

## **UNIT-3 System Analysis and Design**

### **System Analysis**

- Introduction to Structured System Analysis Development Methodology (SSADM)
  - System survey
  - Structured analysis
  - Structured design
  - Hardware study
  - System implementation
  - maintenance
- Tools for analysis-
  - Decision tree
  - Decision table
  - Structured English
  - Data flow diagram
  - Entity relationship diagram
  - Data dictionary

### **System Design**

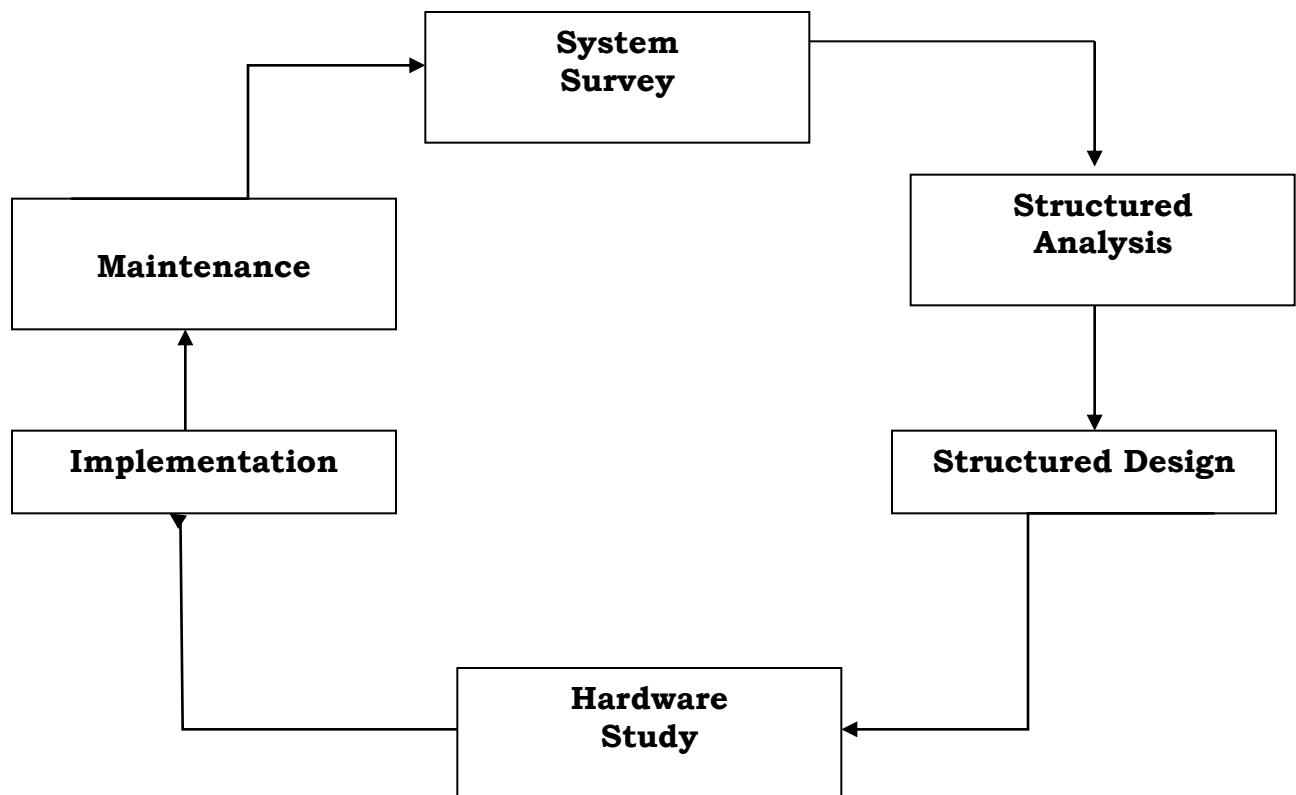
- Systems design process and components
- Designing effective output
- Designing effective input
- Software design alternatives and monitoring

**:Need for Structured Analysis Design(SADDM):****OR****:Limitations/ Disadvantages of SDLC:**

- Convention system developments methods like SDLC having some numbers of the disadvantages or drawbacks or limitations which are as follows:
  1. Interaction with the user is limited. The users interact with the systems analyst at the problem identifications and feasibility study stages. After that they are expected to use the system when it is implemented. Hence, for the user, the new system is a black box. Further in SDLC, there is no proper mode for the user to express his requirements clearly to the analyst at different stages.
  2. The systems analyst is constantly overwhelmed with the business and technical details of the system.
  3. The analytical tools of SDLC like system flowchart, program chart are concerned more with physical aspect of the system rather than the logical ones. It becomes difficult for the user to understand how the system parts fit together.
  4. The specification of SDLC is difficult to understand and modify. If we change the one part of the system the one part of the system, then changes are propagates to all others parts so we have to modify all others parts so we have to modify all others parts of the system.
  5. Software development is bottom-up. So the complete package of the system is viewed after the fully implementation of the product.
  6. All the system documents are prepares at the end of the project which is not correct way.
- In short, SDLC lacks a structured approach to system development. The structure methodology has overcome most of the disadvantages of the conventional method.

**:Meaning of Structured System Analysis and Design  
Methodology(SSADM):****OR****What is SSADM?**

- Structured system analysis and design is a well defined approach in, the form of methodology. It is not new. SSADM is in fact a modified form of SDLC. So it is called as SSADM as SDLC using structured techniques.
- SSADM consists of:
  1. System survey
  2. Structured analysis
  3. Structured design
  4. Hardware study
  5. Implementation
  6. Maintenance



**[Figure of Structured System Analysis & Design Methodology]**

- The structured analysis uses symbols instead of narrative description and creates a graphic model of the system.
- Structured analysis uses tools like:
  1. Data Flow Diagram
  2. Data Dictionary
  3. Structured analysis
  4. Decision Trees
  5. Decision Tables

### **:SSADM Methodology:**

#### **1. System Survey:**

- The first step in SSADM is system survey. This step is similar to feasibility study in SDLC. The sub activities in survey are:
  1. Identify the scope of the current system.
  2. Identify and list the deficiencies in the current system according to user requirements.
  3. Establish new system goals and identify the constraints.
  4. Prepare a document consisting of:
    - a. Goals and objectives
    - b. Customized profile life cycle
    - c. Constraints regarding technical and procedural aspect
    - d. Cost benefit analysis

#### **2. Structured Analysis:**

- This is second stage. It is techniques and graphical tools like data dictionary and data flow diagrams.
  - System analyst develops the new system specification by using this tools and user of the system can easily understand the system.
  - Next, system analyst prepare the model which concentrates on “What to implements” not “How to implements”.
  - Following is the description of sub process of structured analysis’s figure.  
**Sub process 2.1: To study current system:** While studying the current system, the analyst identifies:
    - The external entities
    - The list of processes performed in the current system
    - Sequence of these processes
    - Data used for the processes
    - How the processes are performed etc.
- In short, a physical model of the existing system in the form of DFD evolves.

**Sub process 2.2: To derive logical equivalent DFD:**

- To understand system properly, It is requires a pictorial representation of the system that shows what processes must be performed, the flow of data through the system and the data stores that are required.
- That is logical DFD of the working of the current system is needed.

**Sub process 2.3: Develop logical model of new system:**

- The logical DFD obtained in 2.2 is modified on the basis of the survey conducted and according to users' requirements.
- This new DFD will inform the user which requirements of his will be met.
- The output of this sub process is a new logical DFD of the proposed system.
- The output will also include data elements, files, outputs, inputs etc. i.e a new data dictionary.

**Sub process 2.4: Establish man-machine interface:**

- The output of 2.3 is the logical DFDs and DDs for the proposed system.
- But to bring this conceptual idea to the real life world, we need DFDs relating physical things like people, forms, computers and their relationships.
- This is done by preparing physical DFDs for proposed system in this sub process.
- The output of this sub process is the new physical DFDs which will serve as a input for next sub process.

**Sub process 2.5: Quantity costs and benefits:**

- The output of this sub process will be cost benefit analysis report which will serve as the input for sub process 2.6.
- Various options are identified in terms of costs and benefits.

**Sub process 2.6: Select the best of option:**

- The inputs for this sub process are cost benefit analysis report and physical DFD of proposed system.
- In this sub process, the most important activity of taking the decision of selecting the best option is carried out.
- The outputs of this sub process are:
  - Estimated budget
  - Physical requirements
  - DFDs for the selected option

**Sub process 2.7: Package Specifications:**

- Now that all the conceptual thinking of the analysis and organize them into finished structured specifications.
- This process is called “Packaging”.
- The final result of this is the structured specifications which consist of an interpreted set of DFDs, DDs and process descriptions.
- This is done through the usage of Structured English, Decision Tables and Decision Trees.

**3. Structured Design:**

- Structured Design is a data-flow based methodology. The input for structured design is structured specifications which is the output of structured analysis. It also receives input from the hardware study.
- In short, system design involves transforming a logical design into physical design. This step is much more exacting than designing logical design specifications.
- Here the important activity is “Software Packaging”.
- The software packaging includes:
  - Input-output design
  - Files and database design
  - Program design
  - Control design
- Activities that run parallel to this detailed design steps of software packaging are:
  - Equipment specifications
  - Test specifications
  - User interface specifications
- The main input, structured specifications (i.e. logical design specifications) is used to derive structure charts.
- The most difficult step in SSADM is that of converting DFSs of structured specifications into software packages.
- To do this, we need to construct what is called Structured Chart.
- While a DFD considers a sequential order of processes the structured chart begins with the most important process (BOSS) and then goes on to its subordinate processes.
- The top level of structured chart shows the most important division of work, the lowest level at the bottom shows the details.
- This is essential to divide the total design process into smaller independent modules which in turn help to have flexibility in the design, i.e. any changes made in one particular module will not affect the other modules.
- So, this technique provides a top-down, flexible design which is easier to maintain.
- A structured walk through is an organized step by step tracing through of a design by a group or users.
- The purpose of walk through is to find where improvements can be made in the system or in the development process.
- There are two types of structured walk throughs:

1. A Preliminary Design Walkthrough
  2. A Detailed Design Walkthrough
- Walkthroughs are conducted:
    1. To catch design errors in advance
    2. To improve communication
    3. To fine tune a design
  - The process 3 in figure is exploded into detailed sub process in below figure:

#### **4. Configuring Hardware study:**

- This step considers the physical requirements of the proposed system It is based on the new physical DFDs, DD of step 2.
- Here we should specify the details of the configuration to be used in the implementation stage.
- These configuration details go as input for equipment specification process in Step-3.
- The cost involved and the present worth of the benefits to be filled are considered here for hardware specifications.

#### **5. Constructing the System and Implementation:**

- The implementation process begins after the management has accepted the new system.
- System implementation consists of FIVE components:
  1. System Acquisition
  2. Programming
  3. Testing
  4. Conversion
  5. Documentation

##### **1. System Acquisition:**

- It involves the purchase of hardware, packaged software and software services.
- Here the systems analyst and designer work together to determine the best place to make these outside purchases.
- Another important part of system acquisition is the actual purchase of goods and services.

