Analysis modelling

Analysis modelling……

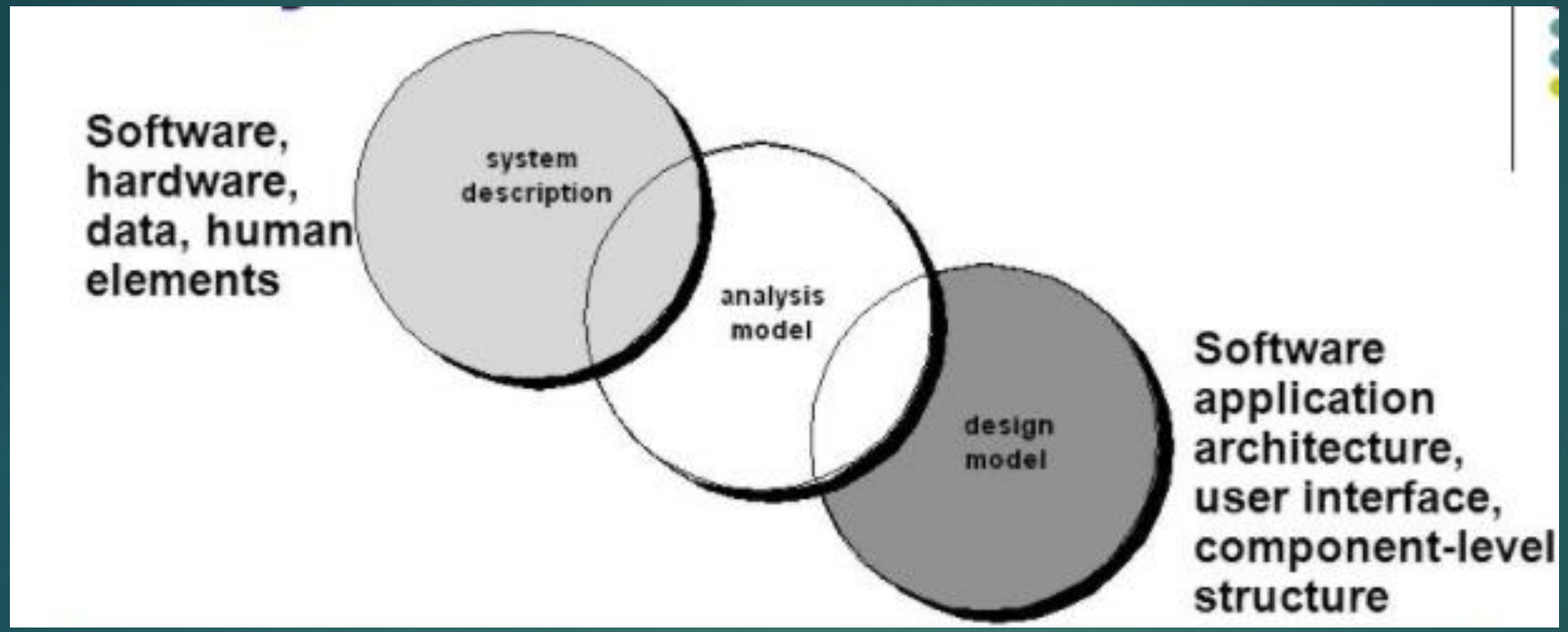
- **Analysis Model** is a technical representation of the system.
- It acts as a link between system description and design model.
- In Analysis Modelling, information, behavior, and functions of the system are defined and translated into the architecture, component, and interface level design in the design modeling.

Analysis modelling……

- The development process starts with the analysis phase.
- This phase results in a specification document that shows what the software will do without specifying how it will be done.
- The analysis phase can use two separate approaches, depending on whether the implementation phase is done using a procedural programming language or an object-oriented language.

Analysis principles

- The information domain must be represented and understood.
- Models should be developed to give emphasis on system information, function and behavior.
- Models should uncover and give details of all the layers of the development process.
- The function and the problem statement must be defined.
- The various analysis models are flow oriented modelling, scenario based modelling, class based modelling, and behavioral modelling.


Software, hardware, data, human elements
system description
analysis model
design model
Software application architecture, user interface, component-level structure

