UNIT 3

Database Design

MSC CS

GAC CBE

Table of Contents

Design Phases

Consists of three parts namely the conceptual design, the logical design and the physical design.

Design Phases

- **Initial phase** - characterize fully the data needs of the prospective database users.
- Eg. The requirements can be stated as Every employee works for exactly one department and a department can have many employees. New department need not have any employee

Design Phases

- **Second phase** - choosing a data model
  - Applying the concepts of the chosen data model
  - Translating these requirements into a conceptual schema of the database.
  - A fully developed conceptual schema indicates the functional requirements of the enterprise.
    - Describe the kinds of operations (or transactions) that will be performed on the data.
    - Eg. A relational Model is chosen.

Design Phases

- **Final Phase** - Moving from an abstract data model to the implementation of the database
  - **Logical Design** - Deciding on the database schema.
    - Database design requires that a "good" collection of relation schemas is found.
    - Eg. Emp and Dept Schema. ER diagram is drawn.
    - Business decision - What attributes should be recorded in the database?
    - Eg. Eno, Ename, Salary, Dno, Dname and Address

Design Phases (cont.)

- Computer Science decision - What relation schemas should we have and how should the attributes be distributed among the various relation schemas?
- Eg. Eno, Ename, Salary and Dno in Emp Schema. Dno, Dname and Address in Dept Schema

- **Physical Design** - Deciding on the physical layout of the database

Design Alternatives

Choosing the best among the best.

Design Alternatives

- While designing a database schema, two major pitfalls needs to be avoided:
  - **Redundancy**: a bad design may result in repeat information.
    - Redundant representation of information may lead to data inconsistency among the various copies of information
  - **Incompleteness**: a bad design may make certain aspects of the enterprise difficult or impossible to model.
- Avoiding bad designs is not enough. There may be a large number of good designs from which a choice could be made.

The Entity Relationship Model

To facilitate database design

The ER Model

- The ER data model was developed to facilitate database design by allowing specification of an **enterprise schema** that represents the overall logical structure of a database.
- The ER data model employs three basic concepts:
  - entity sets
  - relationship sets
  - attributes.
- The ER model also has an associated diagrammatic representation, the **ER diagram**, which can express the overall logical structure of a database graphically.

Entity Sets

- An **entity** is an object that exists and is distinguishable from other objects.
  - Example: specific employee,department, project, event, plant
- An **entity set** is a set of entities of the same type that share the same properties.
  - Example: set of all employees, set department, trees, holidays
- An entity is represented by a set of attributes; i.e., descriptive properties possessed by all members of an entity set. Example:

  emp = (eno, ename, dno,salary)
  dept= (dno, dname, address)
- A subset of the attributes form a **primary key** of the entity set; i.e., uniquely identifying each member of the set

Entity Sets

11	Aswatha	11	10000
13	Brindha	10	15000
22	Parthiban	10	20000
24	Mugilan	14	25000

Employee(Emp)

10	Marketing	Coimbatore
11	Accounting	Salem
12	Purchase	Perundurai
13	Sales	Tiruppur
14	Auditing	Erode

Department(Dept)

Representing Entity sets (cont.)

- Entity sets can be represented graphically as follows:
  - Rectangles represent entity sets.
  - Attributes listed inside entity rectangle
  - Underline indicates primary key attributes

Relationship Sets

- A **relationship** is an association among several entities

  Example:

  13 (Brindha) works_for 10 (Marketing)

  emp entity relationship set *dept* entity

- A **relationship set** is a mathematical relation among $n \geq 2$ entities, each taken from entity sets

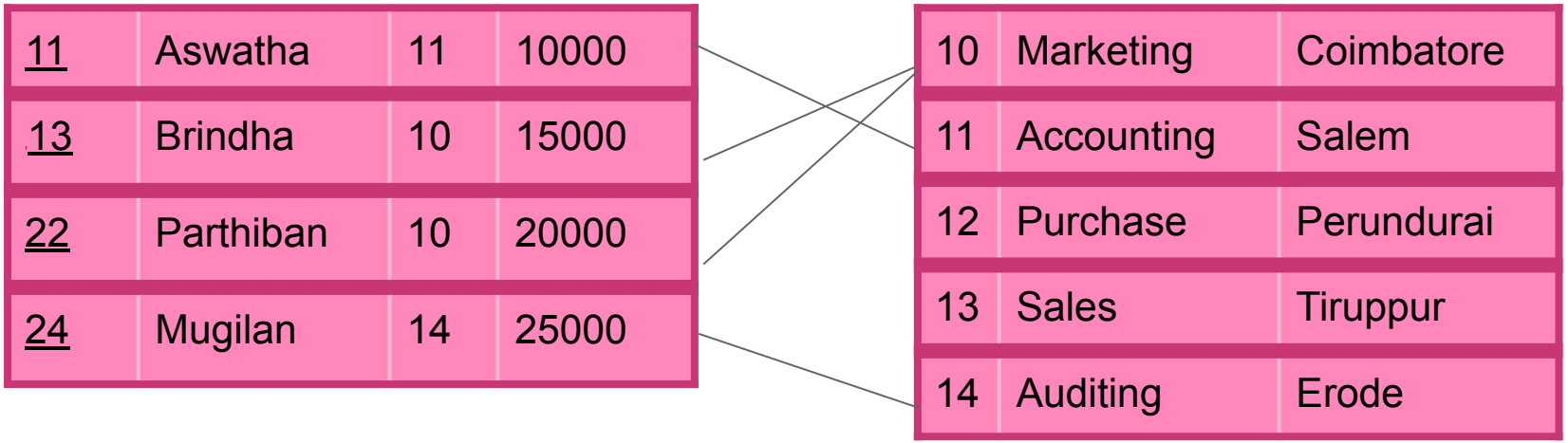
$$\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$$

  where $(e_1, e_2, \ldots, e_n)$ is a relationship

  - Example:

$$(13,10) \in \text{works for}$$

Relationship Sets



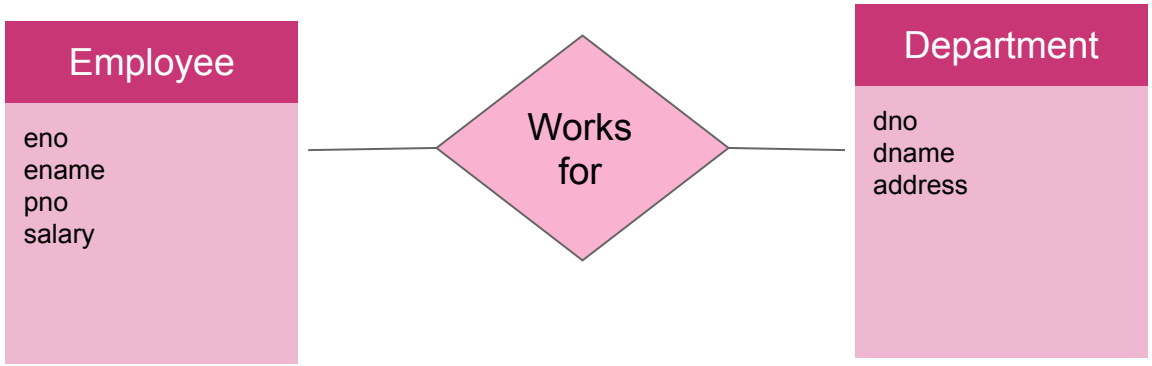
11	Aswatha	11	10000
13	Brindha	10	15000
22	Parthiban	10	20000
24	Mugilan	14	25000

10	Marketing	Coimbatore
11	Accounting	Salem
12	Purchase	Perundurai
13	Sales	Tiruppur
14	Auditing	Erode

- Example: The relationship set works_for is defined to denote the associations between employee and the department which he works.
- Pictorially, a line is drawn between related entities.

Representing Relationship Sets via ER Diagrams

- Diamonds represent relationship sets.