##### **2. Programming:**

- It is the writing of instructions to be read and executed by a computer.
- Programming is performed by computer programmer or programmer/analyst rather than by systems analyst or designers.
- Tasks in programming include writing the coded instructions, testing each segment of the code and testing the entire computer program once it is completed.

##### **3. Testing:**

- It consists of putting together the various coded pieces of a design, testing them and correcting the parts of the code or the design that are not correct.

- At this stage some errors are introduced purposely to test whether they will be spotted by the program.

#### 4. Conversion:

- Once the system has been tested successfully then the part which remains is that of putting them into the operation.
- The conversion team must manage the smooth changeover from the old system to the new system.
- This requires:
  - i. Training Personnel
  - ii. Modifying parts of the old system
  - iii. Running parallel system or dual system until everything goes as planned.

#### 5. Documentation:

- Documentation means putting it in the written form about how a system is designed or functions.
- The documentation includes:
  - 1. Design Documentation:** It describes the overall system design and includes system flowcharts, all input/output formats, file description, control requirements and report specifications.
  - 2. Program Documentation:** It consists of programming specifications like program logic, graphic aids, input-output formats etc.
  - 3. Training Documentation:** It includes user training manuals and materials to be used in the conversion and the installation of new system.
  - 4. Operations Documentation:** It contains instructions for normal operations as well as directions for handling problems and breakdowns.
  - 5. User reference Documentation:** It carries on after training is over and the system is installed. It should provide quick, clear answers like a dictionary.

#### 6. Maintenance:

- This is the last step in the system life cycle. However it takes the longest duration.
- **Maintenance may be corrective, adaptive or perfective.**
- In corrective maintenance errors or bugs are rectified.
- In adaptive maintenance the user requirements if any are still considered and the necessary changes are made.
- In perfective maintenance efforts will be constantly going on to perfect the system in terms of response time and resource requirements.

### :Advantages of SSADM:



Some of the important advantages of SSADM are:

**1. Good Documentation:**

In the structured methodology well defined documentation takes place. So, it is easy for the analysts, users and programmers to understand and use.

**2. Better Communication:**

Since structured methodology is graphic it provides easy to understand presentation of the application. The DFD, for example, presents a better picture than any other comparative tool.

**3. Standardization:**

Before the emergence of the structured methods, the system analyst used to have their own methods of designing computerized system. But structured methodology offers very little scope for individual approach.

**4. Modularization:**

The process is partitioned so that we have clear picture of the smaller modules which is essential to understand the system thoroughly.

**5. Logical Design:**

The SSADM is more logical than physical. The elements of the system do not depend on vendor or hardware.

**6. User Oriented:**

The SSADM consults user at every stage of development thereby leaving no scope for rejection after the system is implemented.

**7. Maintainability:**

The need for maintenance arises due to errors, modified user requirements and enhancements. The structured methodology takes into account this aspect, hence maintenance becomes cheaper.

## :Decision Table:

- **Definition:** Data flow diagram is a graphical aid for defining systems inputs, processes and outputs. It represents flow of data through the system.

Parul Captive Portal X decision table in system X https://iptnow.wikispaces X W Decision table - Wikipedia X IT532 - SYSTEMS ANALYSIS X

www4.desales.edu/~d1m1/it532/class04/mathdes.html

To: [\(Type\)](#) [\(Purpose\)](#) [\(Components\)](#) [\(Related\)](#) [\(Example\)](#) [\(Comments\)](#)

**Model Type** [\(Top\)](#)

Mathematical.

**Purpose** [\(Top\)](#)

A decision table organizes and presents complex logic that selects one or more actions that should be performed given a set of conditions and their values.

**Components and Structure** [\(Top\)](#)

The general structure of a decision table is illustrated by the following template:

Table Name	RULE			
	1	2	:::	N
Condition 1	A	B	:::	C
Condition 2	Y	Y	:::	N
:::	:::	:::	:::	:::
Condition N	Y	N	:::	:::
Action 1	X	X	:::	
Action 2		X	:::	
:::			:::	X
Action N	X		:::	

← Condition Values

← X for each action

**Related Models** [\(Top\)](#)



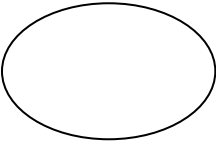
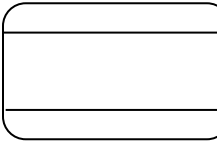


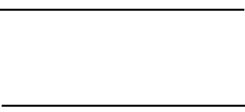
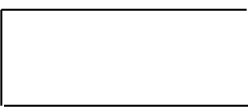
- A decision table can be added to the [rule dictionary](#) to express a complex business or transform rule.

mod6.PPT Show all downloads...

2:04 PM 3/2/2016

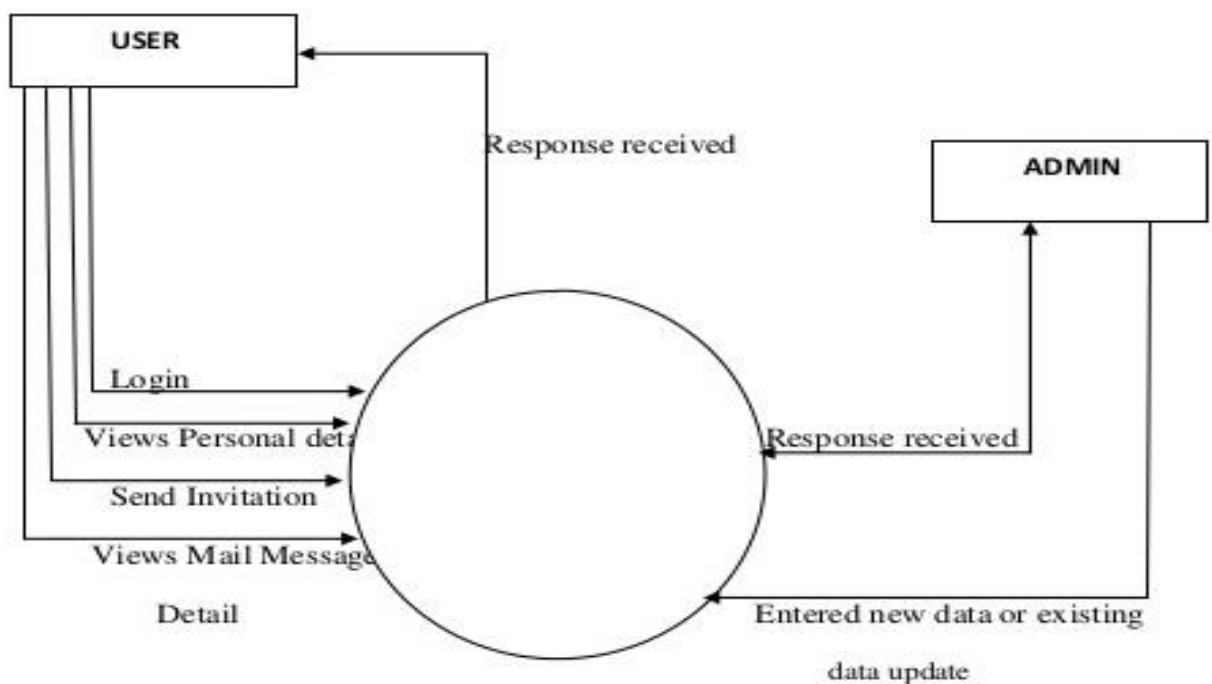
### :Data Flow Diagrams:

- **Definition:** Data flow diagram is a graphical aid for defining systems inputs, processes and outputs. It represents flow of data through the system.
- DFDs serve two purposes:
  1. Provide a graphic tool which can be used by the analyst to explain his understanding of the system to the user
  2. They can be readily converted into a structured chart which can be used in design.
- **Symbols used in DFDs:** DFD use four types of symbols which represent system components such as:
  1. External entity
  2. Process
  3. Data flow
  4. Data store
- The use of specific items associated with each element depends on whether Yordon or Gane and Sarson approach is used.

Sr No.	Symbol Name	Symbol	
		Yordon	Gane & Sarson
1.	<b>External entity:</b> a source or destination of data which is external to the system. E.g. supplier, customer etc.		
2.	<b>Process:</b> Here flow of data is transformed. E.g. verify credits, update inventory file.		
3.	<b>Data flow:</b> it is a packet of data. It may be in the form of a document, letter, telephone call etc.		
4.	<b>Data Store:</b> any stored data but with no reference to the physical method of storing. E.g. inventory master file, customer master file etc.		

- **Rules for Constructing a DFD:** The following rules may be used while constructing DFDs:

1. Processes should be named and numbered for easy reference.
2. The direction of flow is from top to bottom and from left to right. Data traditionally flow from the source to the destination although they may flow back to the source.
3. When a process is exploded into lower level details, they are numbered.
4. The names of data stores, sources and destinations are written in capital letters. Process and data flow names have the first letter of each word capitalized.



**[ZERO LEVEL DAGRAM FOR SOCIAL NETWORKING SITES]**

- **Difference between Physical and Logical DFD:**

Physical DFD		Logical DFD	
1.	Data flow names include the implementation facts as names, numbers, media, timing etc.	1.	Data flow names describe the data they contain. They do not refer to the form or document on which they reside.
2.	Process names include the name of the processor i.e. person, department, computer system etc.	2.	Process names describe the work done without referring to e.g. Account receivable, order processing etc.
3.	Data stores identify their computer and manual implementation.	3.	Physical location of data stores is irrelevant. Many times, the same data store may be shared by many subsystems and processes.
4.	This is more realistic and implementation oriented. The PDFD are more detailed in nature.	4.	This is more logical in format. This is more abstract than PDFD and less dependent on implementation steps.

### **:Entity Relationship Modeling:**

#### **Introduction:**

- The Entity-Relationship data model grew out of the exercise of using commercially available DBMS to the model application databases. Earlier commercial systems were based on the hierarchical and network approach.
- The entity-relationship model is a generalization of these models. It allows the representation of explicit constraints as well as relationships.
- Even though the E-R model has some means of describing the physical database model, it is basically useful in the design and communication of the logical database model.
- In this model, objects of similar structures are collected into an entity set. The relationship between entity sets is represented by a named E-R relationship and is 1:1, 1:M or M:N, mapping from one entity set to another.
- The database structure, employing the E-R model is usually shown pictorially using Entity-Relationship (E-R) diagrams.

