Database Design & Applications

Enhanced ERD

Objectives

After completing this module, you will be able to:

- Understand the need of Enhanced ERD
- Representing Supertype and Subtype
- Generalization
- Specialization
- Specifying constraints in supertype/subtype relationships
- Aggregation
- Conversion of Specialization to Table
- Conversion of Aggregation into Tables

Need of Enhanced ERD

- ERD was introduced in mid 1970s.
- It was suitable for modeling most common business problems.
- But in modern business environment business data and business relationships are more complex.
- The technology also developed during these years.
- The Enhanced ERD (EERD)model help in representing complex data.
- EER model is semantically similar to object-oriented data modeling.

Representing Supertypes And Subtypes

- One of the major challenges in data modeling is to recognize and clearly represent entities that are almost the same.
- Entity types that share common properties, but also have one or more distinct properties that are of interest to the organization.
- ERD has been extended to include subtype and supertype relationships
- A subtype is a subgrouping of the entities in an entity type that is meaningful to the organization.
- A supertype is a generic entity type that has a relationship with one or more subtypes.

Representing Supertypes And Subtypes

- Suppose that an organization has two basic types of employees:
  - Hourly employees
  - Salaried employees
- Some of the important attributes for each of these types of employees:
  - **Hourly employees:** Employee Number, Employee Name, Address, Date Hired, Hourly Rate
  - **Salaried employees:** Employee Number, Employee Name, Address, Date Hired, Annual Salary
- Both types of employees have several attributes common.
- Each type has one or more attributes distinct from the attributes of other types.

Representing Supertypes And Subtypes

- To develop a conceptual model for the problem:
  - ✔ Define a single entity type called EMPLOYEE.
  - ✔ Define a separate entity type for each of the two entities
  - ✔ EMPLOYEE entity is called the supertype
  - ✔ HOURLY EMPLOYEE and SALARIED EMPLOYEE is called the subtype

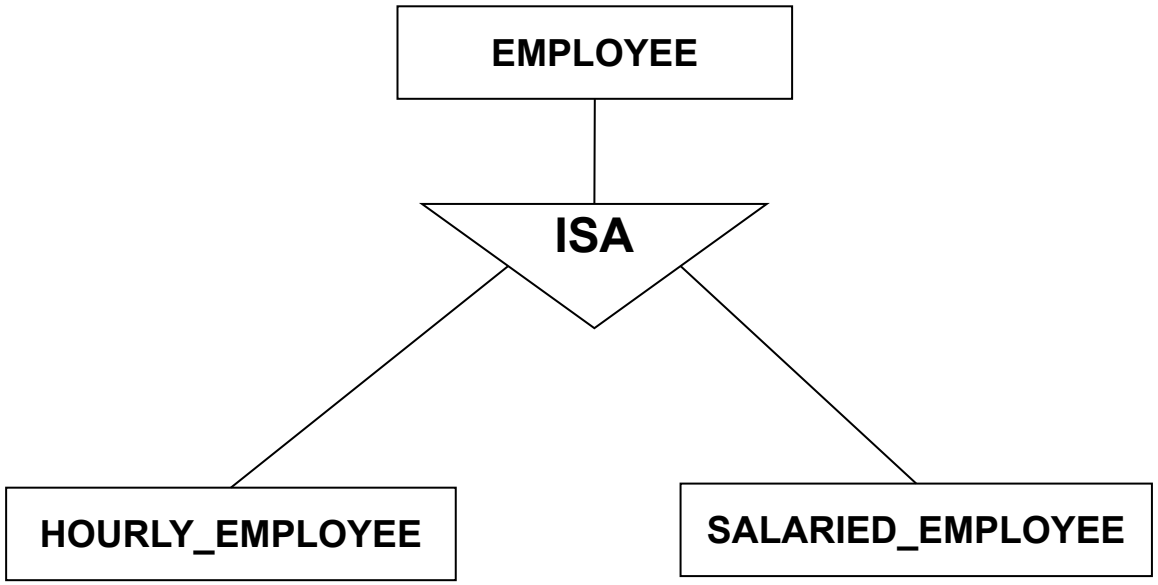


Figure : 3.1

Generalization

- Generalization is the process of defining a more general entity type from a set of more specialized entity types.
- It is a bottom-up process.
- It 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 become the subtype.
- Subtypes are dependent entities.

