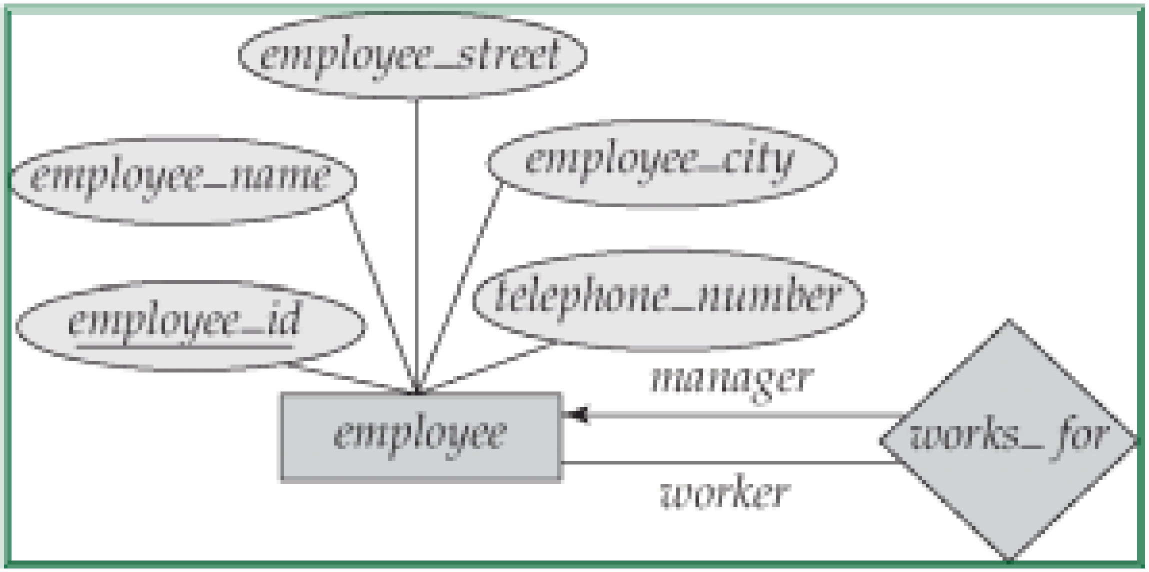
¨ Projects have subtasks

¨ Departments manage one or more projects

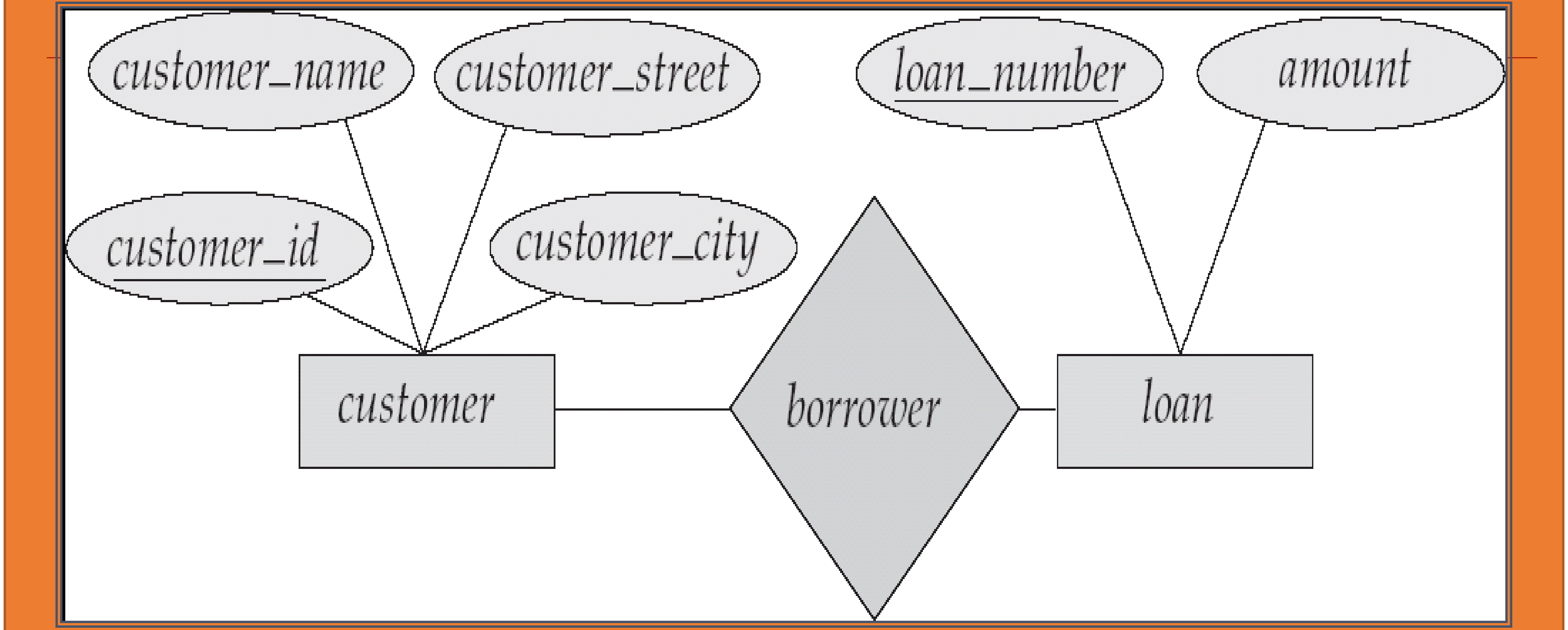
Degree of a Relationship

- The degree of a relationship is the number of entities associated with the relationship. The n-ary relationship is the general form for degree n. Special cases are the binary, and ternary, where the degree is 2, and 3, respectively.
- Binary relationships, the association between two entities are the most common type in the real world.
- A recursive binary relationship occurs when an entity is related to itself. An example might be "some employees are married to other employees".
- A ternary relationship involves three entities and is used when a binary relationship is inadequate. Many modeling approaches recognize only binary relationships. Ternary or n-ary relationships are

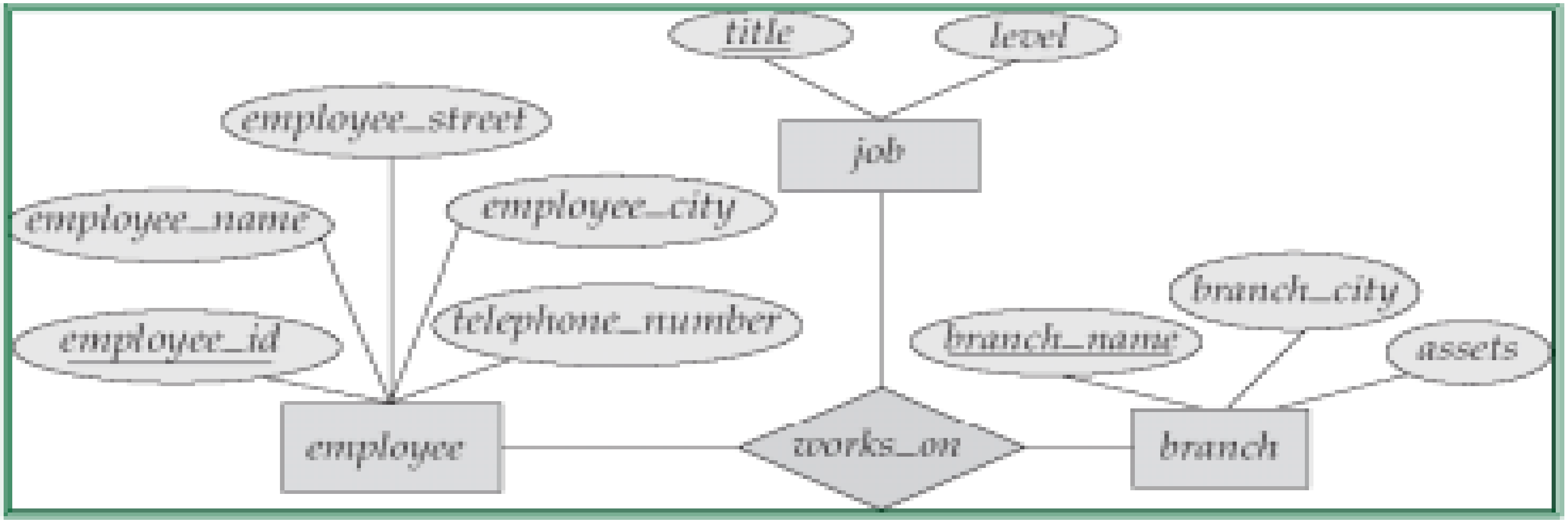
Recursive Relationship



Binary Relationship



Ternary Relationship



Connectivity and Cardinality

The connectivity of a relationship describes the mapping of associated entity instances in the relationship. The values of connectivity are "one" or "many". The cardinality of a relationship is the actual number of related occurrences for each of the two entities.

The basic types of connectivity for relations are:

¨ One to One (1:1)

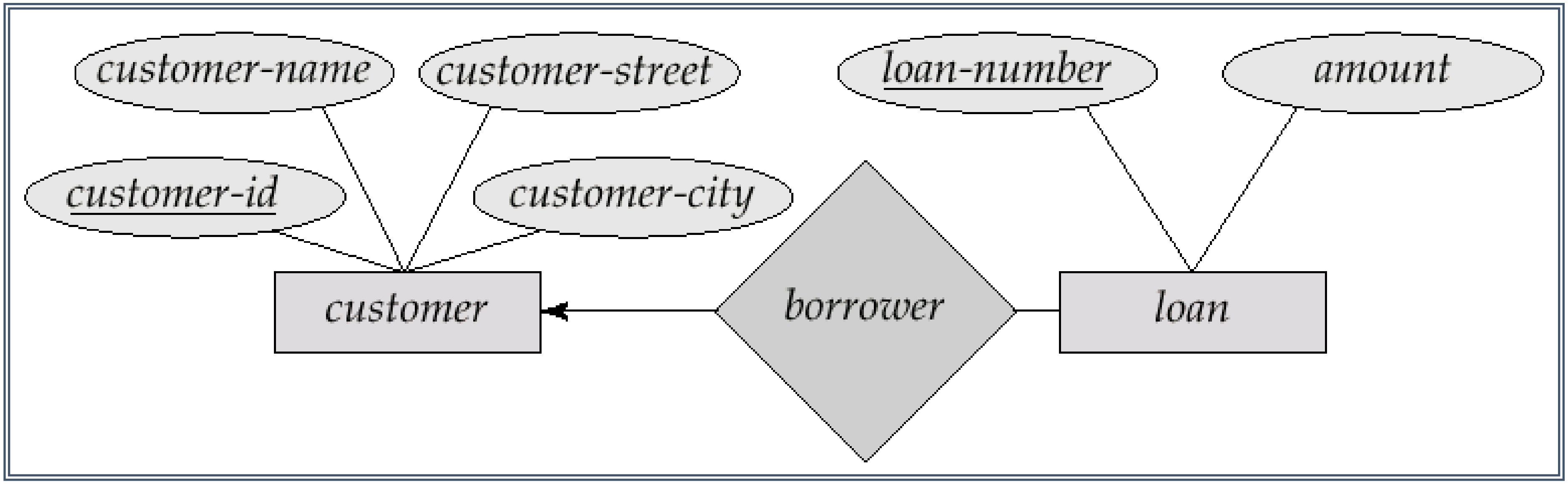
¨ One to Many (1:M)

¨ Many to One (M:1)

¨ Many to Many (M:M)

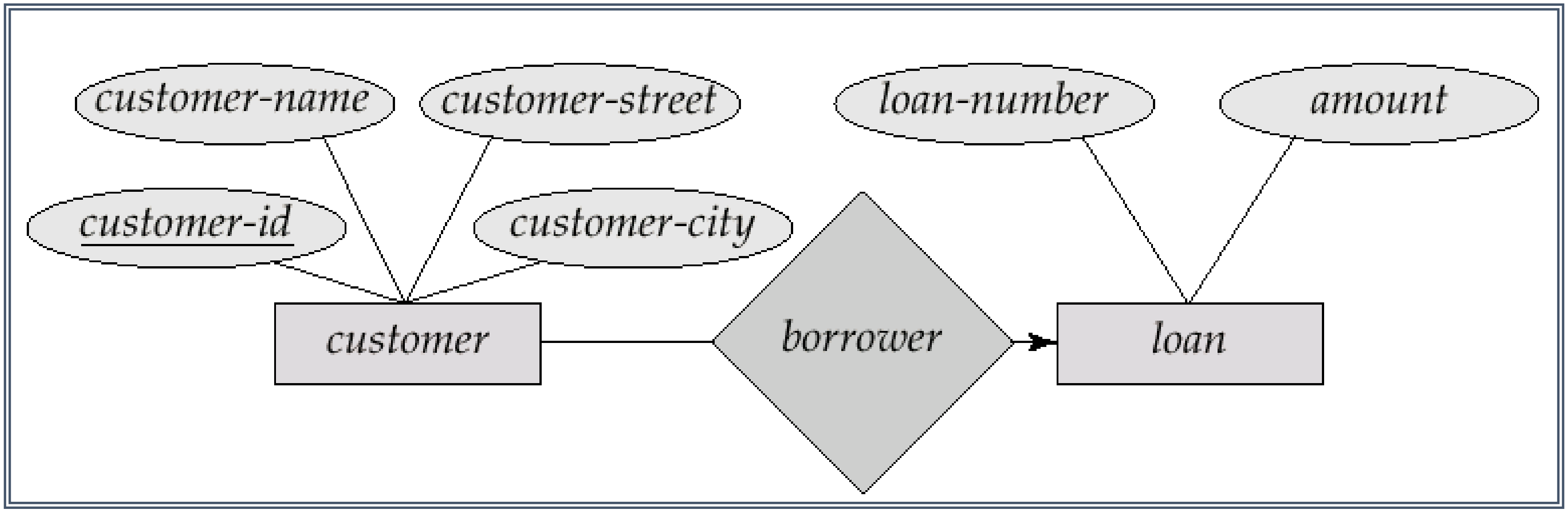
One-To-Many Relationship

- In the one-to-many relationship a loan is associated with at most one customer via *borrower*, a customer is associated with several (including 0) loans via *borrower*