#### **Entity:**

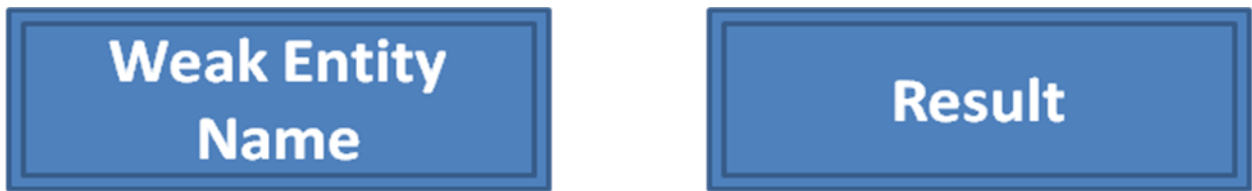
- Entity is an object of concern to an organization for which it maintains the data entities have concrete existence.
- e.g specific person, company, event, plant building, room, employee, customer, pen, computer etc.
- There are two types of entities:
  - 1. Strong Entity:** The Entity which has primary key is called Strong Entity. Example: The entity set EMPLOYEE would qualify as a strong entity because it has an attribute Employee\_Id that uniquely identifies an instance of the entity EMPLOYEE, no two instances of the entity have the same value for the attribute Employee\_Id.

**Strong Entity**  
**Name**

**Student**

#### **Example of Strong Entity**

- 2. Weak Entity:** The entity which has no primary key is called Weak Entity. Entities may not be distinguished by their attributes but their relationship to another entity. We then establish a relationship, DEDUCTIONS, between the modified entity EMPLOYEE\* and DEPENDENTS as shown in figure.



### Example of Weak Entity

In this case, the instances of the entity from the set DEPENDENTS are distinguishable only by their relationship with an instance of an entity from the entity set EMPLOYEE. The relationship set DEDUCTIONS is an example of an identifying relationship and the entity set DEPENDENTS is an example of a weak entity.

**Entity Set:** An Entity Set is group of similar objects of concern to an organization for which it maintains data.

For Example: set of all persons, companies, trees, holidays

### :Attributes:

- The properties that characterize an entity set are called its attributes. Attribute is also called data item, data element, data field, item, elementary item or object property.
- For Example, Employee entity has attributes like id, name, address, phone no
- Types of Attributes:
  1. Single valued attributes
  2. Composite attributes
  3. Derived attributes
  4. Multi valued attributes
  5. Attributes with null values

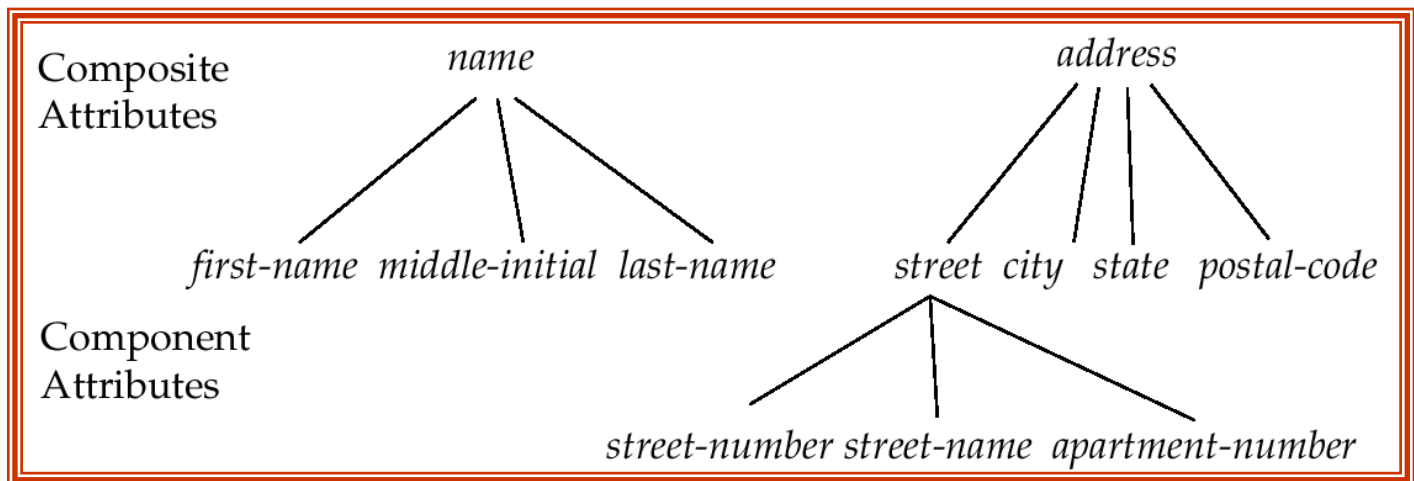
#### 1. Single values attributes: Are not divided into subparts

- Course name
- Capacity of this classroom
- Price of sugar at a shop
- Distance from gandhinagar to ahmedabad

#### 2. Composed attributes: Can be divided into subparts

- Donor-address
  - Street
    - House number
    - Street name
  - City
  - State
  - PIN (postal index number)
  - Country

- (Future) planet and solar system



**3. Derived attributes:** An attribute whose value is derived from other (independently defined) attribute(s) (can be computed from other attributes)

- Age of a person
  - Correct only on the date of recording
  - Better to store date of birth
- Total salary of an employee
  - Sum of basic pay, allowances, (-) deductions
  - Better to store separate attribute values

**4. Multi-valued attributes:** A multi-valued attribute is one that may take on more than one value.

- Number of courses taken by a student
- Number of teachers teaching a course
- Number of hobbies of a student

**5. Attributes with null values:** Value of an attribute is null when

- Not known
  - Such as % or Average grade during first semester of studies

#### **:Relationship:**

- An association among entities is called relationship.
- A collection of relationships of the same type is called a relationship set.
- Types of
- The relationship set is used in data modeling to represent an association between entity sets & itself.
- For Example: A grade is an attribute of the enrollment relationship set between the entity sets cause and students.

**Degree of Relationship set:** Is the number of entity types that participate in it.

- The three most common relationship degrees are unary (degree 1), binary (degree 2) and ternary (degree 3)
- Higher degree relationships are possible but rarely encountered in practice



**1. Unary Relationship:** Is between the instances of a single entity type (also called recursive relationships)

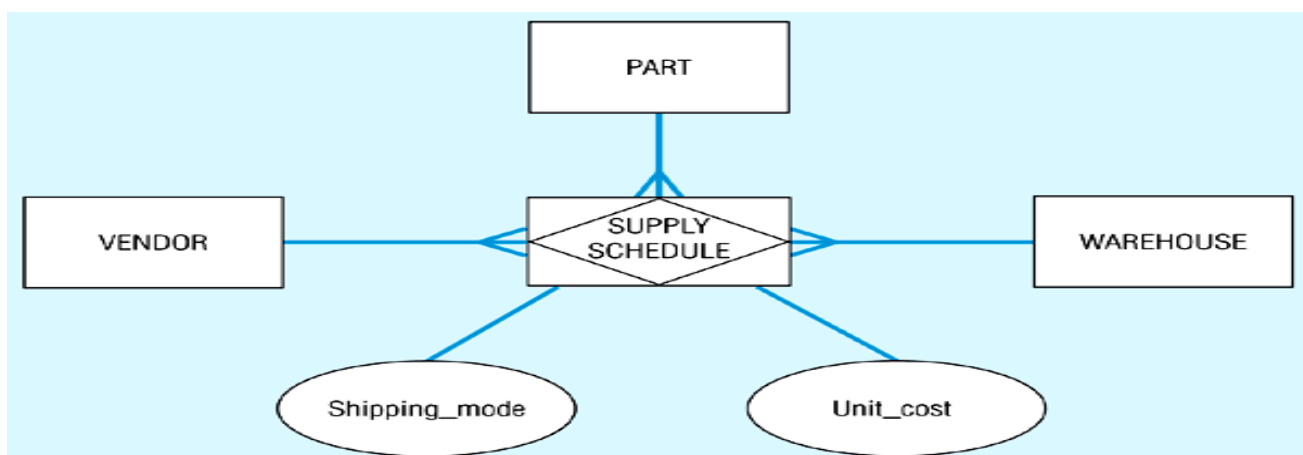
- 'Is\_Married\_To' is a one-to-one relationship between instances of the PERSON entity type
- 'Manages' is a one-to-many relationship between instances of the EMPLOYEE entity type

**2. Binary Relationship:** Between the instances of two entity types, and is the most common type of relationship encountered in data modelling. e.g. (one-to-one) an EMPLOYEE is assigned one PARKING\_PLACE, and each PARKING\_PLACE is assigned to one EMPLOYEE

- e.g. (one to many) a PRODUCT\_LINE may contain many PRODUCTS, and each PRODUCT belongs to only one PRODUCT\_LINE
- e.g. (many-to-many) a STUDENT may register for more than one COURSE, and each COURSE may have many STUDENTS

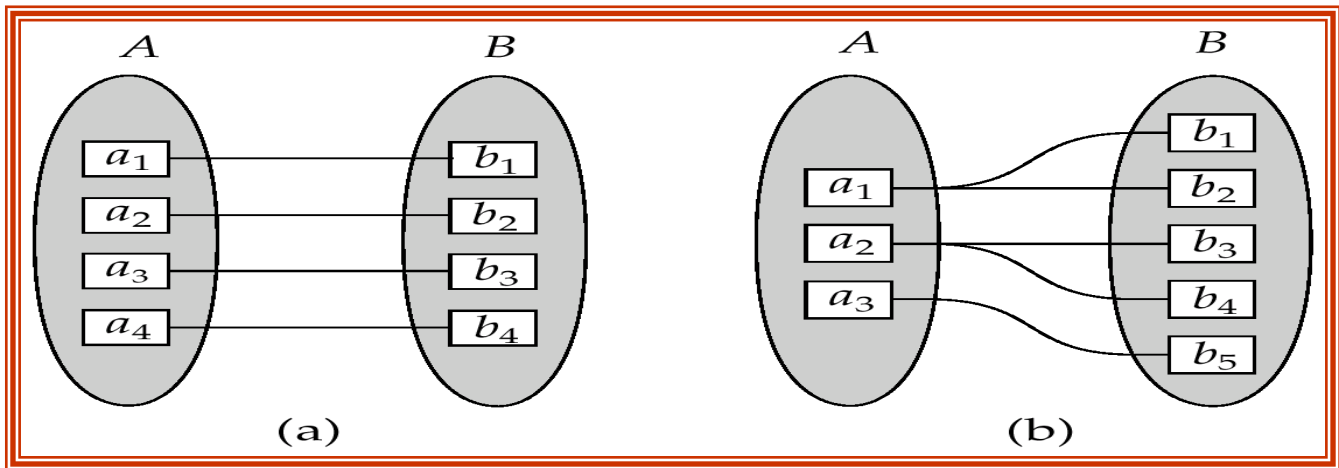
**3. Ternary Relationship:** A ternary relationship is a simultaneous relationship among the instances of 3 entity types.

