SCSD2613
System Analysis and Design

UTM
UNIVERSITI TEKNOLOGI MALAYSIA

PART IV
The Analysis Process #1:
Introduction to Data Flow Diagram (DFD)

update: August 2019 (sharinhh)

OBJECTIVES

1. Comprehend the importance of using logical and physical data flow diagrams (DFDs) to graphically depict movement for humans and systems in an organization.
2. Create, use, and explode logical DFDs to capture and analyze the current system through parent and child levels.
3. Develop and explode logical DFDs that illustrate the proposed system.
4. Produce physical DFDs based on logical DFDs you have developed.
5. Understand and apply the concept of partitioning of physical DFDs.

MAJOR TOPICS

DATA FLOW DIAGRAM	• What is DFD? • DFD Symbols
CREATING DFD	• Steps to create DFD • Context Diagram • Diagram 0 • DFD Level
LOGICAL & PHYSICAL DFD	• From logical to physical DFD
PARTITIONING	• Partitioning DFD • CRUD matrix
COMMUNICATING DFD	

MAJOR TOPICS

DATA FLOW DIAGRAM

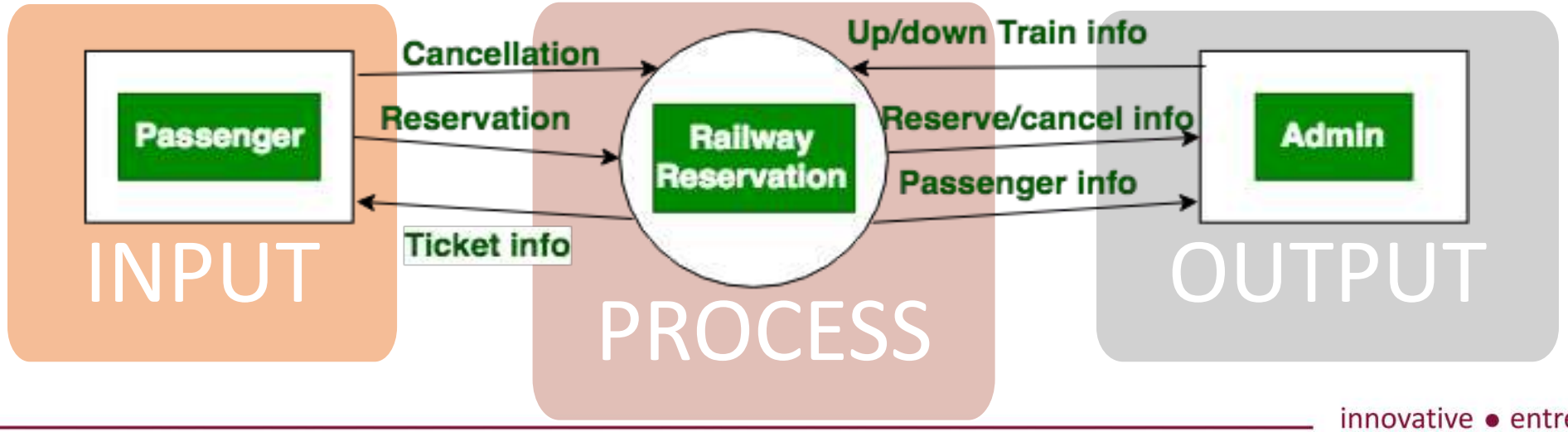
- What is DFD?
- DFD Symbols

DATA FLOW DIAGRAM

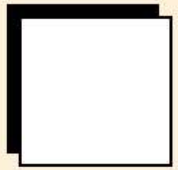
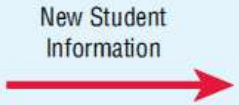
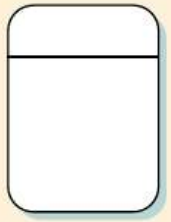
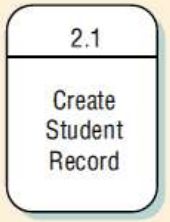
- Graphically characterize data processes and flows in a business system
- Depict:
  i. System inputs
  ii. Processes
  iii. Outputs

ADVANTAGE OF DATA FLOW APPROACH

- Freedom from committing to the technical implementation too early
- Understanding of the interrelatedness of systems and subsystems
- Communicating current system knowledge to users
- Analysis of the proposed system

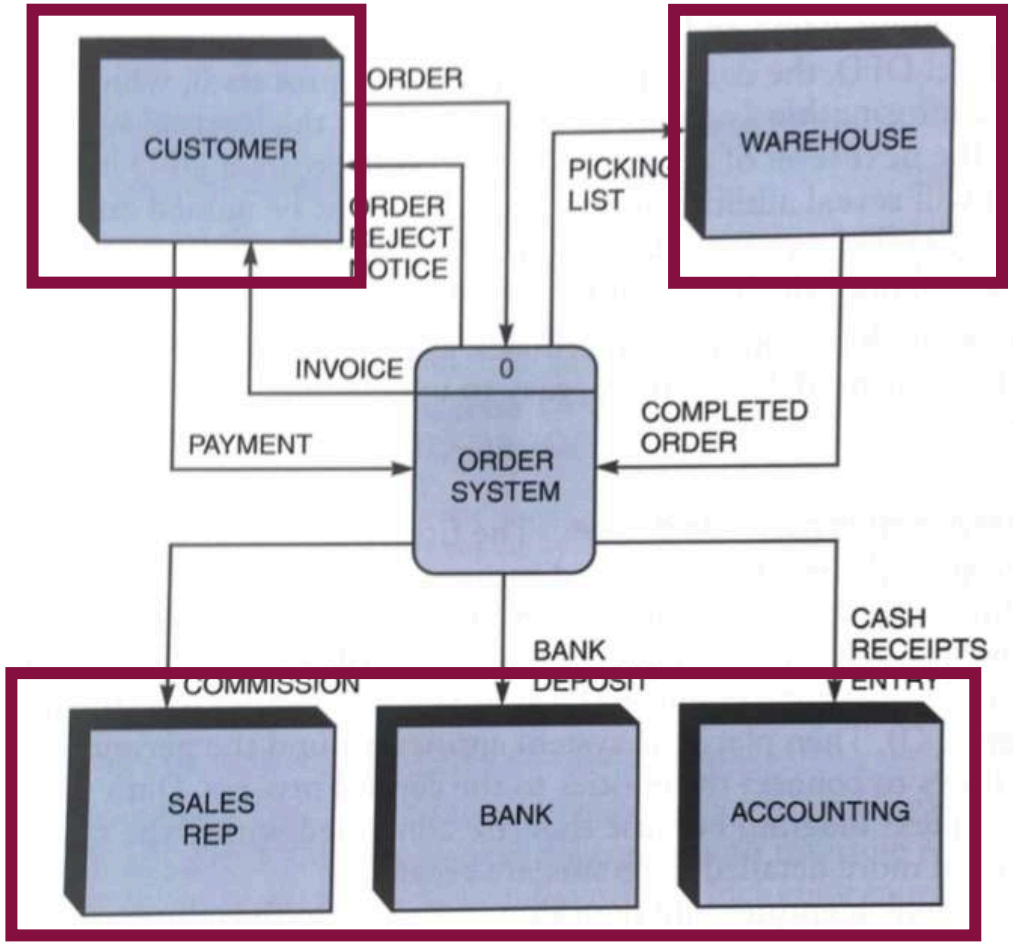


DFD BASIC SYMBOLS

	Symbol	Meaning	Example
A double square for an external entity		Entity	
An arrow for movement of data from one point to another		Data Flow	
A rectangle with rounded corners for the occurrence of a transforming process		Process	
An open-ended rectangle for a data store		Data Store	