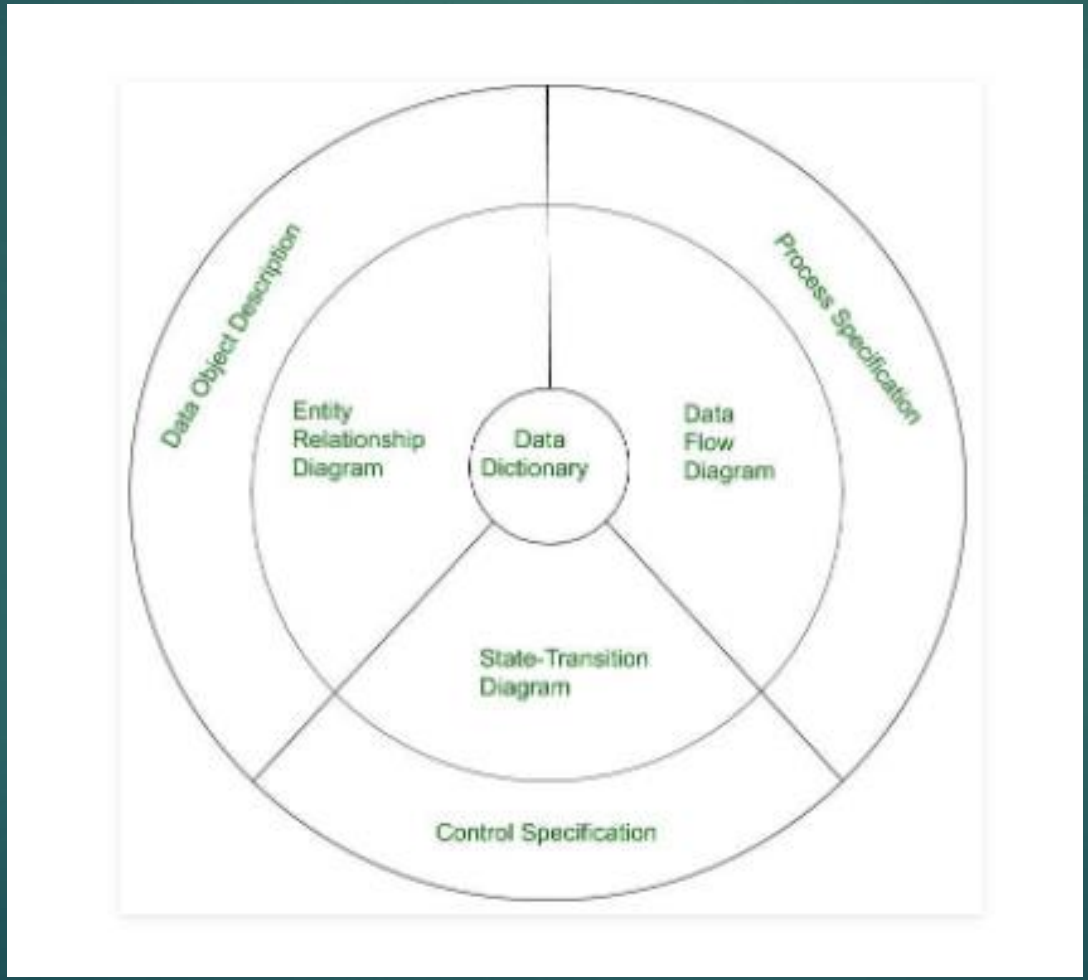
Activities involved

- Analysis modelling describes the operational and behavioral characteristics.
- Shows the relationship between software interface and other software elements.
- Provides the software developer the representation of the information, function, and behavior.
- Coverts the design into the more descriptive models like use case, ER diagram.
- Provides the customer and the developer the means to maintain the quality.

Objective

- Describe what the customer requires.
- Establish a basis for the creation of a software design.
- Devise a set of requirements that can be validated once the software is built.
- Analysis model bridges the gap between system level description and the overall system functionality.

Elements of analysis model



Must be achieved

- The model should focus on requirements that are visible within the problem or business domain and be written as a relatively high level of abstraction.
- Each element of the analysis model should add to the understanding of the requirements and provide insight into the information domain, function, and behaviour of the system.
- Delay consideration of infrastructure and other non-functional models until design.
- Minimize coupling throughout the system.
- Be certain the analysis model provides value to all stakeholders.
- Keep the model as simple as possible.

Analysis Modeling Approaches

Structured analysis

- Considers data and processes that transform data as separate entities.
- Structure analysis is a top down approach.
- It focuses on refining the problem with the help of the functions performed on the problem domain.

Object-oriented analysis

- Focuses on the definition of classes and the manner in which they collaborate to effect the customer requirements.
- Defines the system as a set of objects which interact with each other with the services provided.
- Analyses the problem domain and then partitions the problem with the help of objects.
- The concept of object, attributes, class, operation, inheritance, and polymorphism should be known to work on object oriented modelling

Domain Analysis

- Domain Analysis is the process that identifies the relevant objects of an application domain.
- The goal of Domain Analysis is Software Reuse.
- The higher is the level of the life-cycle object to reuse, the larger are the benefits coming from its reuse, and the harder is the definition of a workable process.

Concept and technical application domain of the software

- Frameworks are excellent candidates for Domain Analysis: they are at a higher level than code but average programmers can understand them.
- Umbrella activity involving the Identification, analysis, and specification of common requirements from a specific application domain, typically for reuse in multiple projects
- Object-oriented domain analysis involves the identification, analysis, and specification of reusable capabilities within a specific application domain in terms of common objects, classes, subassemblies, and frameworks

Input and Output Structure of domain analysis

- The main goal is to create the analysis classes and common functions.
- The input consists of the knowledge domain.
- The input is based on the technical survey, customer survey and expert advice.
- The output domain consists of using the input as the reference and developing the functional models