- It is the most common relationship encountered in data modelling
- The following Fig. shows a typical ternary relationship
- Here, vendors can supply various parts to warehouses
- The relationship 'Supplies' is used to record the specific PARTs supplied by a given VENDOR to a particular WAREHOUSE
- There are two attributes on the relationship 'Supplies', Shipping\_Mode and Unit\_Cost
- e.g. one instance of 'Supplies' might record that VENDOR X can ship PART C to WAREHOUSE Y, that the Shipping\_Mode is 'next\_day\_air' and the Unit\_Cost is 5-00 per unit



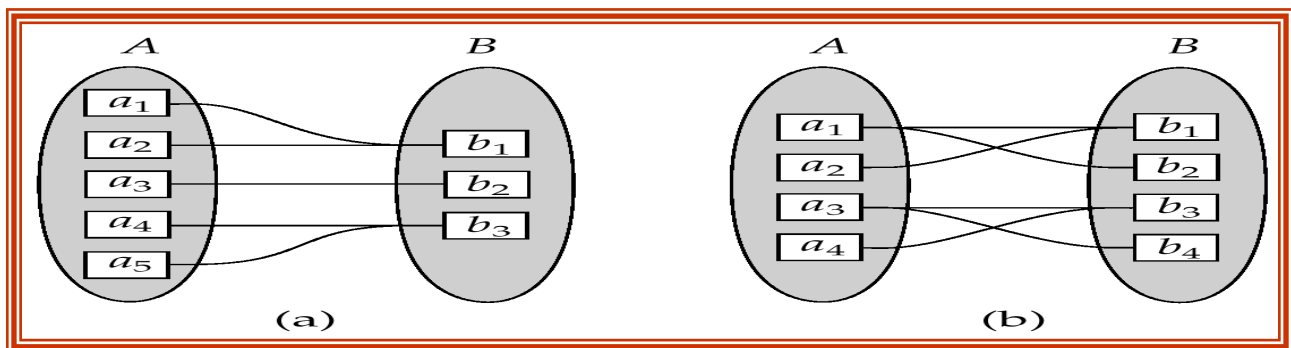
**Relationship Types / Mapping Cardinality:**

- Express the number of entities to which another entity can be associated via a relationship set.
- For a binary relationship set the mapping cardinality must be one of the following types:
  1. One to one
  2. One to many
  3. Many to one
  4. Many to many



(a) One to One

(b) One to Many

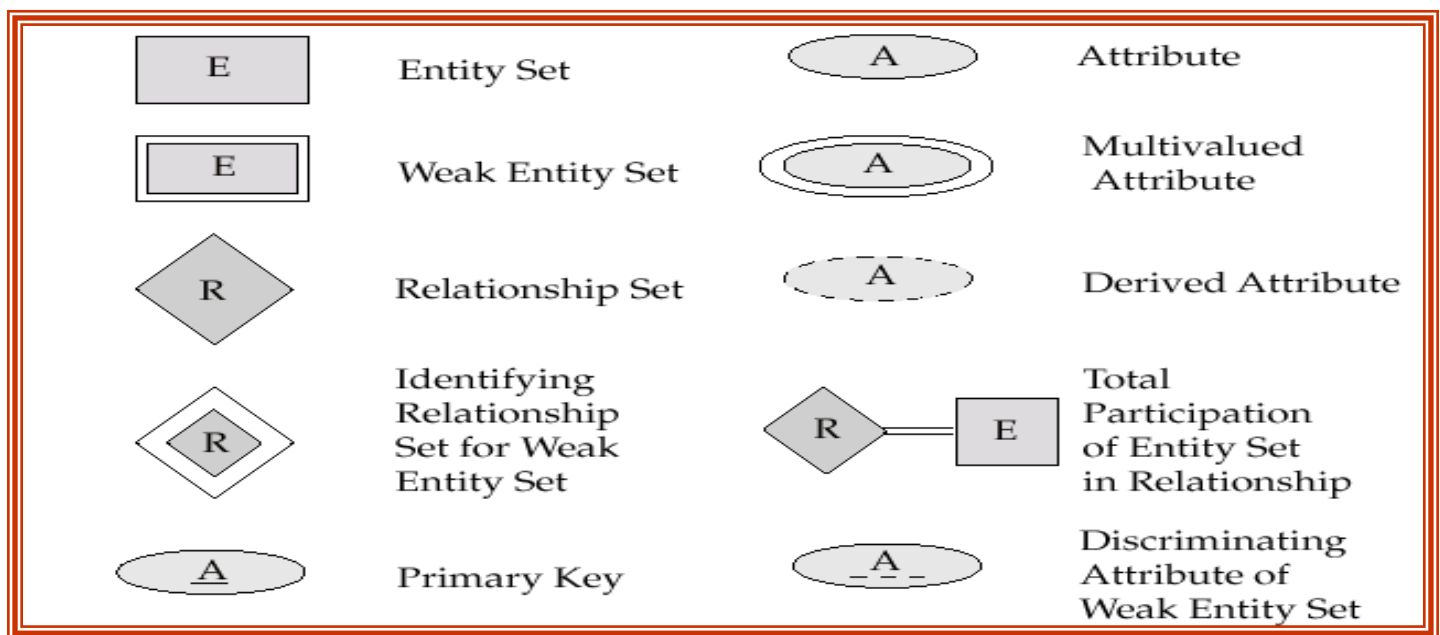


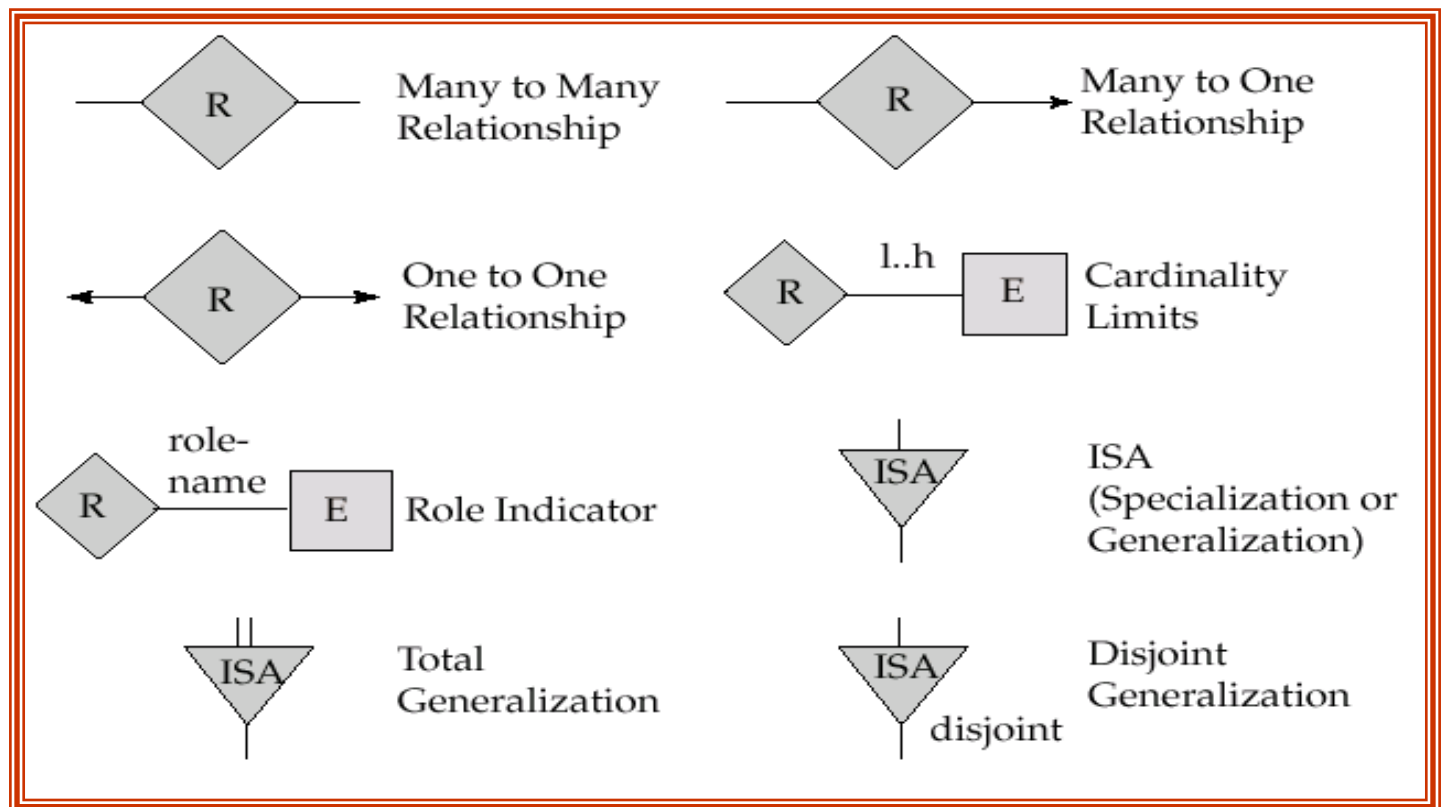
(c) Many to one

(d) Many to many

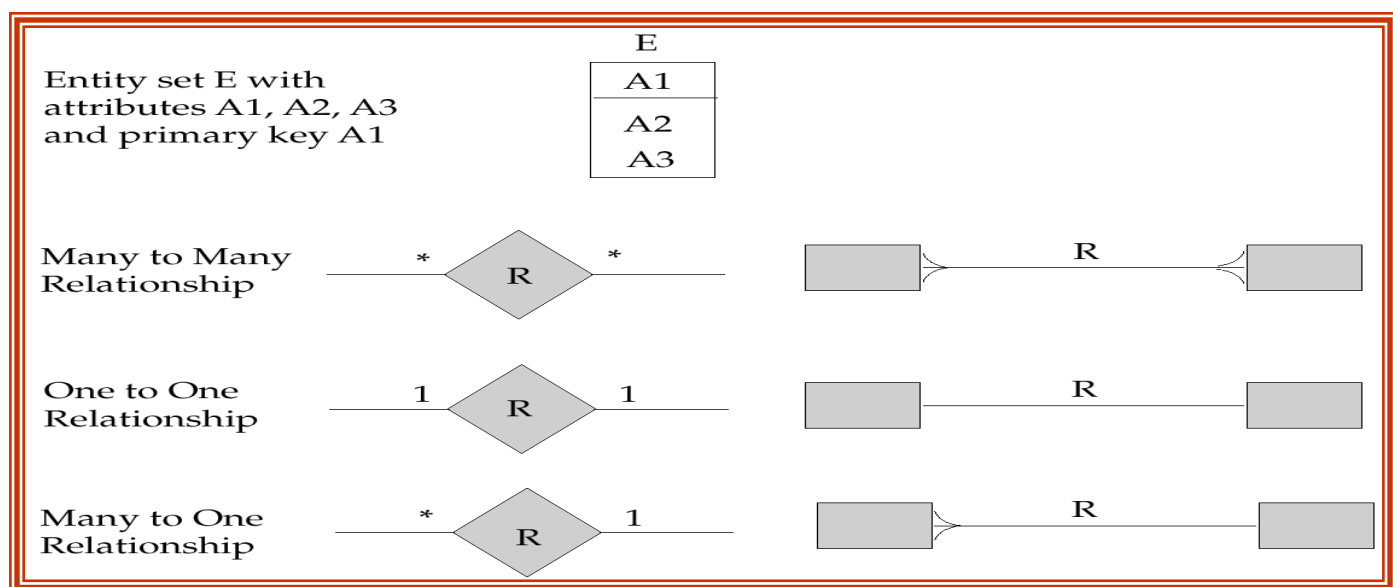
**:Entity-Relationship Diagram:**

- E-R diagram or Entity-Relationship diagrams basically represent the overall logical structure of a database. The data models of a database represent a collection of conceptual tools for describing data, data relationship, data semantics and consistency constraints.
- The various data models fall into three categories:
  1. Object-based logical model
  2. Record-based logical model
  3. Physical data model
- Entity-relationship model is a kind of object-based logical model. This model is based on a perception of a real world, which consists of a collection of basic objects entities and relationship among these objects.
- In addition, the E-R model represents certain constraints to which the contents of a database must confirm. The overall logical structure of a database can be expressed graphically by an E-R diagram, which consists of the following components:
  - Rectangles represent entity sets.
  - Diamonds represent relationship sets.
  - Lines link attributes to entity sets and entity sets to relationship sets.
  - Ellipses represent attributes
  - Double ellipses represent multi-valued attributes.
  - Dashed ellipses denote derived attributes.
  - Underline indicates primary key attributes