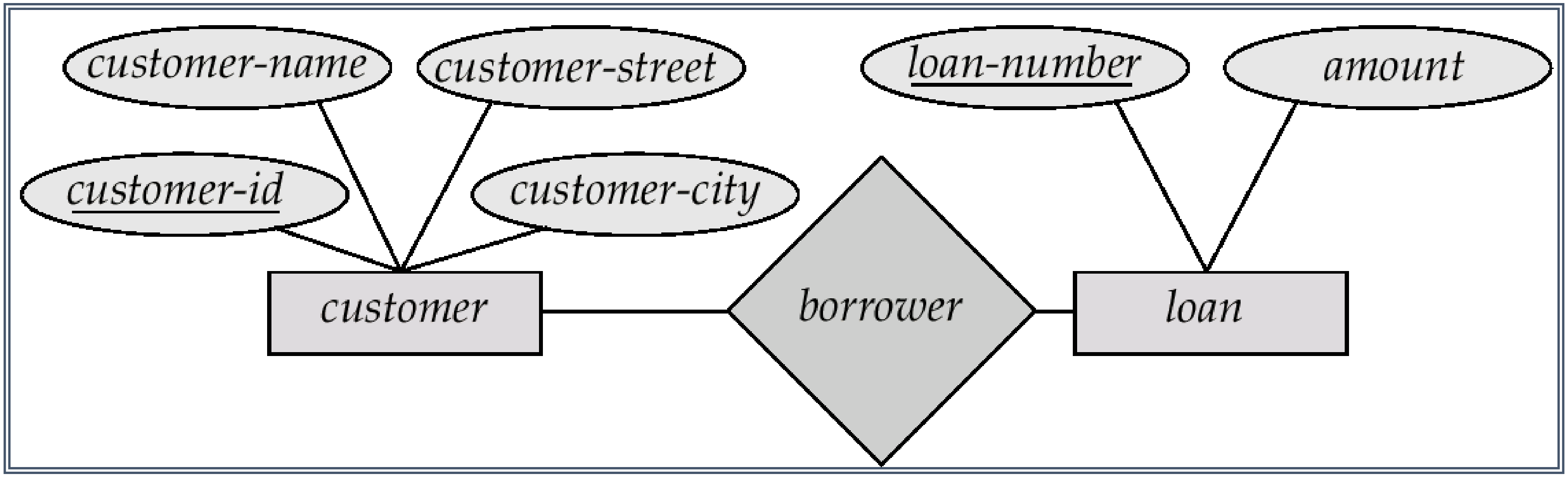
Many-To-One Relationships

- In a many-to-one relationship a loan is associated with several (including 0) customers via *borrower*, a customer is associated with at most one loan via *borrower*



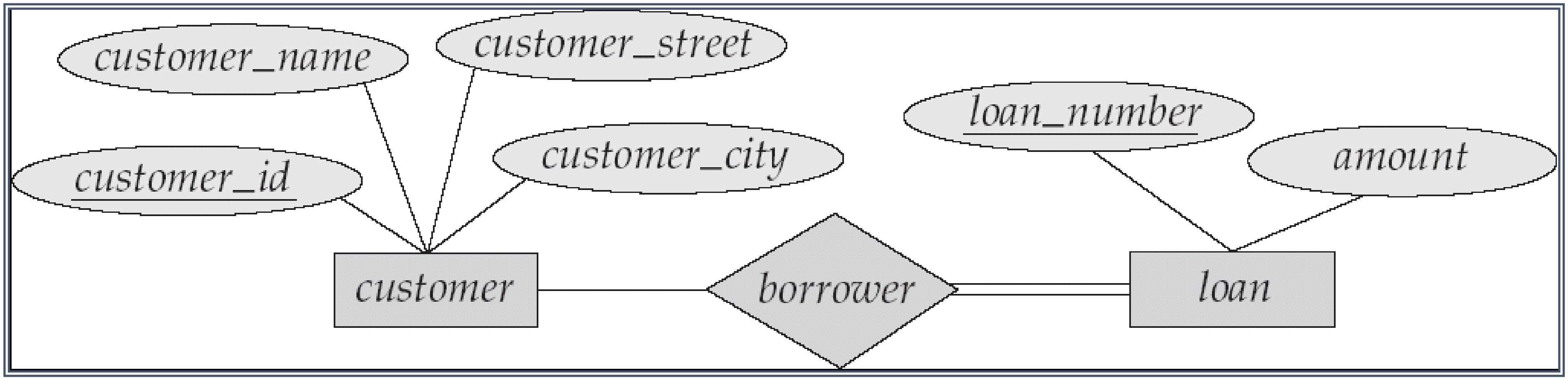
Many-To-Many Relationship

- A customer is associated with several (possibly 0) loans via borrower
- A loan is associated with several (possibly 0) customers via borrower



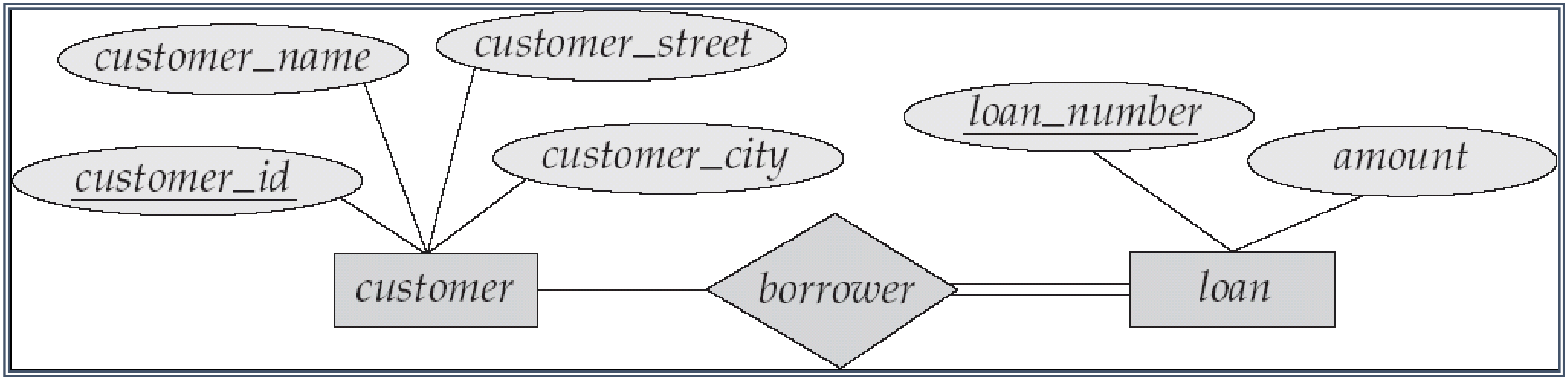
Participation of an Entity Set in a Relationship Set

- **Total participation** (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set
  - E.g. participation of loan in borrower is total
    - every loan must have a customer associated to it via borrower



Participation of an Entity Set in a Relationship Set

- **Partial participation** (indicated by single line): some entities in the entity set may not participates in any relationship in the relationship set
  - Example: participation of customer in borrower is partial

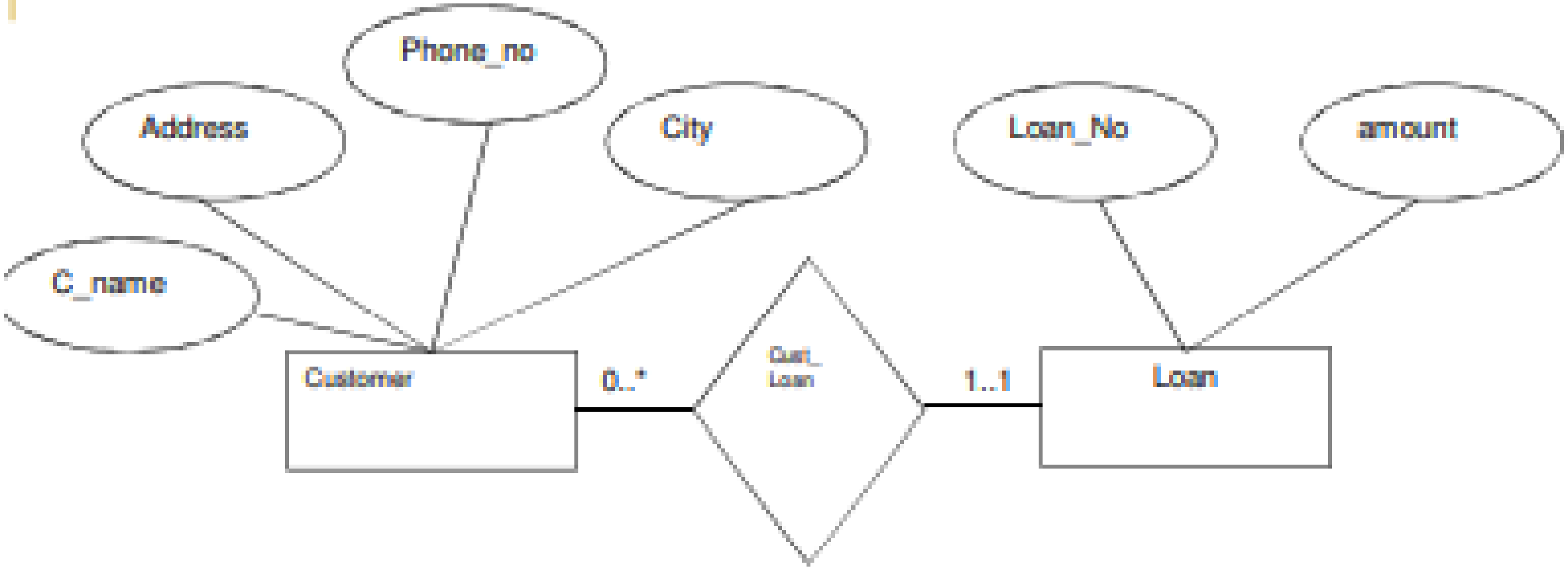


Represenation of Cardinality in ER

The cardinality of a relationship is the actual number of related occurrences for each of the two entities. E-R diagrams also provide a way to indicate cardinality of the relationship. An edge between an entity set and a relationship set can have an associated minimum and maximum cardinality, shown in the form l..h, where l is the minimum and h the maximum cardinality. A minimum value of 1 indicates total participation of the entity set in the relationship set. A maximum value of 1 indicates that the entity participates in at most one relationship, while a maximum value * indicates no limit. The label 1..* on an edge is equivalent to a double line

Represenation of Cardinality in ER

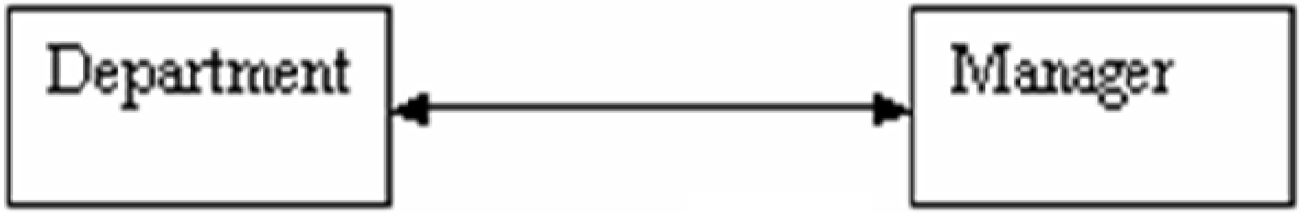
the edge between loan and Cust_Loan has a cardinality constraint of 1..1, meaning the minimum and the maximum cardinality are both 1. That is, each loan must have exactly one associated customer. The limit 0..* on the edge from customer to Cust_Loan indicates that a customer can have zero or more loans. Thus, the relationship Cust_Loan is one to many from customer to loan, and further the participation of loan in Cust_Loan is total.