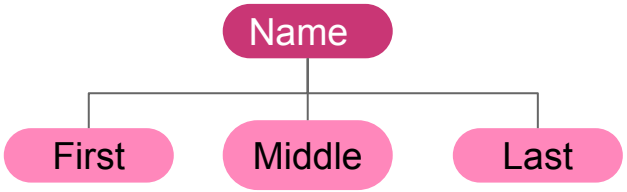


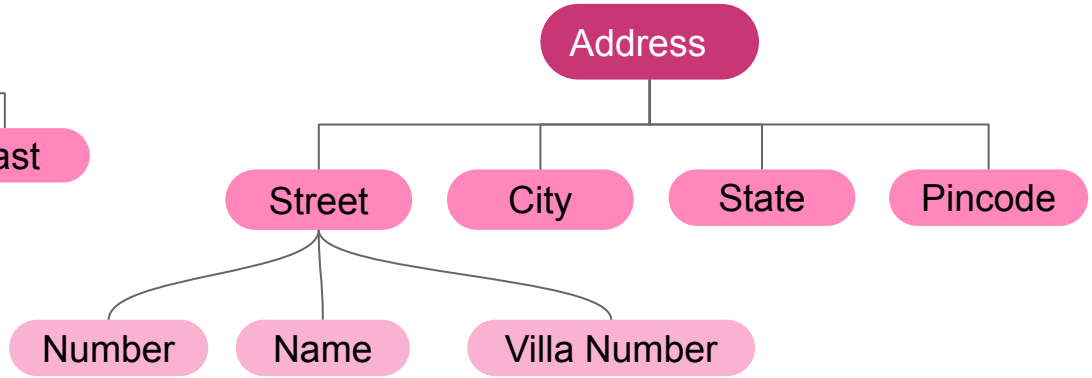
Attributes

- Attribute types:
  - **Simple** and **composite** attributes.
  - **Single-valued** and **multivalued** attributes
    - Example: multivalued attribute: *phone_no, Address*
  - **Derived** attributes
    - Can be computed from other attributes
    - Example: age, given date_of_birth
- **Domain** - the set of permitted values for each attribute

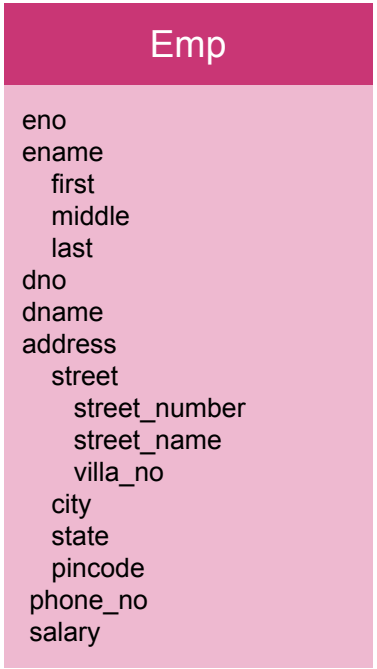
Composite Attributes

Composite attributes allows to divide attributes into subparts (other attributes).





Complex Attributes



Constraints

Express the number of entities to which another entity can be associated via a relationship set.

Mapping Cardinality Constraints

- Most useful in describing binary relationship sets.
- For a binary relationship set the mapping cardinality must be one of the following types:
  - One to one
  - One to many
  - Many to one
  - Many to many

Participation using Single line / Double line Representation

- **Total participation** (indicated by double line): every entity in the entity set participates in at least one relationship in the relationship set
- **Partial participation**: some entities may not participate in any relationship in the relationship set

Structural Constraints

- Structural Constraints includes
  - Cardinality ratio also called as Maximum cardinality
  - Participation or Existence is also called Minimum cardinality

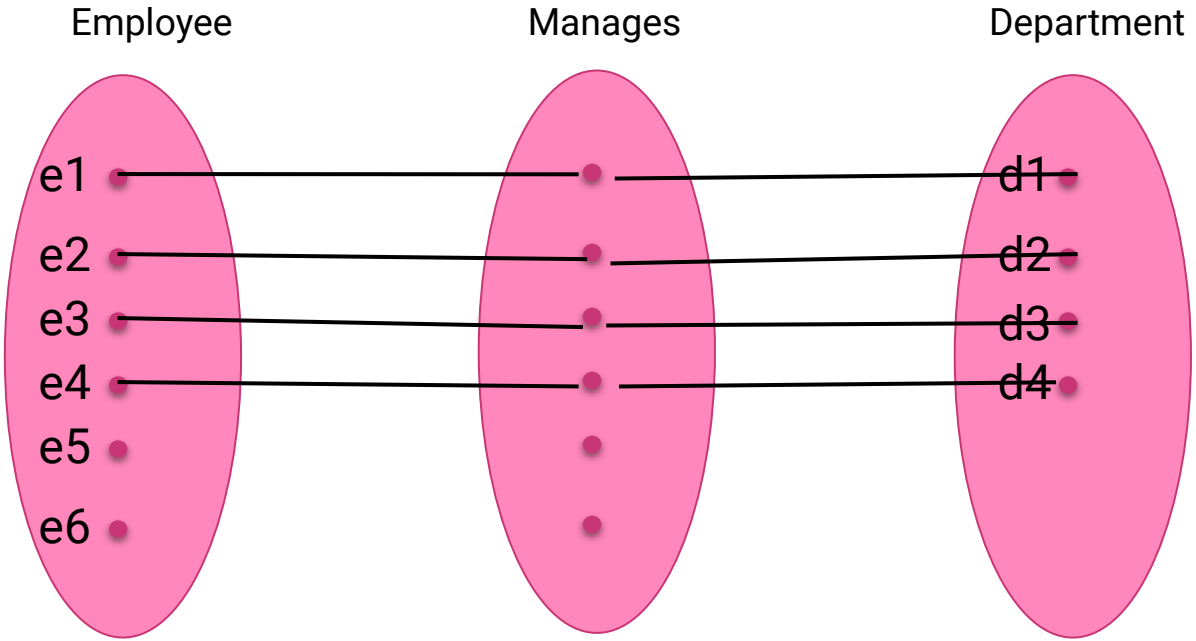
Min Max Representation

- A line may 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.
  - A maximum value of 1 indicates that the entity participates in at most one relationship
  - A maximum value of * indicates no limit.

Relationship 1:1

- **Requirements Analysis** : Every department should have a manager and only one employee manages a department and an employee can manage only one department
- Identify entities from noun. Here employee and department are the entities. The verb manages is used as a relationship.

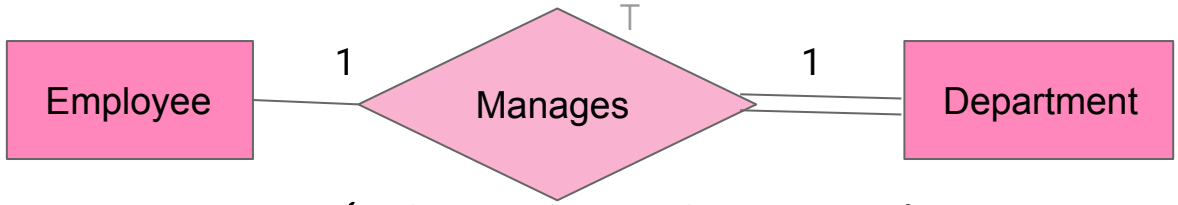
Relationship 1:1



Degree and Cardinality

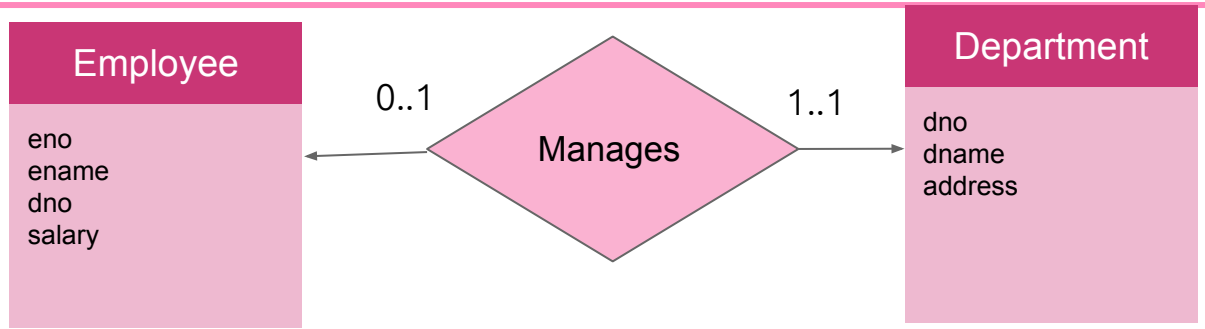
- Degree is 2 as only two entities participate. For binary relationship the degree is 2.
- Maximum number of relationship an entity employee can participate is called cardinality
- The cardinality is 1 for employee because the employee can manage only one department.
- The cardinality for department is also 1 because the department has to be managed by at least one and exactly one employee.