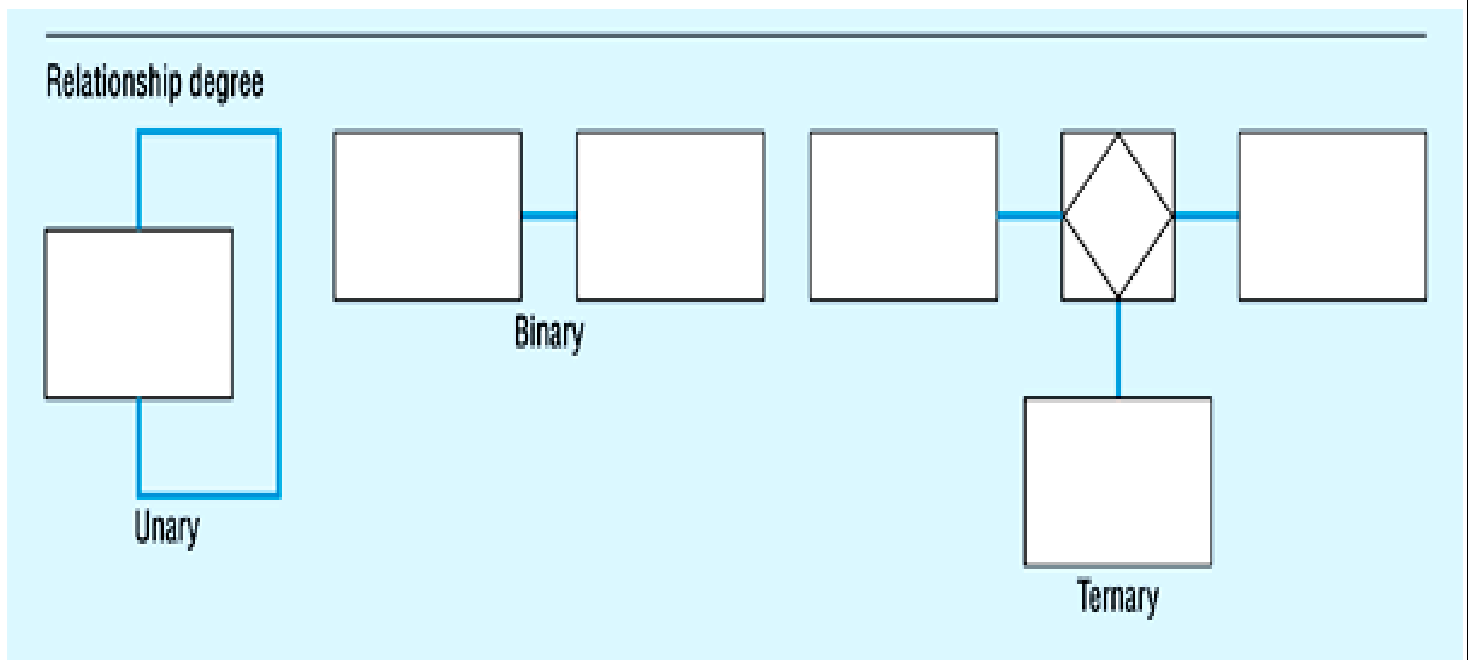
**Summary of Symbols Used in E-R Notation:**



### Alternative E-R Notations:



### Relationship Degree:

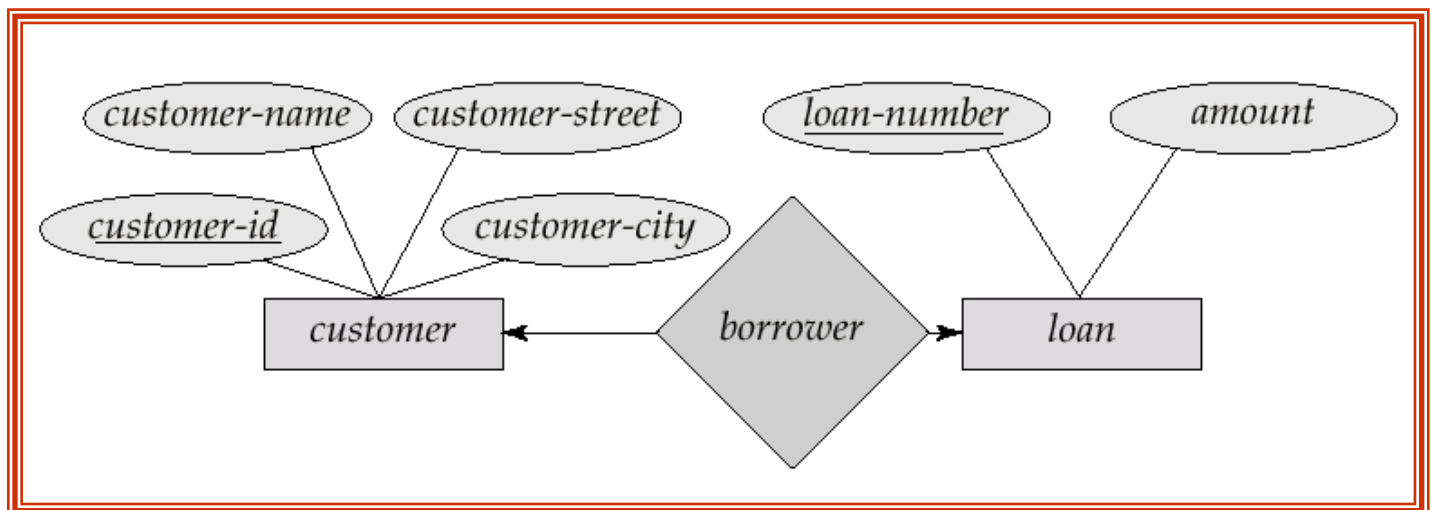


### **Relationship Types:**

We express cardinality constraints by drawing either a directed line ( $()$ ), signifying “one,” or an undirected line ( $—$ ), signifying “many,” between the relationship set and the entity set.

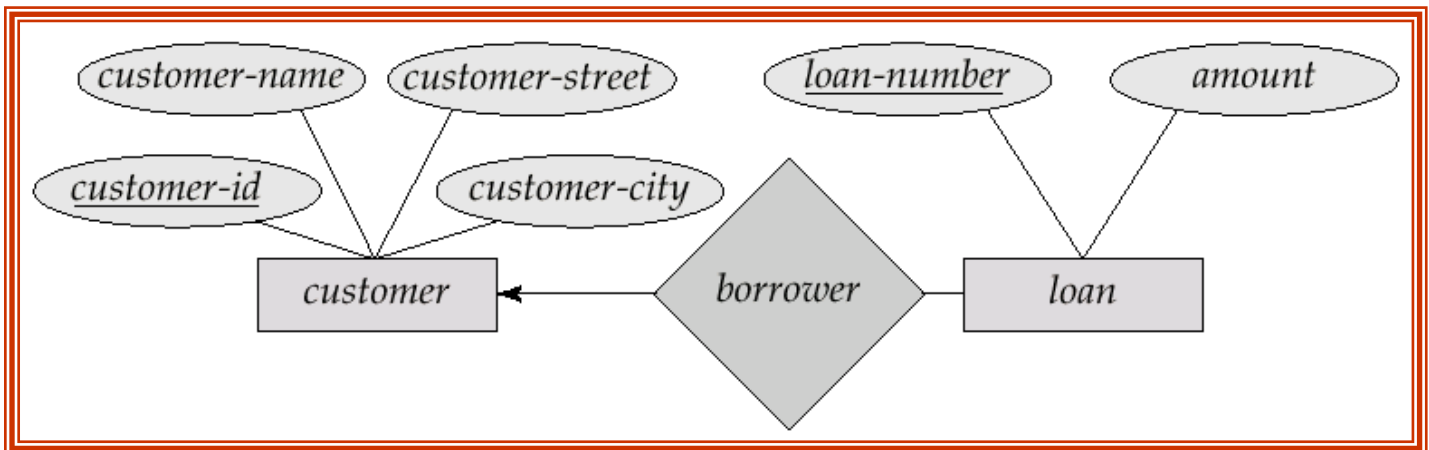
#### **One-to-one relationship:**

- A customer is associated with at most one loan via the relationship borrower
- A loan is associated with at most one customer via borrower