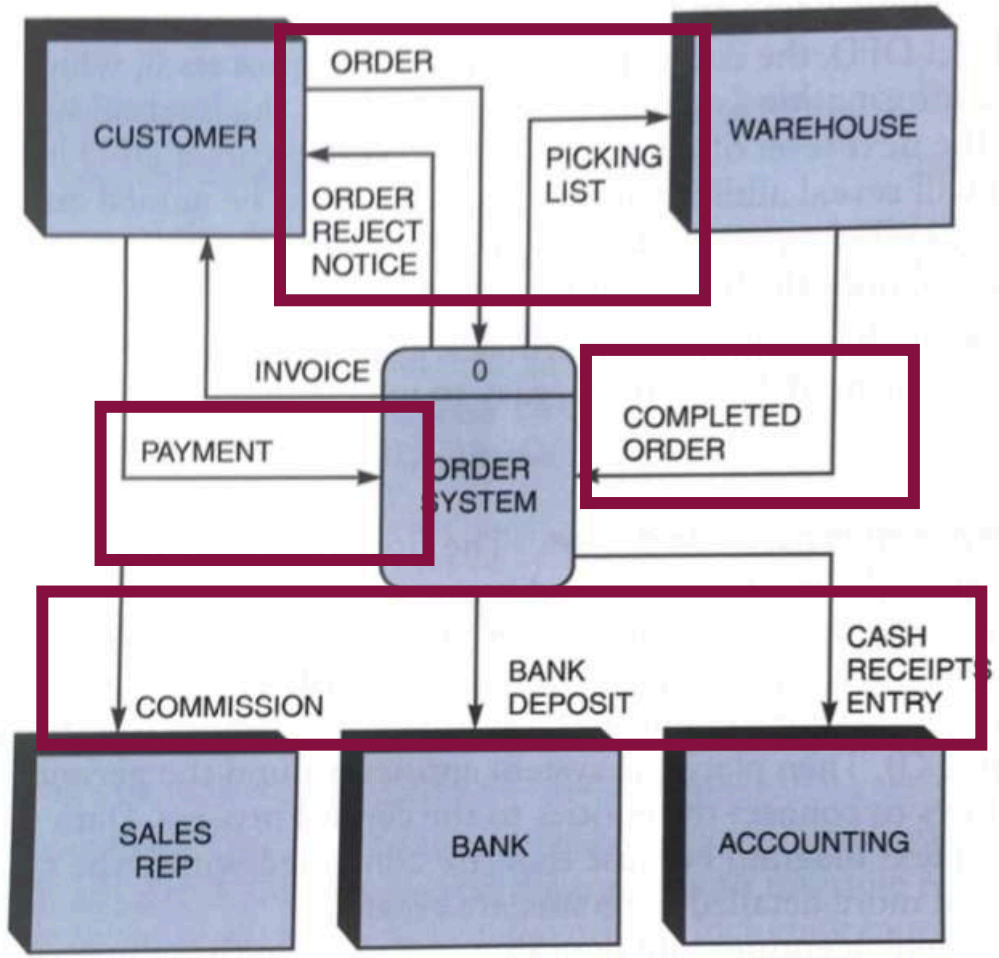
DFD – EXTERNAL ENTITIES

- Represent another department, a business, a person, or a machine
- A source or destination of data, outside the boundaries of the system
- Should be named with a noun



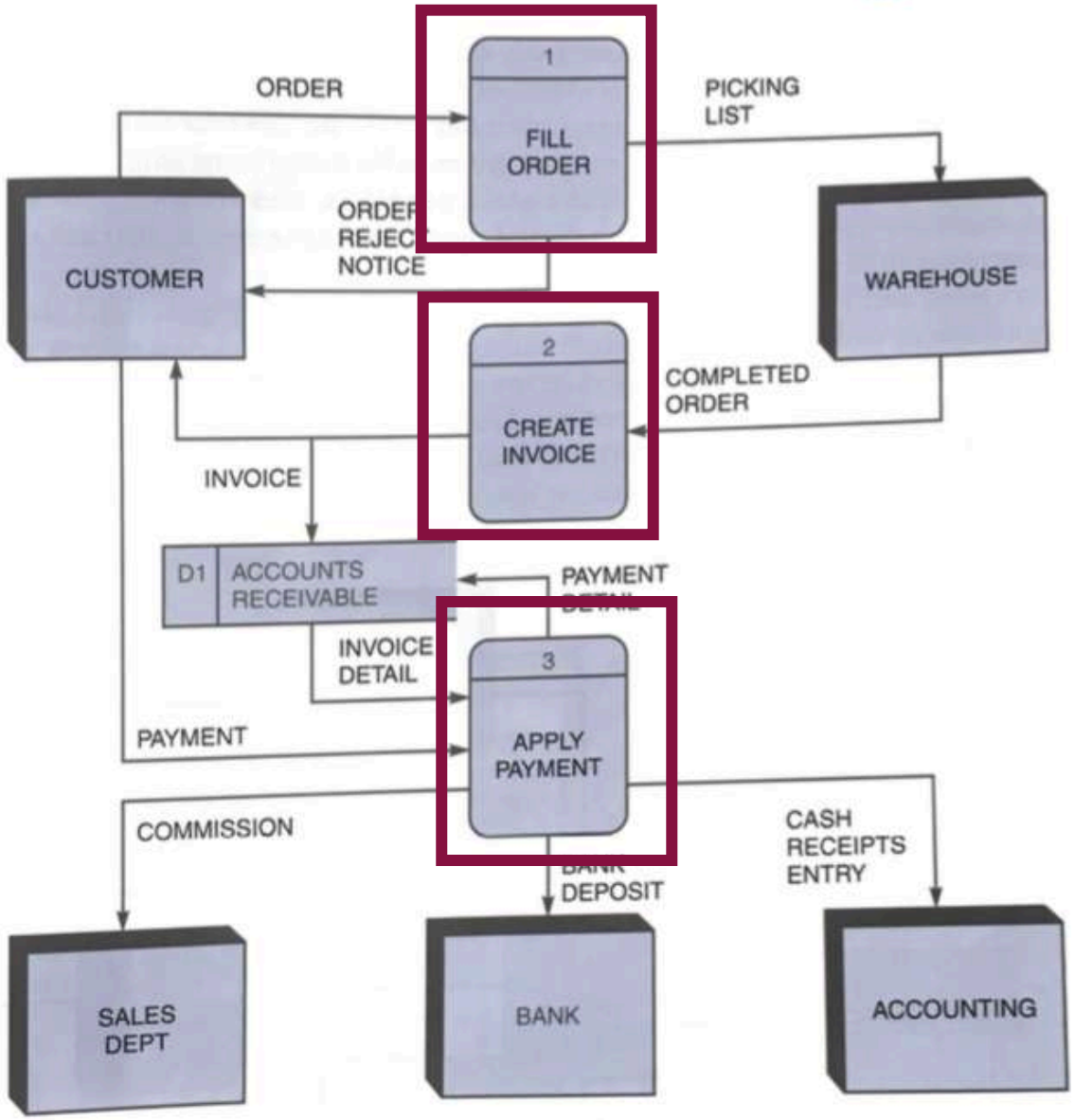
DFD – DATA FLOW

- Shows movement of data from one point to another
- Described with a noun
- Arrowhead indicates the flow direction
- Represents data about a person, place, or thing