Participation using Single line - Double line Representation



- **Total participation** (indicated by double line): Every entity in the entity Department participates in at least one relationship in the relationship set Manages.
- **Partial Participation** (indicated by single line). Every entity in the entity set Employee is not participating in at least one relationship in the relationship set Manages.

Min Max Representation for Employee and Department

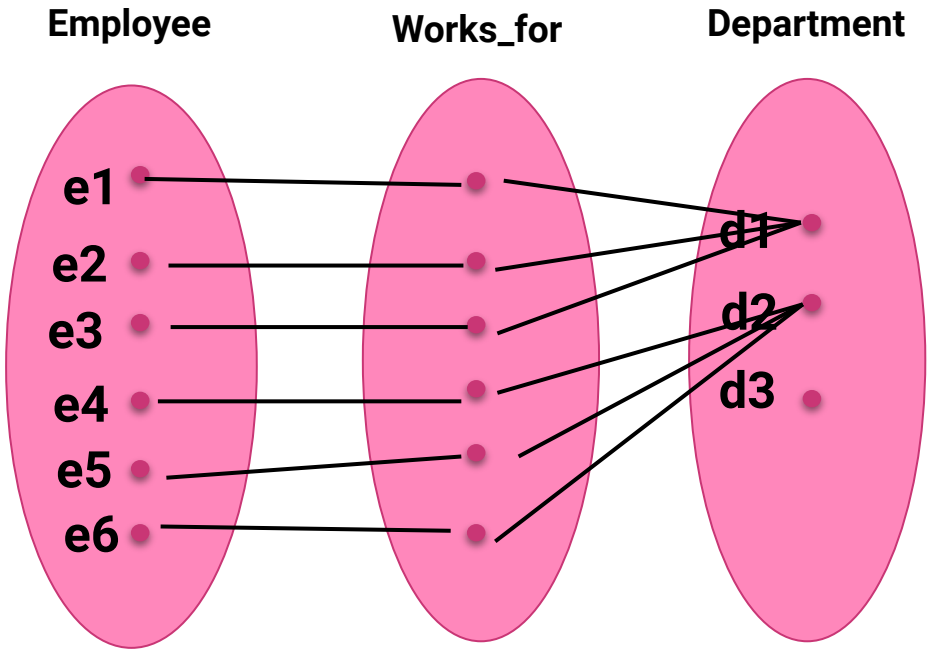


- The min value 1 in Department entity set indicates total participation.
- The minimum value 0 in Employee entity set indicates partial participation.
- A maximum value of 1 indicates that the entities Employee and Department participates in at most one relationship

One to Many Relationship 1:M

- **Requirements Analysis**: Every Employee works for a Department and a Department have many employees. New Department need not have any employee
- Identify the nouns. Employee and Department are the nouns considered for entity. Works for is the verb identified for relationship. It is binary relationship.

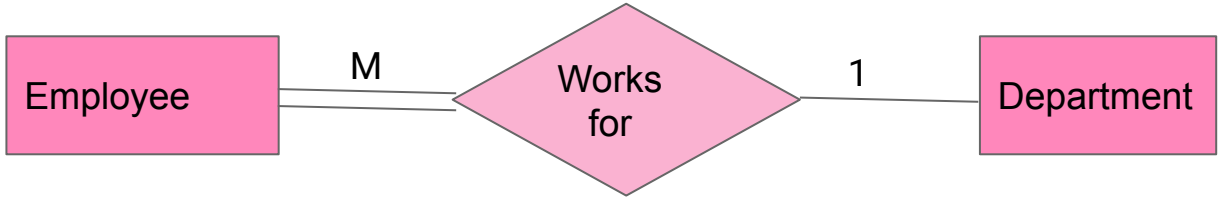
One to many relationship 1:M



Degree and Cardinality Ratio

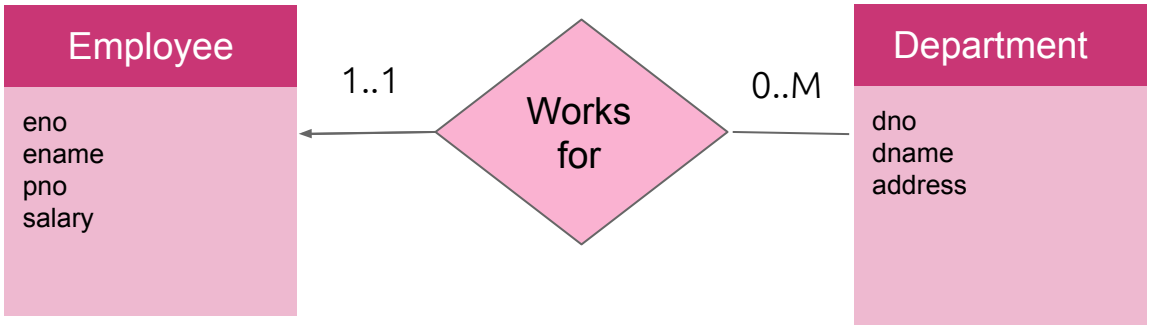
- Two entities Employee and Department are participating in a relation. So the degree is 2.
- **Cardinality ratio** is the maximum number of relationship an entity can participate. The cardinality of Employee entity is 1.The cardinality of Department is M as the participation of Department entity is more than 1.

Participation of Employee and Department



- **Total participation** (indicated by double line): Every entity in the entity set Employee participates in at least one relationship in the relationship set Works_for.
- **Partial participation** (indicated by single line): Every entity in the entity set Department is not participating in at least one relationship in the relation set Works_for.

Min Max Representation for Employee and Department



- The min value 1 in Employee indicates total participation and 0 in Department indicates partial participation.
- The max value of 1 for employee indicates that the entity is participating in at most one relation.
- The max value of M indicates that the Department is participating in more than one relationship which is M.