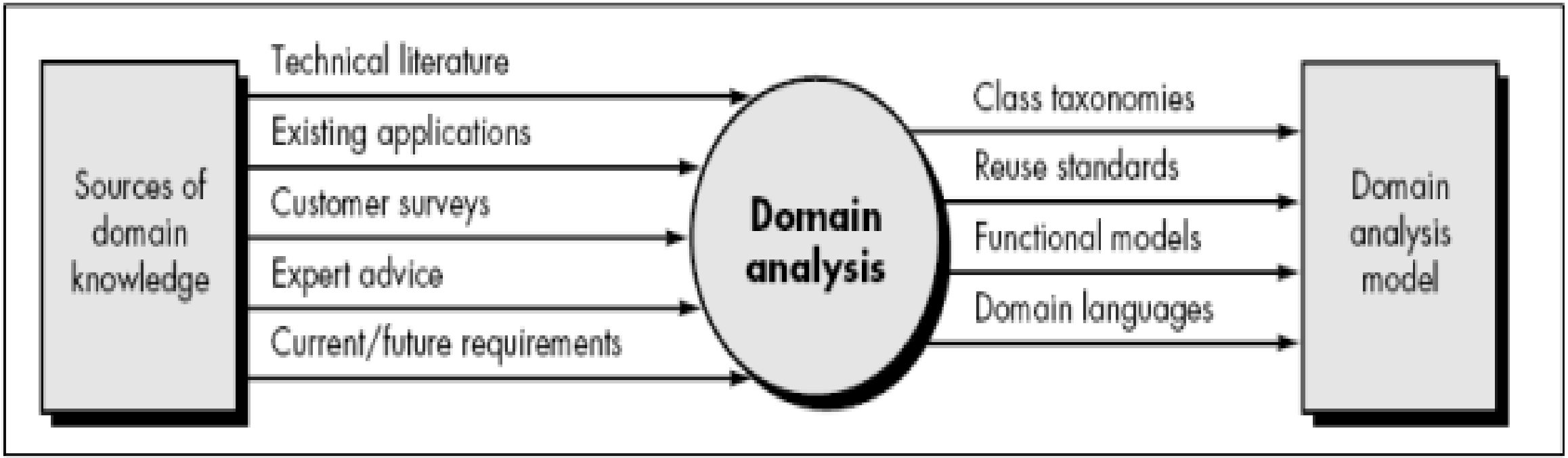


Figure 5: Domain Analysis

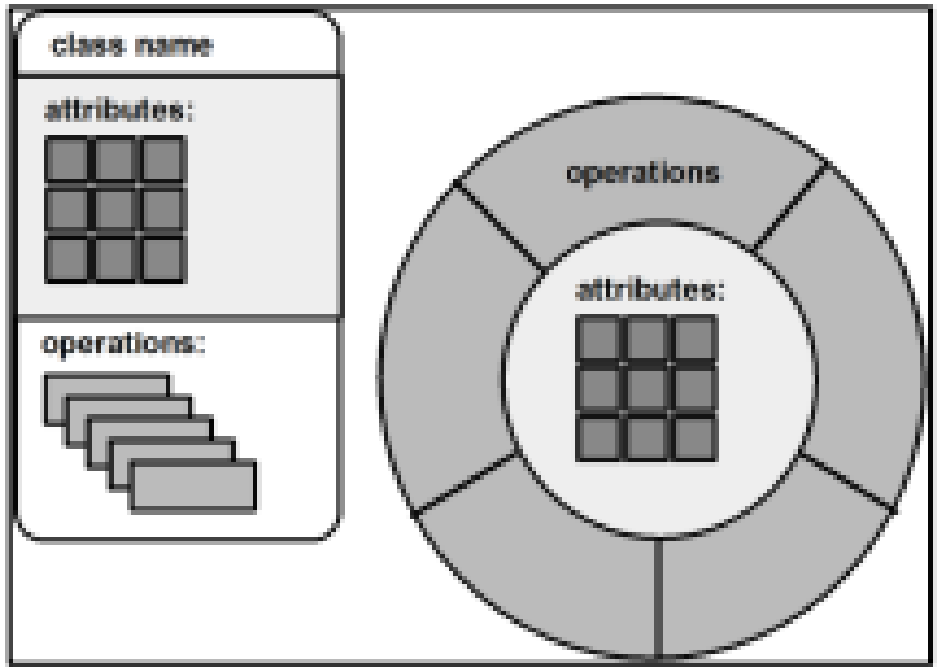


Figure 3: Class Domain

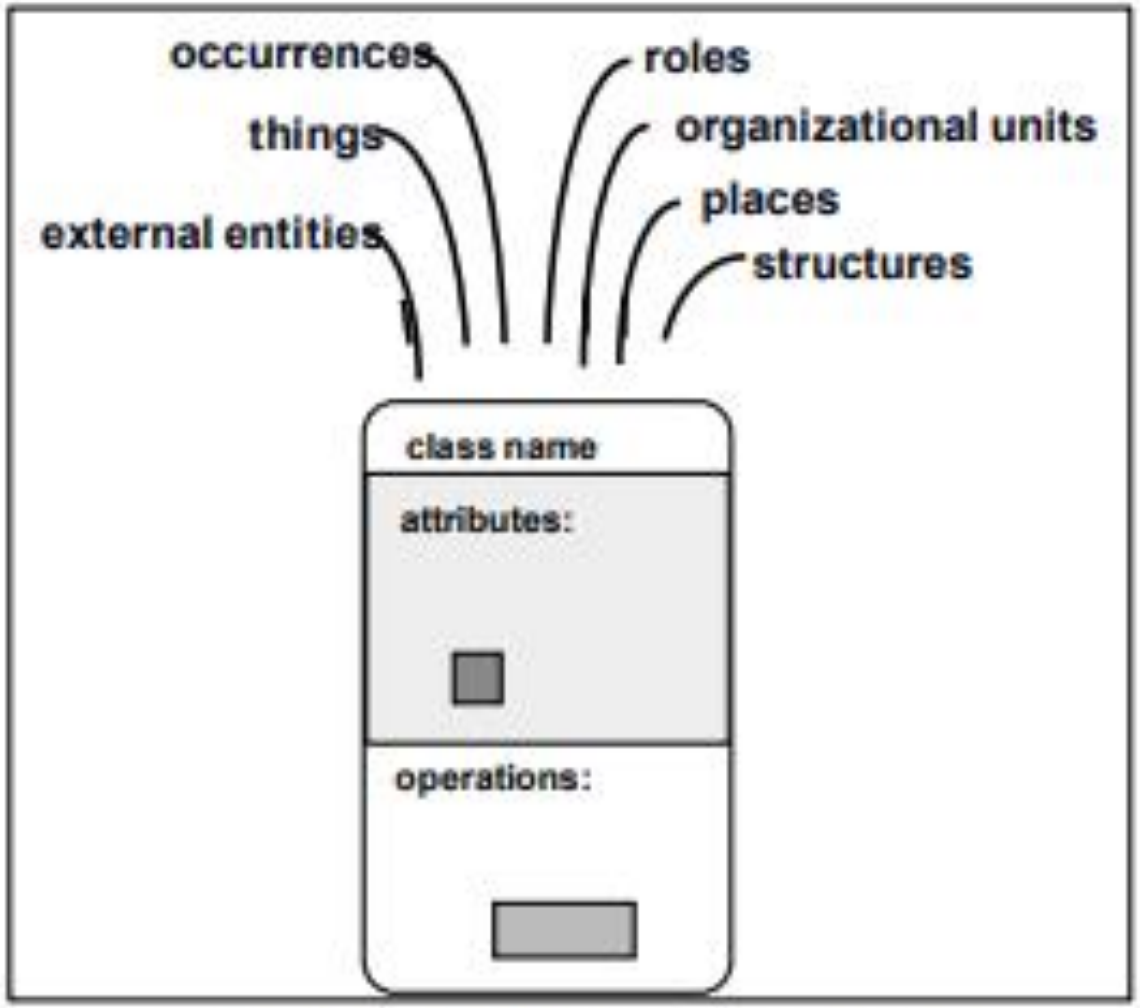


Figure 4: Class, attributes and operations

Building The Analysis Model

Involves

- Data Modelling
- Flow oriented modelling
- Scenario Based modelling

Data Modeling

- Data modeling is the analysis of data objects that are used in a business or other context and the identification of the relationships among these data objects.
- Data modeling is a first step in doing object-oriented programming.
- A data model can be thought of as a diagram or flowchart that illustrates the relationships between data.
- Data modelers often use multiple models to view the same data and ensure that all processes, entities, relationships and data flows have been identified.
- Data objects are modeled to define their attributes and relationships.

There are several different approaches to data modeling, including:

- Conceptual Data Modeling - identifies the highest-level relationships between different entities.
- Enterprise Data Modeling - similar to conceptual data modeling, but addresses the unique requirements of a specific business.
- Logical Data Modeling - illustrates the specific entities, attributes and relationships involved in a business function. Serves as the basis for the creation of the physical data model.
- Physical Data Modeling - represents an application and database-specific implementation of a logical data model

Data objects (Entities)