Generalization

- Two entities has common attributes: VehicleId, LicencePlateNo, Price

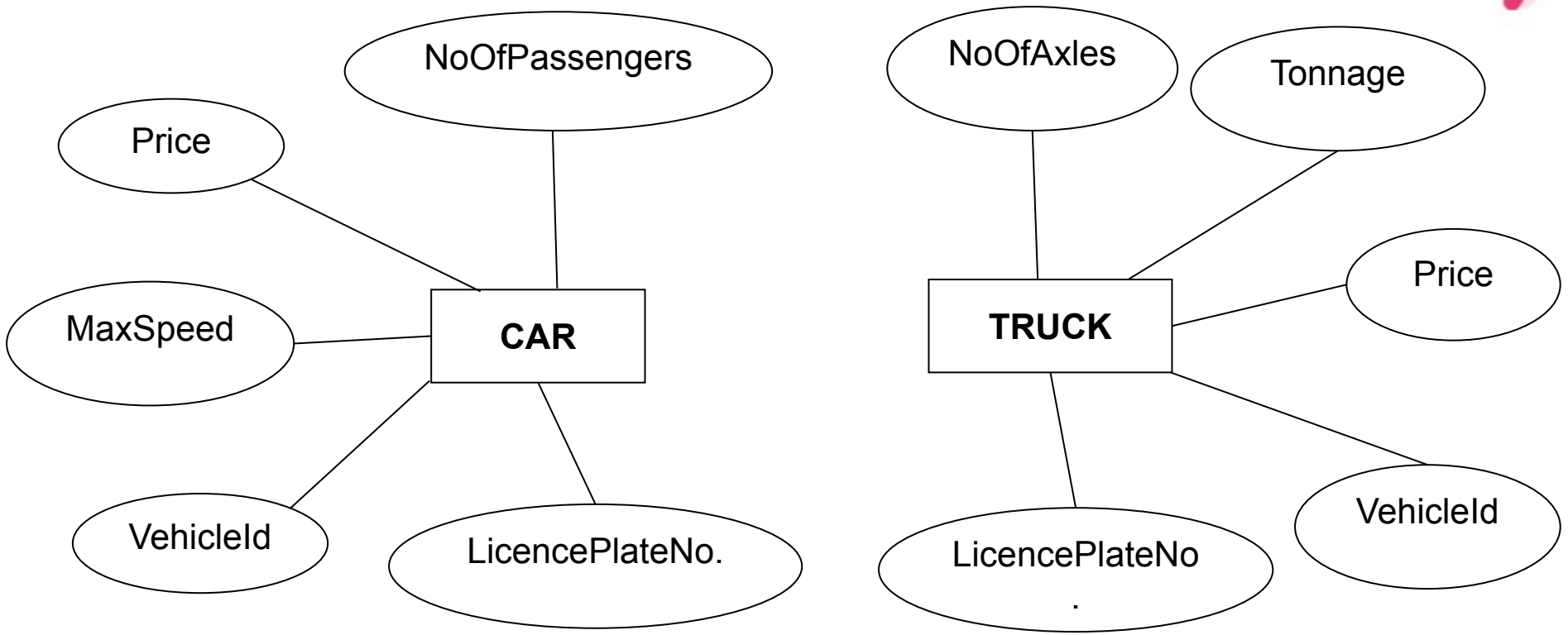


Figure : 3.2

Generalization

- Two entities **CAR** and **TRUCK** are really a version of more general entity type : VEHICLE
- The common attributes shared by both entity will become the attributes of VEHICLE.
- Generalization allows us to group entity types with their common attributes.
- Also preserves the specific attributes that are specific to each subtype entity.

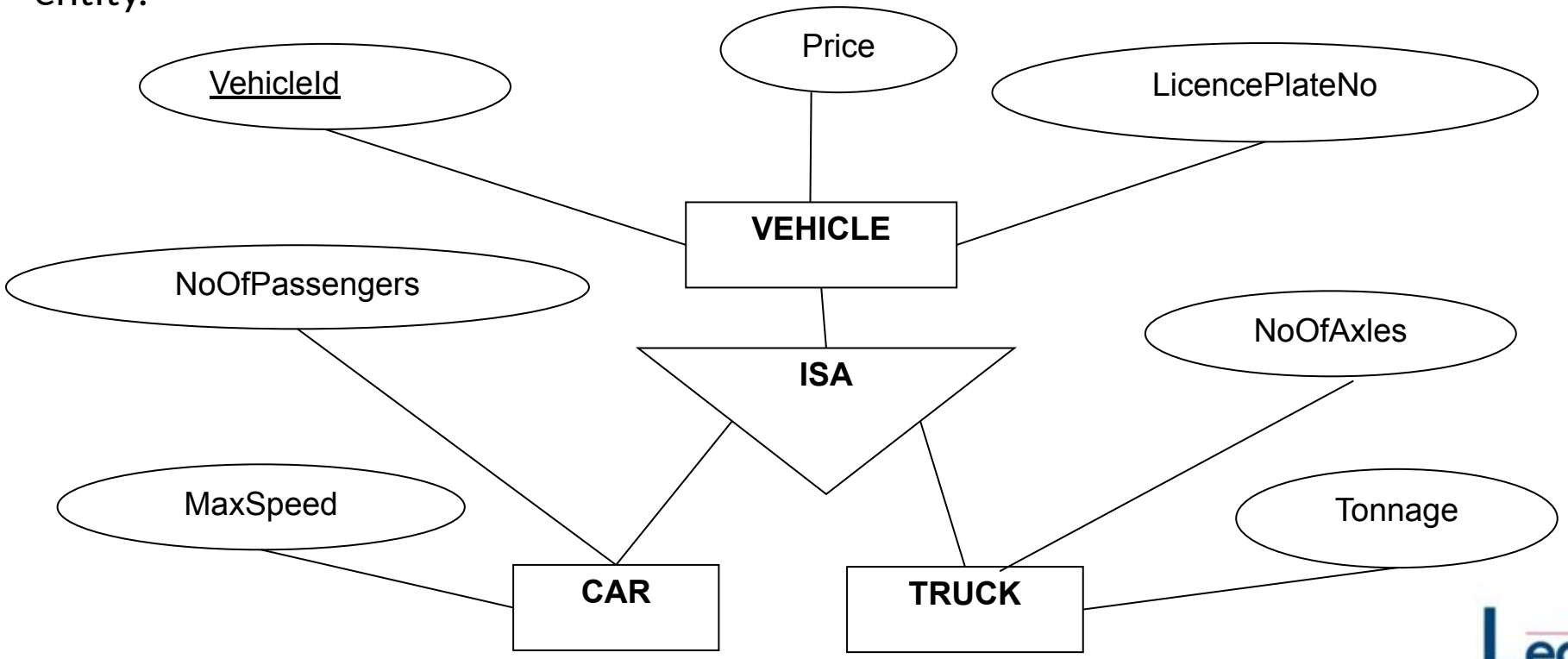


Figure : 3.3

Specialization

- Top-down process.
- Reverse of Generalization.
- It is a process of defining one or more subtypes of the supertypes and forming supertype/subtype relationships.
- Each subtype is formed based on some distinguishing characteristic, such as attributes or relationships specific to the subtype.

Specialization

- An entity set EMPLOYEE with attributes
  - Name, Employeeid, Address, Birthdate, Jobtype.
- An Employee may be further classified as one of the following (figure :3.4) :
  - Secretary
  - Engineer
  - Technician
- Each of these employee types is described by a set of attributes that includes all the attributes of entity set EMPLOYEE plus possibly additional attributes.
  - Secretary : Typingspeed
  - Technician :Tgrade
  - Engineer Engtype.