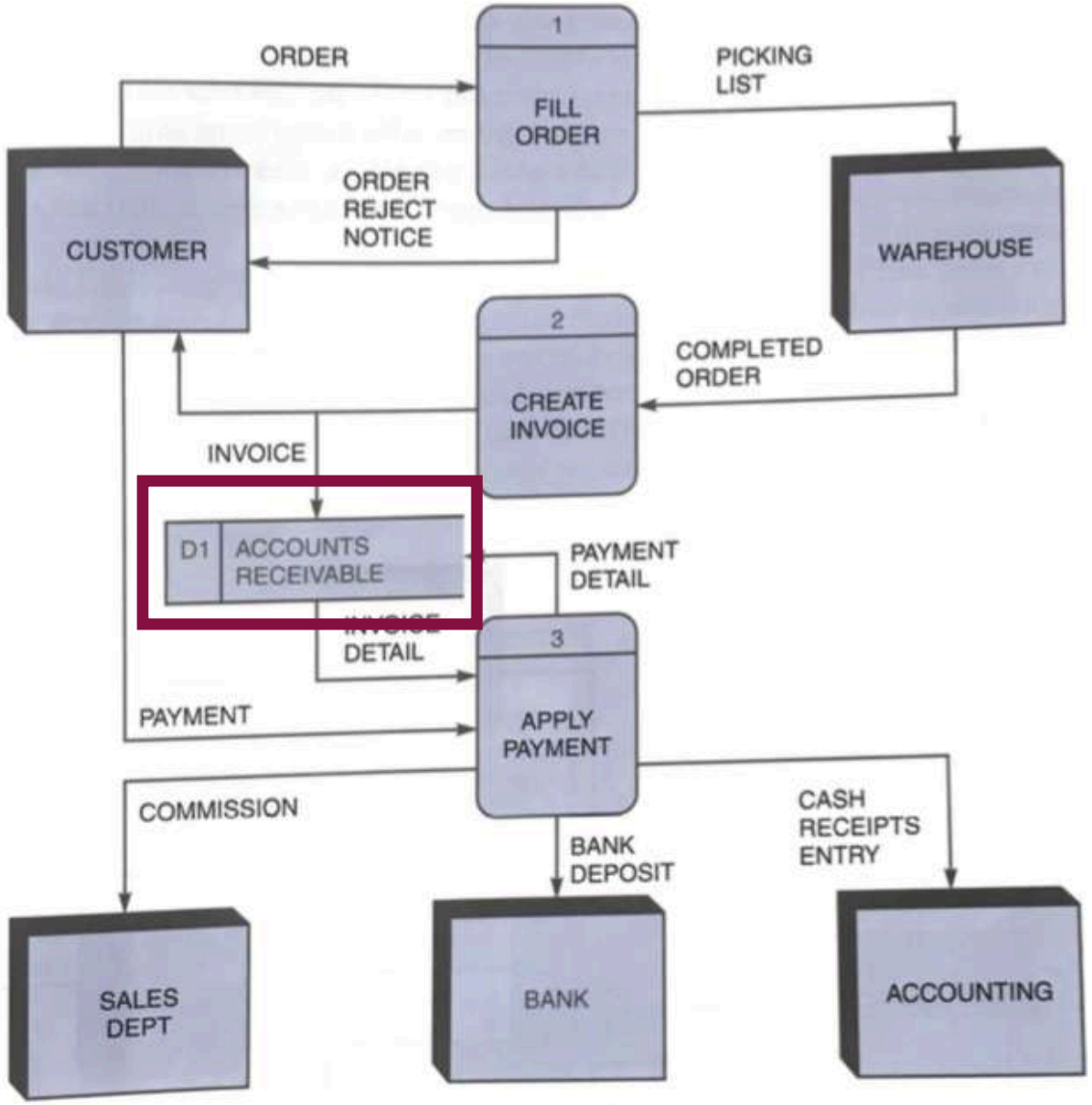
DFD – PROCESS

- Denotes a change in or transformation of data
- Represents work being performed in the system
- Naming convention:
  - Assign the name of the whole system when naming a high-level process (refer previous diagram)
  - To name a major subsystem attach the word subsystem to the name
  - Use the form verb-adjective-noun for detailed processes



DFD – DATA STORE

- A depository for data that allows examination, addition, and retrieval of data
- Named with a noun, describing the data
- Data stores are usually given a unique reference number, such as D1, D2, D3
- Represents a:
  - Database
  - Computerized file
  - Filing cabinet



MAJOR TOPICS

CREATING DFD

- Steps to create DFD
- Context Diagram
- Diagram 0
- DFD Level

STEPS TO DEVELOP DFD

Developing Data Flow Diagrams
Using a Top-Down Approach

1. Make a list of business activities and use it to determine various
   - External entities
   - Data flows
   - Processes
   - Data stores
2. Create a context diagram that shows external entities and data flows to and from the system. Do not show any detailed processes or data stores.
3. Draw Diagram 0, the next level. Show processes, but keep them general. Show data stores at this level.
4. Create a child diagram for each of the processes in Diagram 0.
5. Check for errors and make sure the labels you assign to each process and data flow are meaningful.
6. Develop a physical data flow diagram from the logical data flow diagram. Distinguish between manual and automated processes, describe actual files and reports by name, and add controls to indicate when processes are complete or errors occur.
7. Partition the physical data flow diagram by separating or grouping parts of the diagram in order to facilitate programming and implementation.