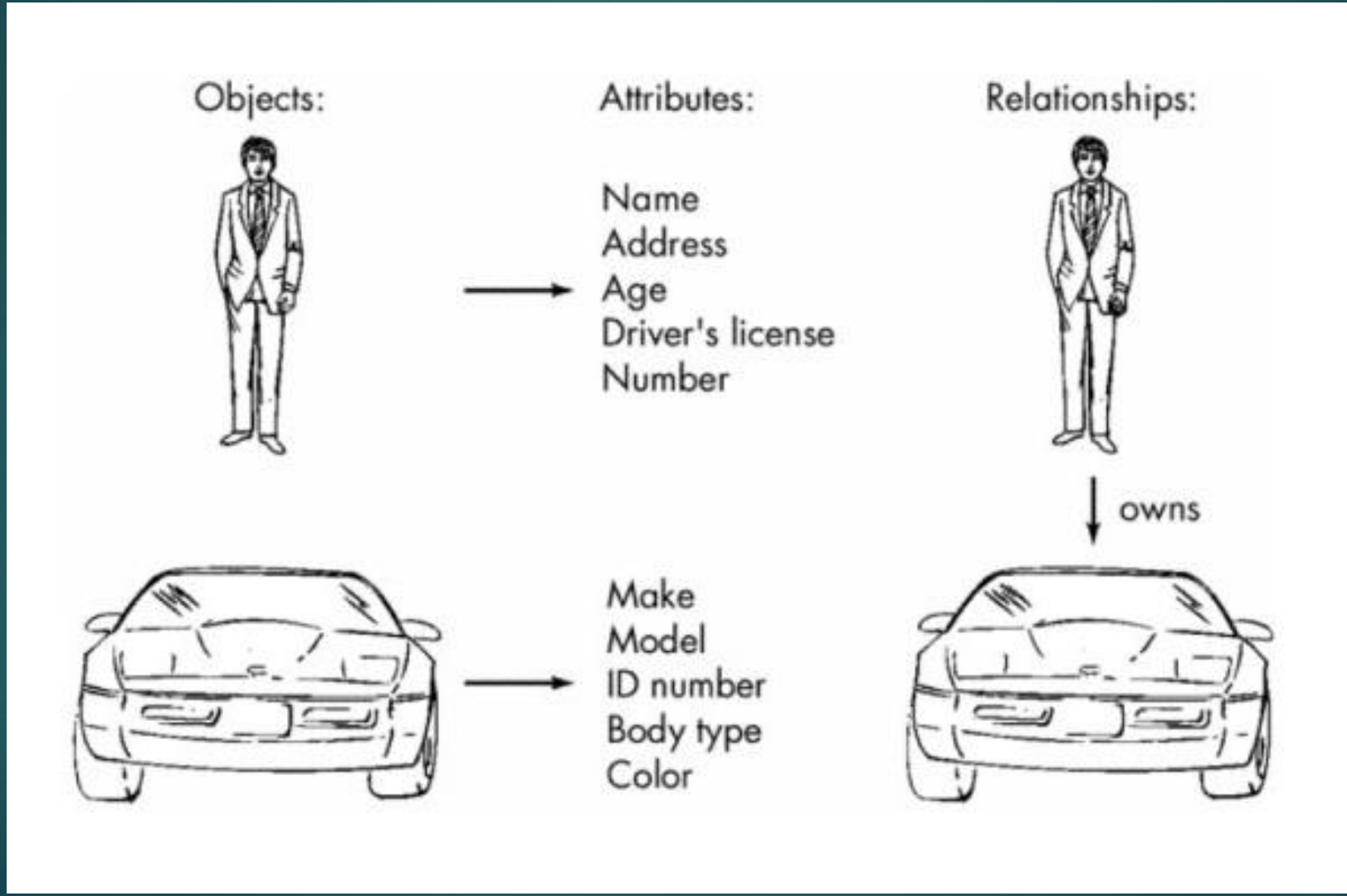
- Data objects are the representation of the most composite information of the system.
- Data object description incorporates the data object and all of its attributes.
- Data objects are all related to each other

Data attributes

- Attributes define the properties of the data object.
- Attributes are used to name the instances of the data. They describe the instance of the data.
- It helps to make reference to the other instance in another table

Relationships

- Data objects are linked with each other in different ways. These links and connections of data objects are known as relationships.


Objects:
Attributes:
Relationships:
Name
Address
Age
Driver's license
Number
owns
Make
Model
ID number
Body type
Color

Instance

Naming attributes

Ties one data object to another, in this case, owner

Identifier

Descriptive attributes

Referential attributes

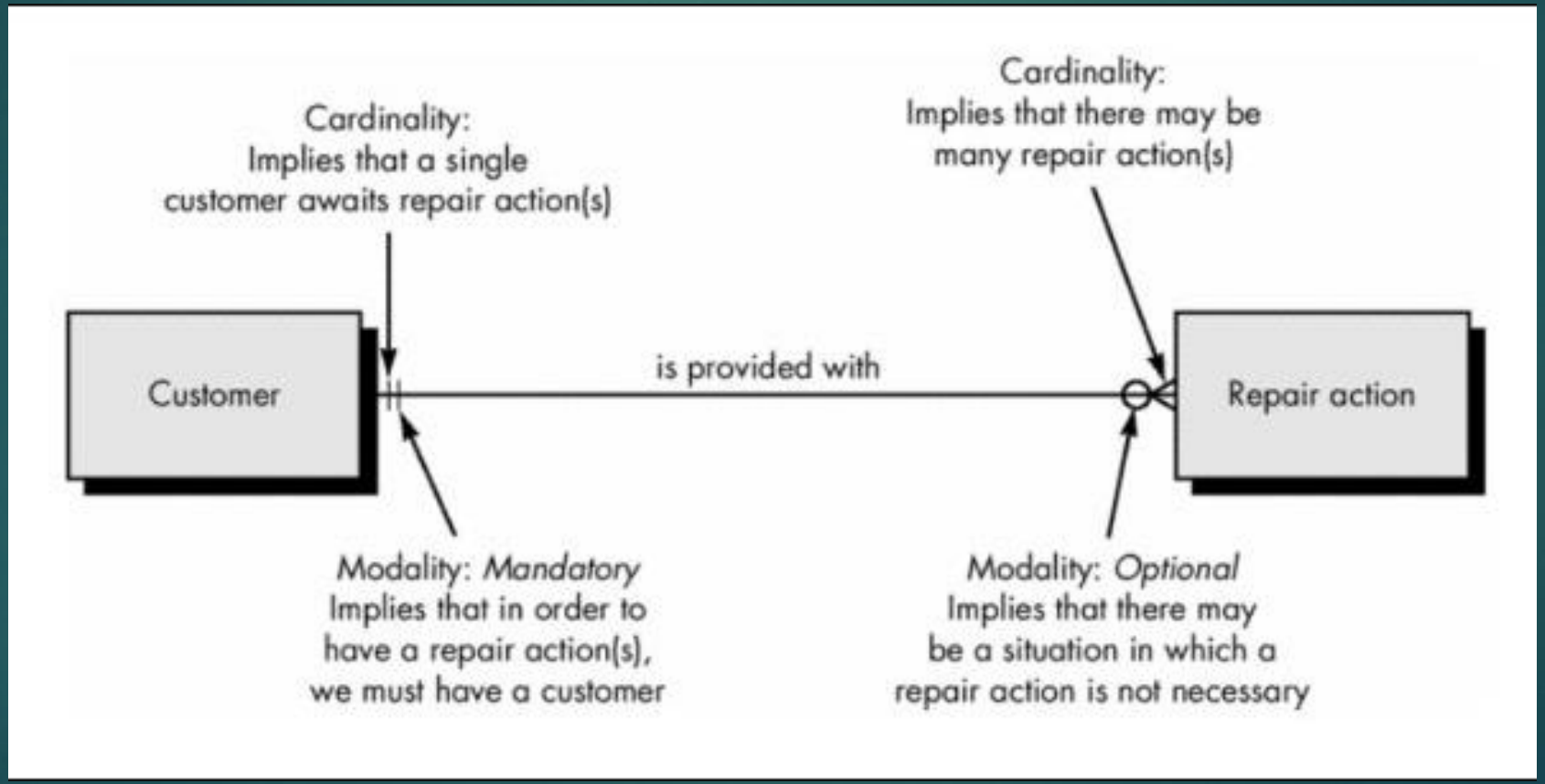
Make	Model	ID#	Body type	Color	Owner
Lexus	LS400	AB123. . .	Sedan	White	RSP
Chevy	Corvette	X456. . .	Sports	Red	CCD
BMW	750iL	XZ765. . .	Coupe	White	LJL
Ford	Taurus	Q12A45. . .	Sedan	Blue	BLF