Specialization and Generalization

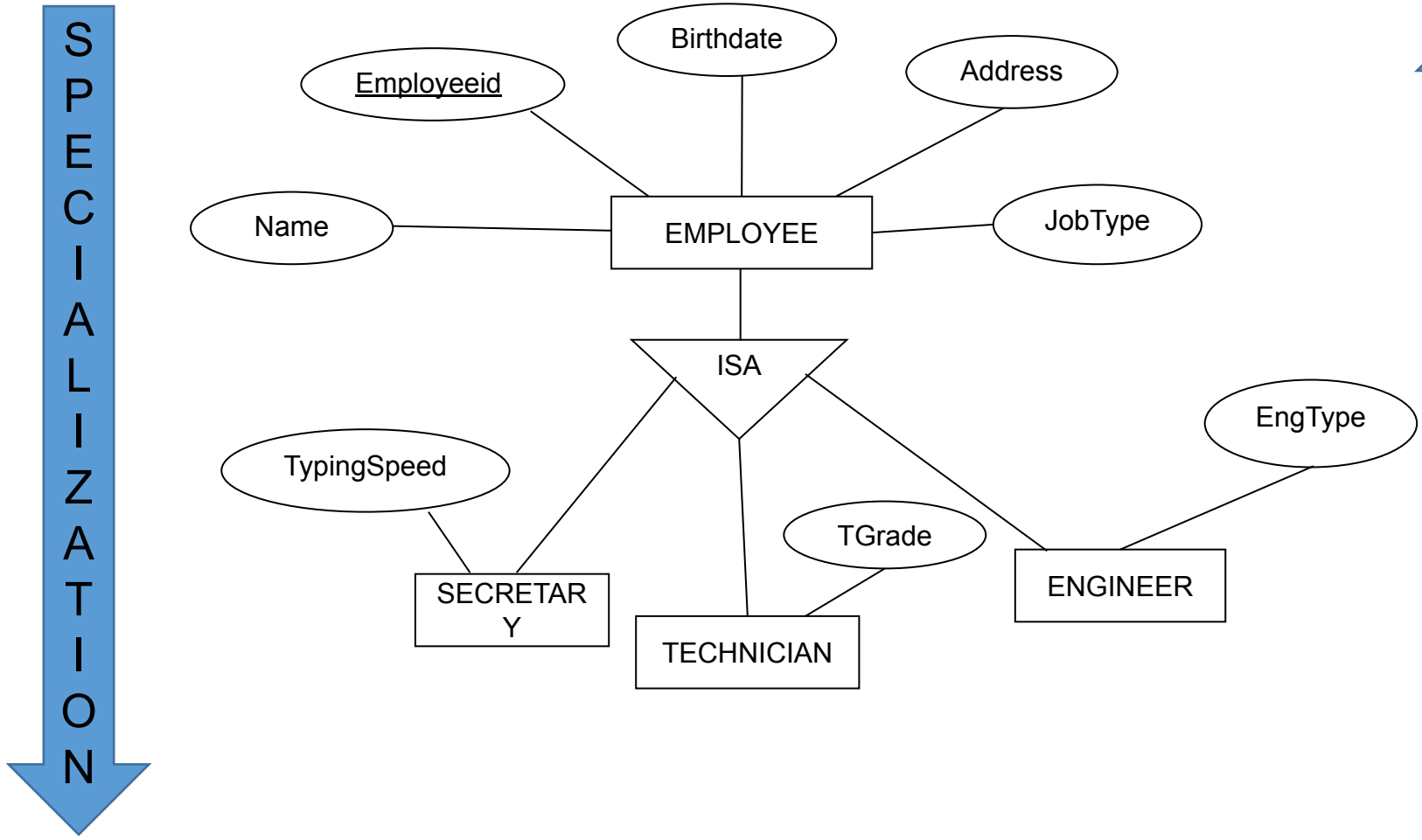


Figure : 3.4

Specifying Constraints In Supertype / Subtype Relationships

- **Condition-defined VS User-defined Constraint**
- **Condition-defined (figure: 3.4):**
  - Lower-level or subtype entity sets, membership is evaluated on the basis of whether or not an entity satisfies an explicit condition or predicate.
  - The value of jobtype attribute will decide which entity will belong to which subtype.
  - For example : those entities of the EMPLOYEE entity type whose attribute value for jobtype is ‘secretary’ belong to the subtype SECRETARY.

Specifying Constraints In Supertype/Subtype Relationships

- **Condition-defined VS User-defined Constraint**
- **User-defined(figure:3.5):**
  - Lower-level or subtype entity set's membership is specified individually for each entity by the user
  - The membership is specified individually for each entity by the user, not by any condition
  - For instance, let us assume that, after 3 months of employment, the employees of a software company are assigned to one of three work teams
  - We therefore represent the teams as three lower-level entity sets of the higher-level employee entity set.