Roles

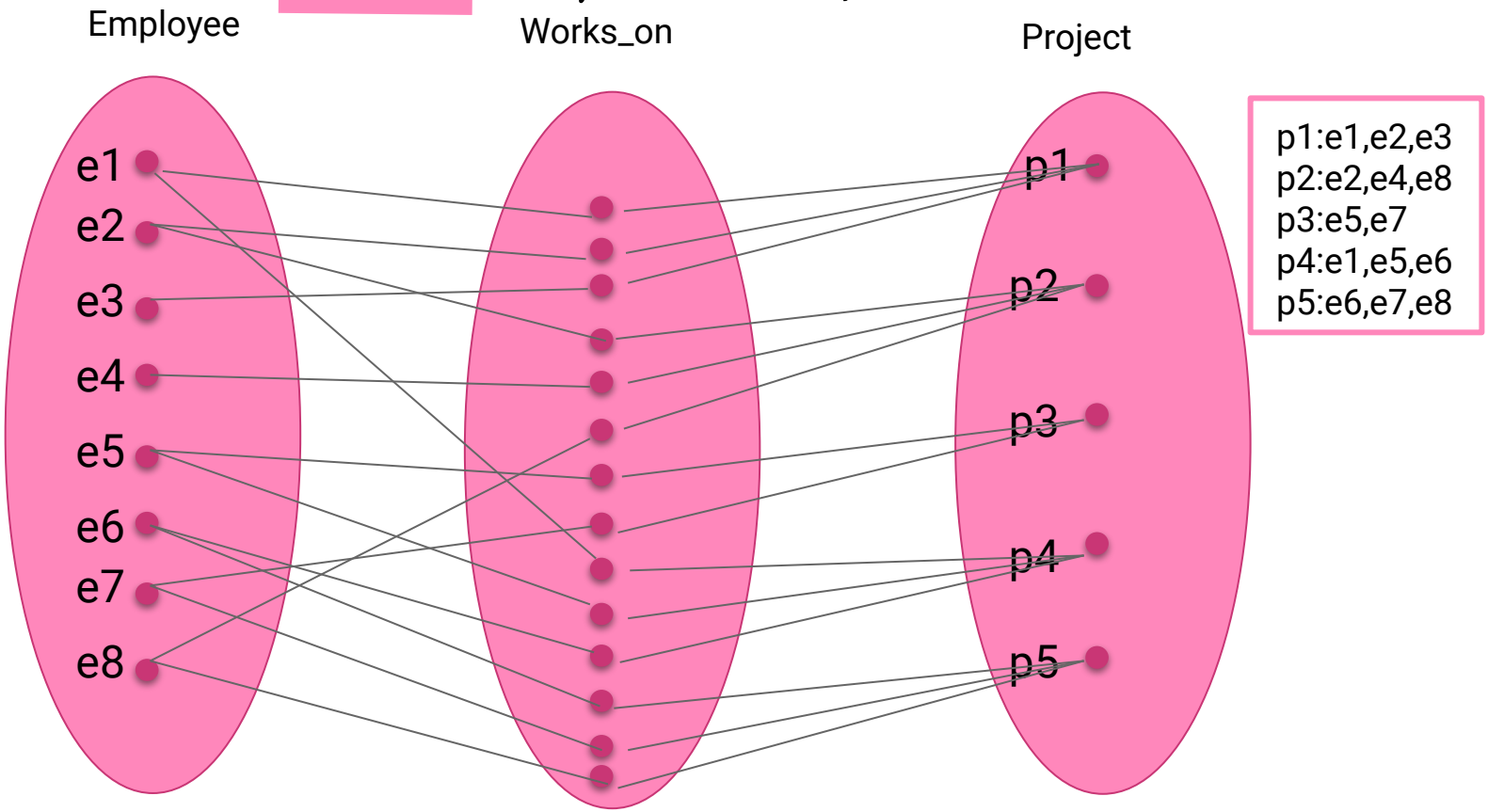
- Each entity type that participates in a relationship type plays a particular role in the relationship. The **role name** signifies the role that a participating entity from the entity type plays in each relationship instance, and helps to explain what the relationship means.
- In the Works_for relationship type, Employee plays the role of *employee* or *worker* and Department plays the role of *department* or *employer*.

Relationship M:N

Requirements Analysis

Every Employee is supposed to work for at least one project and an employee can work on many projects. Similarly a project can have many employees and every project is to have at least one employee.

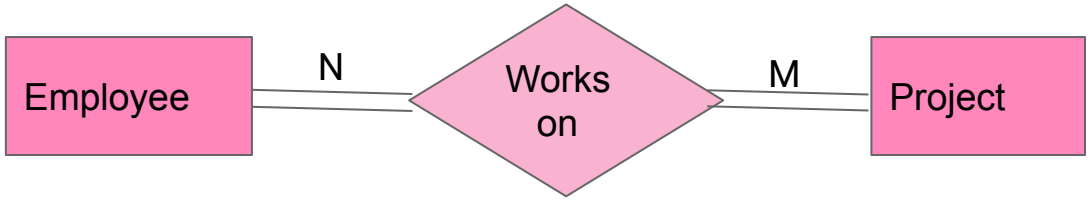
Many to Many Relationship M:N



Degree and Cardinality

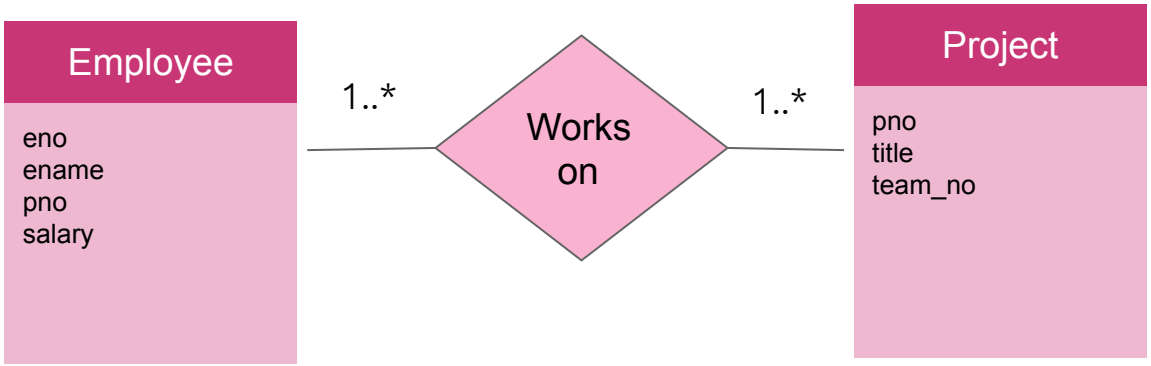
- From the nouns identify the entity. Let the entities be employee and project. From the verb identify the relationship works on.
- Number of entities participating in this relationship is 2. so the **degree** is 2
- **Cardinality** is the maximum number of relationship in which each entity can participate. The employee has participated in M relationship and project has participated in N relationship

Total Participation of Employee and Project



- **Total participation** (indicated by double line): Every entity in the entity set Employee participates in at least one relationship in the relationship set Works_on. Similarly every entity in the entity set Project participates in at least one relationship in the relationship set Works_on.

Min Max Representation for Employee and Project



The min value 1 indicates total participation.

* indicates no limit.

Primary Key

- Primary keys provide a way to specify how entities and relations are distinguished. We will consider:
  - Entity sets
  - Relationship sets.
  - Weak entity sets

Primary key for Entity Sets

- By definition, individual entities are distinct.
- From database perspective, the differences among them must be expressed in terms of their attributes.
- The values of the attribute values of an entity must be such that they can uniquely identify the entity.
  - No two entities in an entity set are allowed to have exactly the same value for all attributes.
- A key for an entity is a set of attributes that suffice to distinguish entities from each other

Primary key for Relationship Sets

- To distinguish among the various relationships of a relationship set we use the individual primary keys of the entities in the relationship set.
  - Let R be a relationship set involving entity sets E1, E2, .. En
  - The primary key for R is consists of the union of the primary keys of entity sets E1, E2, ..En
  - If the relationship set R has attributes a1, a2, .., am associated with it, then the primary key of R also includes the attributes a1, a2, .., am
- Example: relationship set "works_for".
  - The primary key consists of emp.eno and dept.dno
- The choice of the primary key for a relationship set depends on the mapping cardinality of the relationship set.

Choice of Primary key for Binary Relationship

- Many-to-Many relationships. The preceding union of the primary keys is a minimal superkey and is chosen as the primary key.
- One-to-Many relationships . The primary key of the "Many" side is a minimal superkey and is used as the primary key.
- Many-to-one relationships. The primary key of the "Many" side is a minimal superkey and is used as the primary key.
- One-to-one relationships. The primary key of either one of the participating entity sets forms a minimal superkey, and either one can be chosen as the primary key.