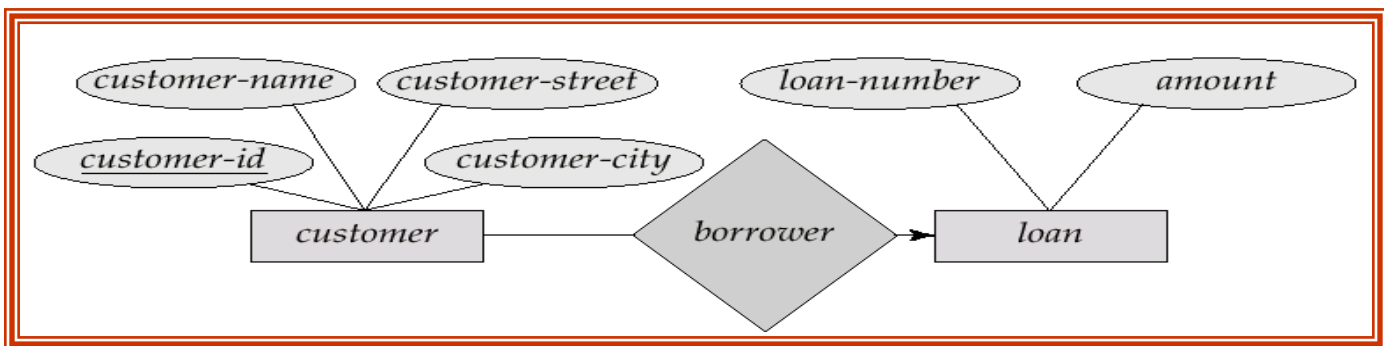


#### **One-to-many relationship:**

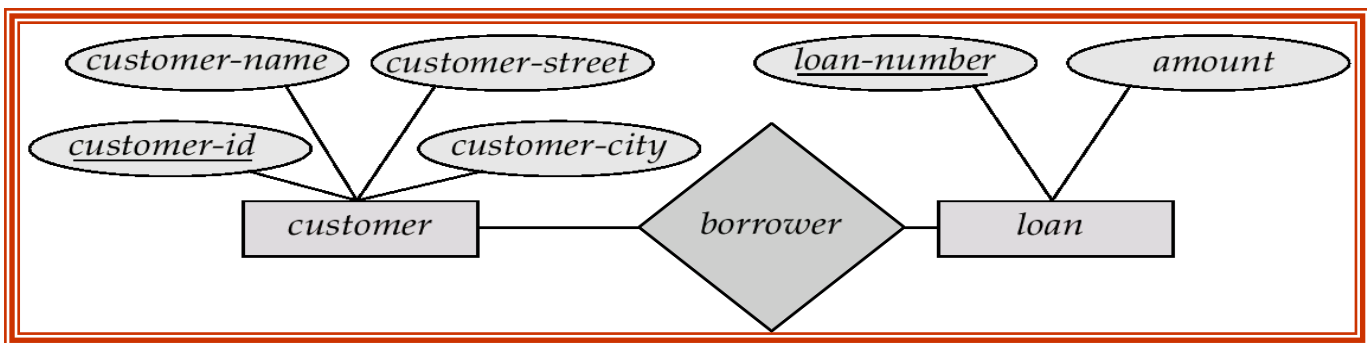
In the one-to-many relationship a loan is associated with at most one customer via borrower, a customer is associated with several (including 0) loans via borrower

**Many-to-one relationship:**

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

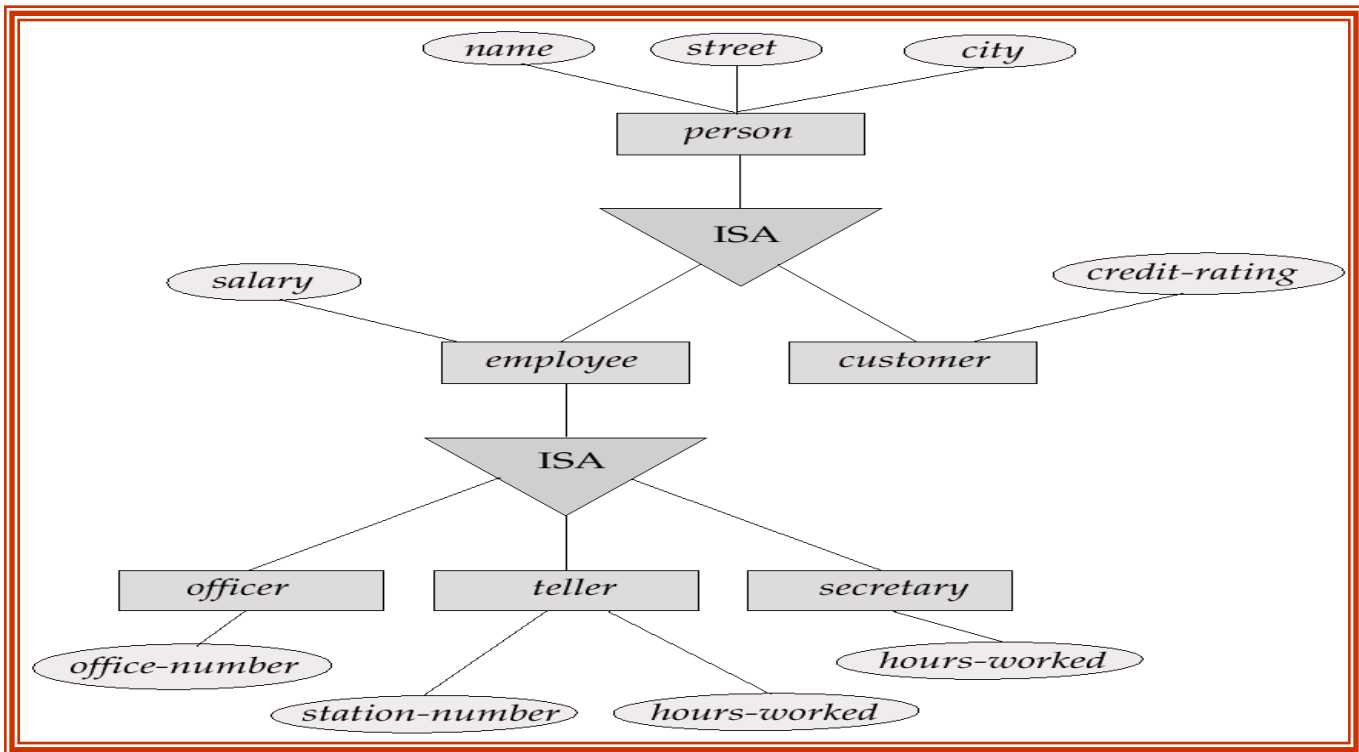
**Many-to-many relationship:**

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

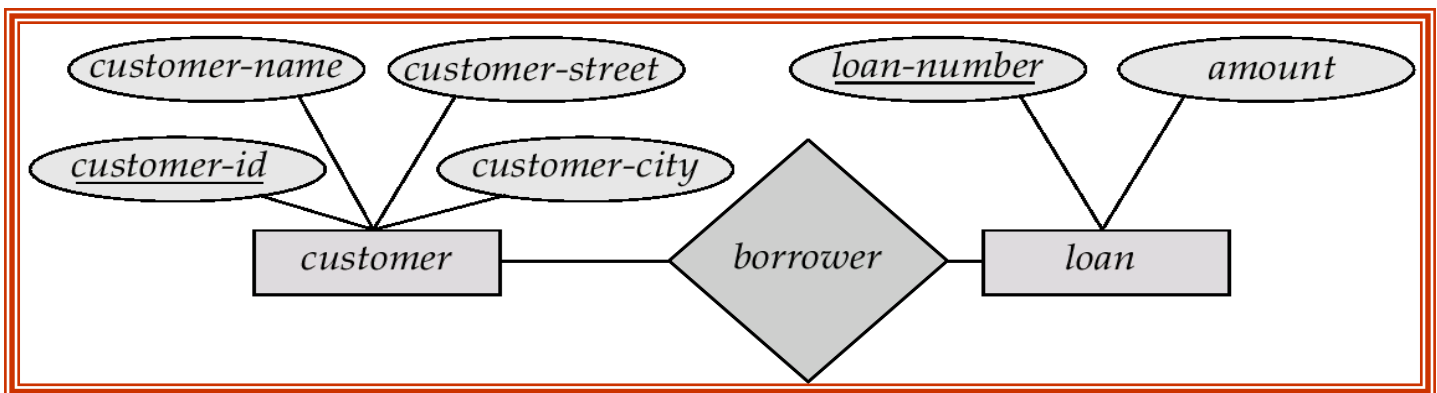
**Specialization:**

- Top-down design process; we designate subgroupings 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

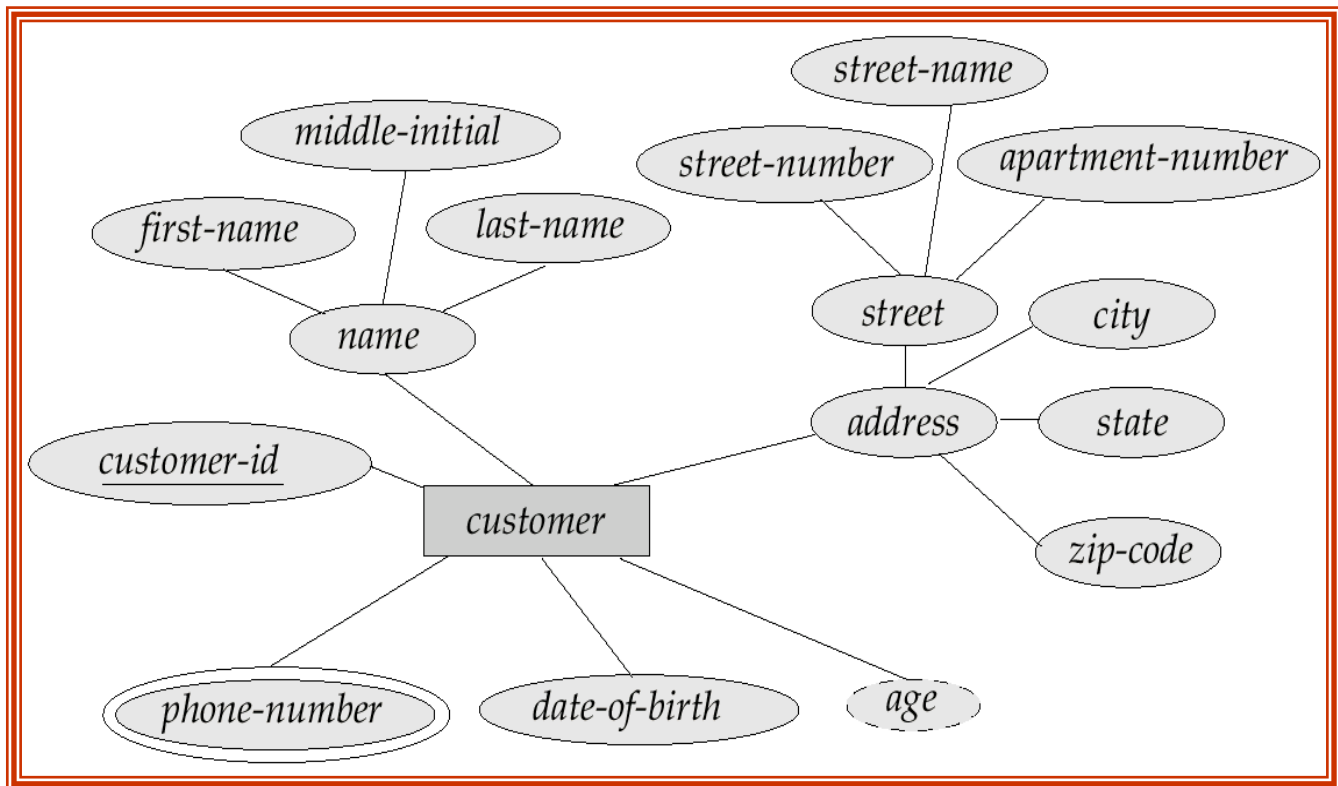
- participate in relationships that do not apply to the higher-level entity set.
- Depicted by a triangle component labeled ISA (E.g. customer “is a” person).
- Attribute inheritance – a lower-level entity set inherits all the attributes and relationship participation of the higher-level entity set to which it is linked.