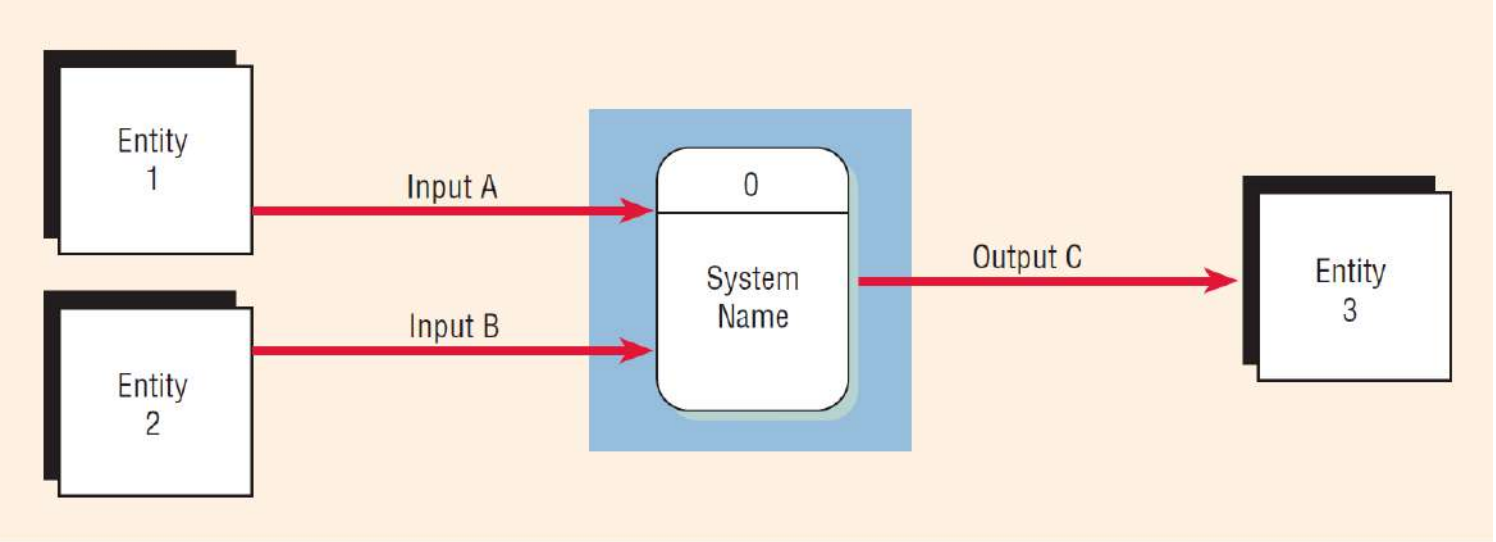
BASIC RULES

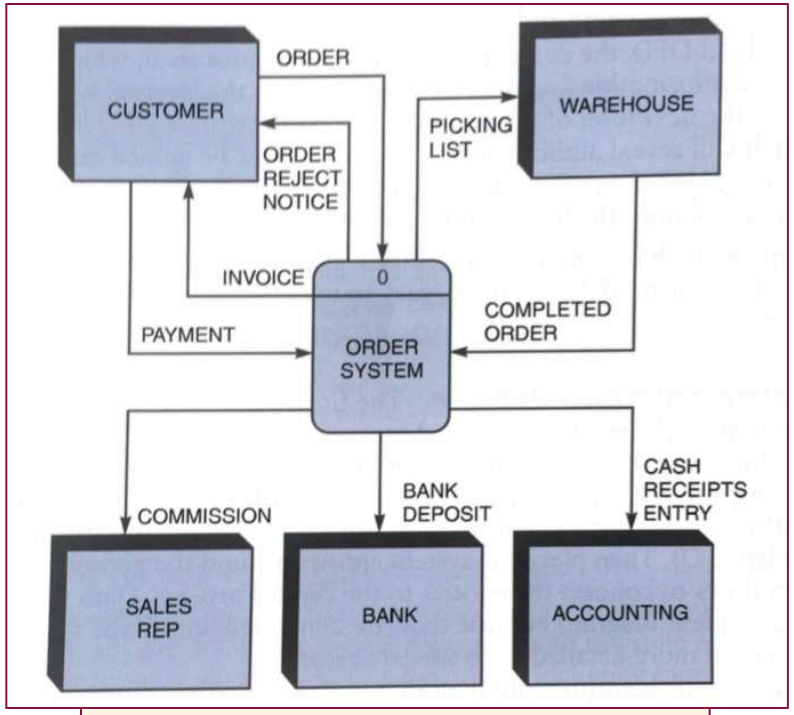
- The data flow diagram must have at least one process
- Must not be any freestanding objects
- A process must have both an input and output data flow
- A data store must be connected to at least one process
- External entities should not be connected to one another

CREATING CONTEXT DIAGRAM

- The highest level in a data flow diagram
- Contains only one process, representing the entire system
- The process is given the number 0
- All external entities, as well as major data flows are shown
- Basically the context diagram consists of:
  - one process—depicting the entire system
  - external entities
  - data flows from the external entities to the process
- The diagram does not contain any data stores.

