



**Department of ICT**

**System Analysis and Design (SYAD-2432)**

**CHAPTER 4: System Analysis**

**2<sup>nd</sup> Lt. Lencho A.**

**Masters in Computer Science and Engineering**

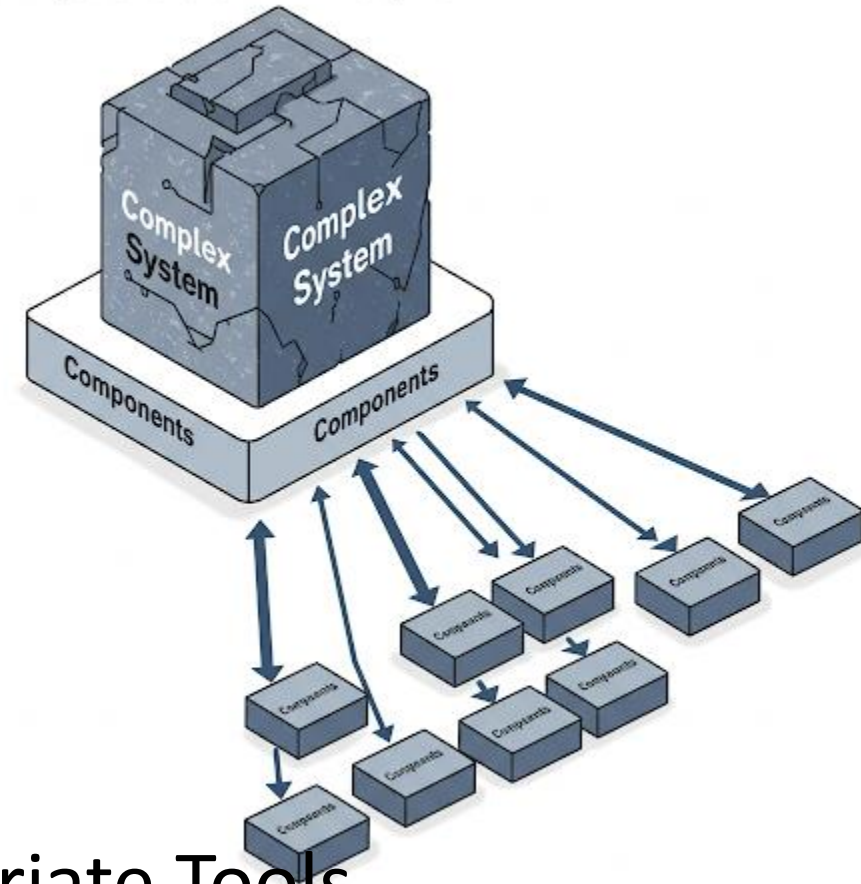
**E-Mail: [lenchoajema@gmail.com](mailto:lenchoajema@gmail.com)**

**March, 2025**

# CHAPTER 4: SYSTEM ANALYSIS

- 4.1 What is Structured Analysis
- 4.2 Structured Analysis Tools
  - 1. Data Flow Diagrams (DFD)
  - 2. Data Dictionary
  - 3. Decision Trees
  - 4. Decision Tables
  - 5. Structured English
  - 6. Pseudo code
- 4.9 Guidelines for Selecting Appropriate Tools

Decomposition Principle



# Introduction

- What is system analysis?
  - System analysis is a methodical approach to understanding and evaluating complex systems.
  - It involves gathering information, identifying requirements, and proposing solutions.
  - System analysts collaborate with stakeholders and communicate effectively to ensure successful system development and implementation.
  - The purpose of system analysis is to understand the current system and identify areas for improvement.
  - System analysis is a critical step in the system development life cycle.

# Structured Analysis

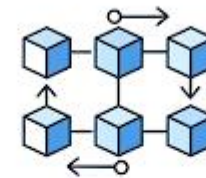
- Structured analysis is a systematic approach to system analysis that uses a set of graphical tools to represent the system.
- Structured analysis is a powerful tool for understanding complex systems.

## Prunice vis stages

elotors in sip sign



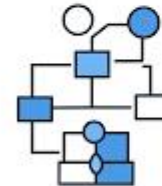
Decomposition



Data Flow  
Representation



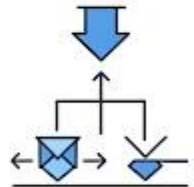
Logical Modeling



Graphical  
Representation



Logical  
Understanding



Top-Down  
Approach

# Structured Analysis Tools

- There are a number of structured analysis tools that can be used to represent the system.
- Some of the most common tools include:
  1. Data flow diagrams (DFDs)
  2. Data dictionaries
  3. Decision trees
  4. Decision tables
  5. Structured English
  6. Pseudo code

## PROS



IMPROVED UNDERSTANDING  
UNDEER STIANING



ENHANCED COMMUNICATION  
COSPEATION SYSTEM

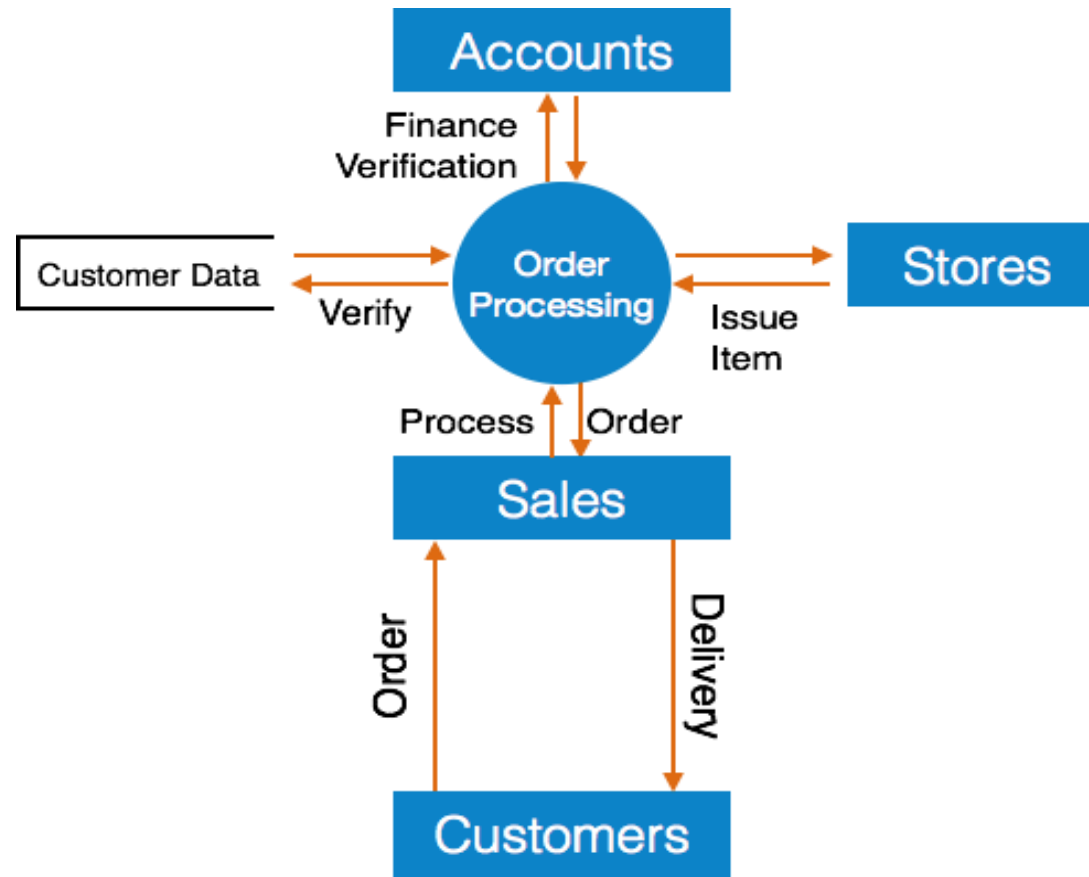


EARLY ERROR DETECTION



# 1. Data Flow Diagrams (DFDs)

- A DFD is a graphical representation of the flow of data through a system.
  - DFDs can be used to understand the system's data flows, processes, and data stores.