**Example1: E-R Diagram of customer get loan from the bank.**



**Example2: E-R Diagram of customer with composite, multi-valued and derived attributes.**

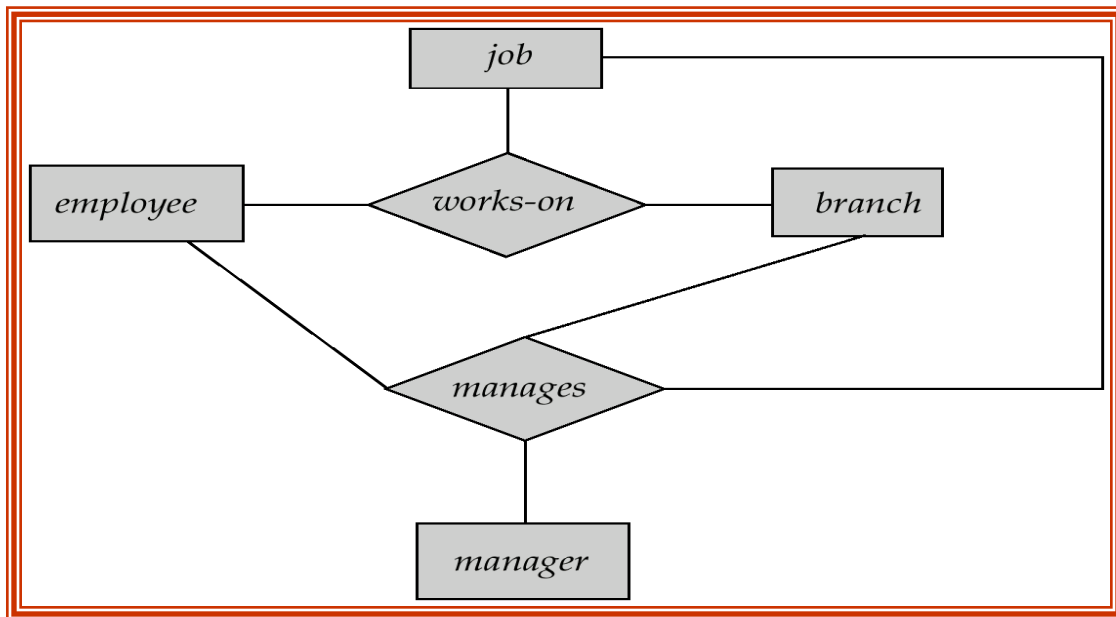
**Generalization:**

- 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.

**Aggregation:**

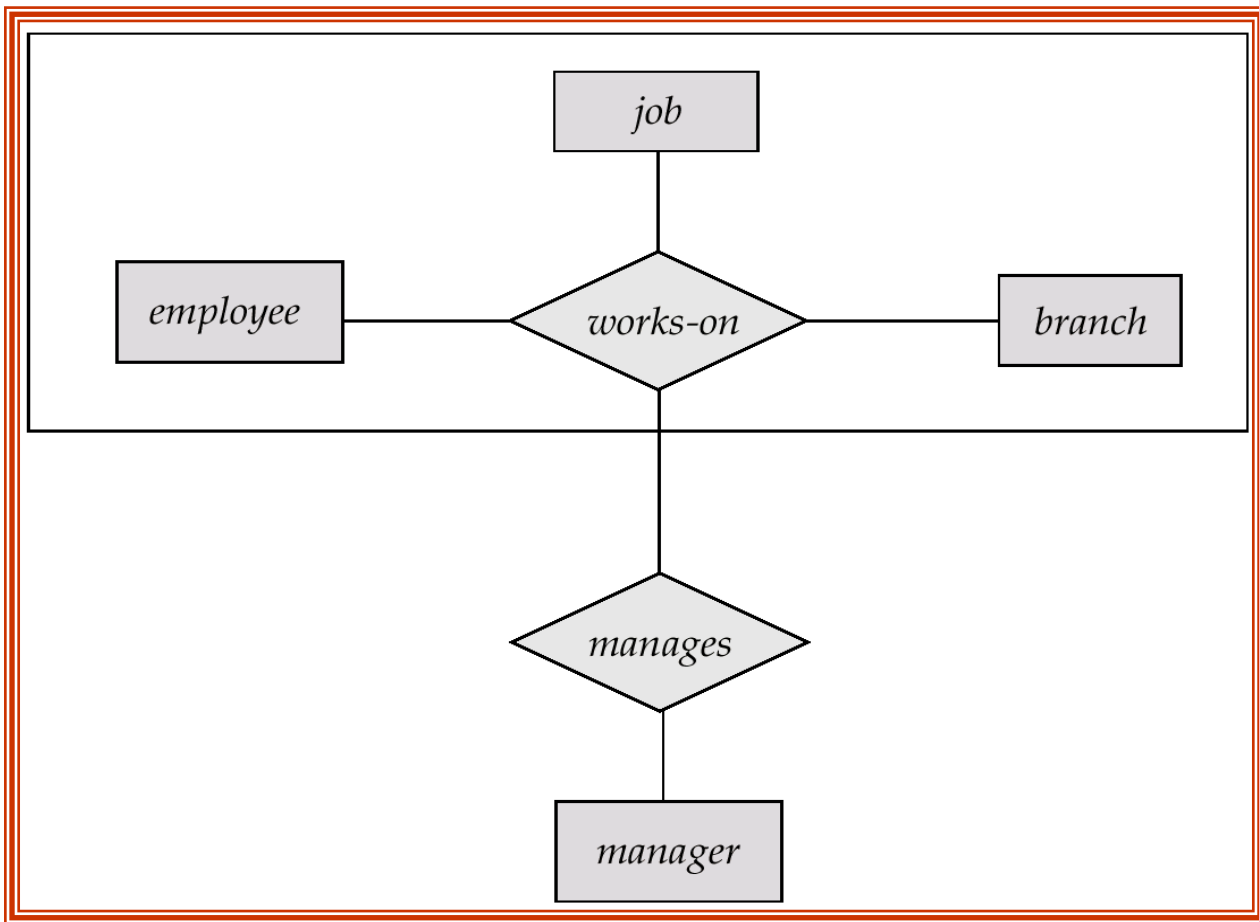
- Consider the ternary relationship works-on, which we saw earlier
- Suppose we want to record managers for tasks performed by an employee at a branch



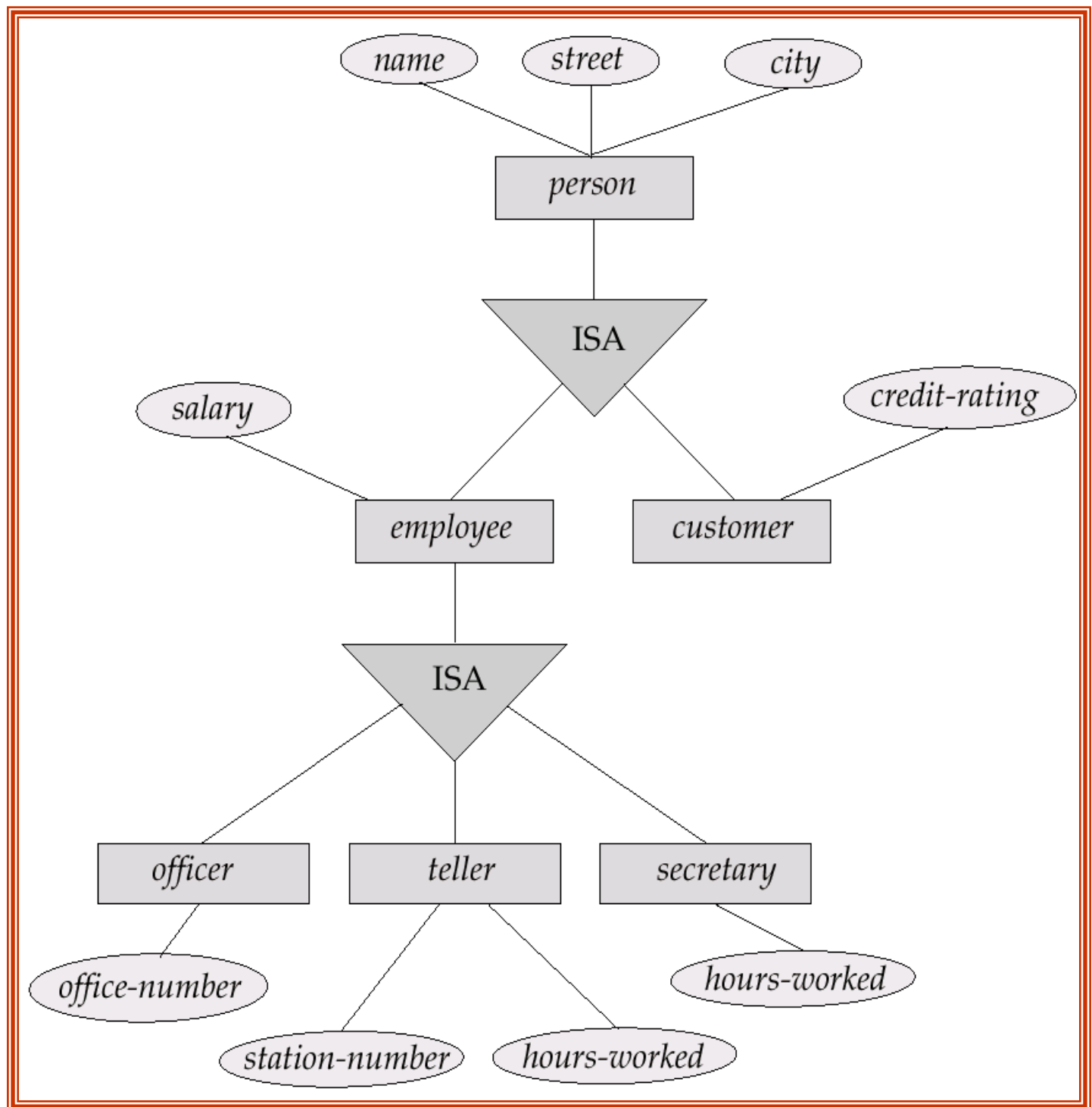


- Relationship sets *works-on* and *manages* represent overlapping information
  - Every *manages* relationship corresponds to a *works-on* relationship
  - However, some *works-on* relationships may not correspond to any *manages* relationships
  - So we can't discard the *works-on* relationship
- Eliminate this redundancy via *aggregation*
  - Treat relationship as an abstract entity
  - Allows relationships between relationships
  - Abstraction of relationship into new entity
- Without introducing redundancy, the following diagram represents:
  - An employee works on a particular job at a particular branch
  - An employee, branch, job combination may have an associated manager