CREATING CONTEXT DIAGRAM



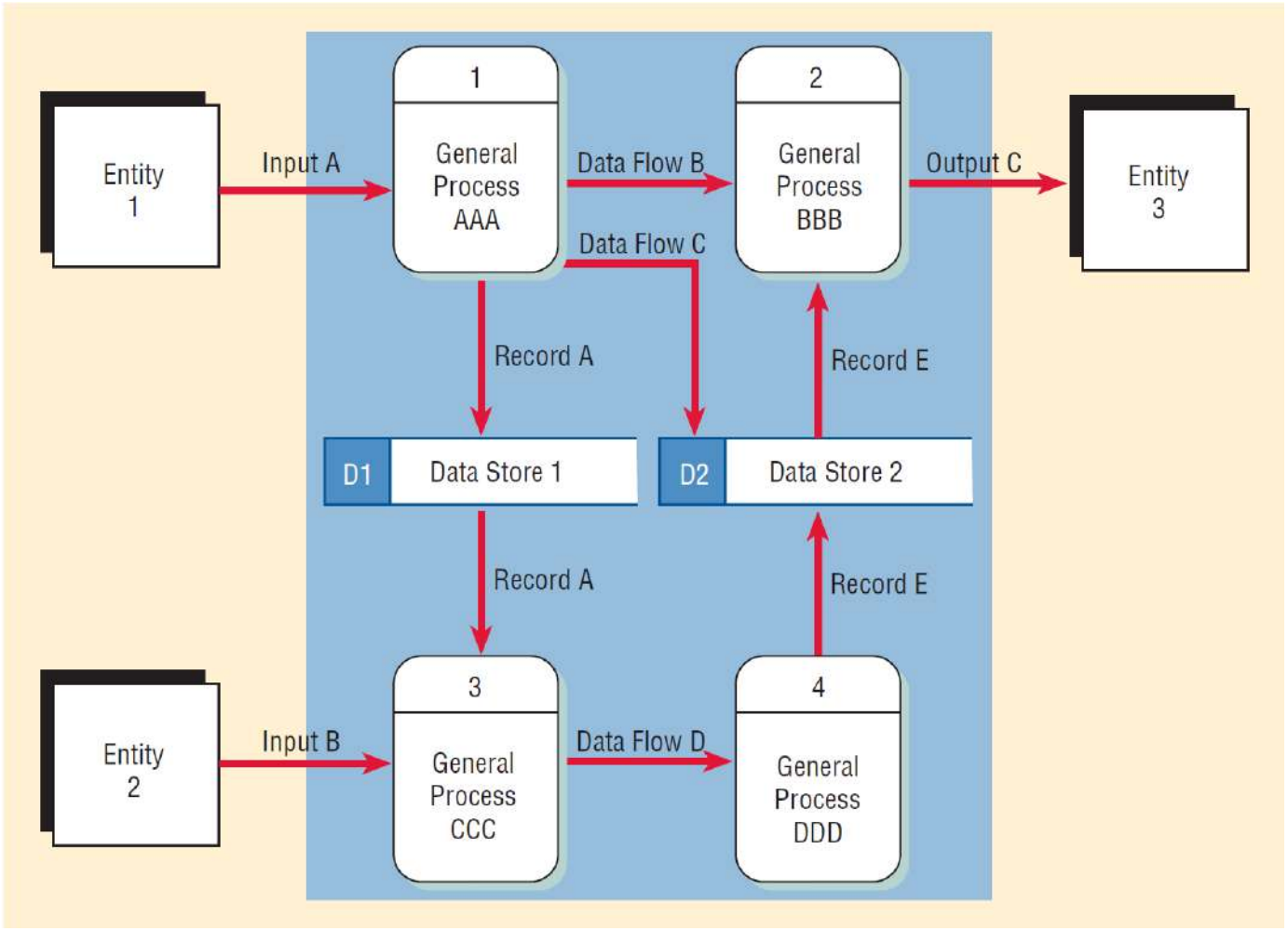


CONTEXT DIAGRAM EXAMPLE

DRAWING DIAGRAM 0

- The explosion of the context diagram
- May include up to nine processes
- Each process is numbered
- Major data stores and all external entities are included
- Start with the data flow from an entity on the input side
- Work backward from an output data flow
- Examine the data flow to or from a data store
- Analyze a well-defined process
- Take note of any fuzzy areas

DRAWING DIAGRAM 0



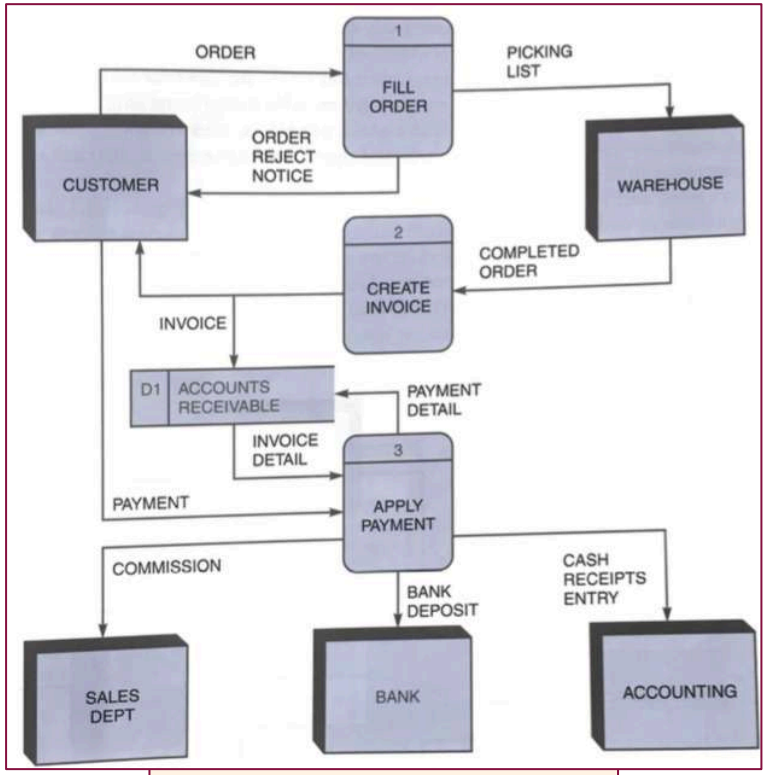


DIAGRAM 0 EXAMPLE

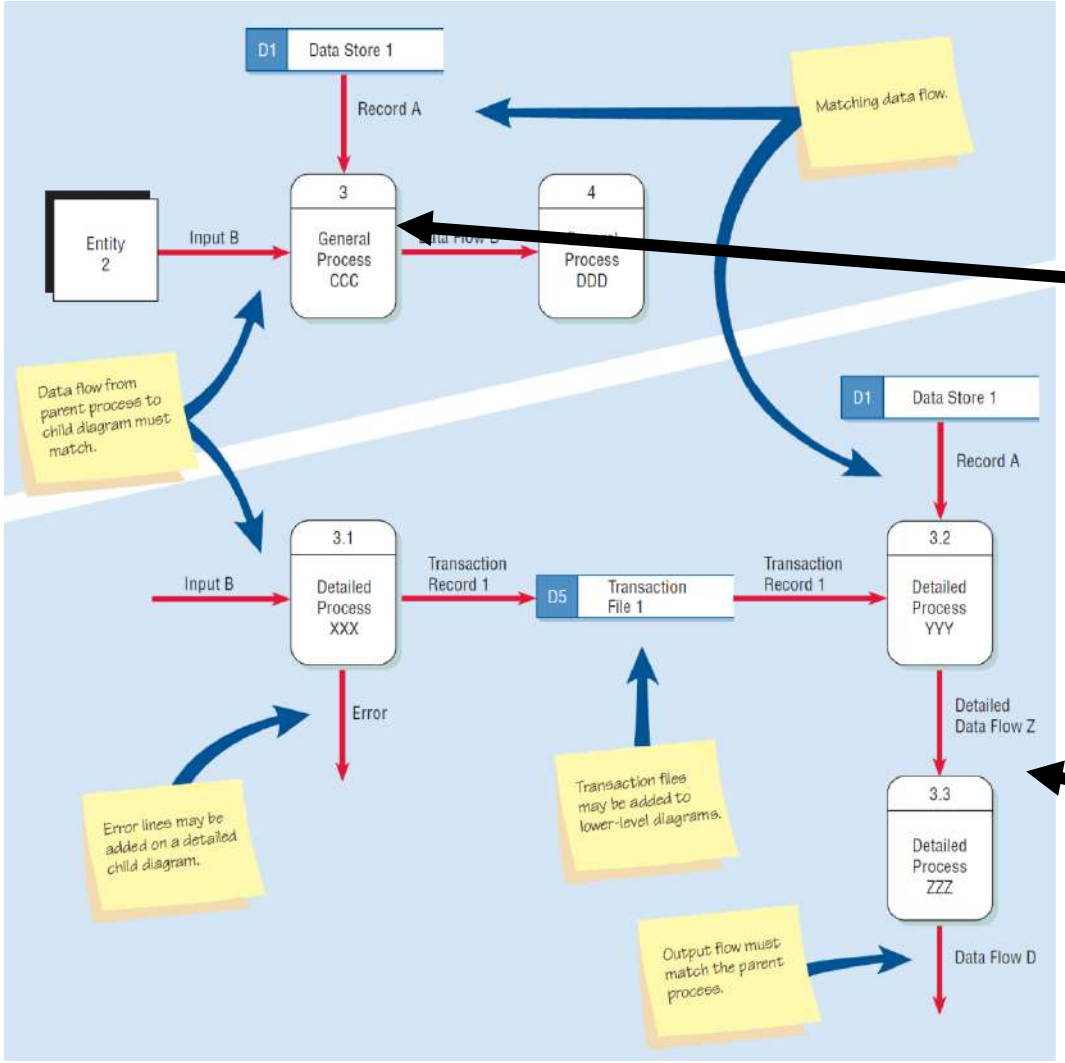
DFD LEVELS

- Data flow diagrams are built in layers
- The top level is the context level
- Each process may explode to a lower level
- The lower level diagram number is the same as the parent process number
- Processes that do not create a child diagram are called primitive