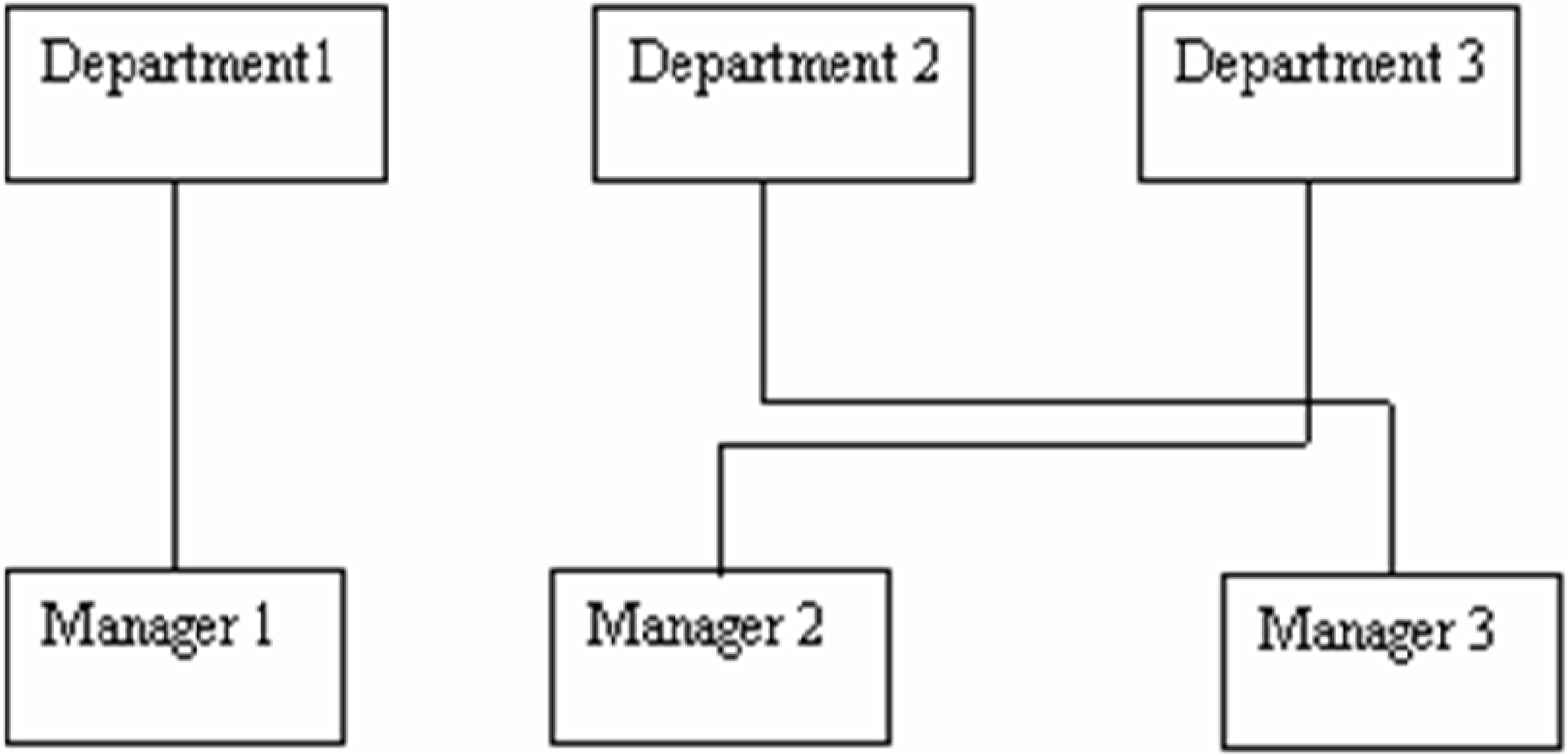
Direction

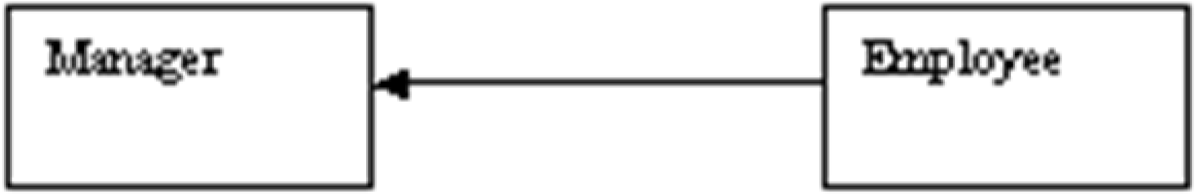
The direction of a relationship indicates the originating entity of a relationship. The entity from which a relationship originates is the parent entity; the entity where the relationship terminates is the child entity.

The type of the relation is determined by the direction of line connecting relationship component and the entity. To distinguish different types of relation, we draw either a directed line or an undirected line between the relationship set and the entity set. Directed line is used to indicate one occurrence and undirected line is used to indicate many occurrences in a relation as shown in next case.

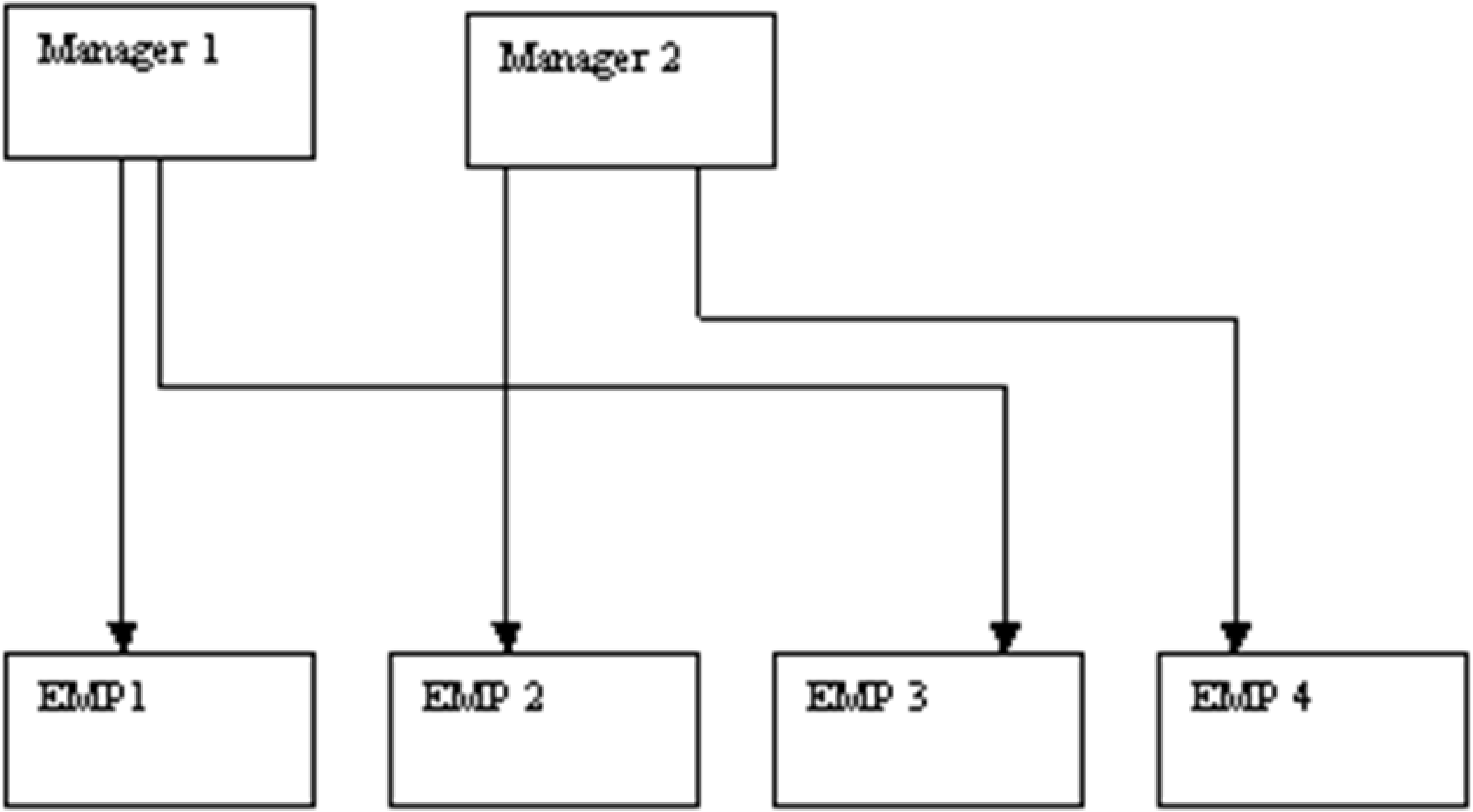


One-to-One relationship





One-to-Many relationship



E-R Notation

¨ Entities are represented by labeled rectangles. The label is the name of the entity. Entity names should be singular nouns.

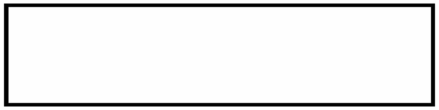
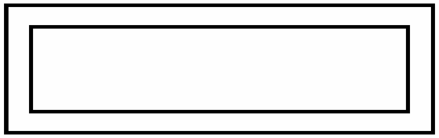
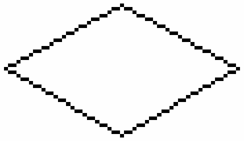
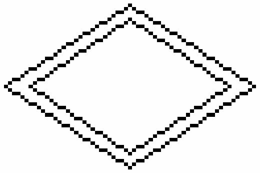
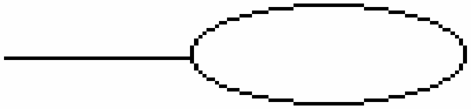
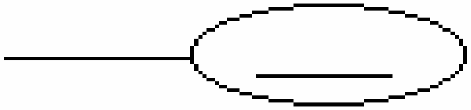
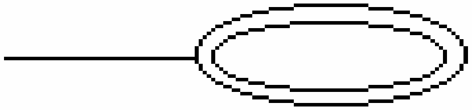
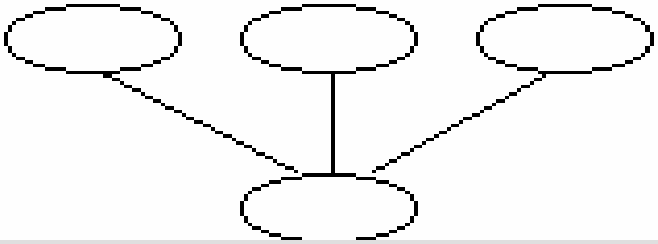
¨ Attributes are represented by Ellipses.

¨ A solid line connecting two entities represents relationships. The name of the relationship is written above the line. Relationship names should be verbs and diamonds sign is used to represent relationship sets.

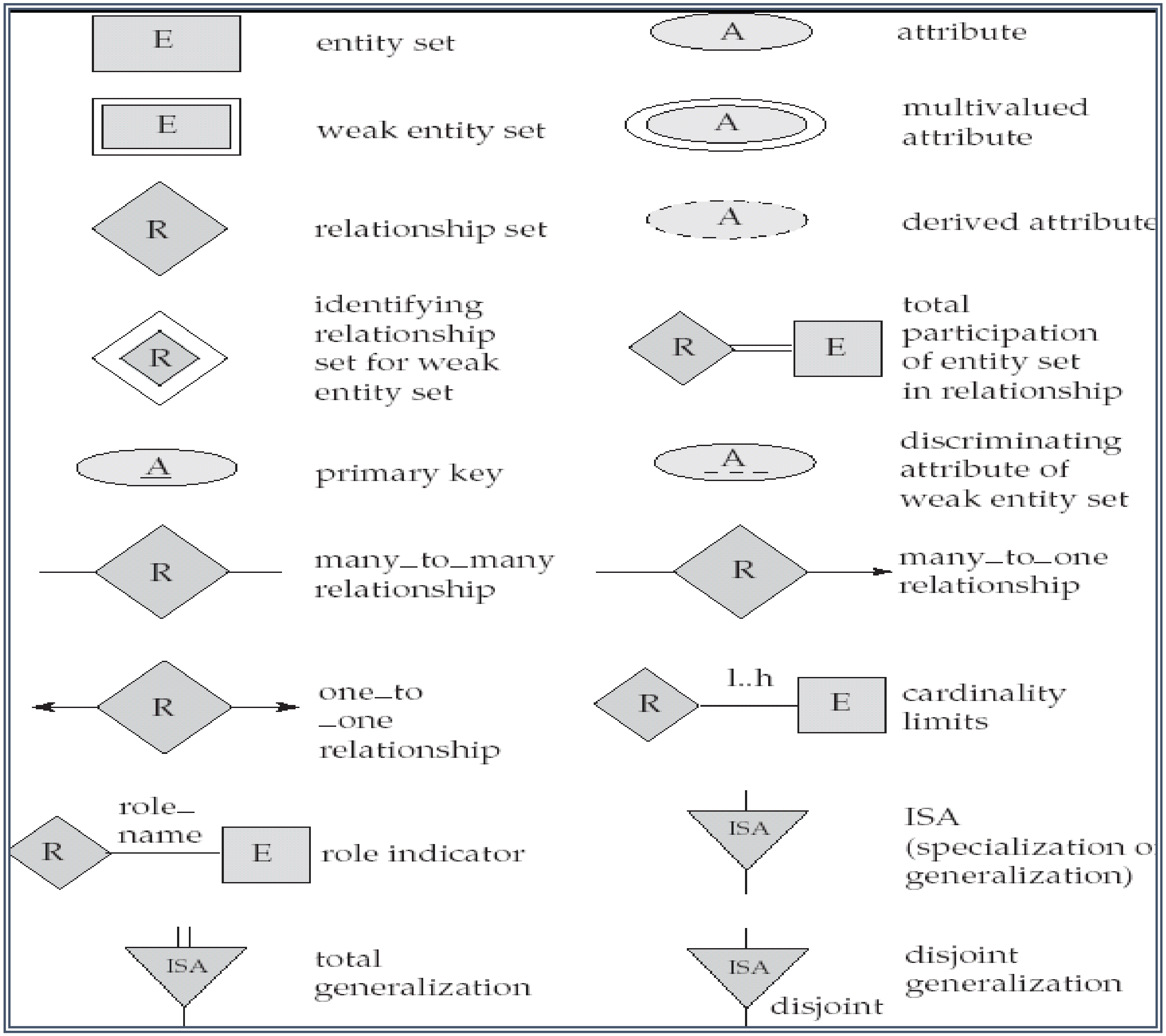
¨ Attributes, when included, are listed inside the entity rectangle. Attributes, which are identifiers, are underlined. Attribute names should be singular nouns.