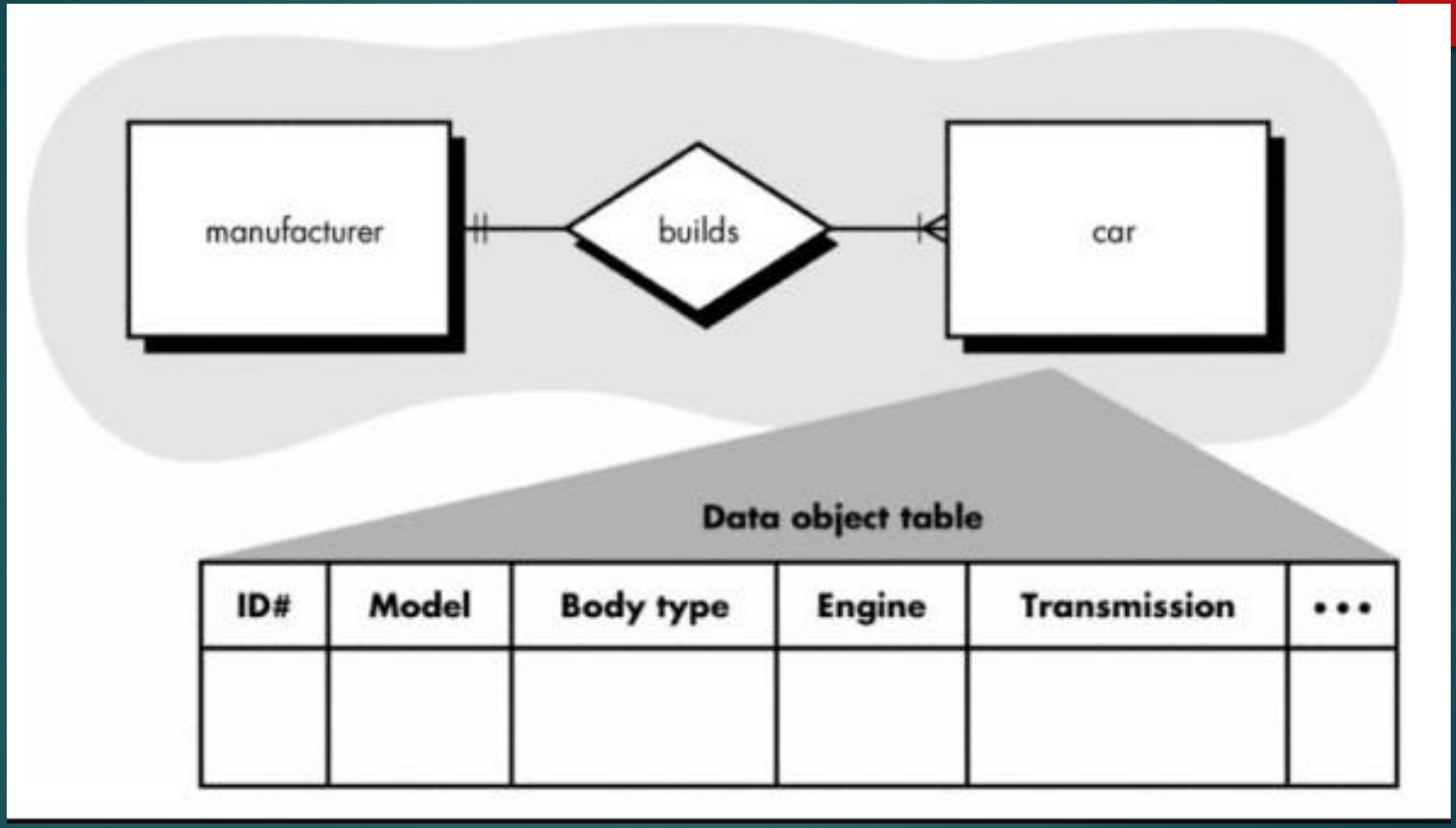
Cardinality (number of occurrences)