# DFD Components



- DFD can represent **Source, destination, storage** and **flow of data** using:-
  1. **Entities** -
    - Entities are source and destination of information data.
    - Entities are represented by a rectangles with their respective names.
  2. **Process** -
    - Activities and action taken on the data are represented by Circle or Round-edged rectangles.
  3. **Data Storage** -
    - There are two variants of data storage - it can either be represented as a rectangle with absence of both smaller sides or as an open-sided rectangle with only one side missing.
  4. **Data Flow** -
    - Movement of data is shown by pointed arrows.
    - Data movement is shown from the base of arrow as its source towards head of the arrow as destination.


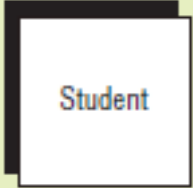

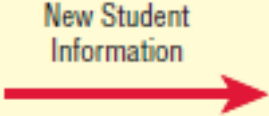

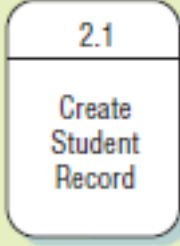

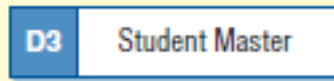
# Conventions Used in Data Flow Diagrams

- Four basic symbols are used to chart data movement on data flow diagrams:
  -  a double square,
  -  an arrow,
  -  a rectangle with rounded corners, and
  -  an open-ended rectangle (closed on the left side and open ended on the right).
- An entire system and numerous subsystems can be depicted graphically with these four symbols in combination.



# Conventions Used in Data Flow Diagrams...cont

The four basic symbols used in data flow diagrams, their meanings, and examples.

Symbol	Meaning	Example
	Entity	
	Data Flow	
	Process	
	Data Store	

# Conventions Used in Data Flow Diagrams...cont

## ➤ PROCESS

- ✍ A process is the work or actions performed on data so that they are transformed, stored, or distributed.
- ✍ A process **receives input data and produces output** that has a different content, form, or both.
- ✍ The symbol of it is **a rectangle with rounded corners**.
- ✍ The name of the process appears inside the rectangle.
- ✍ It identifies **a specific function** and consists of a **verb (and an adjective, if necessary)** followed by a singular noun.
- ✍ If additional levels of detail is needed, the analyst can zoom in on a process symbol and create a more in-depth DFD that shows the process's internal workings — which might reveal even more **processes, data flows, and data stores**. In this manner, the information system can be modeled as a series of increasingly detailed pictures.

# Conventions Used in Data Flow Diagrams...cont

## ➤ ENTITY

- ✍ Entity is the origin and/or destination of the data.
- ✍ It is sometimes called source/sink
- ✍ It is outside the system that sends or receives data, communicating with the system being diagrammed.
- ✍ They are the sources and destinations of information entering or leaving the system.
- ✍ Each entity must be connected to a process by a data flow
- ✍ They might be an outside organization or person, a computer system or a business system.
- ✍ They are also known as terminators, sources and sinks or actors.
- ✍ Depicted by double square and is an external entity (another department, a business, a person, or a machine)
- ✍ Named using nouns and appears inside the symbol

# Conventions Used in Data Flow Diagrams...cont

## ➤ DATA FLOW

- ✍ A data flow is a path for data to move from one part of the information system to another.
- ✍ It is **data that are in motion** and moving as a unit from one place in a system to another.
- ✍ A data flow in a DFD represents one or more data items.
- ✍ The symbol for a data flow is a line with a single or double arrowhead.
- ✍ The data flow name appears above, below, or alongside the line.
- ✍ A data flow name consists of **a singular noun and an adjective**, if needed.

# Conventions Used in Data Flow Diagrams...cont

## **DATA STORE**

- A data store is **data at rest**. A data store may represent one of many different physical locations for data, including a file folder, one or more computer-based file(s), or a notebook.
- A data store is used in a DFD to represent data that the system stores because one or more processes need to use the data at a later time.
- The data store has to have at least one incoming and one outgoing data flow and is connected to a process symbol with a data flow.
- The name of the data store appears between the lines and identifies the data it contains.
- A data store name is **a plural name consisting of a noun and adjectives**, if needed.

# Developing Data Flow Diagrams

- To begin a data flow diagram, collapse the organization's system narrative (or story) into a list with the four categories of external entity, data flow, process, and data store.
- Basic rules to follow are:
  - ✍ The data flow diagram must have at least one process
  - ✍ A process must receive at least one data flow coming into the process and create at least one data flow leaving from the process.
  - ✍ A data store should be connected to at least one process.
  - ✍ External entities should not be connected to each other.

# Context Diagram

- A **system context diagram** in software engineering or systems engineering is a diagram that **defines the boundary** between the system, or part of a system, and its environment, showing the entities that interact with it.
- The objective of the system context diagram is to **focus attention on external factors and events that should be considered** in developing a complete set of systems requirements and constraints.
- It is a **high level view of a system** and **contains only one process**, representing the entire system.
- The process is given the **number zero**.
- **All external entities are shown** on the context diagram, as well as major **data flow to and from them**.