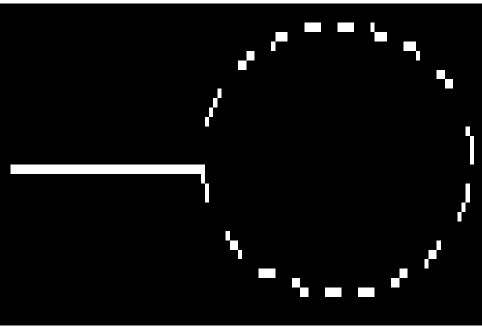
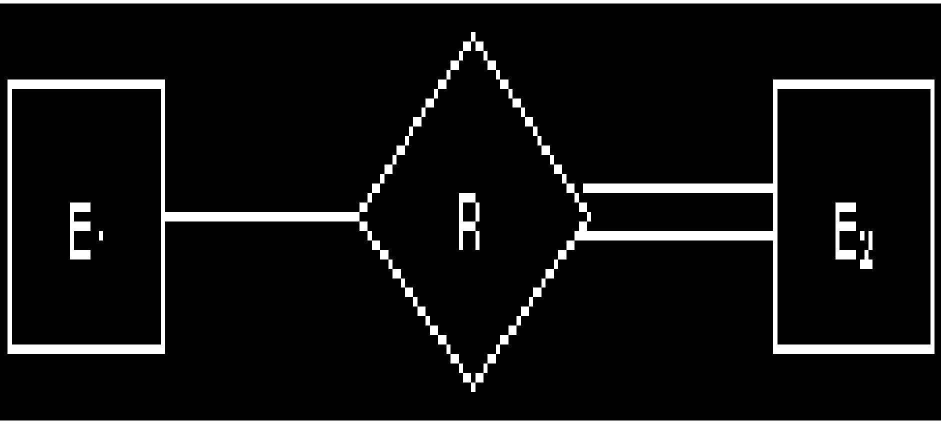
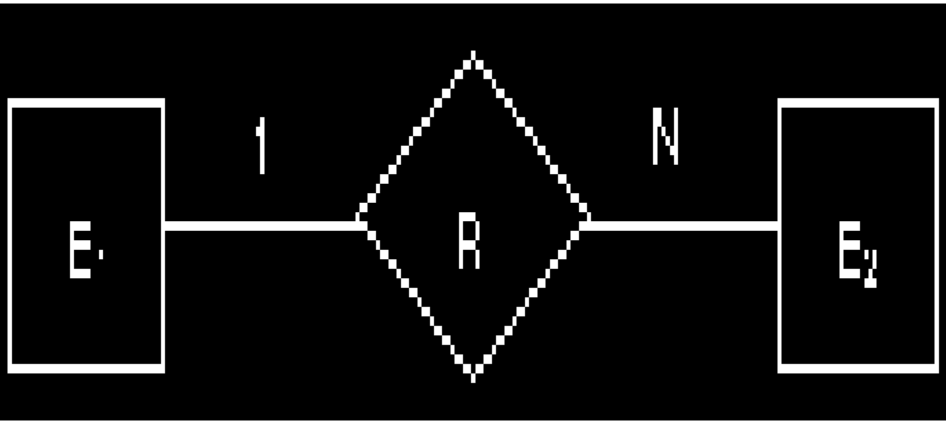
¨ Multi-valued attributes are represented by double ellipses.

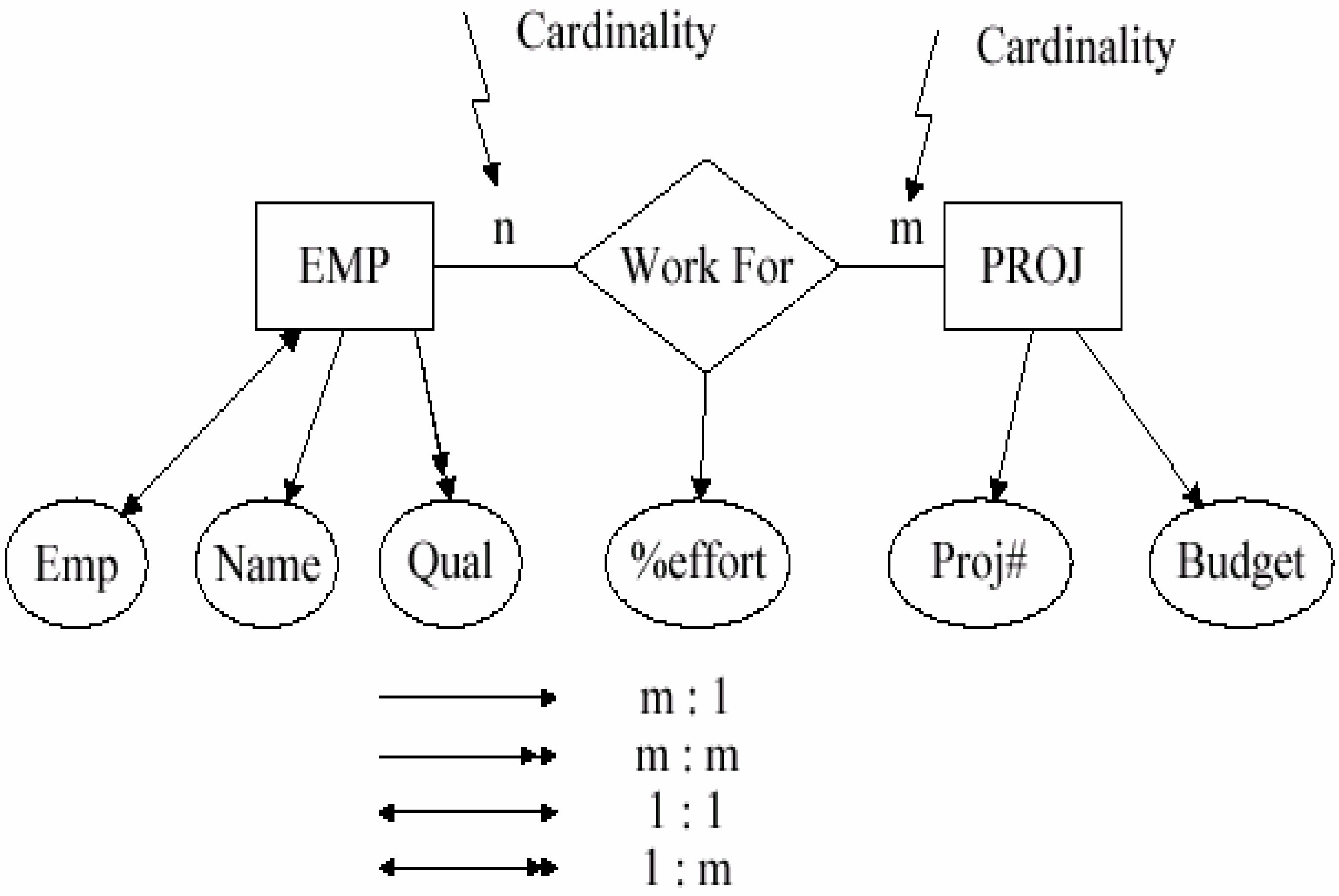
¨ Directed line is used to indicate one occurrence and undirected line is used to indicate many occurrences in a relation.

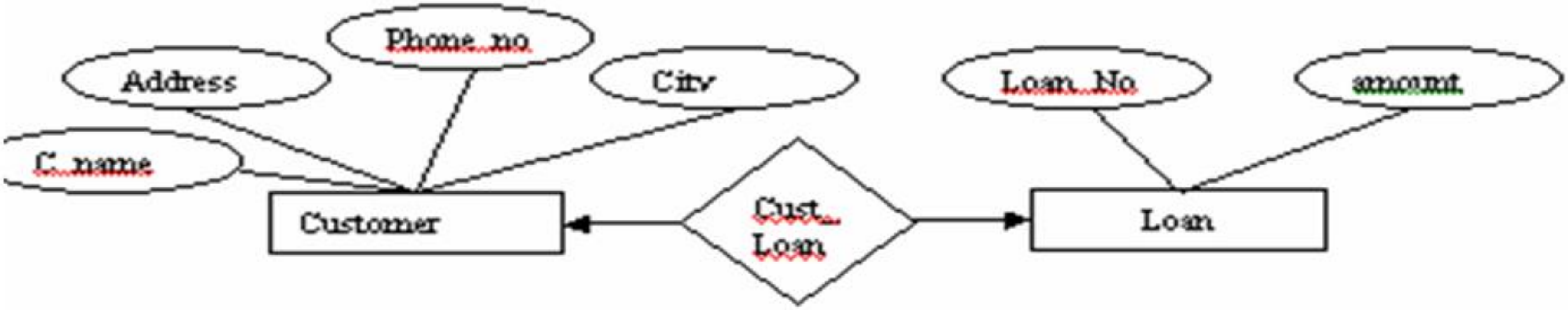
Symbol	Meaning
	Entity Type
	Weak Entity Type
	Relationship Type
	Identifying Relationship Type
	Attribute
	Key Attribute
	Multivalued Atrribute
	Composite Attribute

Summary of Symbols Used in E-R Notation

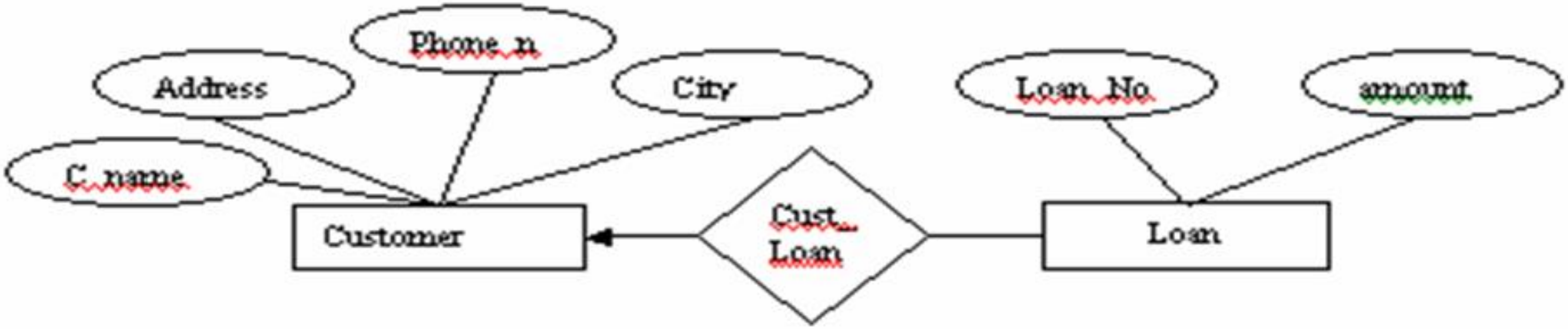


	Derived Atribute
	Total Participation of E2 in R
	Cardinality Relation 1:N for E1:E2 in R

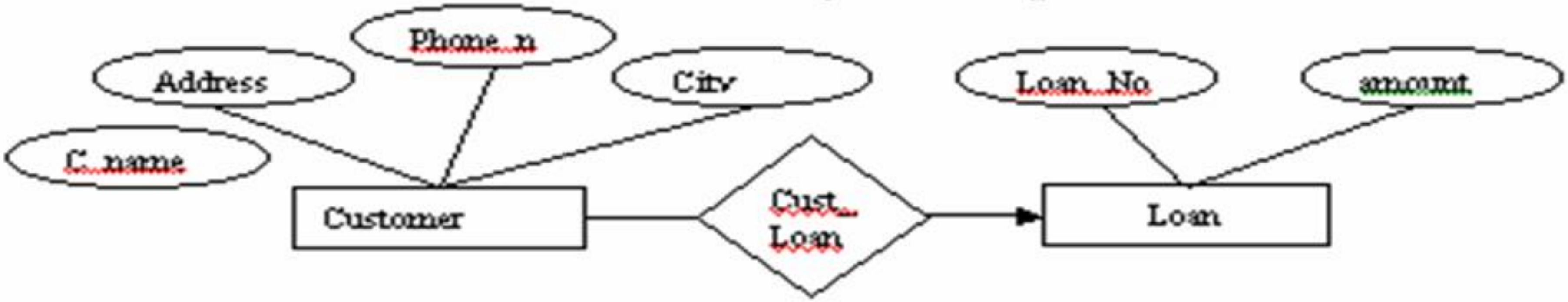

Cardinality
Cardinality
n
m
EMP
Work For
PROJ
Emp
Name
Qual
%effort
Proj#
Budget
m : 1
m : m
1 : 1
1 : m