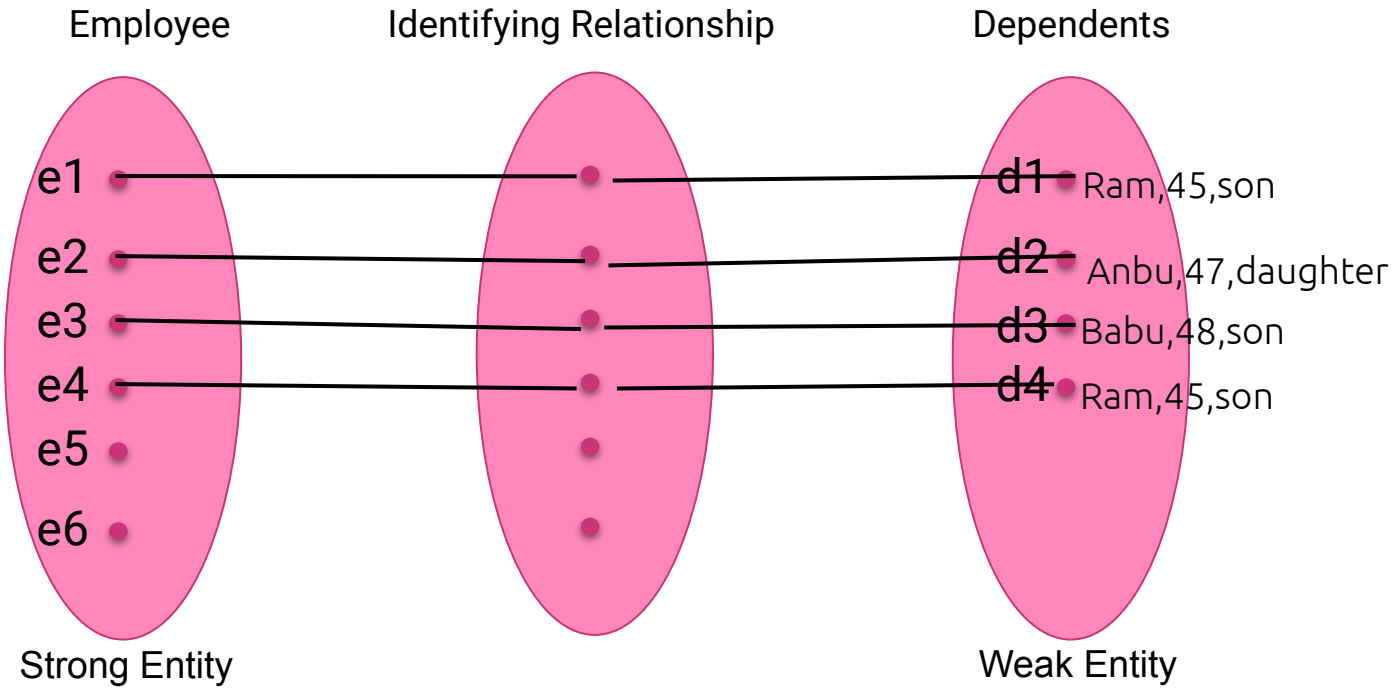
Weak Entity

One whose existence is depend on another entity

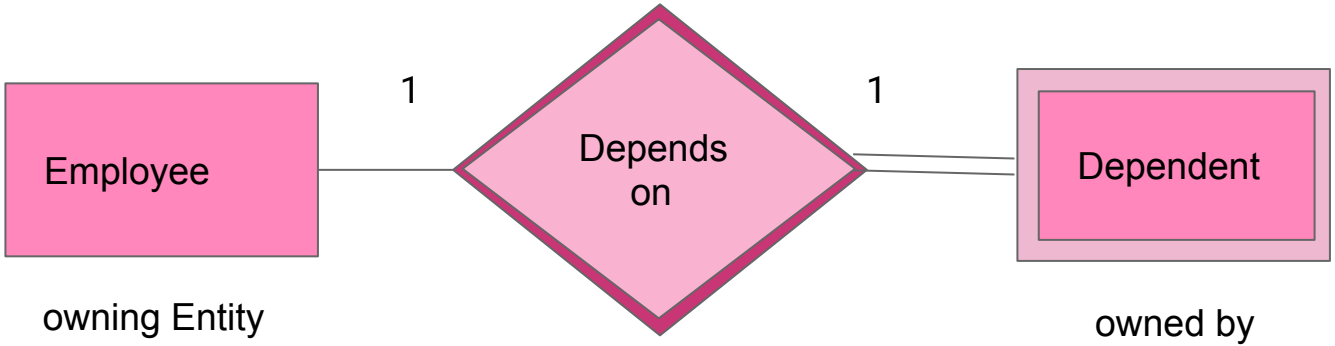
Weak Entity

- The Employee Entity has attributes eno,ename,dno and salary and the dependent entity has name,age and relation_ship attributes.
- The value Ram, 45, son repeats and the entity cannot be identified uniquely as there is no primary key attribute.
- If eno is added to name,age and relation_ship then it becomes strong entity.

Weak Entity

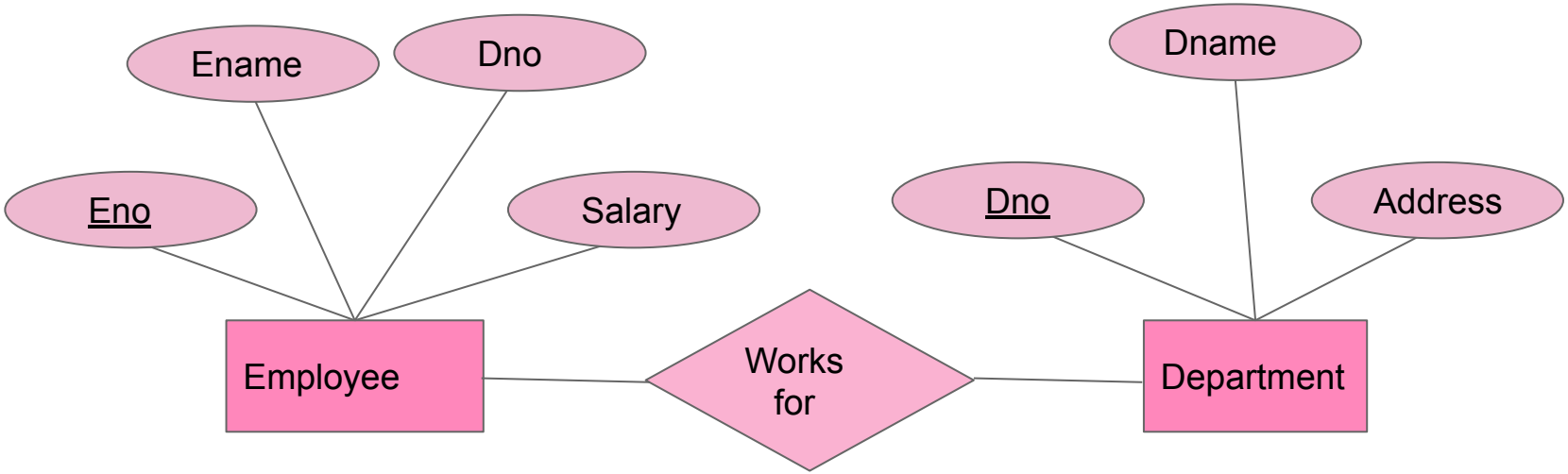

Employee
Identifying Relationship
Dependents
e1
e2
e3
e4
e5
e6
d1
d2
d3
d4
Ram,45,son
Anbu,47,daughter
Babu,48,son
Ram,45,son
Strong Entity
Weak Entity

Participation



- **Total participation** : Every entity in the entity Dependent participates in at least one relationship in the relationship set Depends_on.
- **Partial Participation** :. Every entity in the entity set Employee is not participating in at least one relationship in the relationship set Depends_on.
- The double diamond represents identifying relationship and double rectangle represents weak entity.