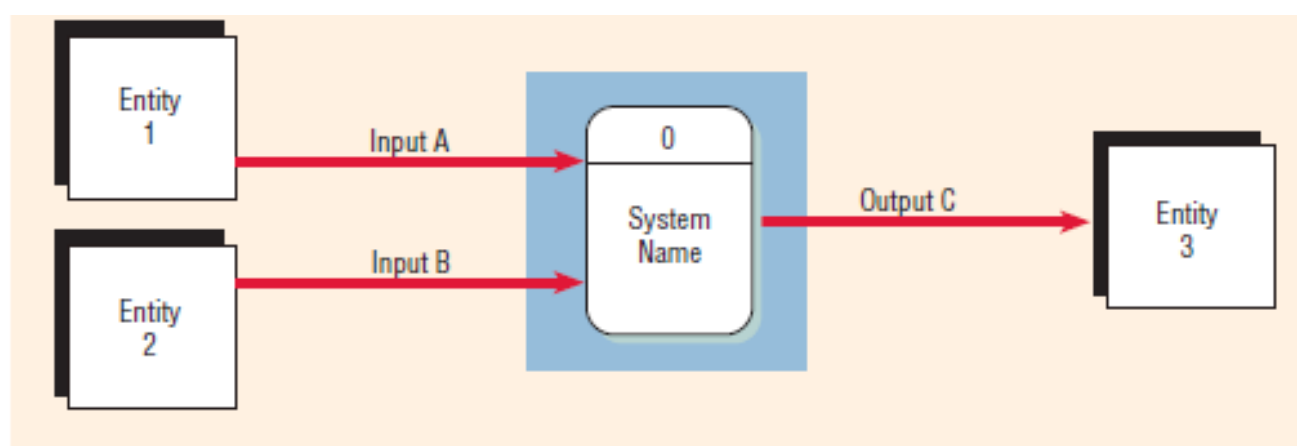
# Context Diagram

- Context diagram is a data-flow diagram of the scope of an organizational system that shows the system boundaries, external entities that interact with the system, and the major information flows between the entities and the system.
- The diagram **does not contain any data stores** and is fairly simple to create, once the external entities and the data flow to and from them are known to analysts.
- **Some of the benefits of a Context Diagram are:**
  - ✍ Shows the scope and boundaries of a system at a glance including the other systems that interface with it
  - ✍ No technical knowledge is required to understand the diagram
  - ✍ Easy to draw and amend due to its limited notation
  - ✍ Easy to expand by adding different levels of DFDs
  - ✍ Can benefit a wide audience including stakeholders, business analyst, data analysts, developers.

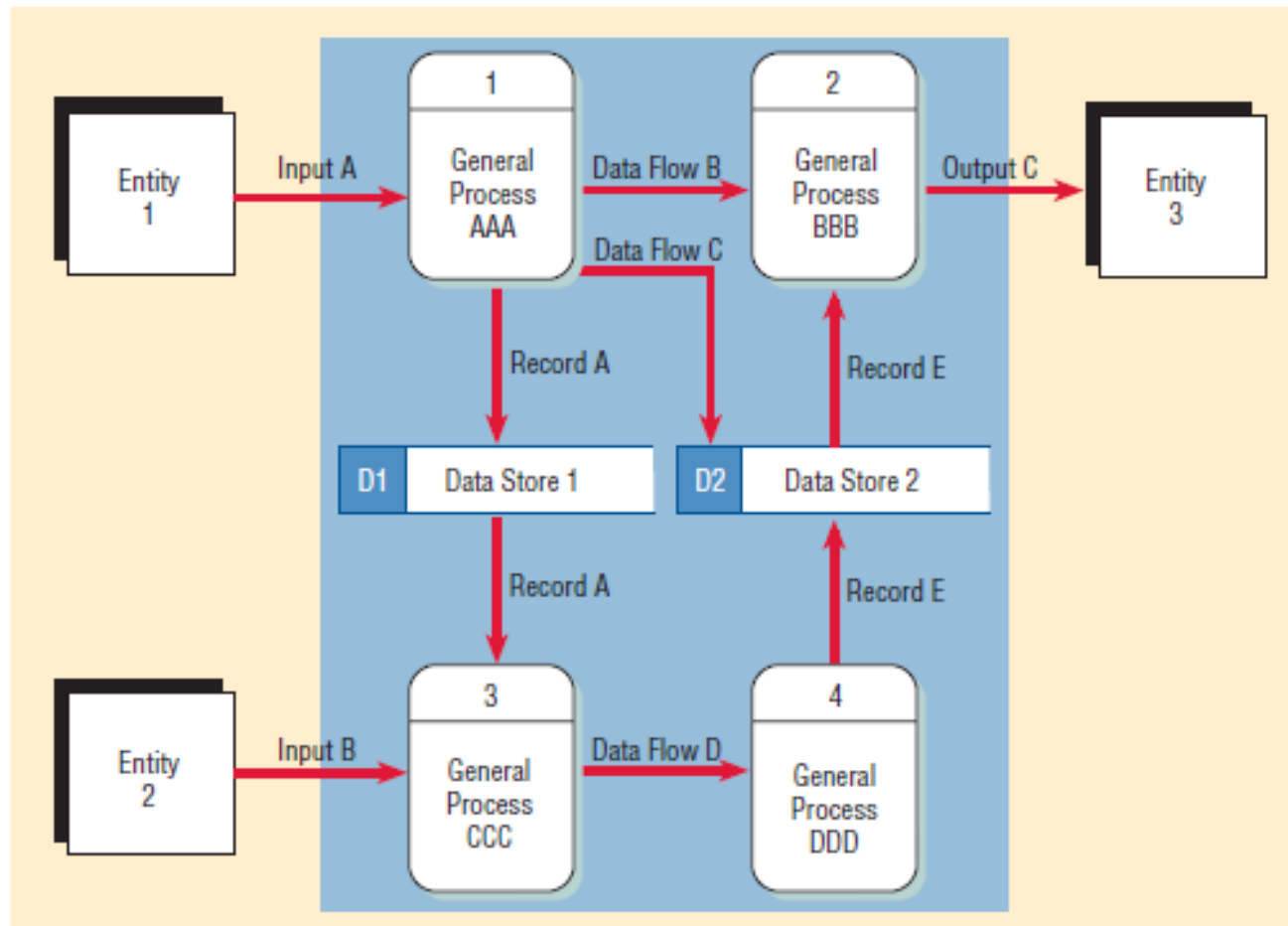


## Drawing Diagram 0 (The Next Level)

- ✍ Diagram 0 is **the explosion of the context diagram** and may include up to nine processes.
- ✍ It is **more detail than the context diagram**
- ✍ Inputs and outputs specified in the first diagram remain constant in all subsequent diagrams.
- ✍ The original diagram is **exploded into three to nine processes** and showing data stores and new lower-level data flows.
- ✍ Each process is numbered with an integer, generally starting from the upper left-hand corner of the diagram and working toward the lower right-hand corner.
- ✍ The major data stores of the system and all external entities are included on Diagram 0.



Context diagrams (above) can be “exploded” into Diagram 0 (below). Note the greater detail in Diagram 0.