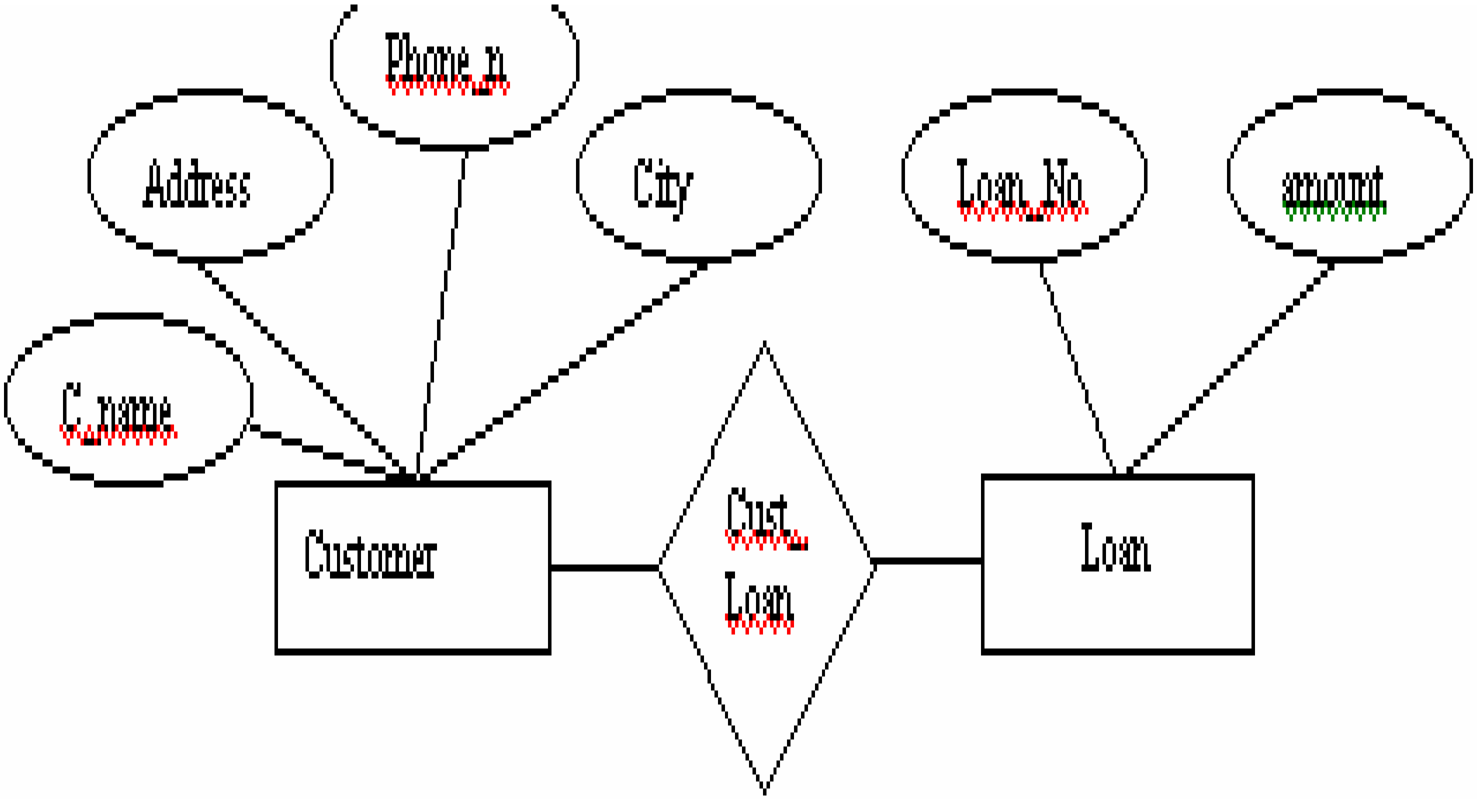
One-to-One Relationship



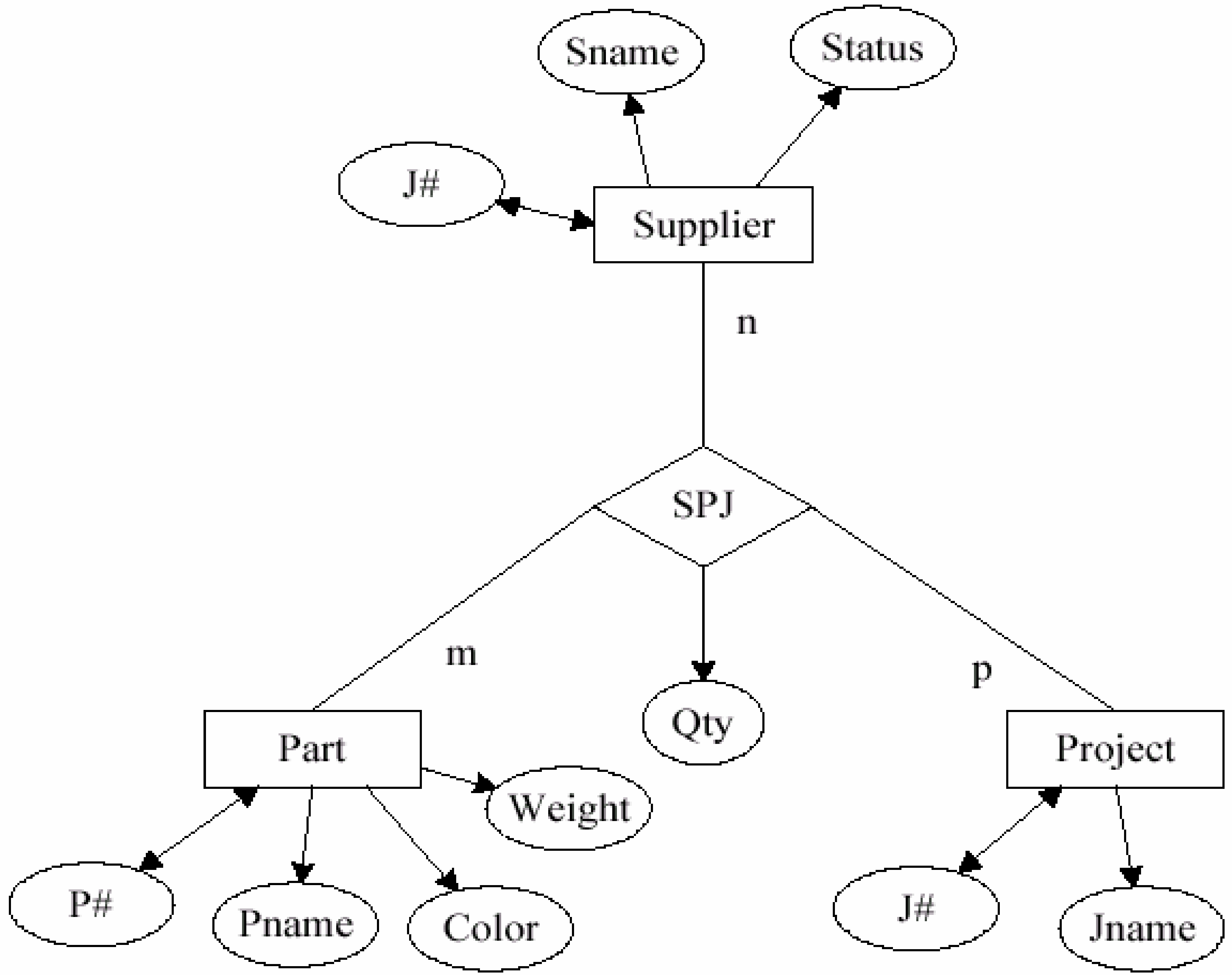
One-to-Many Relationship



Many-to-One Relationship



Many-to-Many Relationship


Sname
Status
J#
Supplier
n
SPJ
m
Qty
p
Part
Weight
Project
P#
Pname
Color
J#
Jname

Strong and Weak Entity Sets

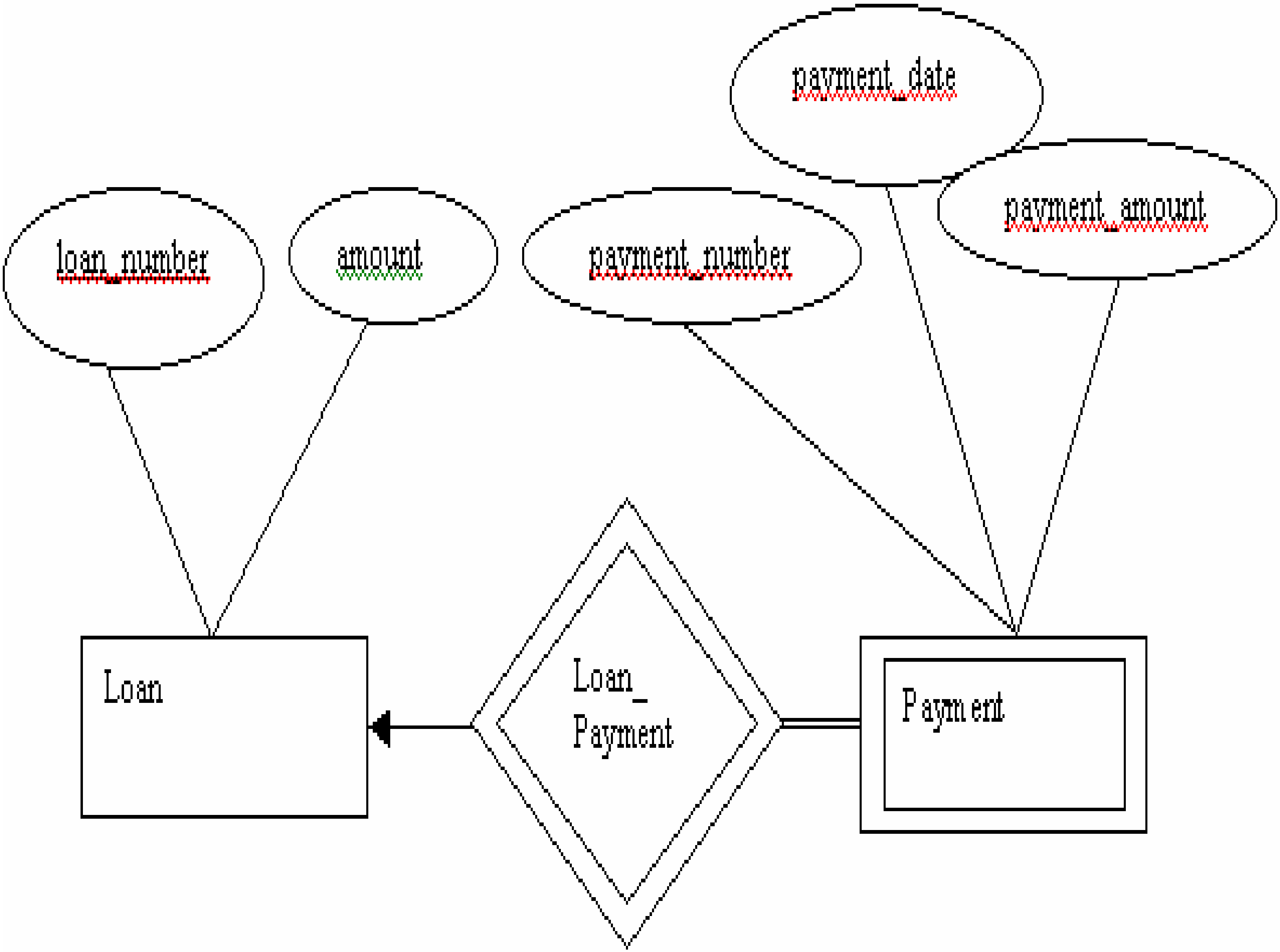
The entity set which does not has sufficient attributes to form a primary key is called as weak entity set.

An entity set that has a primary key is called as Strong entity set.

Consider an entity set Payment which has three attributes: payment_number, payment_date and payment_amount. Although each payment entity is distinct but payment for different loans may share the same payment number. Thus, this entity set does not have a primary key and it is a weak entity set. Each weak set must be a part of one-to-many relationship set.