Specifying Constraints In Supertype /Subtype Relationships

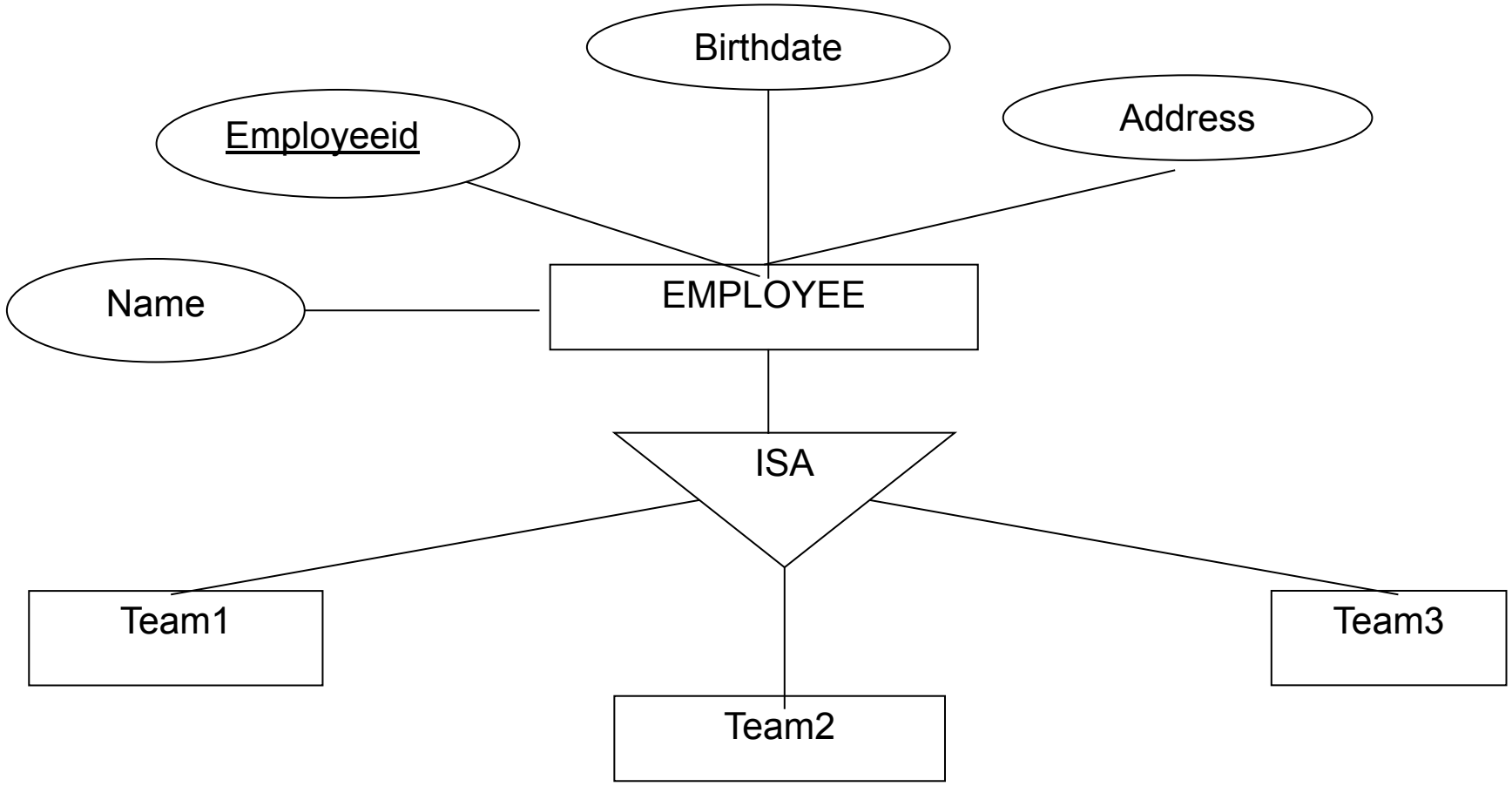


Figure : 3.5

Specifying Constraints In Supertype/Subtype Relationships

- **Completeness Constraints:** addresses the question of whether an instance of a supertype must also be a member of at least one subtype
- **Total Specialization:** A total specialization constraint specifies that every entity in the supertype must be a member of least one subtype in the specialization
- For example, if every employee must be either an **HOURLY_EMPLOYEE** or a **SALARIED_EMPLOYEE**(Figure:3.6).
- Can be represented by using a double line to connect the box representing the higher-level entity set to the triangle symbol.

Specifying Constraints In Supertype/Subtype Relationships

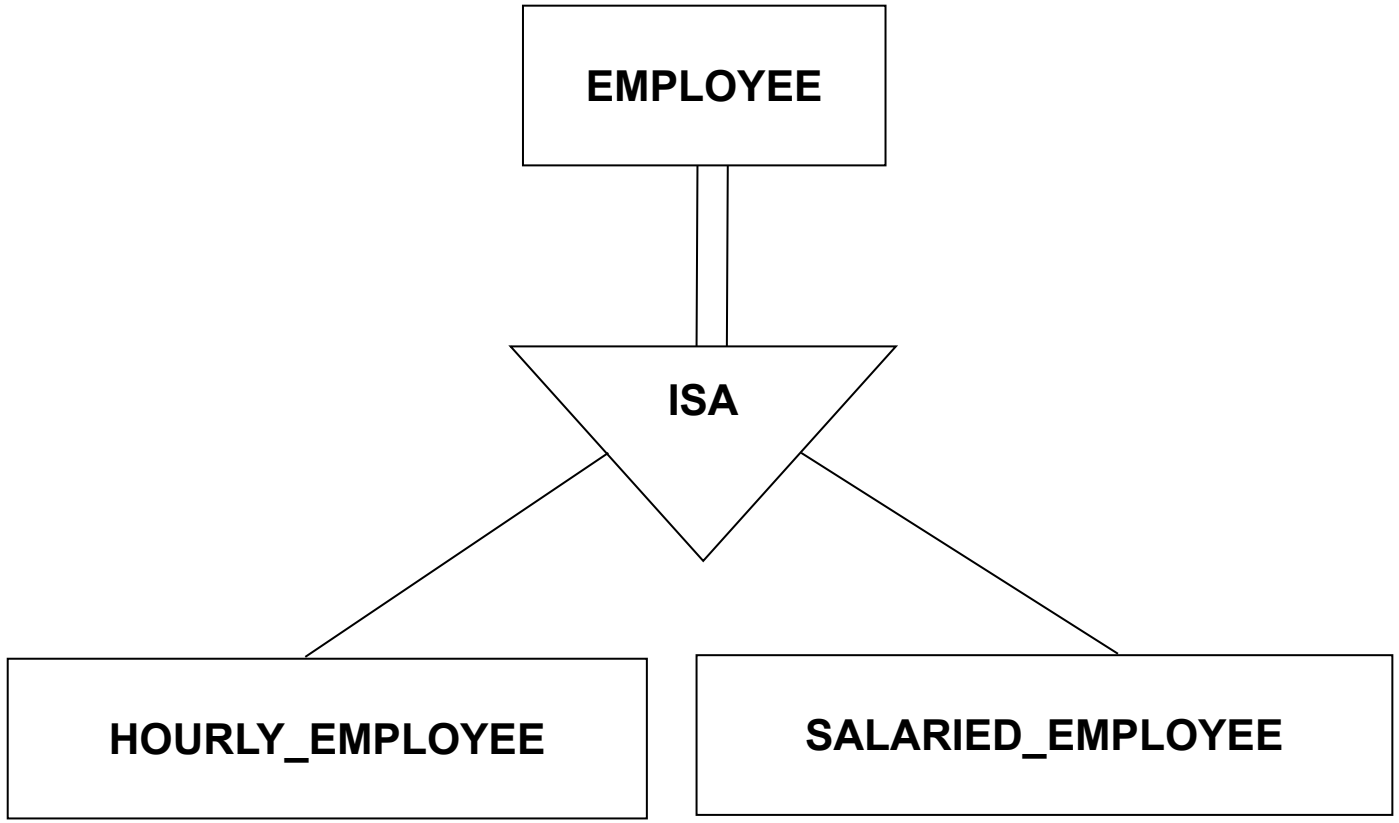


Figure : 3.6

Specifying Constraints In Supertype/Subtype Relationships

- **Partial Specialization:** In partial specialization some of higher-level entities may not belong to any lower-level entity set.
- For example, motorcycle is a type of VEHICLE, but does not represented as a subtype in the data model (figure: 3.7)
- Partial generalization is by default.