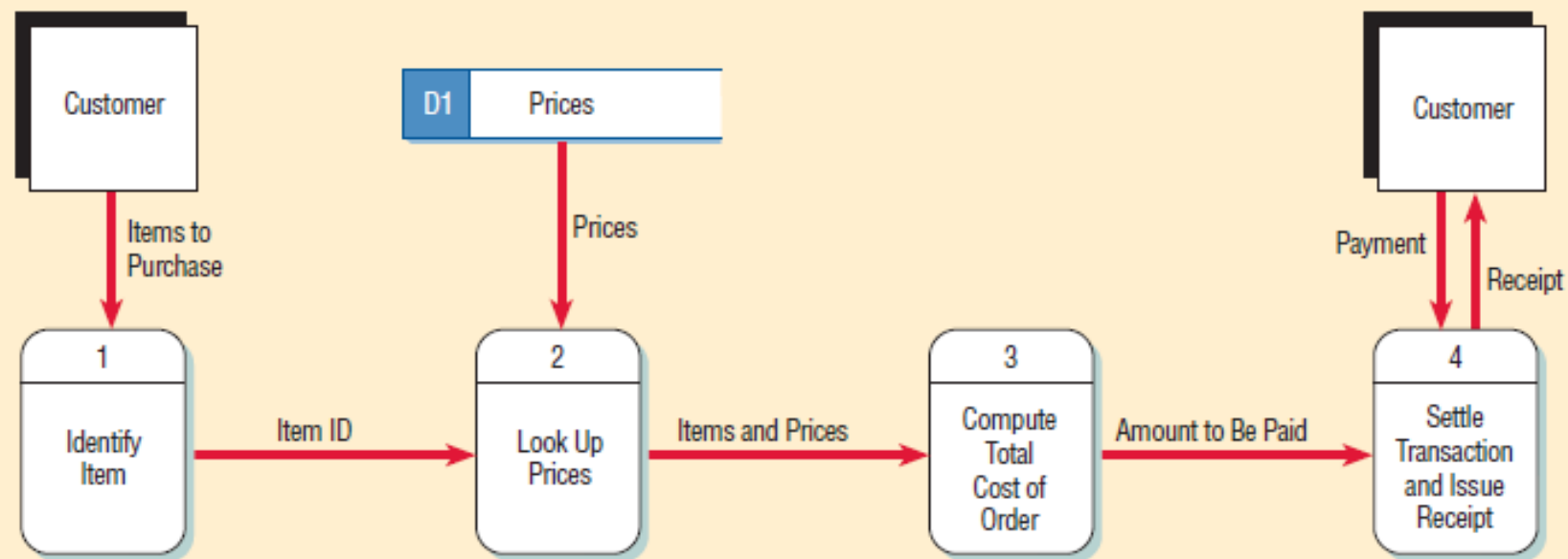


# Logical DFD vs Physical DFD

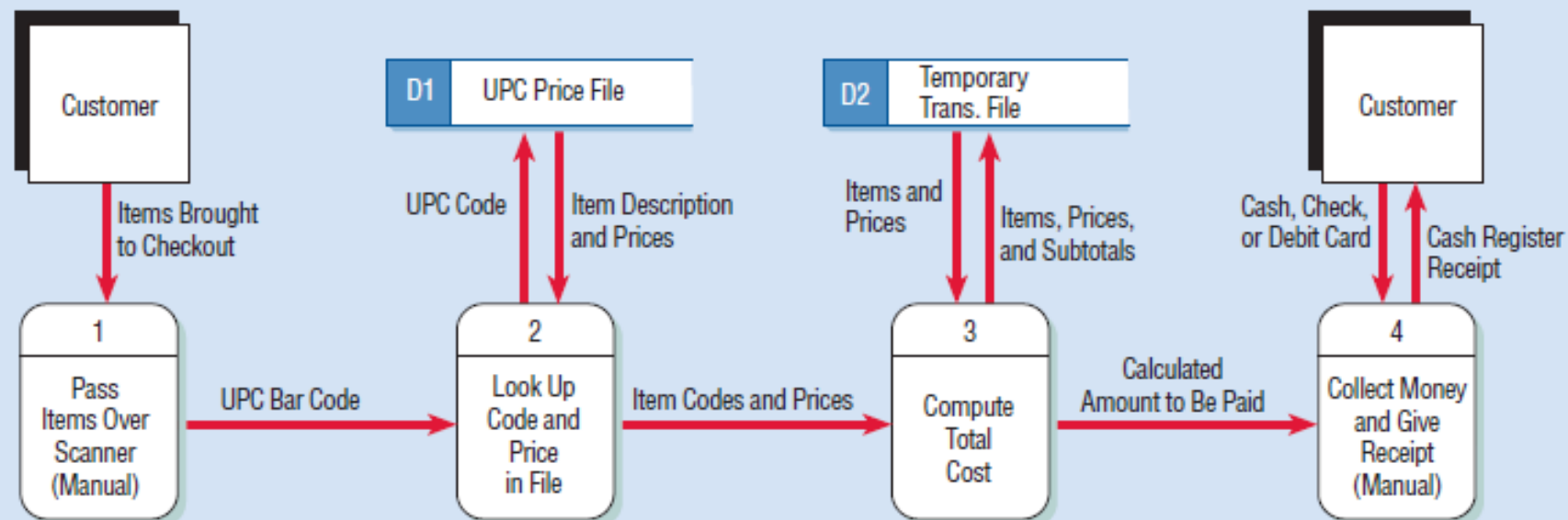
- Data flow diagrams are categorized as either logical or physical.
- A logical data flow diagram
  - ✍ Focuses on the business and how the business operates.
  - ✍ It is not concerned with how the system will be constructed.
  - ✍ It describes the business events that take place and the data required and produced by each event.
- A physical data flow diagram
  - ✍ shows how the system will be implemented, including the hardware, software, files, and people involved in the system.
- In a Logical DFD, the processes would be business activities,
- In a Physical DFD, the processes would be programs and manual procedures.

Design Feature	Logical	Physical
What the model depicts	How the business operates.	How the system will be implemented (or how the current system operates).
What the processes represent	Business activities.	Programs, program modules, and manual procedures.
What the data stores represent	Collections of data regardless of how the data are stored.	Physical files and databases, manual files.
Type of data stores	Show data stores representing permanent data collections.	Master files, transition files. Any processes that operate at two different times must be connected by a data store.
System controls	Show business controls.	Show controls for validating input data, for obtaining a record (record found status), for ensuring successful completion of a process, and for system security (example: journal records).

## Logical Data Flow Diagram



## Physical Data Flow Diagram

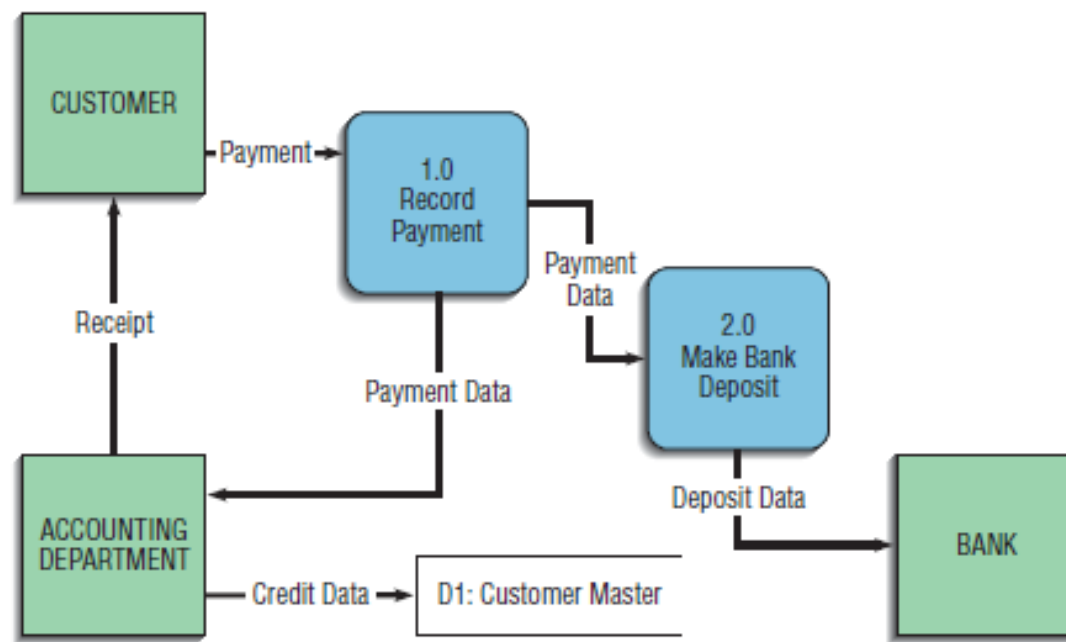


# Advantages of using a Logical Model

- ✍ Better communication with users
- ✍ More stable systems
- ✍ Better understanding of the business by analysts.
- ✍ Flexibility and maintenance
- ✍ Elimination of redundancies and easier creation of the physical model