Strong and Weak Entity Sets

A member of a strong entity set is called dominant entity and member of weak entity set is called as subordinate entity. A weak entity set does not have a primary key but we need a means of distinguishing among all those entries in the entity set that depend on one particular strong entity set. The discriminator of a weak entity set is a set of attributes that allows this distinction to be made. For example, payment_number is acts as discriminator for payment entity set. It is also called as the Partial key of the entity set.


payment_date
loan_number
amount
payment_number
payment_amount
Loan
Loan_
Payment
Paym ent

Steps to design E-R diagram

1. Identify the Entities
2. To find relationships among these entities
3. To identify the key attributes for every Entity.
4. To identify other relevant attributes
5. To draw the complete e-r diagram with all attributes including primary key

Case Study 1

Consider, a university contains many departments. Each department can offer any number of courses. Many teachers can work in a department. A teacher can work only in one department. For each department there is a Head. A teacher can be head of only one department. Each teacher can take any number of courses. A student can enroll for any number of courses. Each course can have any number of students.

Case Study 1

1. To identify the Entities

University, Department, Course, Teacher, Student

Consider University as single instance

2 Relationship:

Each course belong to one department(1:M)

Department & teacher(1:M)

Department Head and teacher(1:1)

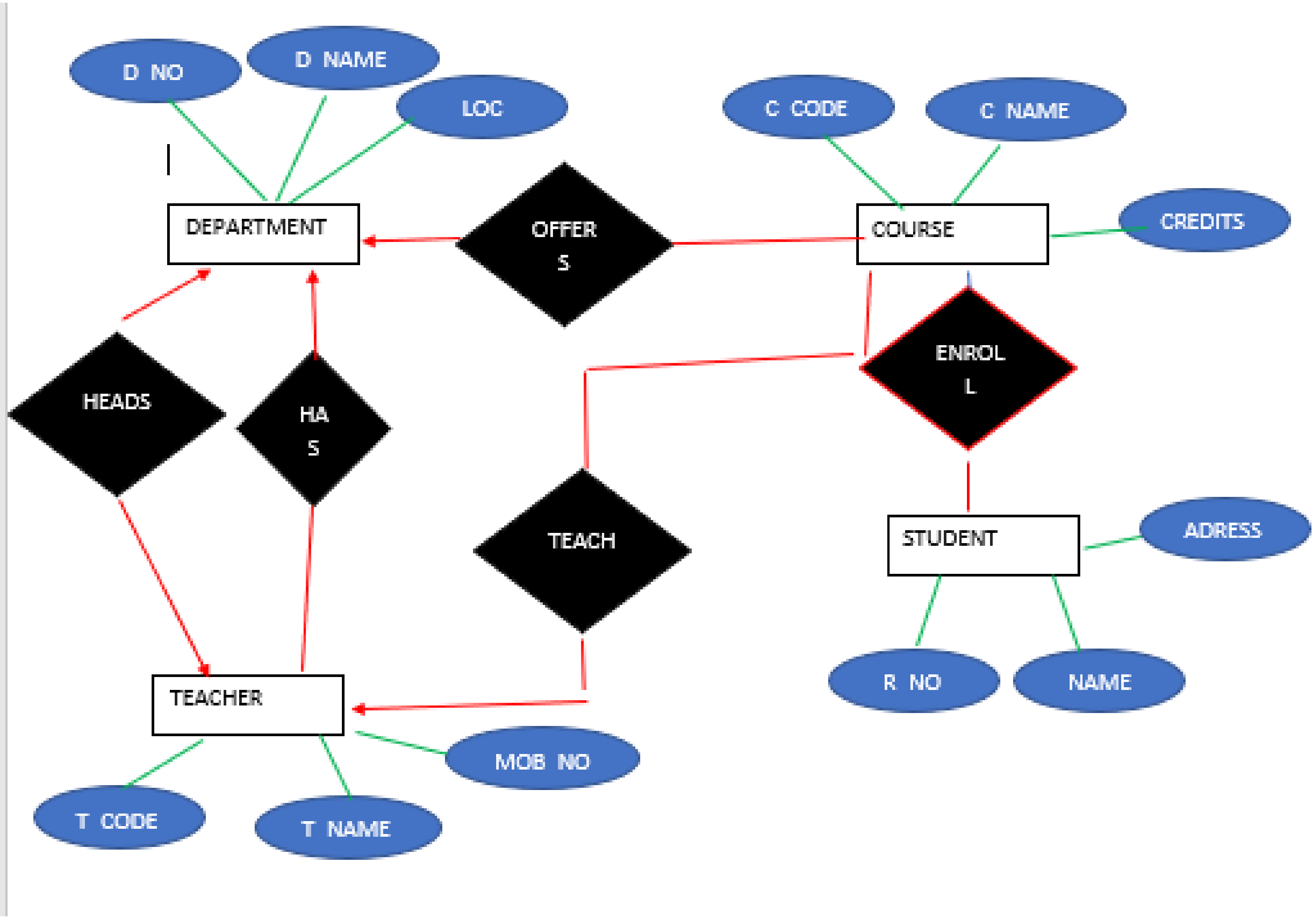
Teacher and course(1:M)

Student & course(M:M)

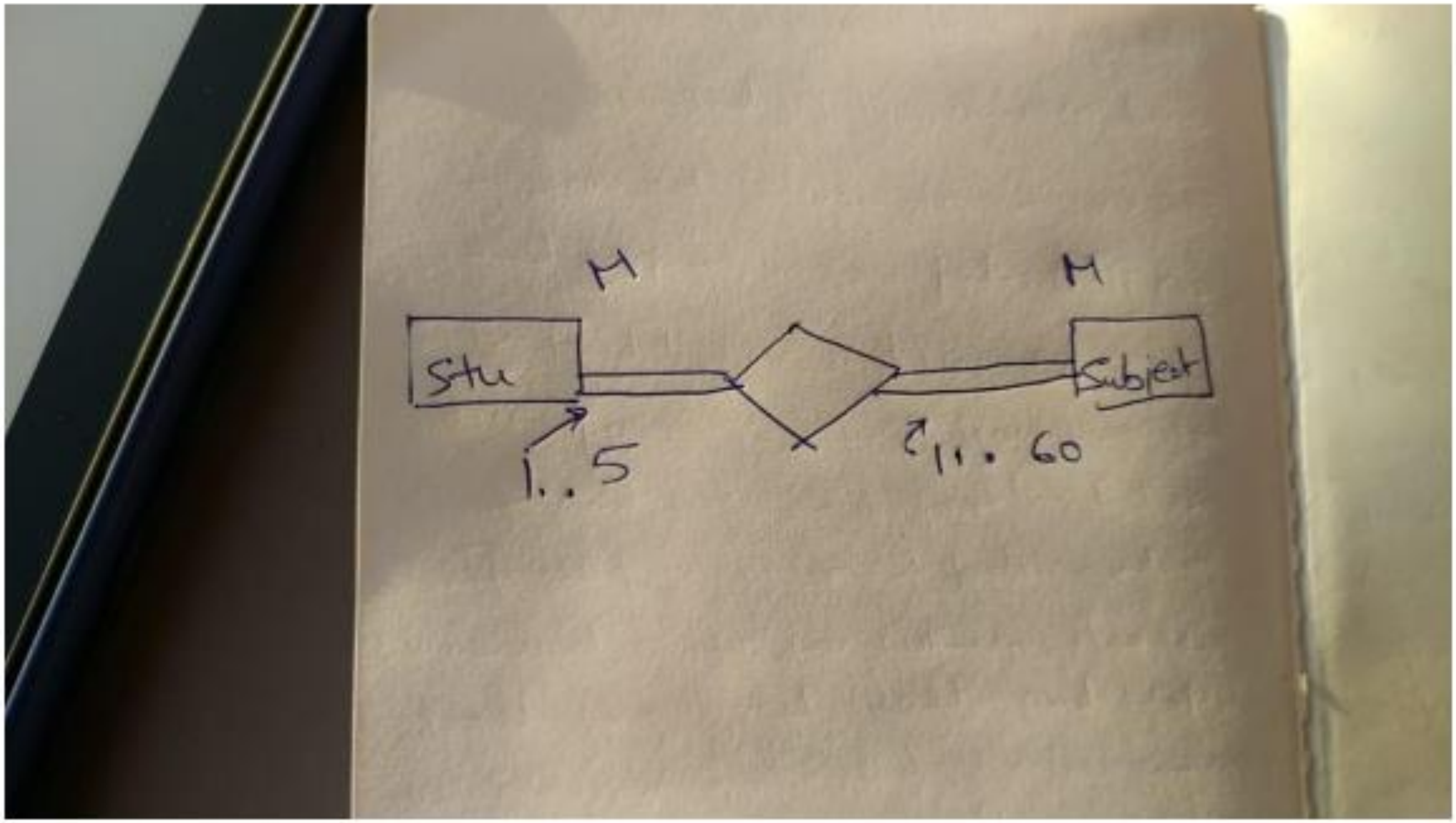
3.Key Attributes:D_no,C_code,Roll_no,t_code

4. Relevant Attributes:d_name,loc,c_name,credits,t_name,mob_no,name,address

Case Study 1 (Solution)



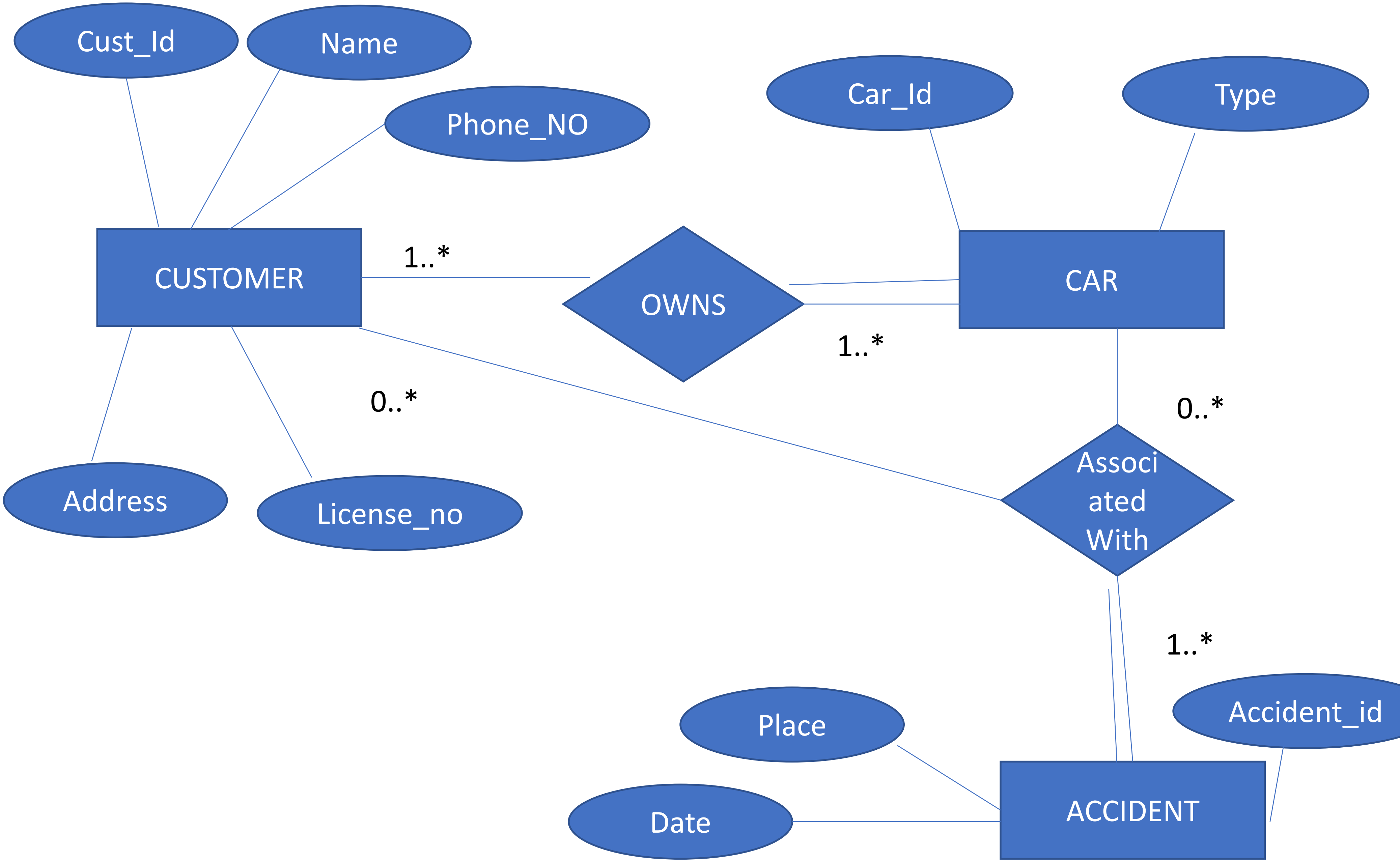
One student can take max of 5 subjects where as one subject can be taken by max of 60 students . Student must take subject and there should be no subject left behind.



Case Study 2

Construct an ER diagram for a car insurance company whose customers own one or more cars each. Each car has associated with it zero to any number of recorded accidents.