Specifying Constraints In Supertype/Subtype Relationships

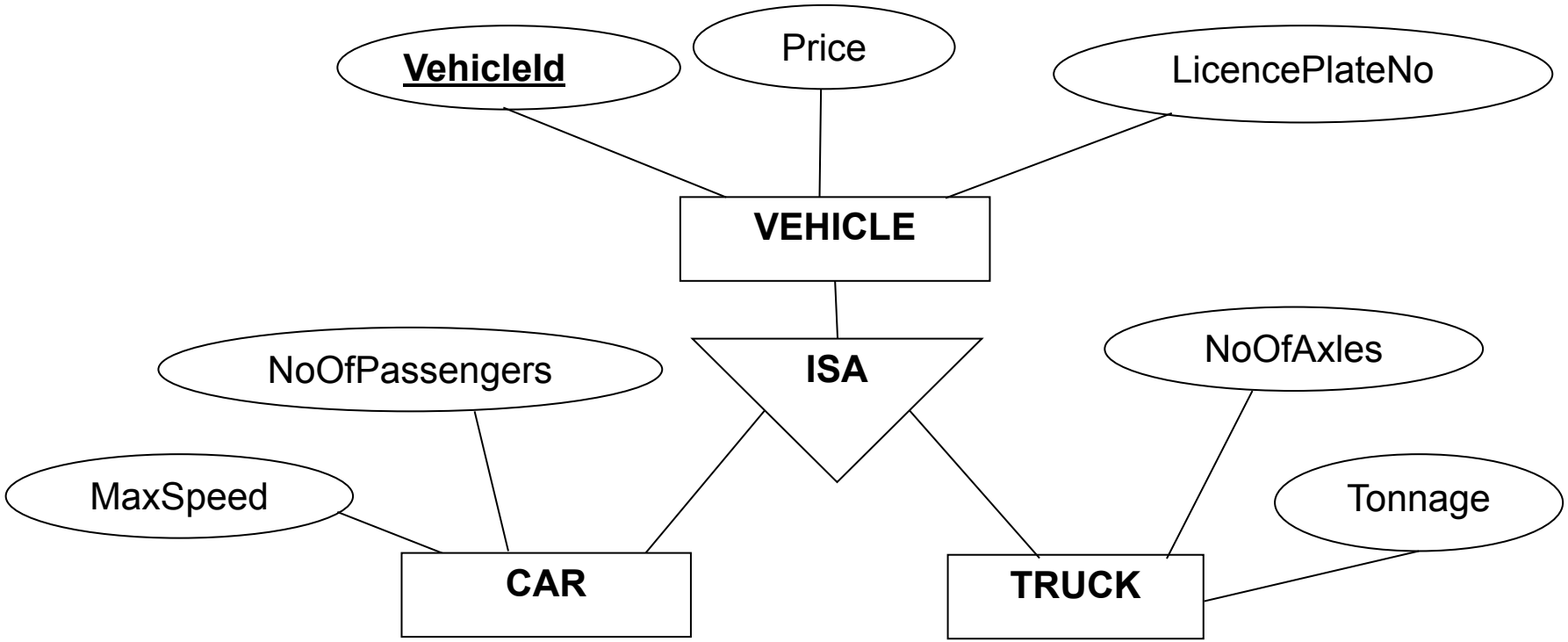


Figure : 3.7

Specifying Constraints In Supertype/Subtype Relationships

- **Disjointness Constraints:** addresses whether an instance of a supertype may simultaneously be a member of two (or more) subtypes.
- **Disjoint:** The disjoint rule specifies that if an entity instance (of the supertype) is a member of one subtype, it cannot simultaneously be a member of any other subtype
- For example, the entity EMPLOYEE, may have two subtypes, CLASSIFIED and WAGES. An employee may be one type or the other but not both (Figure:3.8).

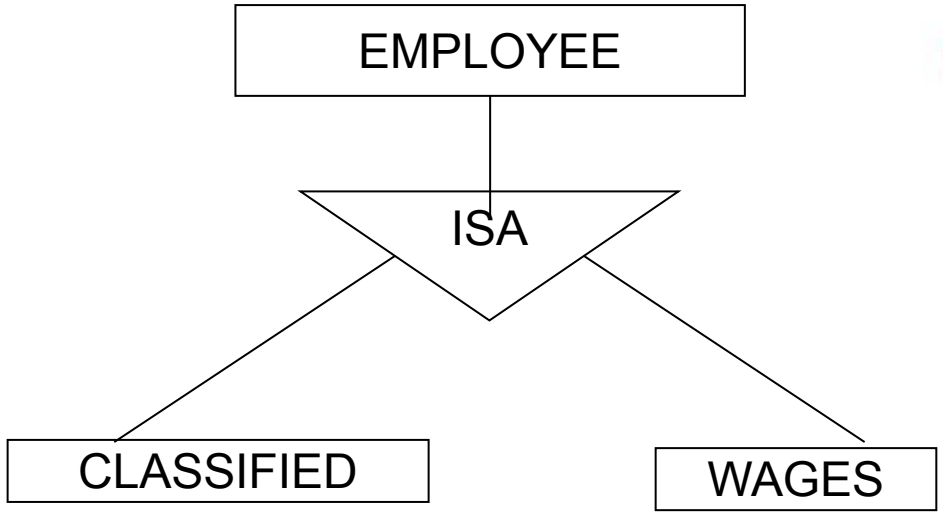


Figure : 3.8

Specifying Constraints In Supertype/Subtype Relationships

- **Disjointness Constraints**
- **Overlapping :** The overlap rule specifies that an entity instance can simultaneously be a member of two (or more) subtypes.
- For example: a person who works for a university could also be a student at that same university(figure: 3.9)