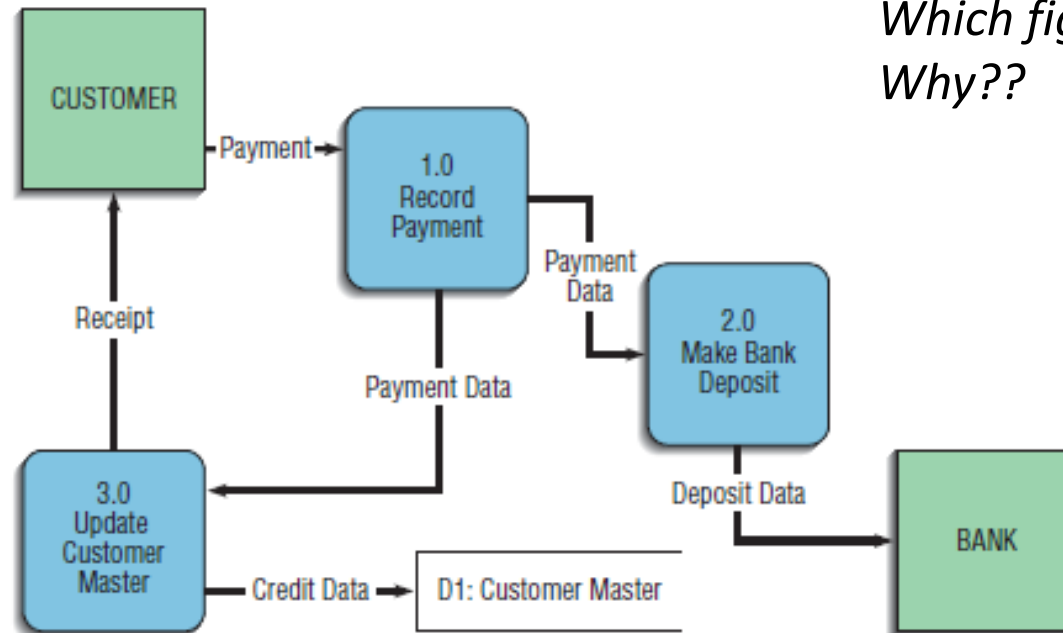
# Advantages of using a Physical Model

- ✍ Clarifying which processes are performed manually and which are automated
- ✍ Describing processes in more detail than logical DFDs
- ✍ Sequencing processes that have to be done in a particular order
- ✍ Identifying temporary data stores
- ✍ Specifying actual names of files, database tables, and printouts
- ✍ Adding controls to ensure the processes are done properly



A

*Which figure , A or B, is incorrect?  
Why??*



B



## 2. Data Dictionary

- A data dictionary is a repository of information about the data used in a system.
- The data dictionary includes information such as the
  - data's name
  - Type
  - format, and
  - usage
- The data dictionary is used to:
  - Document the system's data
  - Ensure that the data is consistent throughout the system
  - Facilitate data sharing between different parts of the system

## Analyzing Systems Using Data Dictionaries...cont

- Data dictionary is a main method for analyzing the data flows and data stores of data-oriented systems
- The data dictionary is a reference work of data about data (metadata). It means that it contains the actual database descriptions used by the DBMS.
- It collects, coordinates, and confirms what a specific data term means to different people in the organization
- The important reason for maintaining a data dictionary is to keep clean data. This means that data must be consistent.
- It can be integrated and automated within the DBMS or be separate. It can be cross-referenced during system design, programming, and by actively-executing programs.

# Reasons for Using a Data Dictionary

➤ The data dictionary is used for the following reasons:

- ✍ Provide documentation
- ✍ Eliminate redundancy
- ✍ Validate the data flow diagram
- ✍ Provide a starting point for developing screens and reports
- ✍ To develop the logic for DFD processes

# The Repository

- A data repository is a large collection of project information
- It includes
  - Information about system data
  - Procedural logic
  - Screen and report design
  - Relationships between entries
  - Project requirements and deliverables
  - Project management information

# Data Dictionary

- Data dictionary should contain information about the following

- Data Flow

- is described by means of DFDs
  - Example

- Address = House No + (Street / Area) + City + State

- Course ID = Course Number + Course Name + Course Level + Course Grades

- Data Structure

- Data Elements

- Data elements consist of Name and descriptions of Data and Control Items, Internal or External data stores etc.