CREATING CHILD DIAGRAM

- Each process on diagram 0 may be exploded to create a child diagram
- A child diagram cannot produce output or receive input that the parent process does not also produce or receive
- The child process is given the same number as the parent process
  - Process 3 would explode to Diagram 3
- Entities are usually not shown on the child diagrams below Diagram 0
- If the parent process has data flow connecting to a data store, the child diagram may include the data store as well
- When a process is not exploded, it is called a primitive process

CREATING CHILD DIAGRAM



Parent Diagram vs. Child Diagram

Parent Diagram

Child Diagram

CREATING CHILD DIAGRAM - example

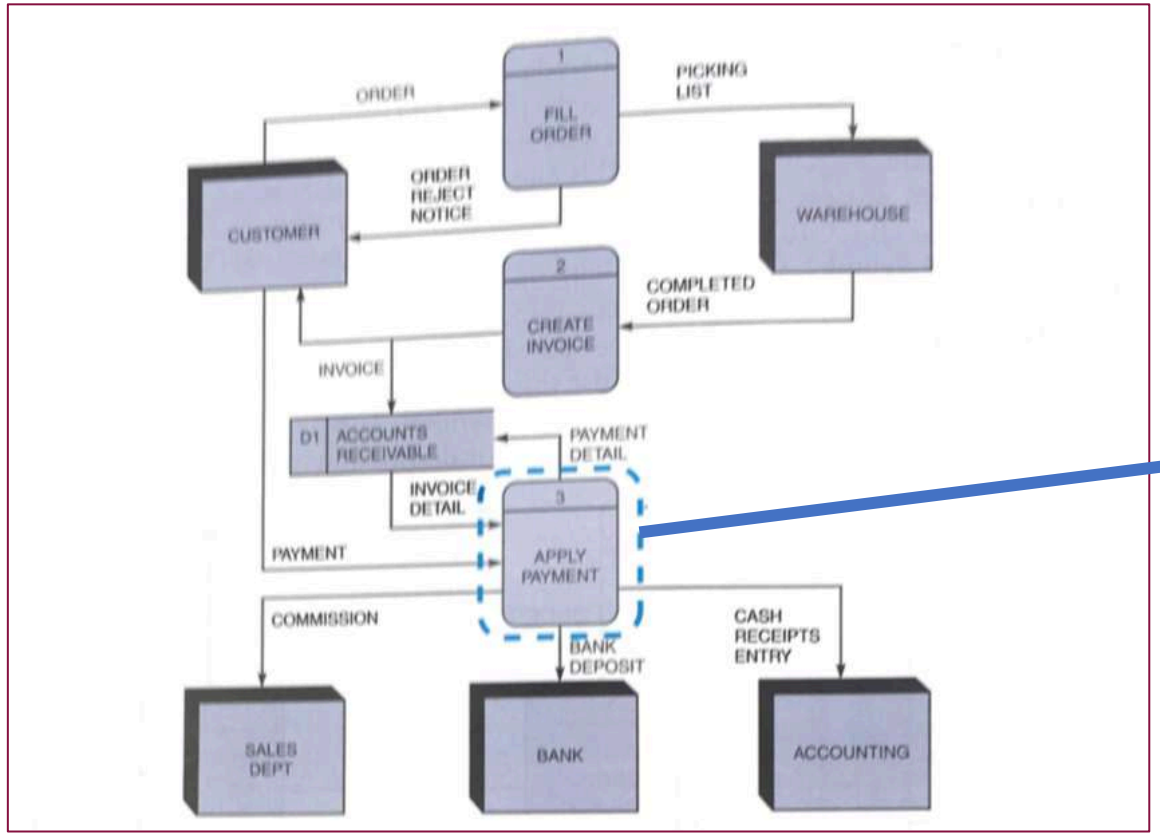
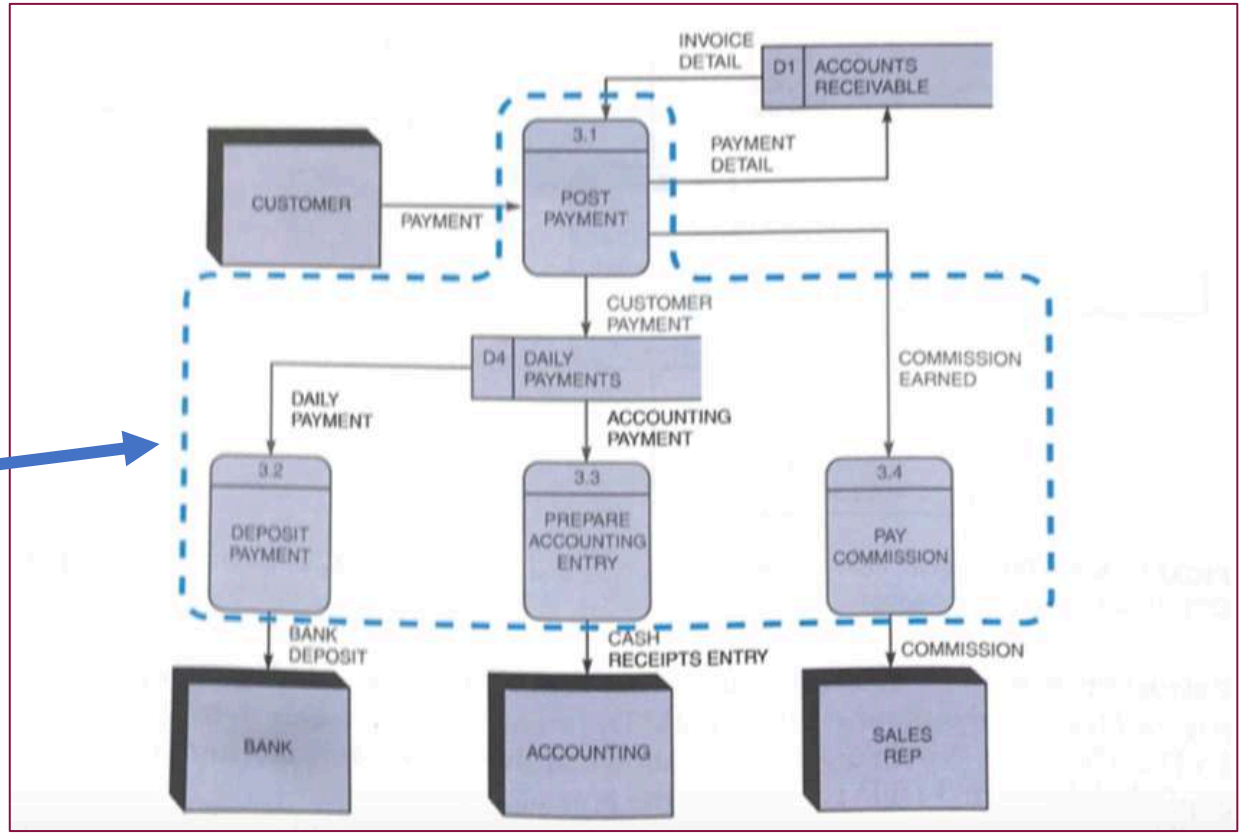