Case Study 2 solution



Enhanced ER MODEL

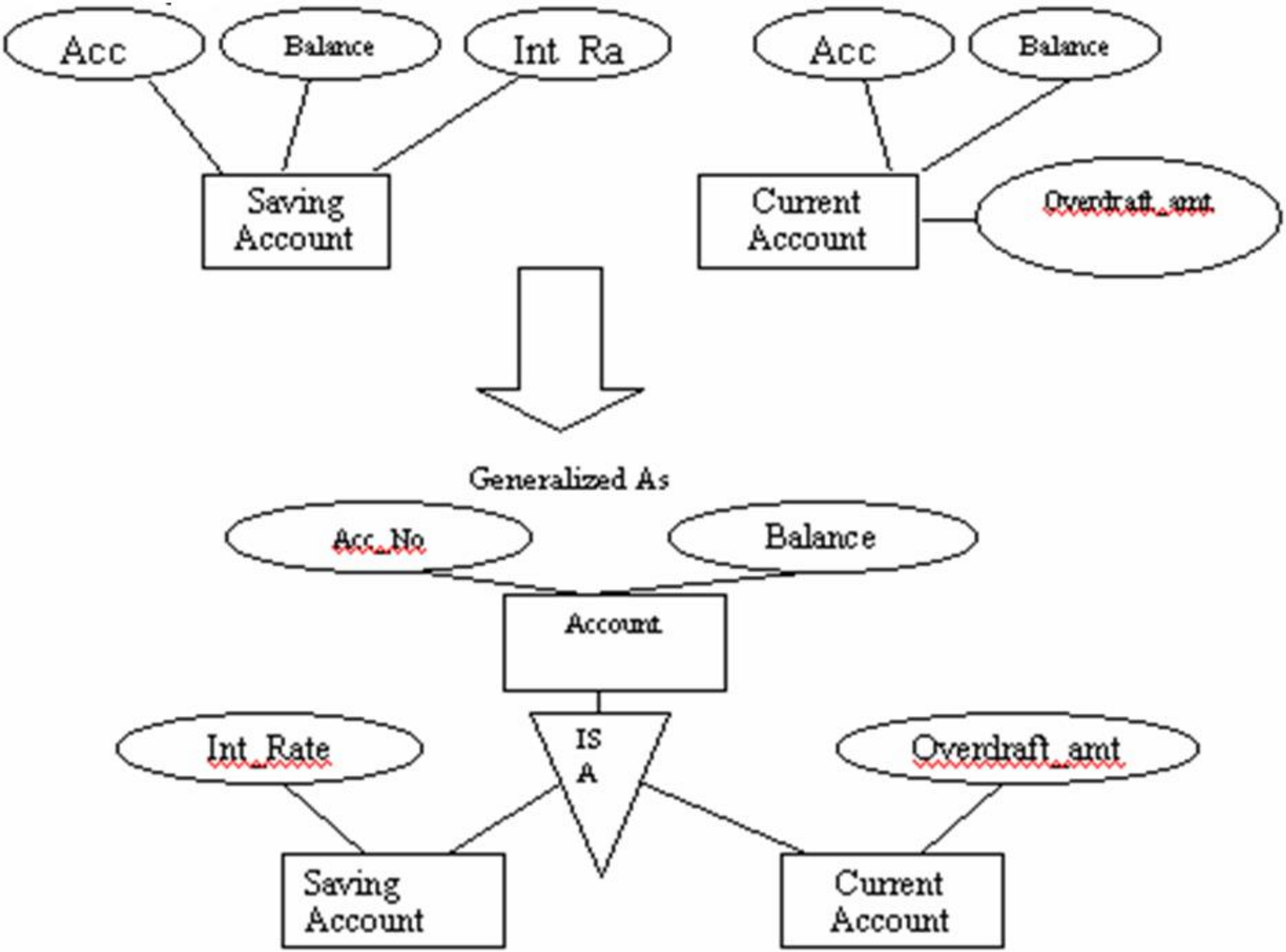
Generalization

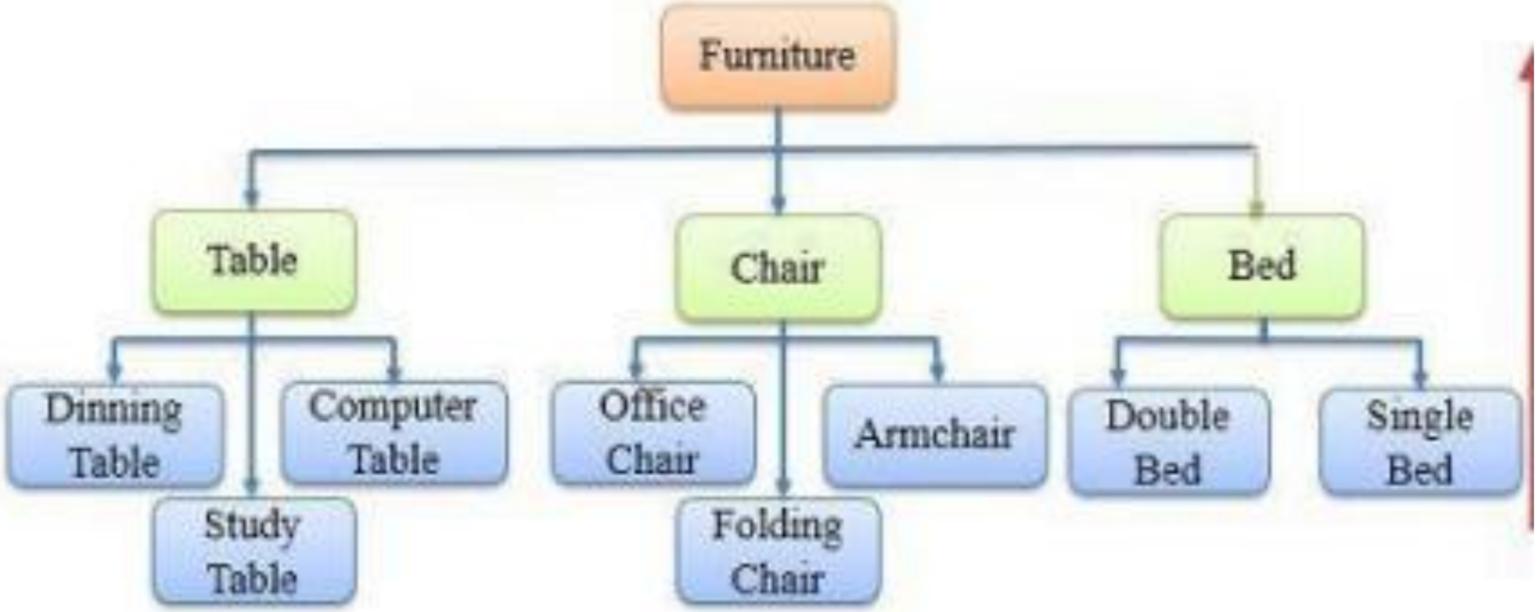
A generalization hierarchy is a form of abstraction that specifies that two or more entities that share common attributes can be generalized into a higher-level entity type called a supertype or generic entity. The lower level of entities becomes the subtype, or categories, to the super type. Subtypes are dependent entities.

Generalization is used to emphasize the similarities among lower-level entity sets and to hide differences.

It makes ER diagram simpler because shared attributes are not repeated. Generalization is denoted through a triangle component labeled ‘IS A”,

Bottom Up Approach


Acc
Balance
Int Ra
Saving Account
Acc
Balance
Current Account
Overdraft_amt
Generalized As
Acc_No
Balance
Account
IS A
Int_Rate
Overdraft_amt
Saving Account
Current Account

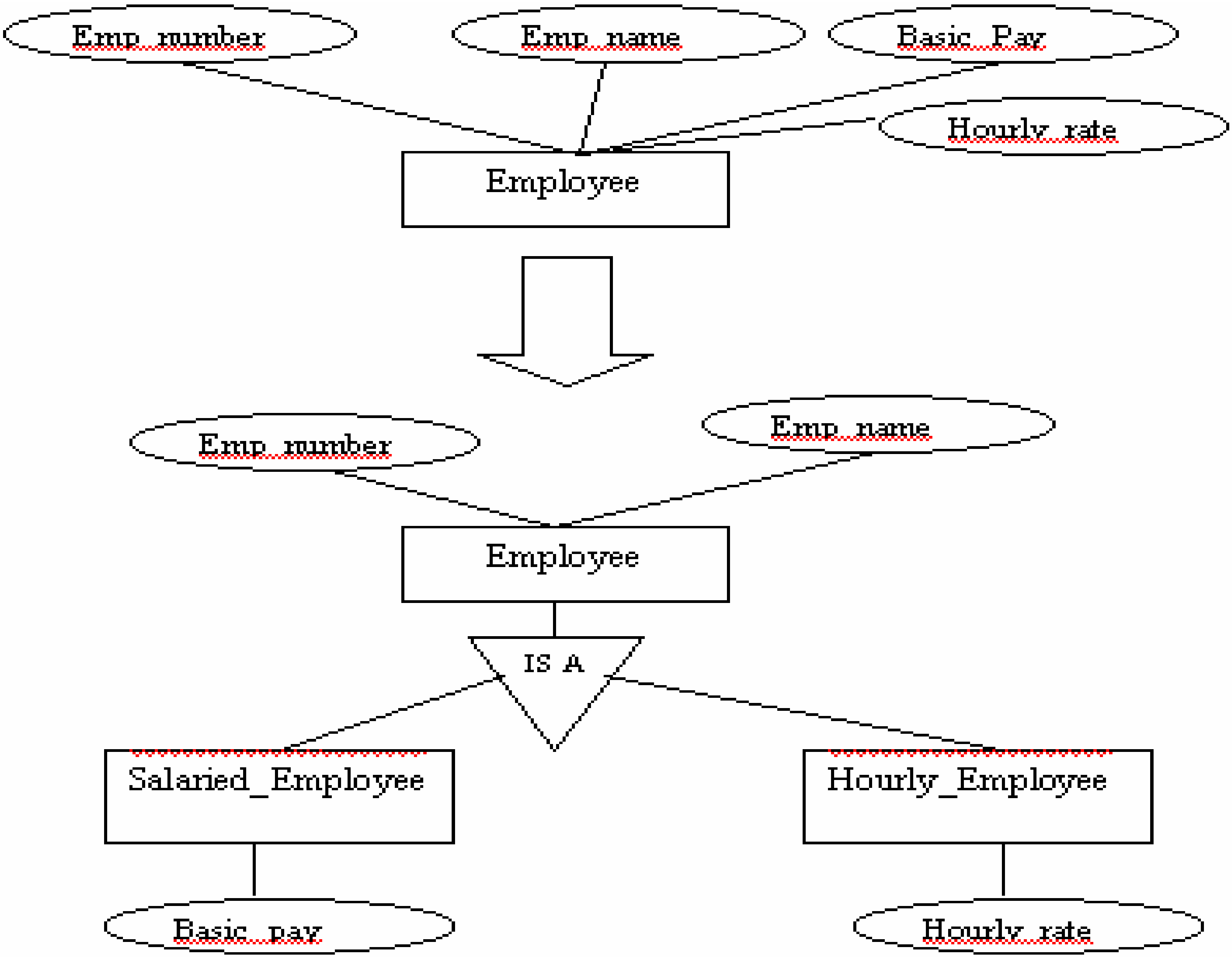

Furniture
Table
Chair
Bed
Dinning Table
Computer Table
Study Table
Office Chair
Folding Chair
Armchair
Double Bed
Single Bed

Specialization

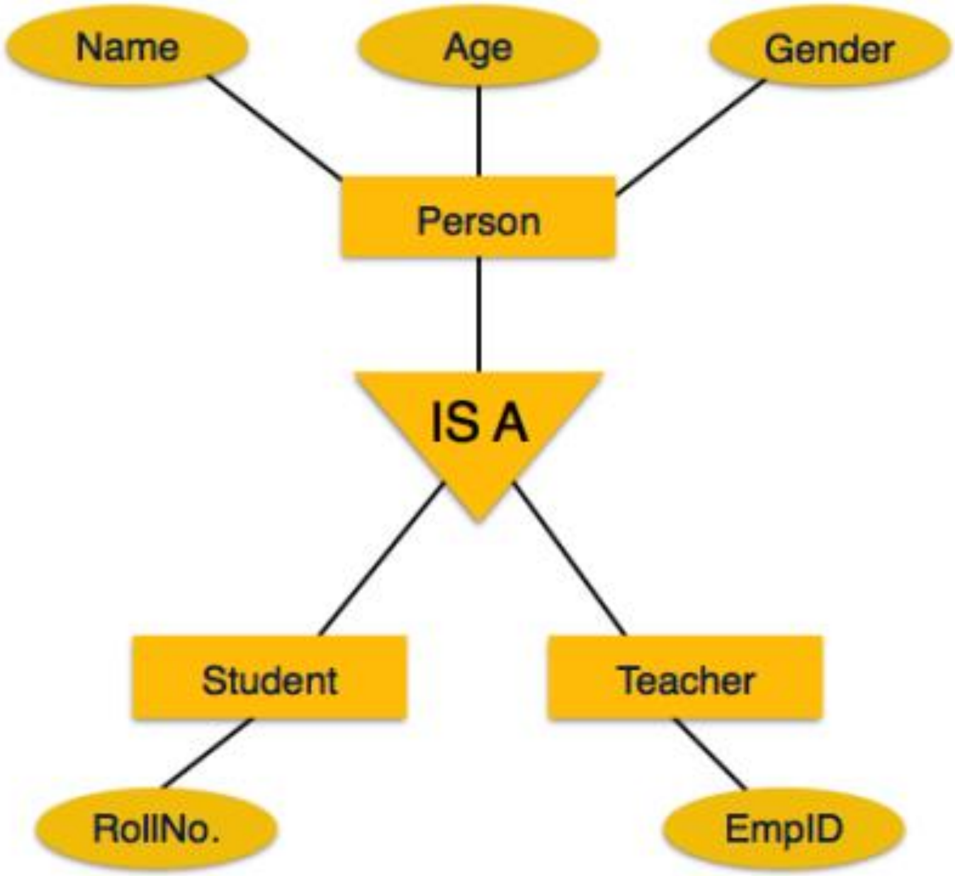
- Specialization is the process of taking subsets of a higher-level entity set to form lower level entity sets. It is a process of defining a set of subclasses of an entity type, which is called as superclas of the specialization. The process of defining subclass is based on the basis of some distinguish characteristics of the entities in the super class.