- Data Stores

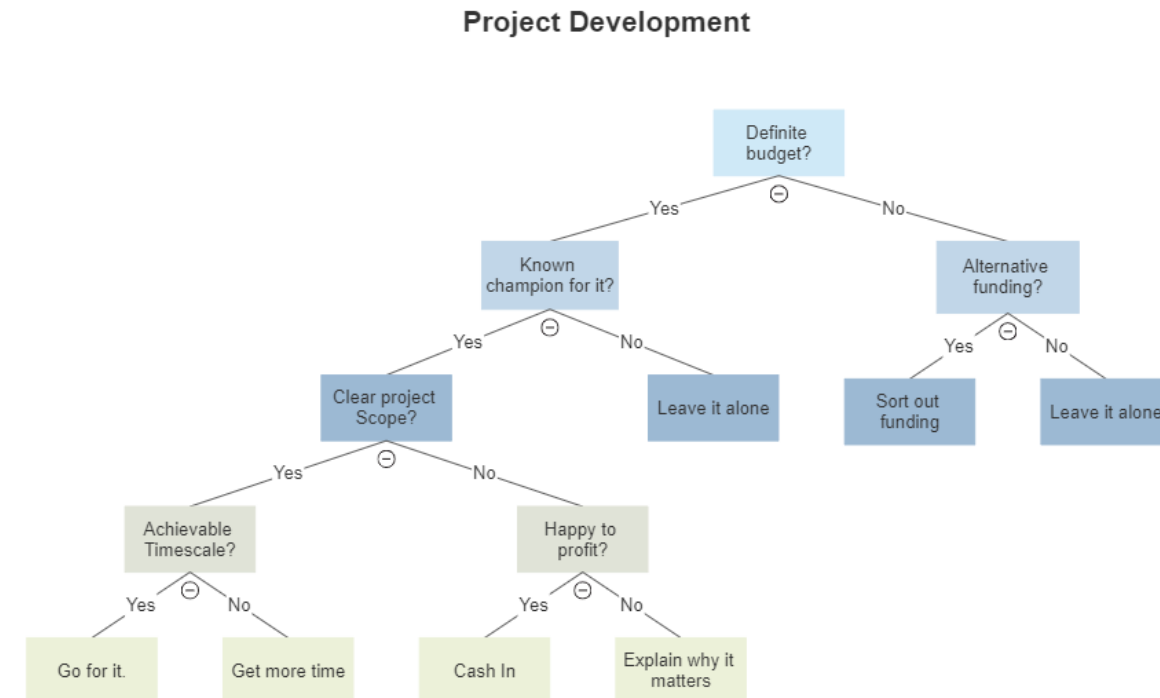
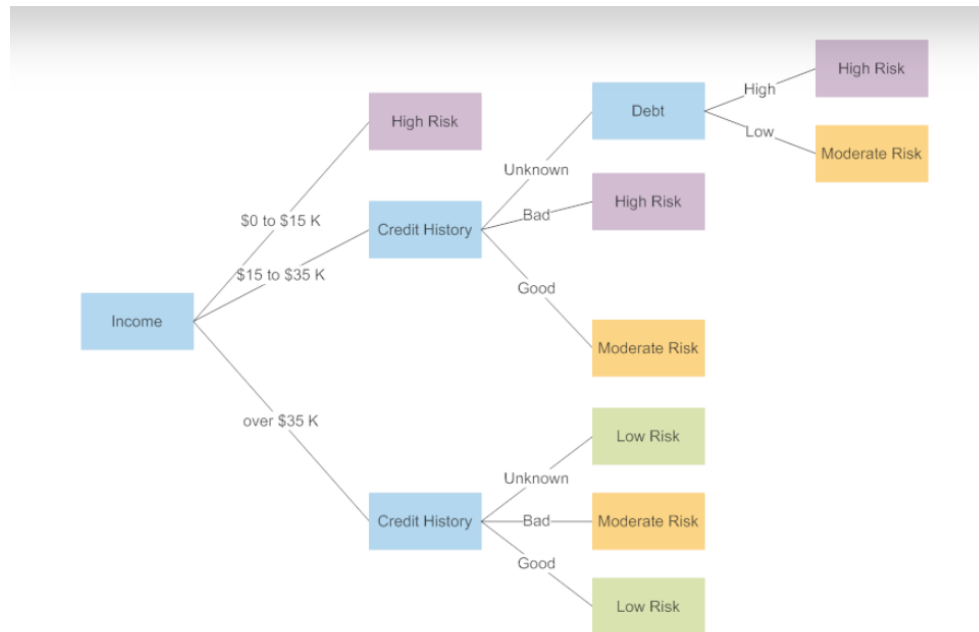
- It stores the information from where the data enters into the system and exists out of the system

- Data Processing

=	Composed of
{ }	Repetition
( )	Optional
+	And
[ / ]	Or

# 3. Decision Trees

- A decision tree is a graphical representation of a decision-making process
- Decision trees can be used to identify the possible outcomes of a decision and the associated costs and benefits.



# Decision Trees

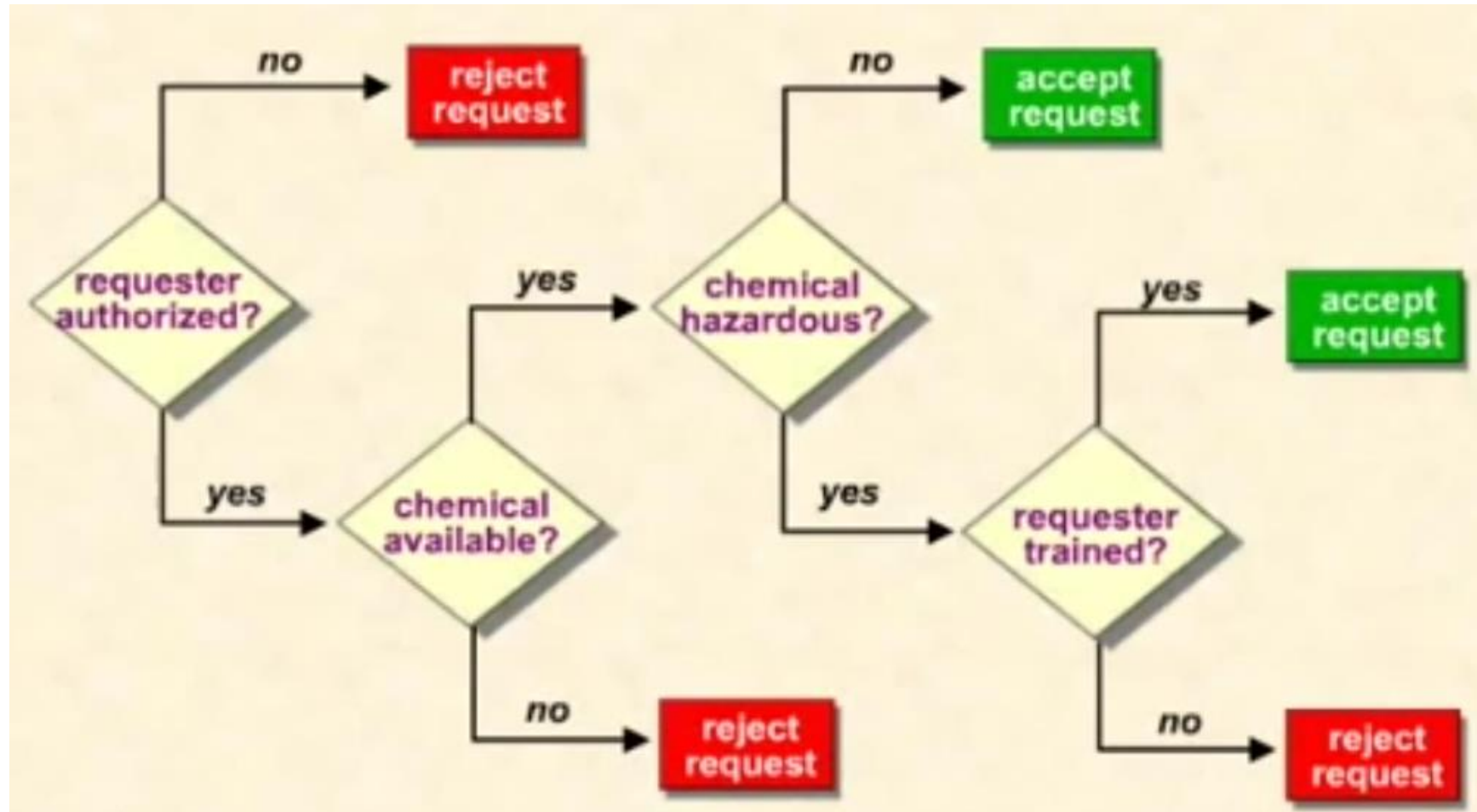
- A decision tree is a graphical representation of the conditions, actions, and rules found in a decision table.
- Decision trees show the logic structure in a horizontal form that resembles a tree with the roots at the left and the branches to the right.
- Like flowcharts, decision trees are useful ways to present the system to management.
- Decision trees and decision tables provide the same results, but in different forms. In many situations, a graphic is the most effective means of communication.