DIAGRAM 0 for ORDER SYSTEM
(Parent Diagram)



LEVEL 1 DIAGRAM for Process 3 APPLY PAYMENT
(Child Diagram)

LET'S TRY

Checkpoint 1: DFD Case Study – Ria Pizza's Ordering System

Ria Pizza is a small restaurant in UTM and most of its customers are the UTM students. In the current manual ordering system, the restaurant employees have to go through three different activities in order to process a pizza order. The first activity, called Receiving the order, consists in getting customers' information (such as name, phone number, and address), and getting orders' information (such as the pizza size, the type of crust, and the ingredients needed to make the pizza). The same process also checks the availability of the ingredients needed before setting the status of the pizza order as "valid". When a pizza order is valid, the employee in charge of Receiving the order provides the valid order information to the cooks who make the pizza. The second activity, Making the pizza, consists in getting the ingredients needed (from containers in the kitchen) and actually cooking the pizza. At the end of that process, the cooks have to update the manual file that keeps track of the ingredients. They, then, provide the drivers in charge of delivering the pizza with the (completed) status of the order. The final activity, Delivering the pizza, done by the drivers consists in getting address and payment information from the employee in charge of Receiving the order, and actually delivering the pizza. The drivers provide the customers with a receipt and get the payment.

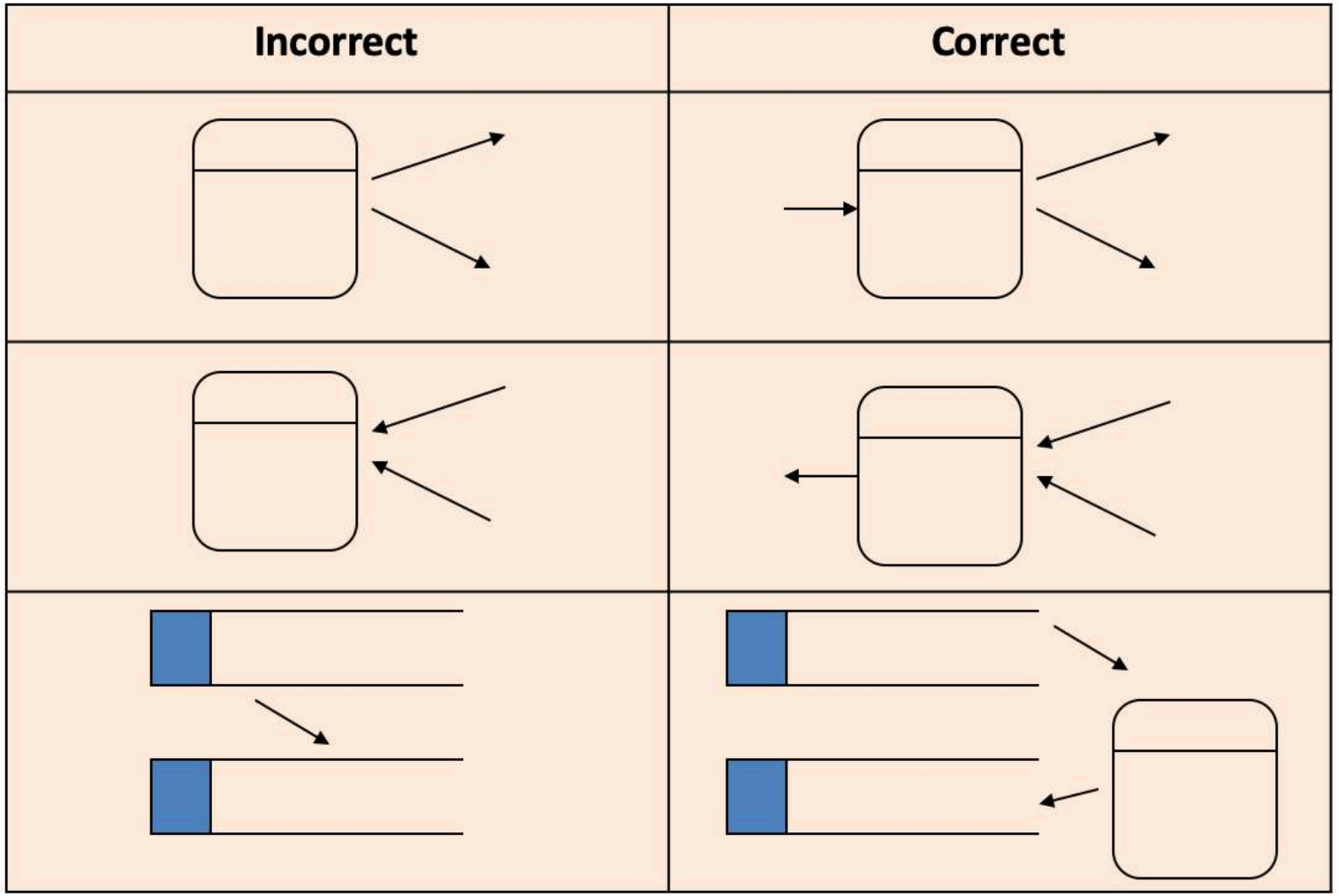
TASKS

1. Name all the processes to be found in the Data Flow Diagram of the system.
2. Name all the external entities to be found in the Data Flow Diagram of the system.
3. Name all the data stores to be found in the Data Flow Diagram of the system.
4. Draw the Data Flow Diagram (Context Diagram, Diagram 0 and/or Child Diagrams) for the new Lincoln Pizza's ordering system.

DATA FLOW DIAGRAM ERRORS

- ✧Forgetting to include a data flow or pointing an arrow in the wrong direction
- ✧Connecting data stores and external entities directly to each other
- ✧Incorrectly labeling processes or data flow
- ✧Including more than nine processes on a data flow diagram
- ✧Omitting data flow
- ✧Creating unbalanced decomposition (or explosion) in child diagrams