- For example, specialization of the Employee entity type may yield the set of subclasses namely Salaried_Employee and Hourly_Employee on the method of pay

- Top Down Approach


Emp_number
Emp_name
Basic_Pay
Hourly_rate
Employee
Emp_number
Emp_name
Employee
IS A
Salaried_Employee
Hourly_Employee
Basic_pay
Hourly_rate

Inheritance is an important feature of Generalization and Specialization. It allows lower-level entities to inherit the attributes of higher-level entities.



For example, the attributes of a Person class such as name, age, and gender can be inherited by lower-level entities such as Student or Teacher.

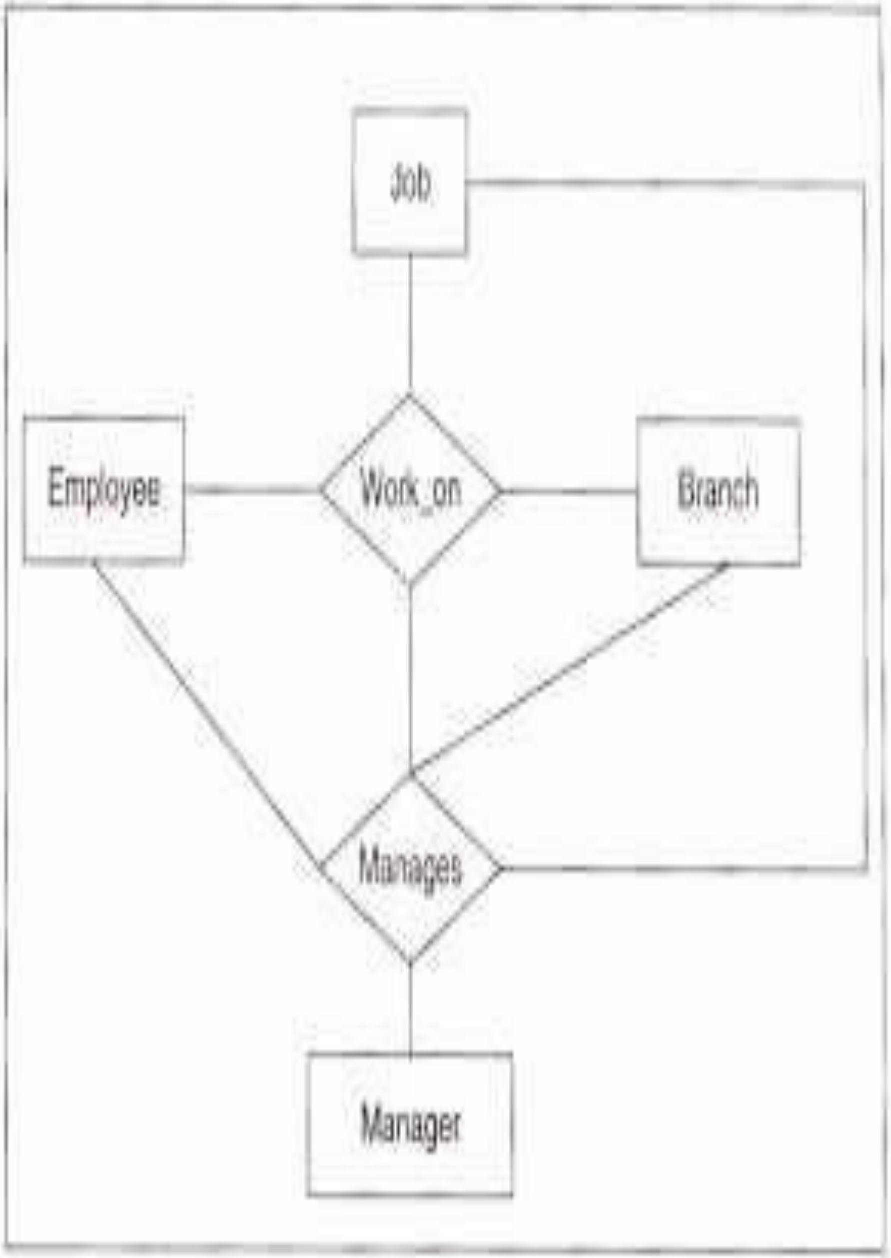
Difference between Specialization and Generalization

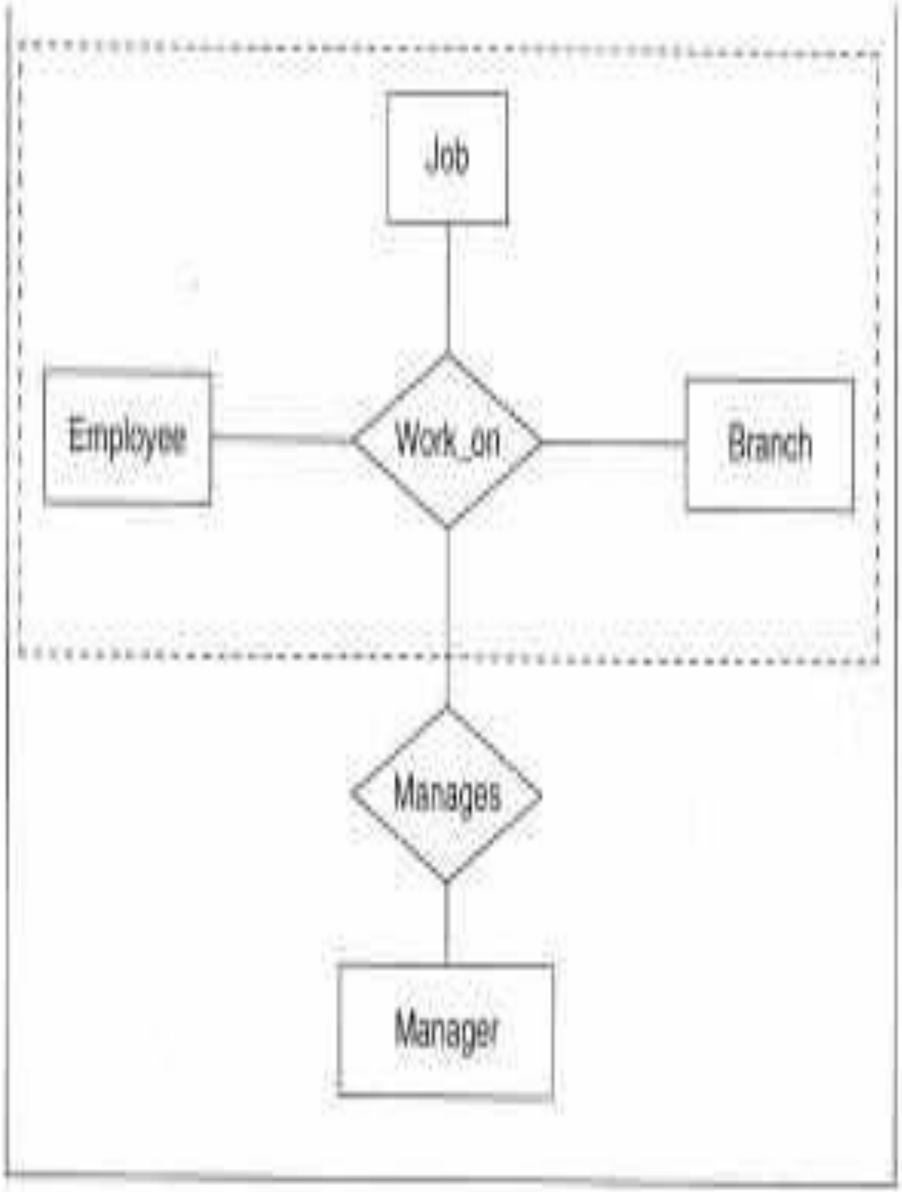
1.Generalization is a bottom-up approach. However, specialization is a top-down approach.
2. Generalization club all the entities that share some common properties to form a new entity. On the other hands, specialization spilt an entity to form multiple new entities that inherit some properties of the spiltted entity.
3. In generalization, a higher entity must have some lower entities whereas, in specialization, a higher entity may not have any lower entity present.
4. Generalization helps in reducing the size of schema whereas, specialization is just opposite it increases the number of entities thereby increasing the size of a schema.
5. Generalization is always applied to the group of entities whereas, specialization is always applied on a single entity.
6. Generalization results in a formation of a single entity whereas, Specialization results in the formation of multiple new

Aggregation

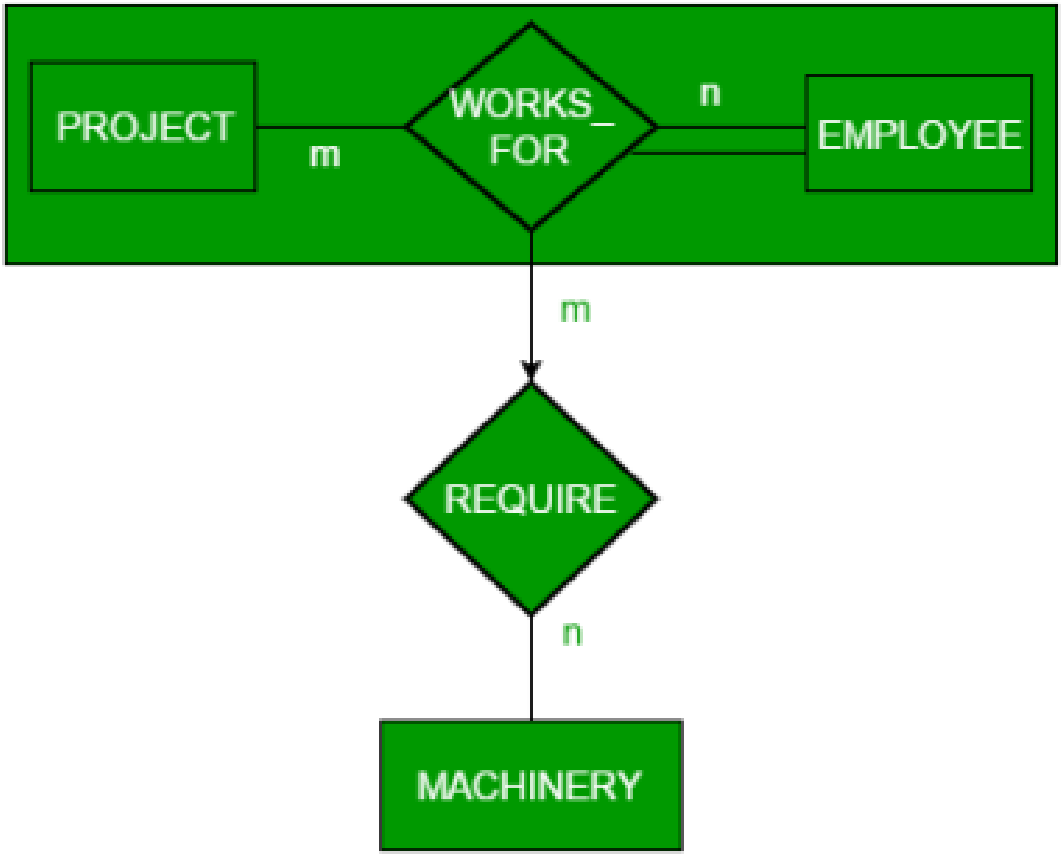
One limitation of the E-R model is that it cannot express relationships among relationships.

The best way to model a situation like this is by the use of aggregation. Thus, the relationship set work_on relating the entity sets Employee, Branch and Job is a higher-level entity set. Such an entity set is treated in the same manner, as is any other entity set. We can then create a binary relationship Manages between work_on and Manager to represent who manages what tasks.


Job
Employee
Work_on
Branch
Manages
Manager


Job
Employee
Work_on
Branch
Manages
Manager

Aggregation



Representing aggregation via schema- To represent aggregation, create a schema containing:

1. primary key of the aggregated relationship
2. primary key of the associated entity set
3. descriptive attribute, if exists.

Thank You