#### Example of E-R Diagram with Aggregation:

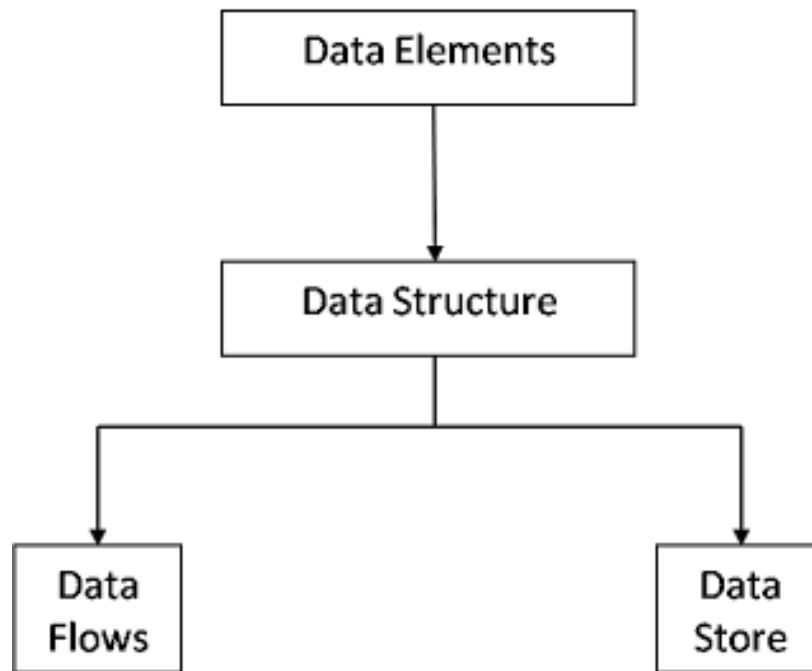


**E-R Diagram for Banking Example:**



**:Data Dictionary:**

- **A data dictionary is a collection of data about data.**
- It maintains information about the definition, structure, and use of each data element that an organization uses.
- There are many attributes that may be stored about a data element.
- “Data dictionary contains description & definition consulting the data structure, data elements, their interrelationship & other characteristics of a system.”
- **Objectives of Data Dictionary**
  - A standard definition of all terms in a system i.e. each data item is uniquely identified and defined.
  - ii. Easy cross referencing between subsystem’s program and modules.
  - iii. Simple program maintenance.
  - iv. It contains information about the data of the system and there is an entry in the data dictionary for every element of DFD. Thus DFD and Data Dictionary are compliment of each other.
- **Objectives of Data Dictionary**
  1. A standard definition of all terms in a system i.e. each data item is uniquely identified and defined.
  2. Easy cross referencing between subsystem’s program and modules.
  3. Simple program maintenance.
  4. It contains information about the data of the system and there is an entry in the data dictionary for every element of DFD. Thus DFD and Data Dictionary are compliment of each other.
- **Data Items:**
- There are three classes of data items –
  - a. Data Element** –It is the smallest unit of data which cannot be meaningfully decomposed further.
    - For Example: Employee code, unit of measurement
  - b. Data Structure** –A group of data elements forms a data structure.
    - For Example: Data Structure of employee consist of a group of data elements such employee code, name, age, experience, phone no., address etc.
  - c. Data Flow & Data stores** –Data flows are data structures in motion whereas Data stores are data structure at rest. Data stores may be files, database etc.



○ **Format of Data Dictionary:**

○ A data dictionary is organized into five sections –

1. Data Elements
2. Data Flows
3. Data Store
4. Process
5. External Entities

○ Data Dictionary lists all data elements, flows, stores, process of the system and it gives the detail about each item in following format –  
Data types, data name, data description, data characteristics, data control information, composition, physical location of data, etc.

○ Data dictionary for data element Employee code in dictionary –

- Data Element - Employee code
- Description - Unique code assigned for each employee
- Type - char
- Length - 4
- Range - 0-9999
- Data Stores - Employee table, Payroll table

### **:System design process:**

- Requirements are translated into design specifications.
- Below are different objectives in designing an information system:
- **Specify the logical design elements:**
- Systems design involves first logical design and then physical construction of the system.
- When analysts formulate a logical design, they write the detailed specifications for the new system, they describe its features: output, inputs, files and databases and procedures. The statement of these features is termed the design specifications of the system.
- The logical design of an information system is like the engineering blueprint of an automobile, it shows the major features and how they are related to one another.
- Physical construction, the activity following logical design, produces program software, files and a working system.
- The programmers in turn write the programs that accept input from users, process data, produce the reports, and store data in the files.
- The physical design of program steps, written in a programming language. During physical construction, programmers write the program instructions to compute the changes and produce the results.
- **Support business activities:**
- A fundamental objective in the design of an information system is to ensure that it supports the business activity for which it is developed.
- In other words, the computer and communications technology specified in the design should always be secondary to the results the system is intended to produce.
- Similarly, the design must fit the way a firm does business. If a sales system designed to work best for orders that are paid in cash, when in fact the firm offers a liberal “sales-on-credit” policy, management will not be very happy, nor will customers.
- Even if the needs of the business, an objective that should guide virtually all systems design decisions.
- **Ensure that system features meet user requirements:**
- An information system meets user needs if it accomplishes the following:
  - Performs the right procedures
  - Presents information and instructions in an acceptable and effective fashion
  - Produces accurate results
  - Provides an acceptable interface and method of interaction
  - Is perceived by users as a reliable system
- **Provide a system engineered for ease of use by people:**
- An experienced systems analyst will know that many technical features of an information system-such as reliability, accuracy, and processing speed- are secondary to the human aspects of a system design.
- So analysts try to design the system to be engineered for people and to include ergonomic [physical] features.

- **Human engineering:**

- The analyst should try to formulate a systems design that,
  - Incorporates system features that are easy to understand and use
  - Avoid user error or carelessness
  - Provides enough flexibility to fit a variety of individual user needs
  - Adapts to increasing user familiarity with the system
  - Generally functions in a manner that seems natural to the user

- **Ergonomic design:**

- It refers to the physical factors of an information system that affect the performance, comfort, and satisfaction of direct users.
- The design of terminals, chairs and other equipment influences the amount of weakness and damage involved in using these items.
- These factors in turn affect such concerns as the introduction of errors during data entry, user efficiency, and even absenteeism.

- **Provide detailed software development specifications:**

- The specifications state input, output and the processing functions and algorithms used to perform them.
- Software modules and routines focusing on what function each performs and the procedures for accomplishing them are specified as well.
- Selection of programming languages, software packages and software utilities occurs during the logical design process and the recommendations are included in the software specifications.

- **Conform to design standards:**

- The objectives of systems design are broad and affect many aspects of both the application and the organization in which the system will be used.
- Information systems groups also maintain systems development standards that will generate system design specifications.
- Examples of areas included in design standards:
  - **Data standards:** guidelines for data item names, length, and type specifications that are used for all applications developed by the information systems group.
  - **Coding standards:** Formal abbreviations and designations to describe activities and entities within the organizations.
  - **Structural standards:** Guidelines on how to structure the system the system and software. Policies on software modularization, structured coding, and the interrelation of system components.
  - **Documentation standards:** Descriptions of systems design features, interrelation of components and operating characteristics that can be reviewed to learn the details of the application.

**:Design of effective Output:**

- Requirements are translated into design specifications.
- Below are different objectives in designing an information system:
- **How to represent data:**
- Below are the guidelines for presentation of data:
- Tabular format
- Graphic information
- Use of icons
- Ways to utilize colors
- **Tabular format:**
- In general, end-users are most used to receiving information in tabular format.
- Common examples of tabular reports are inventory control, account payable, general ledger, sales analysis and production scheduling reports.
- In general, the tabular format should be used under the following conditions:
  - Details dominate and few narrative comments or explanations are needed
  - Details are presented in discrete categories
  - Each category must be labeled
  - Total must be drawn or comparisons made between components
- Certain information in a tabular format is more important and should be more visible than other information. This will vary by application, but in general, the following items should stand out:
- Exceptions to the normal expectations
- Major categories or groups of activities or entities
- Summaries of major categories activities
- Unique identification information
- Time-dependent entities
- **Graphic format:**



**:Design of effective Input:****▪ Objectives of input design:**

- Input design consists of developing specifications and procedures for data preparation, those steps necessary to put transaction data into a usable form for processing, and data entry, the activity of putting the data into the computer from processing.

- Five objectives guiding the design of input focus on:

**▪ 1. Controlling amount of input:**

- There are several reasons why an effective design should control the quantity of data for input. First, Data preparation and data entry operations depend on people.
- Second, the input phase of computing can be slow process that can take many times longer than the time needed by computers to carry out their tasks.

**▪ 2. Avoiding delay:**

- A processing delay resulting from data representation or data entry operations is called blockage. Resolve by turnaround documents.

**▪ 3. Avoiding errors in data:**

- The rate at which errors occur depends on the quantity of data, since the smaller the amount of data to input, the fewer the opportunities for errors.
- Still another aspect of error control is the need to detect errors when they do occur.
- Also can do with input validation techniques.

**▪ 4. Avoiding extra steps:**

- When the volume of transactions cannot be reduced, the analyst must be sure the process is as efficient as possible.
- The experienced analyst will also avoid input designs that cause extra steps.

**▪ 5. Keeping the process simple:**

- The best advice to analysts is to achieve all of the objectives mentioned in the simplest manner possible.

**▪ GOOD FORM DESIGN**

- The systems analyst should be capable of designing a complete and useful form. Unnecessary forms that waste an organization's resources should be eliminated.
- Forms are important instruments for steering the course of work. They are preprinted papers that require people to fill in responses in a standardized way.
- Forms elicit and capture information required by organizational members that will often be input to the computer.
- Through this process, forms often serve as source documents for users or for input to ecommerce applications that humans must enter.
- To design forms that people find useful, four guidelines for form design should be observed:
  1. Make forms easy to fill in.
  2. Ensure that forms meet the purpose for which they are designed.
  3. Design forms to ensure accurate completion.
  4. Keep forms attractive.

- **1. Make forms easy to fill in:**
- To reduce error, speed completion, and facilitate the entry of data, it is essential that forms be easy to fill in.
- **FORM FLOW.** Designing a form with proper flow can minimize the time and effort expended by employees in form completion.
- **SEVEN SECTIONS OF A FORM.** A second method that makes it easy for people to fill out forms correctly is logical grouping of information. The seven main sections of a form are the following:
  - 1. Heading.
  - 2. Identification and access.
  - 3. Instructions.
  - 4. Body.
  - 5. Signature and verification.
  - 6. Totals.
  - 7. Comments.
- **GOOD DISPLAY AND WEB FORMS DESIGN**
- The four guidelines for display design are important but not exhaustive. As noted in they include the following:
  - 1. Keep the display simple.
  - 2. Keep the display presentation consistent.
  - 3. Facilitate user movement among display screens and pages.
  - 4. Create an attractive and pleasing display.