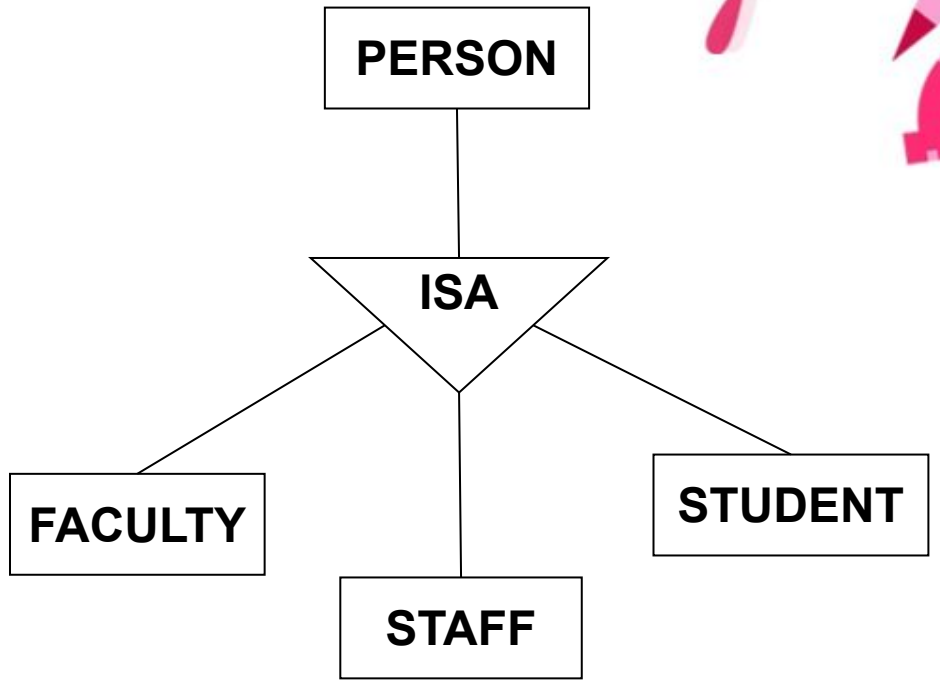


Figure : 3.9

Aggregation

- The E-R model has one limitation that it is not possible to express relationships among relationships. In such situations we use aggregation
- Aggregation, is an approach of modeling a relationship set as a higher level entity set
- Aggregation helps to model the participation of one relationship set into other relationship set
- For example, consider ‘publishing_house’ database in which an author writes a particular book and a publisher publishes the book on a specific date written by a particular author. (Figure:3.9)
- The situation is modelled by associating ‘publisher’ with entity set ‘author’ and ‘book’ through a relationship ‘publish’

Aggregation

E-R Diagram with Redundant Relationship

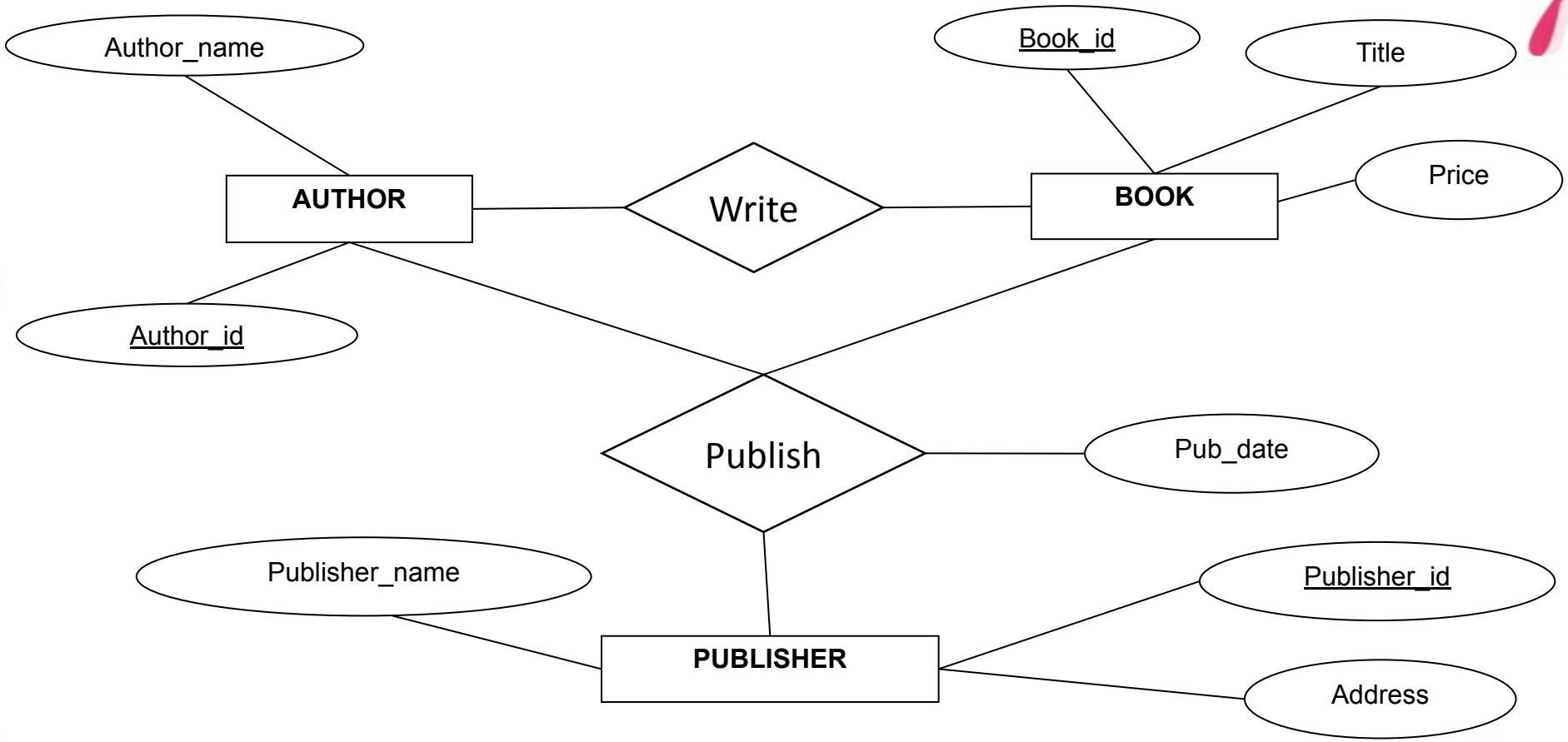


Figure : 3.9

Aggregation

- But this model produces redundant information since every ‘AUTHOR-BOOK’ pair in ‘publish’ is also in ‘Writes’ (Figure:3.9)
- Better method to model the situation is aggregation.
- Aggregation is an abstraction through which relationships are treated as higher level entities.
- We can treat the relationships set ‘Writes’ along with the entity sets AUTHOR and BOOK as a higher level entity set called author-book.
- And associate the Publish relationship with this abstract entity or aggregation(figure:3.10)