## Decision Trees...cont

- The decision tree has three main advantages over a decision table.
  - First, it takes advantage of the sequential structure of decision tree branches so that the order of checking conditions and executing actions is immediately noticeable.
  - Second, conditions and actions of decision trees are found on some branches but not on others, which contrasts with decision tables, in which they are all part of the same table. Those conditions and actions that are critical are connected directly to other conditions and actions, whereas those conditions that do not matter are absent. In other words, the tree does not have to be symmetrical.
  - Third, compared with decision tables, decision trees are more readily understood by others in the organization. Consequently, they are more appropriate as a communication tool.



## Decision Trees...cont



Whether to use a decision table or a decision tree often is a matter of personal preference. A decision table might be a better way to handle complex combinations of conditions. On the other hand, a decision tree is an effective way to describe a relatively simple process.

## 4. Decision Tables

- A decision table is a tabular representation of a decision-making process.
- Decision tables can be used to identify the possible combinations of conditions and actions and the associated outcomes.

### Example

- Let us take a simple example of day-to-day problem with our Internet connectivity.
- We begin by identifying all problems that can arise while starting the internet and their respective possible solutions.

	Conditions/Actions	Rules							
Conditions	Shows Connected	N	N	N	N	Y	Y	Y	Y
	Ping is Working	N	N	Y	Y	N	N	Y	Y
	Opens Website	Y	N	Y	N	Y	N	Y	N
Actions	Check network cable	X							
	Check internet router	X				X	X	X	
	Restart Web Browser								X
	Contact Service provider		X	X	X	X	X	X	
	Do no action								

Table : Decision Table – In-house Internet Troubleshooting

# Decision Tables

- A decision table is **a table of rows and columns, separated into four quadrants.**
  - The upper left quadrant contains the condition(s);
  - The upper right quadrant contains the condition alternatives.
  - The lower half of the table contains the actions to be taken on the left and
    - The rules for executing the actions on the right.
- When a decision table is used to determine which action needs to be taken, the logic moves clockwise beginning from the upper left.

Conditions and Actions	Rules
Conditions	Condition Alternatives
Actions	Action Entries

# Developing Decision Tables

- To build decision tables, the analyst needs to **determine the maximum size of the table; eliminate any impossible situations, inconsistencies, or redundancies; and simplify** the table as much as possible.
- The following steps provide the analyst with a systematic method for developing decision tables:
  1. **Determine conditions** that may affect the decision. The number of conditions becomes the number of rows in the top half of the decision table.
  2. **Determine the number of possible actions** that can be taken. That number becomes the number of rows in the lower half of the decision table.
  3. **Determine the number of condition alternatives for each condition.** In the simplest form of decision table, there would be two alternatives (Y or N) for each condition.

# Developing Decision Tables

4. Calculate the maximum number of columns in the decision table by multiplying the number of alternatives for each condition. If there are four conditions and two alternatives (Y or N) for each of the conditions, there would be 16 possibilities as follows:
  - Condition 1: x 2 alternatives
  - Condition 2: x 2 alternatives
  - Condition 3: x 2 alternatives
  - Condition 4: x 2 alternatives

→  **$2^4 = 16$  = possibilities**
5. Fill in the condition alternatives. Start with the first condition and divide the number of columns by the number of alternatives for that condition.

Condition 1: Y Y Y Y Y Y Y Y N N N N N N N N  
 Condition 2: Y Y Y Y N N N N  
 Condition 3: Y Y N N  
 Condition 4: Y N

and continue the pattern for each condition:

Condition 1 Y Y Y Y Y Y Y Y N N N N N N N N  
 Condition 2: Y Y Y Y N N N N Y Y Y Y N N N N  
 Condition 3: Y Y N N Y Y N N Y Y N N Y Y N N  
 Condition 4: Y N Y N Y N Y N Y N Y N Y N Y N

6. Complete the table by inserting an X where rules suggest certain actions.
7. Combine rules where it is apparent that an alternative does not make a difference in the outcome. For example,

Condition 1:	Y Y
Condition 2:	Y N
<hr/>	
Action 1:	X X

can be expressed as:

Condition 1:	Y
Condition 2:	—
<hr/>	
Action 1:	X

The dash [—] signifies that Condition 2 can be either Y or N, and the action will still be taken.

8. Check the table for any impossible situations, contradictions, and redundancies. They are discussed in more detail later.
9. Rearrange the conditions and actions (or even rules) if it makes the decision table more understandable.

# Simplify the Table

- When all the outcomes have been determined, you are ready to **simplify the table**. In a multi-condition table, some rules might be duplicated, redundant, or unrealistic.
- To simplify the table, the steps are:
  1. Study each combination of conditions and outcomes. When you have rules with three conditions, only one or two of them may control the outcome, and the other conditions simply do not matter.
  2. If you identify conditions that **do not affect the outcome**, mark them with dashes (-)
  3. Now **combine and renumber** the rules

# Simplify the Table

## ➤ Tables with One Condition

- If a process has a **single condition**, there only are **two possibilities - yes or no**. *Either the condition is present or it is not, so there are only two rules.*

For example, to trigger an overtime calculation, the process condition might be: *Are the credit hours greater than 12? If so, the calculation is made. Otherwise, it is not.*

## ➤ Tables with Two Conditions

Suppose you want to create a decision table based on the **Verify Order business process** shown in the figure below



### VERIFY ORDER Business Process with Two Conditions

- An order will be accepted only if the product is in stock and the customer's credit status is OK.
- All other orders will be rejected.

- When documenting a process, it is important to ensure that you list every possibility.

The process description contains two conditions- **product stock status and customer credit status**. *If both conditions are met, the order is accepted. Otherwise the order is rejected.*

VERIFY ORDER Process	1	2	3	4
Credit status is OK	Y	Y	N	N
Product is in stock	Y	N	Y	N
Accept order	X			
Reject order		X	X	X

- Tables with Three Conditions-

Suppose the company now decides that **the credit manager can waive the customer credit requirement**, and that creates a third condition, *so there will be eight possible rules.*

### VERIFY ORDER Business Process with Three Conditions

- An order will be accepted only if the product is in stock and the customer's credit status is OK.
- The credit manager can waive the credit status requirement.
- All other orders will be rejected.

VERIFY ORDER Process	1	2	3	4	5	6	7	8
Credit status is OK	Y	Y	Y	Y	N	N	N	N
Product is in stock	Y	Y	N	N	Y	Y	N	N
Waiver from credit manager	Y	N	Y	N	Y	N	Y	N
Accept order	X	X			X			
Reject order			X	X		X	X	X

If you follow the three simplifying steps, you will see that

Rules 1 and 2 can be combined because credit status is OK in both rules, so the waiver would not matter.

Rules 3, 4, 7, and 8 also can be combined because the product is not in stock, so other conditions do not matter.

The result is that instead of eight possibilities, only four logical rules control the Verify Order process.