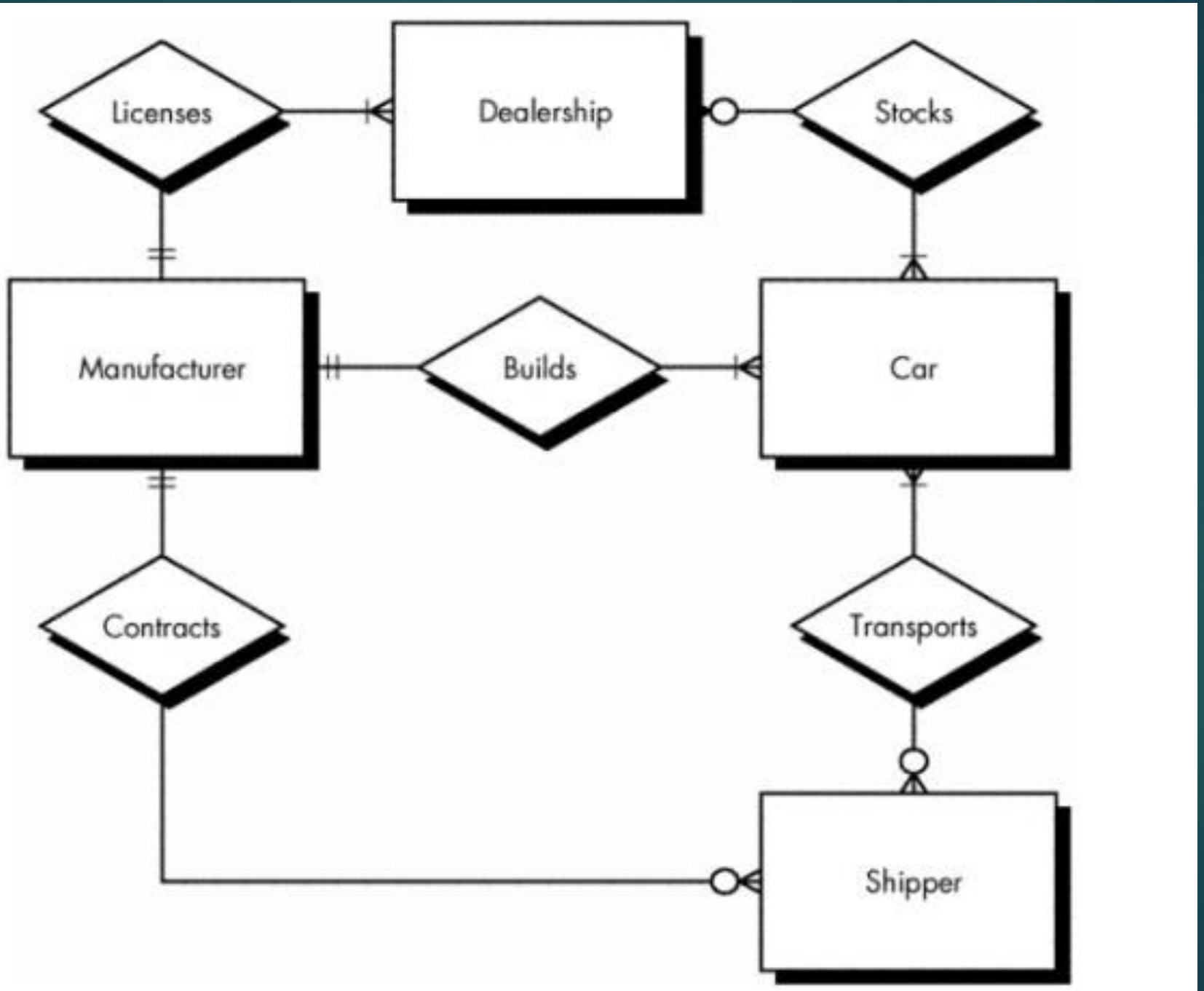
- Cardinality is the specification of the number of occurrences of one object that can be related to the number of occurrences of another object I
- The cardinality is referred to as “one” or “many”, One-to-one (1:1), Oneto-many (1: N), Many-to-many (M: N).
- When one instance of object A relates to one instance of object B, its one to one cardinality.

Modality

- Modality is 1 if an occurrence of the relationship is mandatory.
- Modality is 0 if there is no explicit need for the relationship to occur or the relationship is optional.
- Like :Each faculty member advises many students, each student has only one advisor. iv. Every faculty member may not be advisor, each student must have an advisor


Cardinality:
Implies that a single
customer awaits repair action(s)
Cardinality:
Implies that there may be
many repair action(s)
Customer
is provided with
Repair action
Modality: Mandatory
Implies that in order to
have a repair action(s),
we must have a customer
Modality: Optional
Implies that there may
be a situation in which a
repair action is not necessary


manufacturer
builds
car
Data object table
ID#
Model
Body type
Engine
Transmission
• • •