Aggregation

E-R Diagram with Aggregation

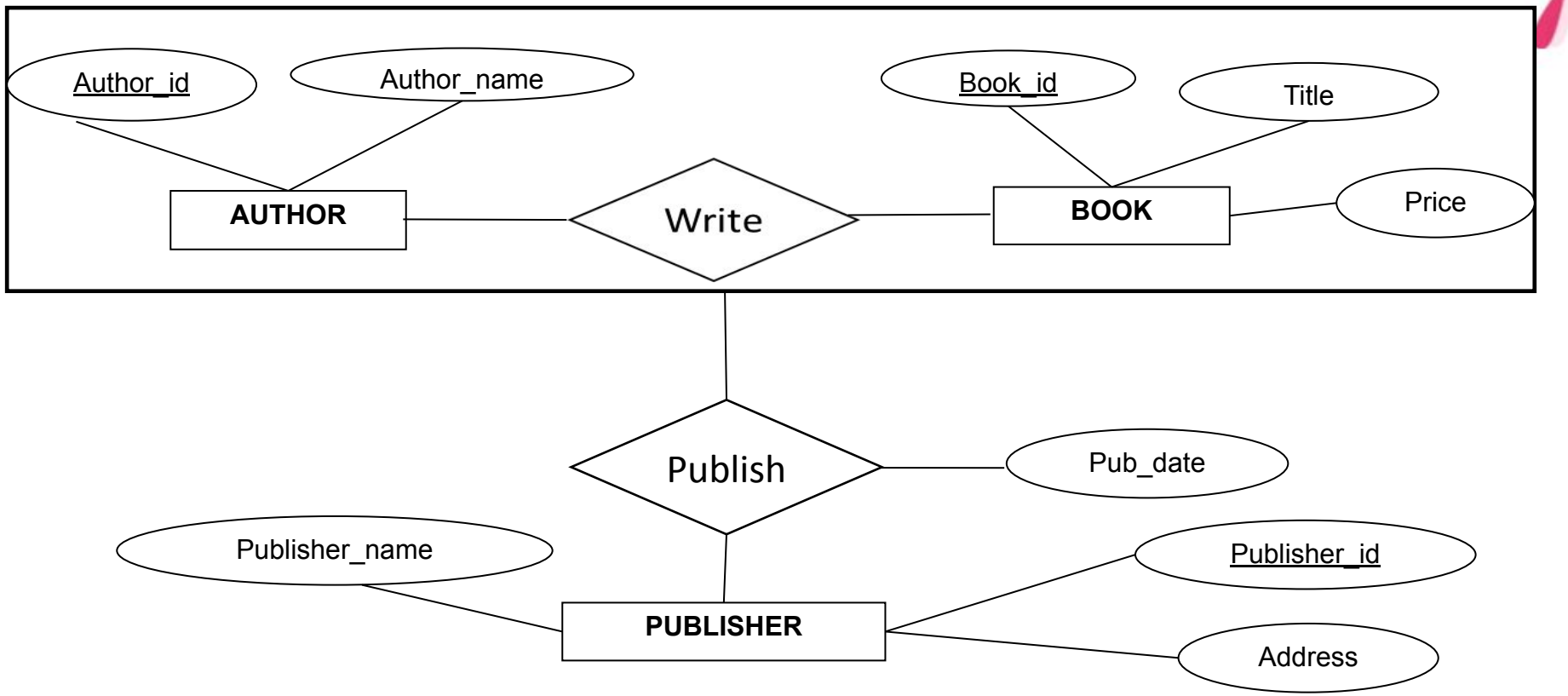


Figure : 3.10

Conversion of Specialization to Table

- Create a separate relation for the supertype and for each of its subtypes.
- Super type entity set is represented by table containing attribute common to lower level entity set, including primary key.
- Sub type entity set is represented by table containing specialized attributes corresponding to that lower level entity set and a column as primary key of higher level entity set.
- Assign one (or more) attributes of the supertype to function as the subtype discriminator.