VERIFY ORDER Process with Credit Waiver (after rule combination and simplification)

VERIFY ORDER Process	1,2	3,4	5	6	7,8
Credit status is OK	Y	Y	N	N	N
Product is in stock	Y	N	Y	Y	N
Waiver from credit manager	-	-	Y	N	-
Accept order	X		X		
Reject order		X		X	X

VERIFY ORDER Process	1,2	5	6	3,4,7,8
Credit status is OK	Y	N	N	-
Product is in stock	Y	Y	Y	N
Waiver from credit manager	-	Y	N	-
Accept order	X	X		
Reject order			X	X

➤ **How does the Chemical Tracking System decide whether to approve or reject a request?**

**Conditions we need to see:**

- ☒ Is the **requester authorized** to request chemical?
- ☒ Is the **chemical available** either in the chemical stockroom or from a vendor?
- ☒ Is the chemical on the list of **hazardous chemical**?
- ☒ Is the **requester trained** in handling hazardous chemicals?

CHEMICAL TRACKING Process	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Requester authorized?	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N
Chemical available?	Y	Y	Y	Y	N	N	N	N	Y	Y	Y	Y	N	N	N	N
Chemical hazardous?	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N
Requester trained?	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N
Accept request	X		X	X												
Reject request		X			X	X	X	X	X	X	X	X	X	X	X	X

CHEMICAL TRACKING Process	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Requester authorized?	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N
Chemical available?	Y	Y	Y	Y	N	N	N	N	Y	Y	Y	Y	N	N	N	N
Chemical hazardous?	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N
Requester trained?	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N
Accept request	X		X	X												
Reject request		X			X	X	X	X	X	X	X	X	X	X	X	X

CHEMICAL TRACKING Process	1	2	3,4	5,6,7,8	9
Requester authorized?	Y	Y	Y	Y	N
Chemical available?	Y	Y	Y	N	-
Chemical hazardous?	Y	Y	N	-	-
Requester trained?	Y	N	-	-	-
Accept request	X		X		
Reject request		X		X	X

CHEMICAL TRACKING Process	1	2	3	4	5
Requester authorized?	Y	Y	Y	Y	N
Chemical available?	Y	Y	Y	N	-
Chemical hazardous?	Y	Y	N	-	-
Requester trained?	Y	N	-	-	-
Accept request	X		X		
Reject request		X		X	X

# Creating Decision Table

- To create the decision table, the developer must follow basic four steps:
  1. Identify all possible conditions to be addressed
  2. Determine actions for all identified conditions
  3. Create Maximum possible rules
  4. Define action for each rule



## 5. Structured English

- Structured English is a subset of the English language that is used to describe the logic of a process.
- Structured English can be used to create clear and concise descriptions of processes.

### Example

- We take example of Customer Authentication in the online shopping environment.
- This procedure to authenticate customer can be written in Structured English as:

```
Enter Customer_Name
SEEK Customer_Name in Customer_Name_DB file
IF Customer_Name found THEN
    Call procedure USER_PASSWORD_AUTHENTICATE()
ELSE
    PRINT error message
    Call procedure NEW_CUSTOMER_REQUEST()
ENDIF
```

# Structured English

- When the process logic involves formulas or iteration, or when structured decisions are not complex, an appropriate technique for analyzing the decision process is the use of structured English.
- As the name implies, structured English is based on
  - structured logic, or instructions organized into nested and grouped procedures, and
  - simple English statements such as add, multiply, and move.
- A word problem can be transformed into structured English by putting the decision rules into their proper sequence and using the convention of IF-THEN-ELSE statements throughout.

# Writing Structured English

- To write structured English, following conventions:
  - ✍ Express all logic in terms of one of these four types:
    - sequential structures,
    - decision structures,
    - case structures, or
    - iterations
  - ✍ Use and capitalize accepted keywords such as IF, THEN, ELSE, DO, DO WHILE, DO UNTIL, and PERFORM.
  - ✍ When words or phrases have been defined in a data dictionary, underline those words or phrases to signify that they have a specialized, reserved meaning.
  - ✍ Be careful when using “and” and “or,” and avoid confusion when distinguishing between “greater than” and “greater than or equal to” and like relationships. “A and B” means both A and B; “A or B” means either A or B, but not both.

# Writing Structured English...cont

Structured English Type	Example
Sequential Structure A block of instructions in which no branching occurs	Action #1 Action #2 Action #3
Decision Structure Only IF a condition is true, complete the following statements; otherwise, jump to the ELSE	IF Condition A is True THEN implement Action A ELSE implement Action B ENDIF
Case Structure A special type of decision structure in which the cases are mutually exclusive (if one occurs, the others cannot)	IF Case #1 implement Action #1 ELSE IF Case #2 Implement Action #2 ELSE IF Case #3 Implement Action #3 ELSE IF Case #4 Implement Action #4 ELSE print error ENDIF
Iteration Blocks of statements that are repeated until done	DO WHILE there are customers. Action #1 ENDDO

## 6. Pseudo code

- Pseudo code is a high-level language that is used to describe the logic of a process.
- Pseudo code is similar to programming languages, but it is not bound by the rules of any particular language.

### Example

Program to print Fibonacci up to n numbers.

```
void function Fibonacci
Get value of n;
Set value of a to 1;
Set value of b to 1;
Initialize I to 0
for (i=0; i< n; i++)
{
    if a greater than b
    {
        Increase b by a;
        Print b;
    }
    else if b greater than a
    {
        increase a by b;
        print a;
    }
}
```

# Guidelines for Selecting Appropriate Tools

- There are a number of factors to consider when selecting appropriate tools for system analysis.
- Some of the factors include:
  - The complexity of the system
  - The experience of the analysts
  - The availability of tools
  - The cost of tools
- Use structured English when
  - There are many repetitious actions, OR
  - Communication to end users is important.
- Use decision tables when
  - Complex combinations of conditions, actions, and rules are found, OR
  - You require a method that effectively avoids impossible situations, redundancies, and contradictions.
- Use decision trees when
  - The sequence of conditions and actions is critical, OR
  - When not every condition is relevant to every action (the branches are different).

# Summary

- Structured analysis is a systematic approach used in software development to understand and analyze complex systems
- It involves using various structured analysis tools to model and represent different aspects of the system
- These tools include Data Flow Diagrams (DFD), Data Dictionary, Decision Trees, Decision Tables, Structured English, and Pseudo code
- The selection of appropriate tools is guided by specific criteria and considerations, such as the nature of the system, the complexity of the problem, and the needs of stakeholders
- These tools help in capturing system requirements, visualizing processes, and facilitating effective communication and documentation throughout the software development lifecycle

# Group assignment

1. Discuss the importance of using structured analysis techniques in system development. Provide examples of how structured analysis can help in understanding and documenting system requirements.
2. Compare and contrast Data Flow Diagrams (DFDs) and Decision Trees as tools used in structured analysis. Discuss their respective strengths and weaknesses, and provide examples of situations where each tool would be most beneficial.
3. Explain the role of a Data Dictionary in structured analysis. Discuss its significance in system development and how it aids in ensuring consistency and accuracy in system documentation.