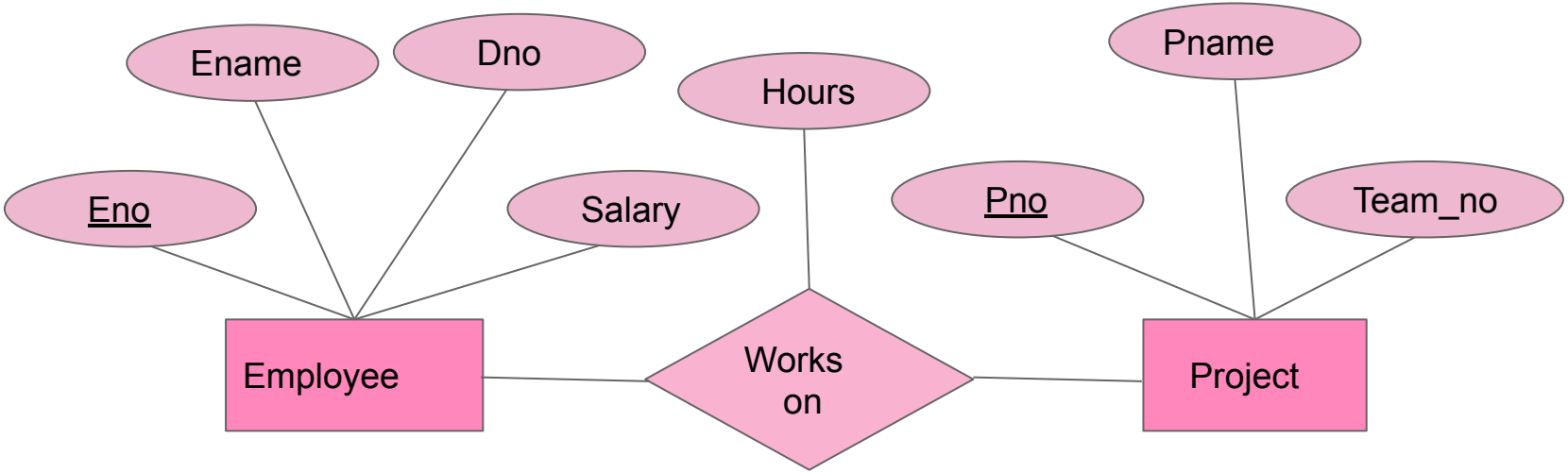
ER Diagram


Ename
Dno
Eno
Salary
Employee
Works for
Dname
Dno
Address
Department

ER Diagram

- Entities are represented using rectangle. In the ER diagram Employee and Department are entities and represented using rectangle symbol.
- Relationships are represented using diamond symbol. The relationship works_on is represented using diamond symbol.
- Attributes are represented using ellipses. Eg, Eno, Ename, Dno etc.
- Entities and attributes are connected using lines.
- Entities and relationship are connected using lines.
- The primary key in Employee entity is Eno and Department is Dno and both are underlined.

Attribute added to a Relationship



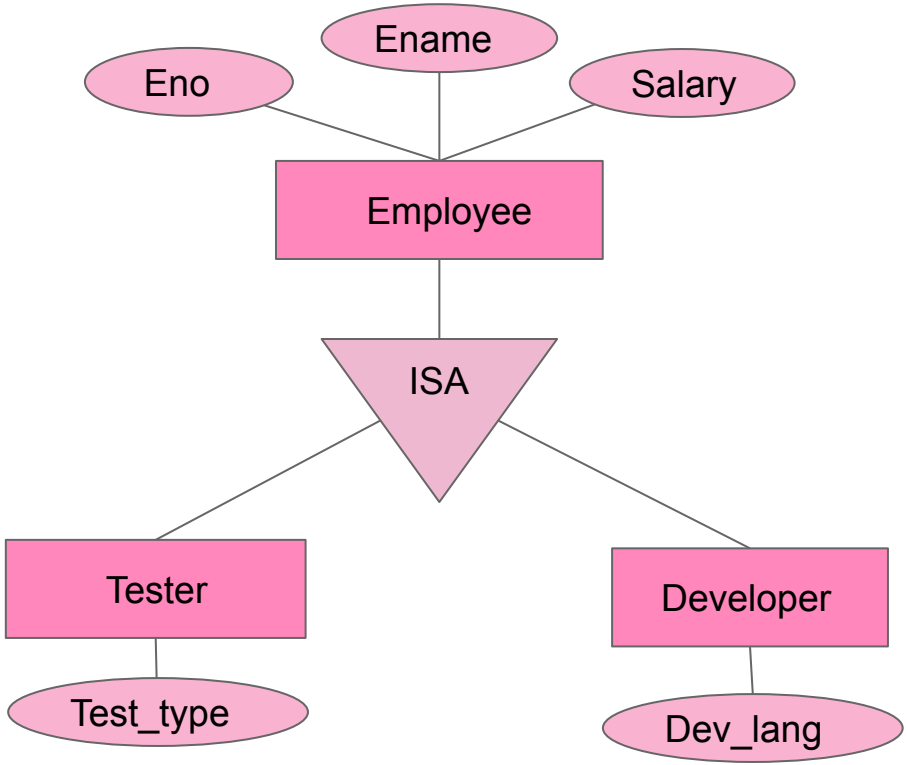
Attribute added to a Relationship

- It is always good practice to place attributes in an entity. Sometimes a need may arise to place the attributes in a relationship. Hours is one such attribute. If it is placed in Employee entity it will give the details of number of hours an employee worked.
- It will not give details on number of hours a project took to complete as it involves many employees.
- Similarly if the attribute hours is placed in project it will give the details of number of hours the project took and not the details about the number of hours the employees worked.
- In this scenario a need arises to place the attribute in relation. So that the details about the number of hours worked by an employee as well as number of hours a project took to complete can be obtained.

Specialization

A process in which an entity is divided into sub-entities

Specialization



Specialization

- Top-down design process; Sub-groupings are designated within an entity set that are distinctive from other entities in the set.
- These sub-groupings become lower-level entity sets that have attributes or participate in relationships that do not apply to the higher-level entity set.
- Depicted by a *triangle* component labeled ISA (e.g., *tester* "is a" *employee*).
- **Attribute inheritance** - a lower-level entity set inherits all the attributes and relationship participation of the higher-level entity set to which it is linked.

Specialization as Schemas

- **Method 1:**
  - Form a schema for the higher-level entity
  - Form a schema for each lower-level entity set, include primary key of higher-level entity set and local attributes

Schema	Description
Employee	Eno, Ename, Salary
Tester	Eno, Test_type
Developer	Eno, Dev_lang

  - **Drawback:** getting information about, an *tester* requires accessing two relations, the one corresponding to the low-level schema and the one corresponding to the high-level schema

Specialization as Schemas

- **Method 2:**
  - Form a schema for each entity set with all local and inherited

Schema	Description
Employee	Eno, Ename, Salary
Tester	Eno, Ename, Salary,Test_type
Developer	Eno, Ename, Salary, Dev_lang

  - **Drawback:** Eno, Ename, Salary may be stored redundantly for people who are both Tester and Developer.

Generalization

Entities are combined to form a more generalized entity

Generalization

- **A bottom-up design process** - combine a number of entity sets that share the same features into a higher-level entity set.
- Specialization and generalization are simple inversions of each other; they are represented in an E-R diagram in the same way.
- The terms specialization and generalization are used interchangeably.