Conversion of Specialization to Table

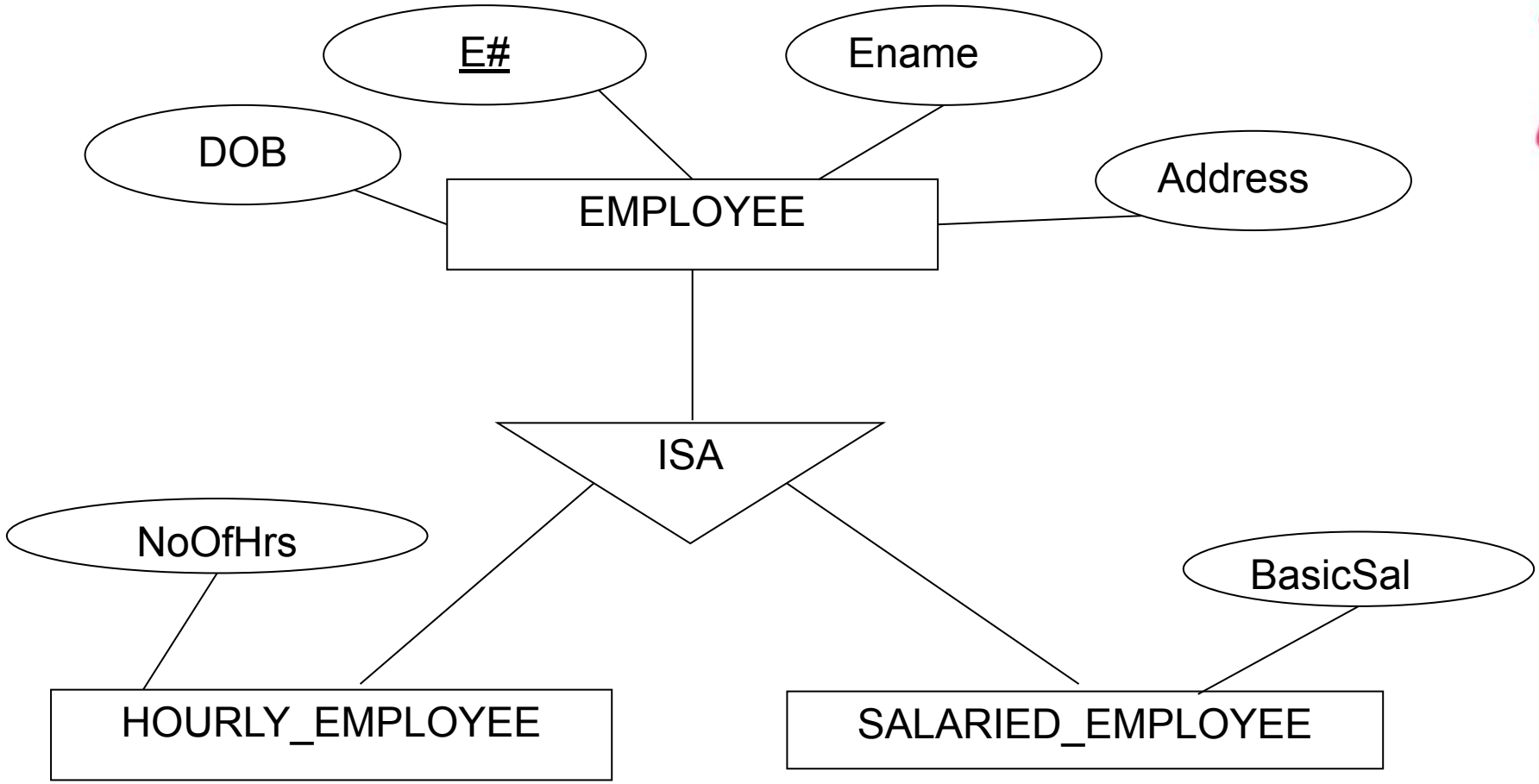
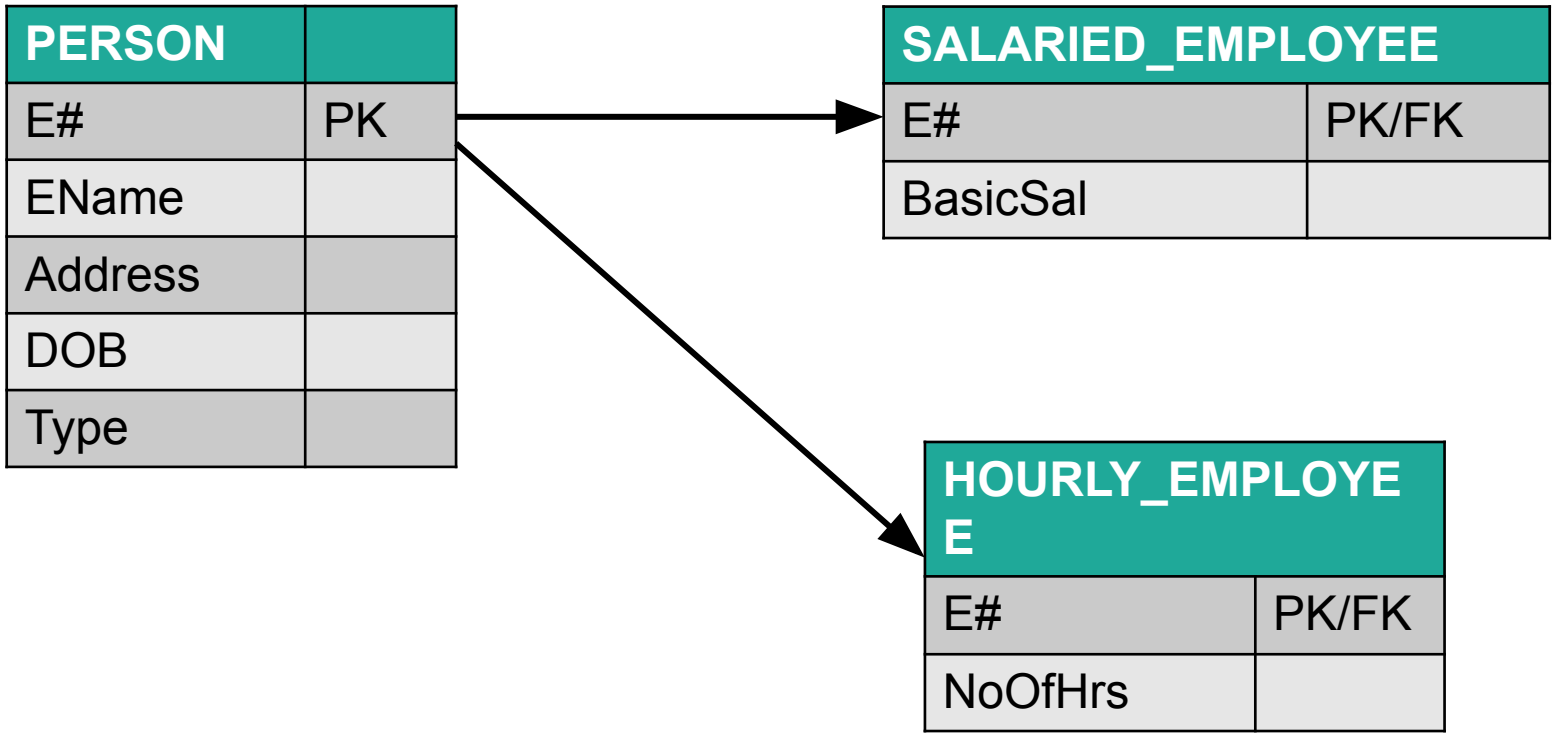


Figure : 3.11

Conversion of Specialization to Table



Conversion of Aggregation into Tables

- If an aggregated entity set combines two entity sets with one relationship and this aggregated entity set is related with one more entity than
- Each entity set is represented by one table
- The relationship between aggregated entity set is represented by one table, and the final table structure depends upon the relationship between the tables.
- The relationship between aggregated entity set and related entity is represented by one table, containing primary key from each table.

THANK YOU!