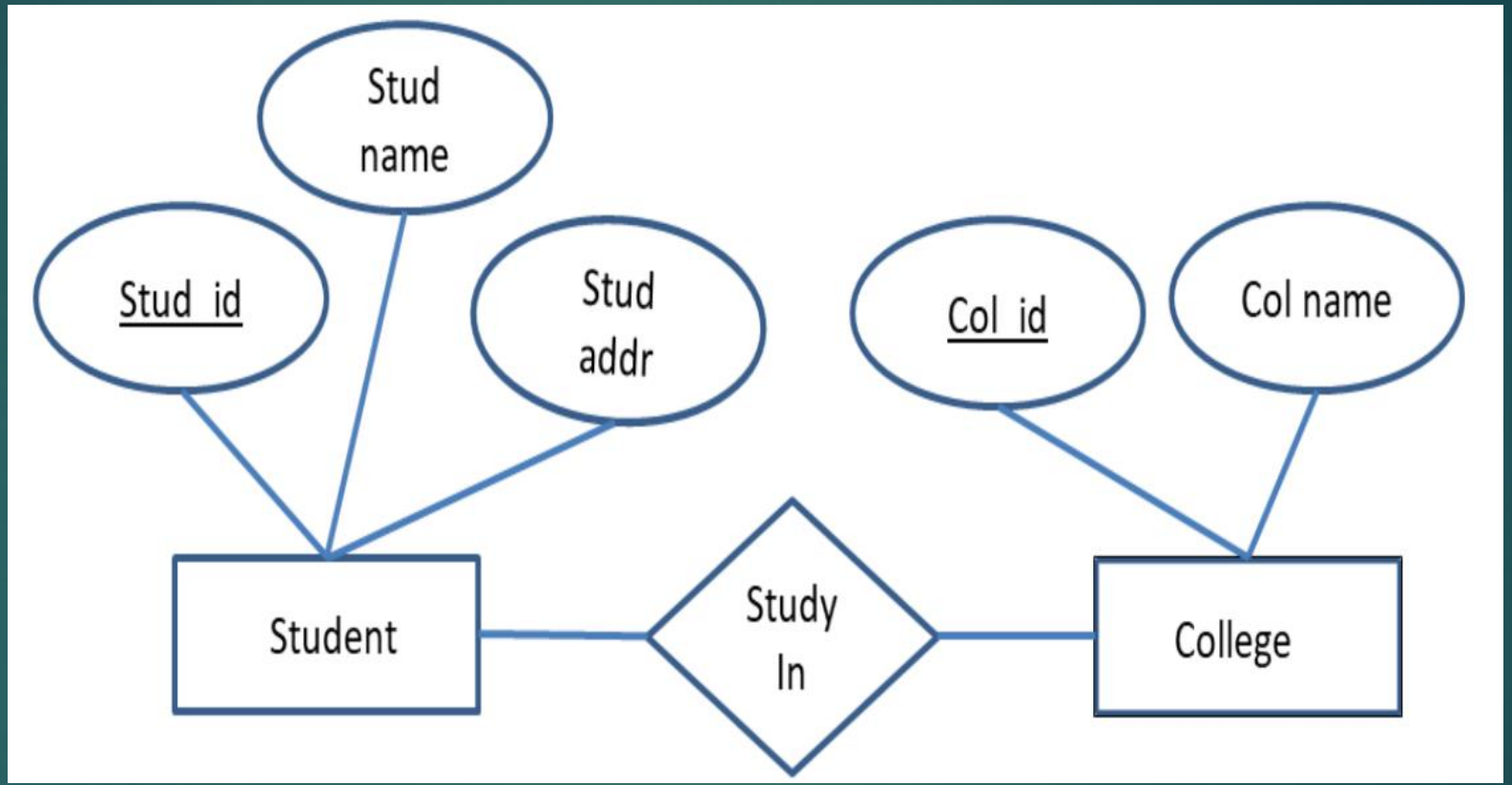

Licenses
Dealership
Stocks
Manufacturer
Builds
Car
Contracts
Transports
Shipper

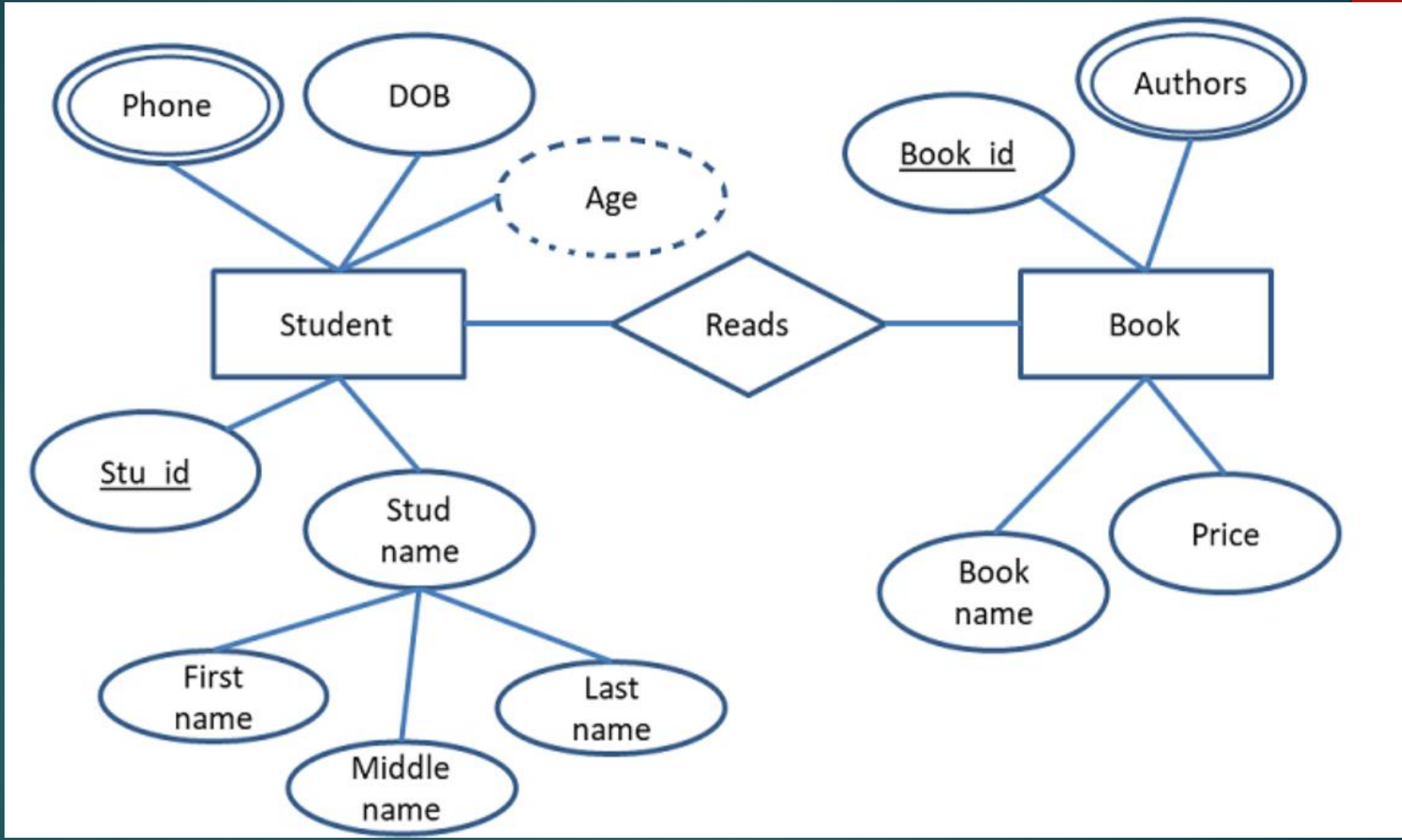
(a) A basic connection between objects

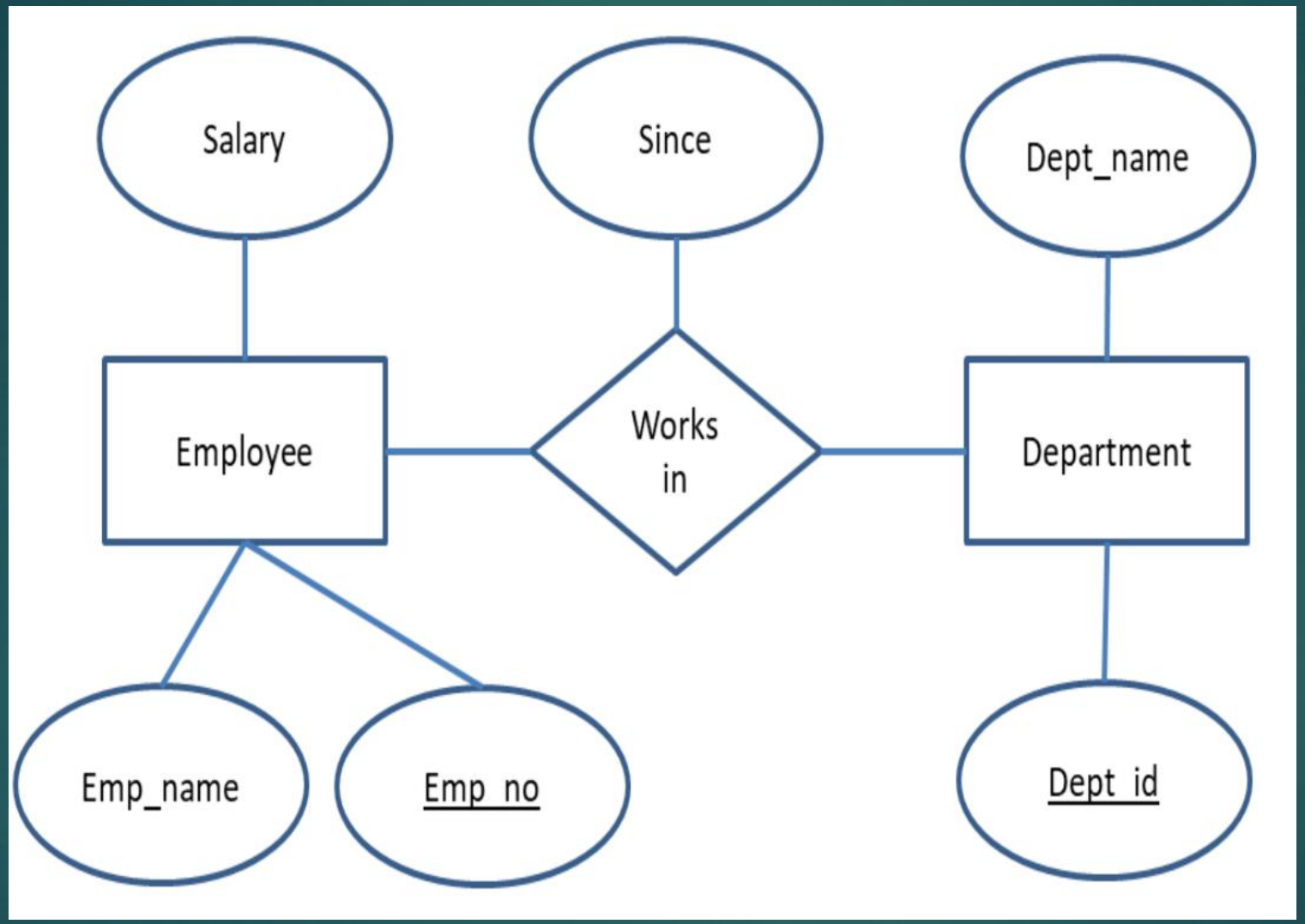
(b) Relationships between objects

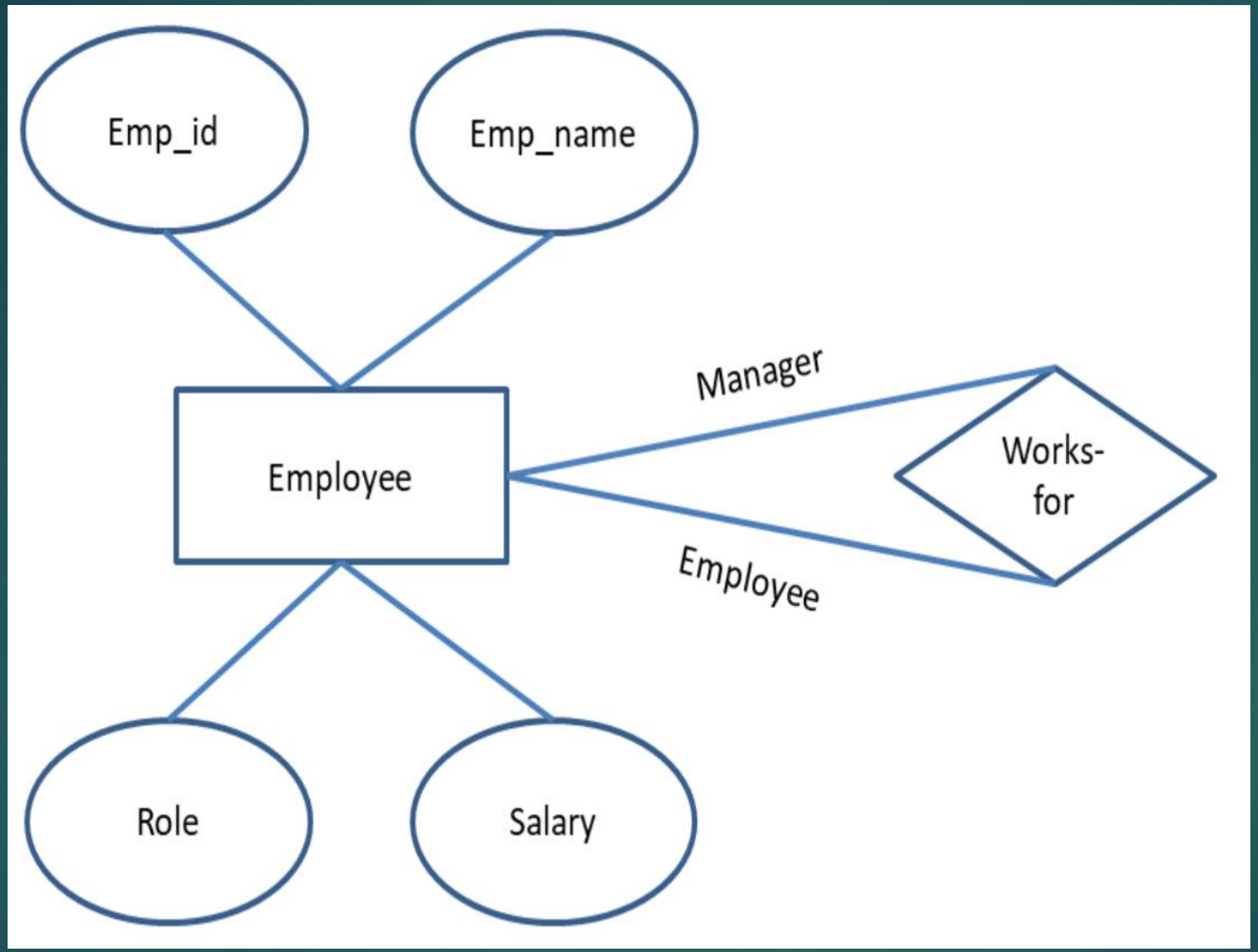
Flow oriented modelling

- This represents how the data objects are transformed as they move through the system. The flow modelling provides the view of the system in the graphical approach


Stud
name
Stud_id
Stud
addr
Col_id
Col name
Student
Study
In
College


Phone
DOB
Age
Authors
Book_id
Student
Reads
Book
Stu_id
Stud name
Price
Book name
First name
Last name
Middle name


Salary
Since
Dept_name
Employee
Works in
Department
Emp_name
Emp_no
Dept_id


Emp_id
Emp_name
Employee
Manager
Works-
for
Employee
Role
Salary

DFD

- The Data Flow Diagram is a graphical technique that depicts information flow and the transforms that are applied as data move from input to output.
- Can be used at any level of abstraction.
- A level 0 DFD, also called a fundamental system model or context diagrams represents the entire software system as a single bubble with input and output data indicated by incoming and outgoing arrows respectively
- A level 1 DFD might contain five or six bubbles with interconnecting arrows; each of the processes represented at level 1 are sub functions of the overall system depicted in the context model.
- Depicts how input is transformed into output as data objects move through a system. vii. Functional modeling and information flow Indicates how data are transformed as they move through the system
- Depicts the functions that transform the data flow.
- Each function description is contained in a process specification