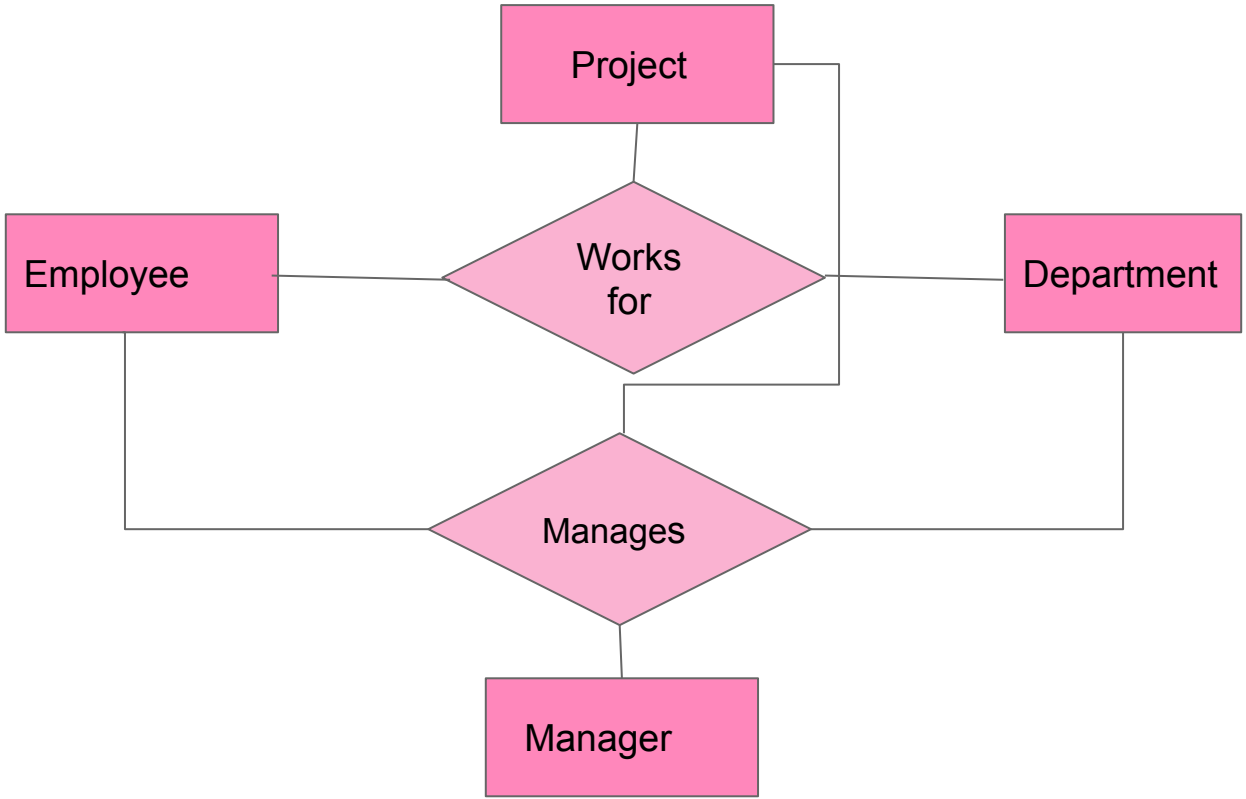
Completeness constraint

- **Completeness constraint** -- specifies whether or not an entity in the higher-level entity set must belong to at least one of the lower-level entity sets within a generalization.
  - **total**: an entity must belong to one of the lower-level entity sets
  - **partial**: an entity need not belong to one of the lower-level entity sets

Aggregation

Relationship with its corresponding entities is aggregated into a higher level entity.

Aggregation

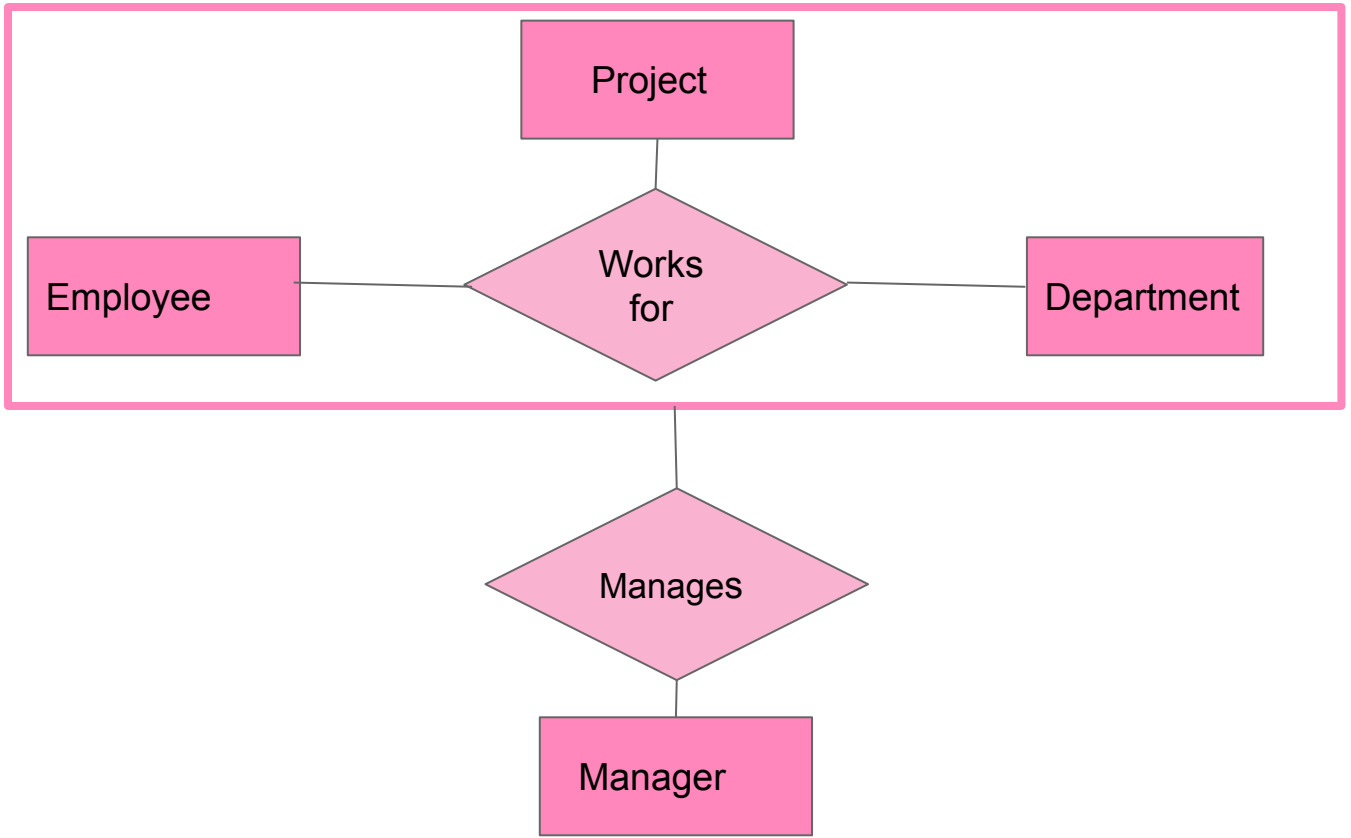


Aggregation

Consider the ternary relationship works_for. The Manager wants to manage department, employee and project

- Relationship sets *works for* and *manages* represent overlapping information
  - Every *works_for* relationship corresponds to a manages relationship
  - However, some *manages* relationships may not correspond to any *works_for* relationships. So we can't discard the *manages* relationship
- Eliminate this redundancy via *aggregation*
  - Treat relationship as an abstract entity
  - Allows relationships between relationships
  - Abstraction of relationship into new entity

Aggregation



Higher Level

Aggregation

- Eliminate this redundancy via aggregation without introducing redundancy, the following diagram represents:
  - A employee works_for particular department on a particular project.
  - A employee, department, project combination may have an associated manager.

Reduction to Relational Schemas

Entity sets and relationship sets can be expressed uniformly as *relation schemas* that represent the contents of the database.

Reduction to Relational Schemas

- Entity sets and relationship sets can be expressed uniformly as *relation schemas* that represent the contents of the database.
- A database which conforms to an E-R diagram can be represented by a collection of schemas.
- For each entity set and relationship set there is a unique schema that is assigned the name of the corresponding entity set or relationship set.
- Each schema has a number of columns (generally corresponding to attributes), which have unique names.

Representing Entity Sets

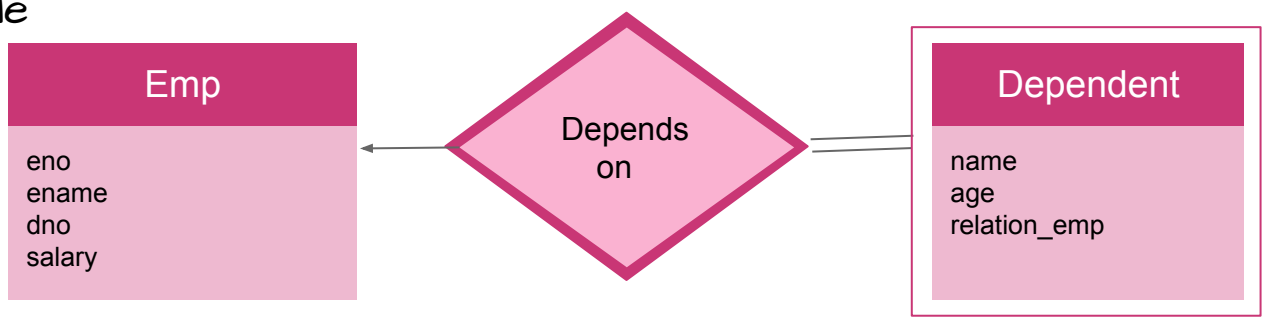
- A strong entity set reduces to a schema with the same attributes

  employee(eno, ename, dno,salary)

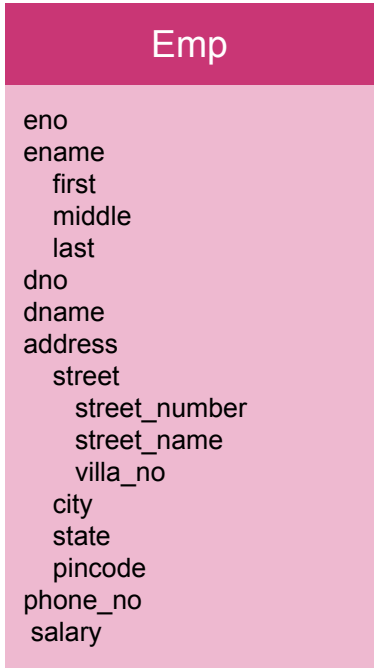
- A weak entity set becomes a table that includes a column for the primary key of the identifying strong entity set

  dependent (eno, name,age,relation_emp)

- Example



Representation of Entity Sets with Composite Attributes



- Composite attributes are flattened out by creating a separate attribute for each component attribute
  - Example: given entity set *emp* with composite attribute *name* with component attributes *first*, *middle* and *last* the schema corresponding to the entity set has three attributes *first*, *middle* and last
- Ignoring multivalued attribute, Extended emp schema is
  - *emp(eno, first, middle, last, dno, dname, street_number, street_name, villa_number,city, state, pincode, salary)*

Representation of Entity Sets with Multivalued Attributes

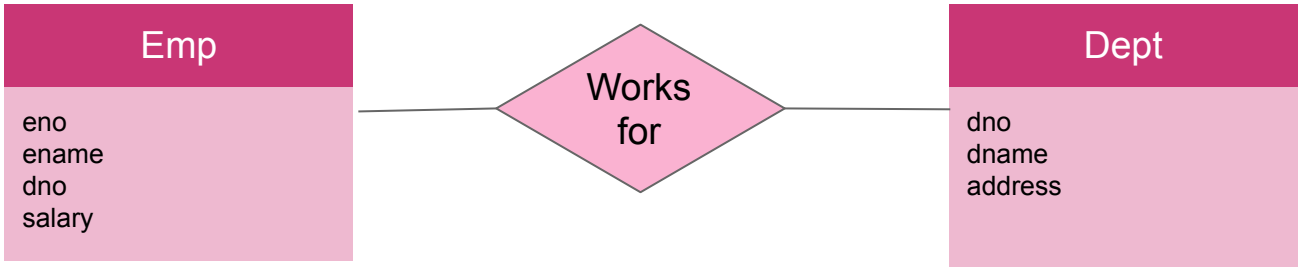
- A multivalued attribute *M* of an entity *E* is represented by a separate schema *EM*
- Schema *EM* has attributes corresponding to the primary key of *E* and an attribute corresponding to multivalued attribute *M*
- Example: Multivalued attribute *phone_number* of *emp* is represented by a schema:

  emp_phone= (*eno*, *phone_no*)
- Each value of the multivalued attribute maps to a separate tuple of the relation on schema *EM*
  - For example, an *emp* entity with primary key 10 and phone numbers 0422-2325553 and 0422-2325555 maps to two tuples: (10, 0422-2325553) and (10, 0422-2325555)

Representing Relationship Sets

- A many-to-many relationship set is represented as a schema with attributes for the primary keys of the two participating entity sets, and any descriptive attributes of the relationship set.
- Example: schema for relationship set works for

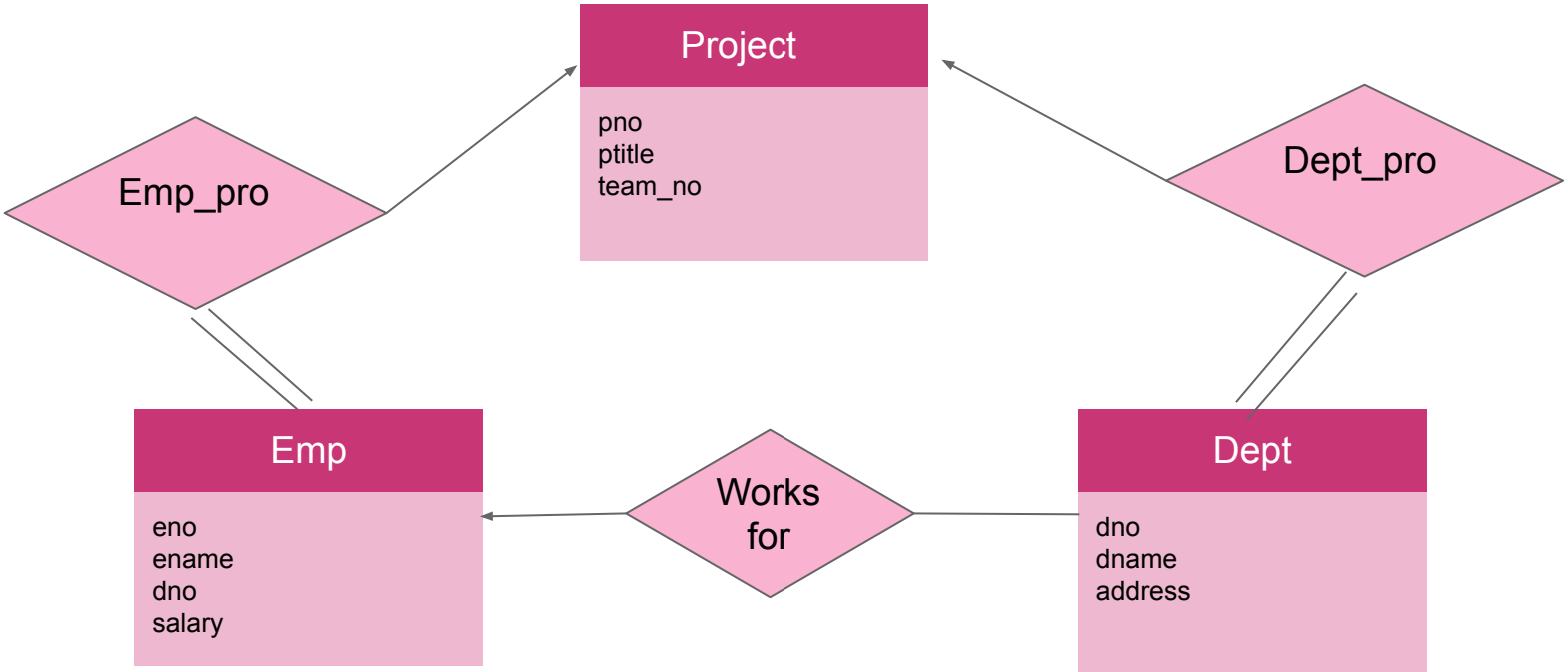
  works for = (<u>eno, dno</u>)



Redundancy of Schemas

- For one-to-one relationship sets, either side can be chosen to act as the "many" side
  - That is, an extra attribute can be added to either of the tables corresponding to the two entity sets
- If participation is *partial* on the "many" side, replacing a schema by an extra attribute in the schema corresponding to the "many" side could result in null values

Redundancy of Schemas



Redundancy of Schemas

- Many-to-one and one-to-many relationship sets that are total on the many-side can be represented by adding an extra attribute to the "many" side, containing the primary key of the "one" side
- Example: Instead of creating a schema for relationship set *emp_pro*, add an attribute *pno* to the schema arising from entity set *emp*

Redundancy of Schemas

- The schema corresponding to a relationship set linking a weak entity set to its identifying strong entity set is redundant.
- Example: The *employee* schema already contains the attribute eno that would appear in the *dependent* schema

Thanks!

Does anyone have any queries?

buvann@gmail.com

Credits

- Abraham Silberschatz ,Henry F. Korth and S. Sudarshan,"Database System Concepts", Seventh Edition, McGRAW Hill Education
- Gate Lectures by Ravindrababu Ravula
- http://slidesgo.com/