DATA FLOW DIAGRAM ERRORS

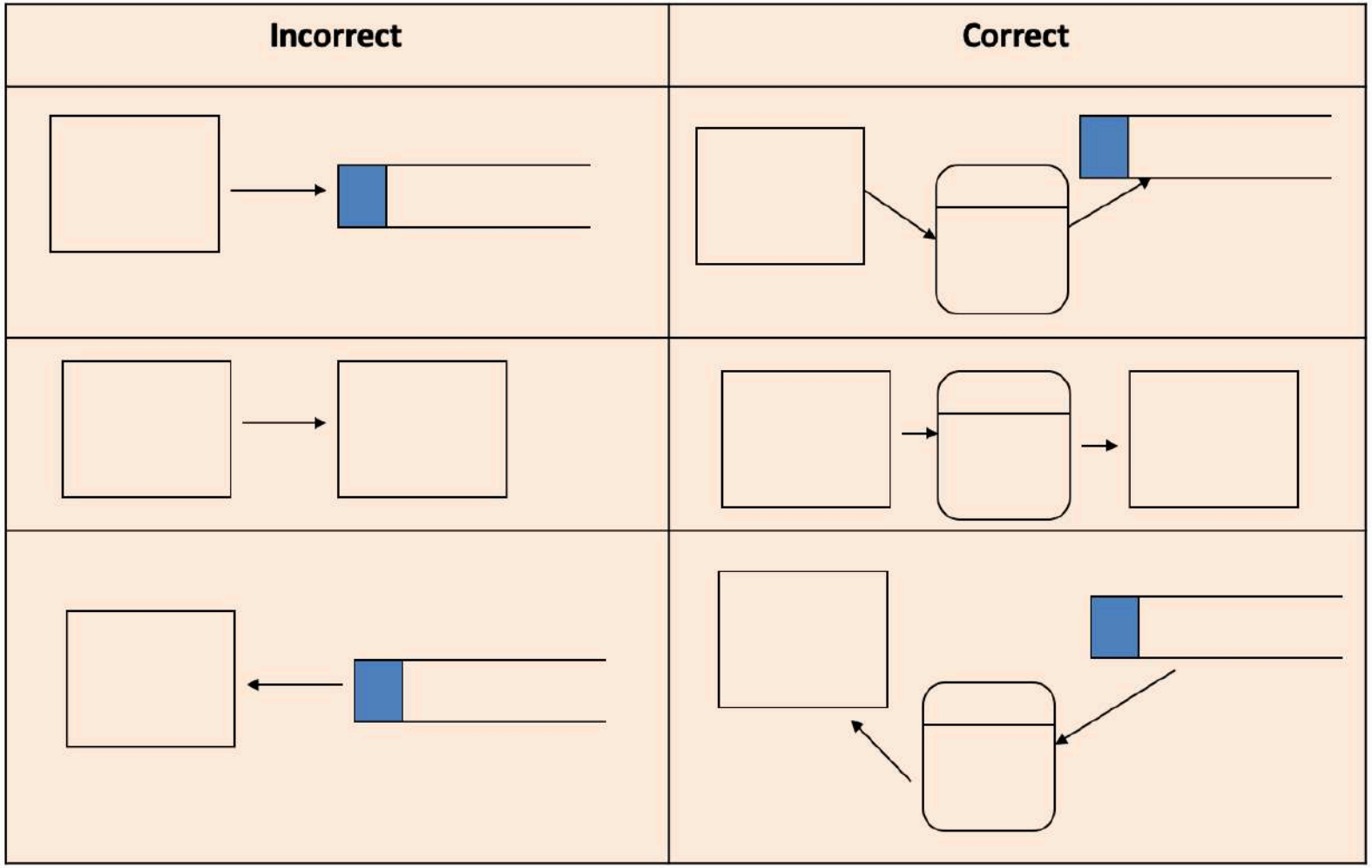


No process can have only output (**MIRACLE**)

No process can have only input (**BLACK HOLE**)

Data cannot move directly from one data store to another data store – must be moved by a process that receives (input) and place it into another data store (output)

DATA FLOW DIAGRAM ERRORS

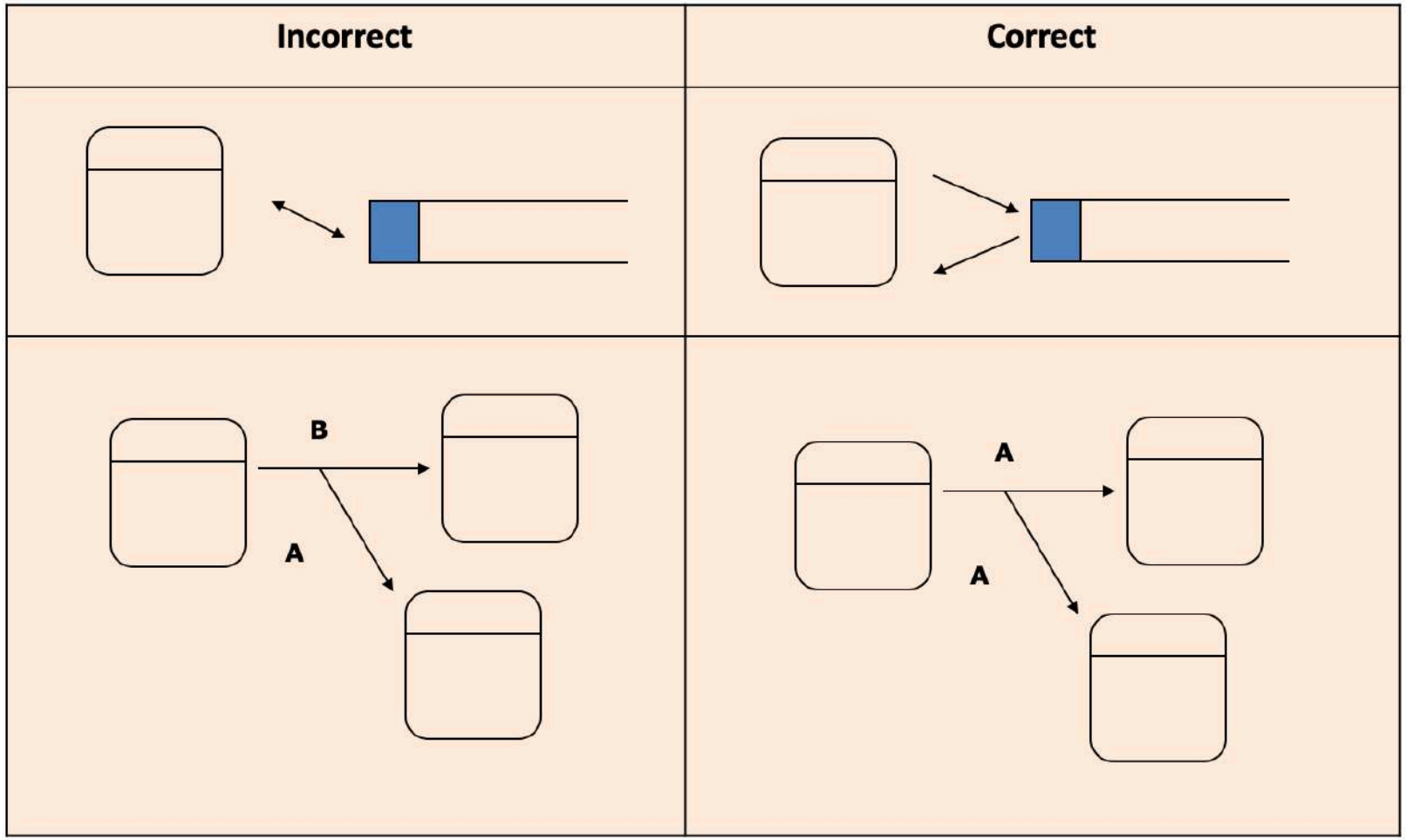


Data cannot move directly from an external entity to a data store – must be moved by a process that receives (input)from entity and place it into the data store (output)

Data cannot move directly from one entity to another entity – must be moved by a process that receives (input entity) and place it into another entity (output)

Data cannot move directly from a data store to an external entity – must be moved by a process that receives (input)from data store and place it into the entity (output)

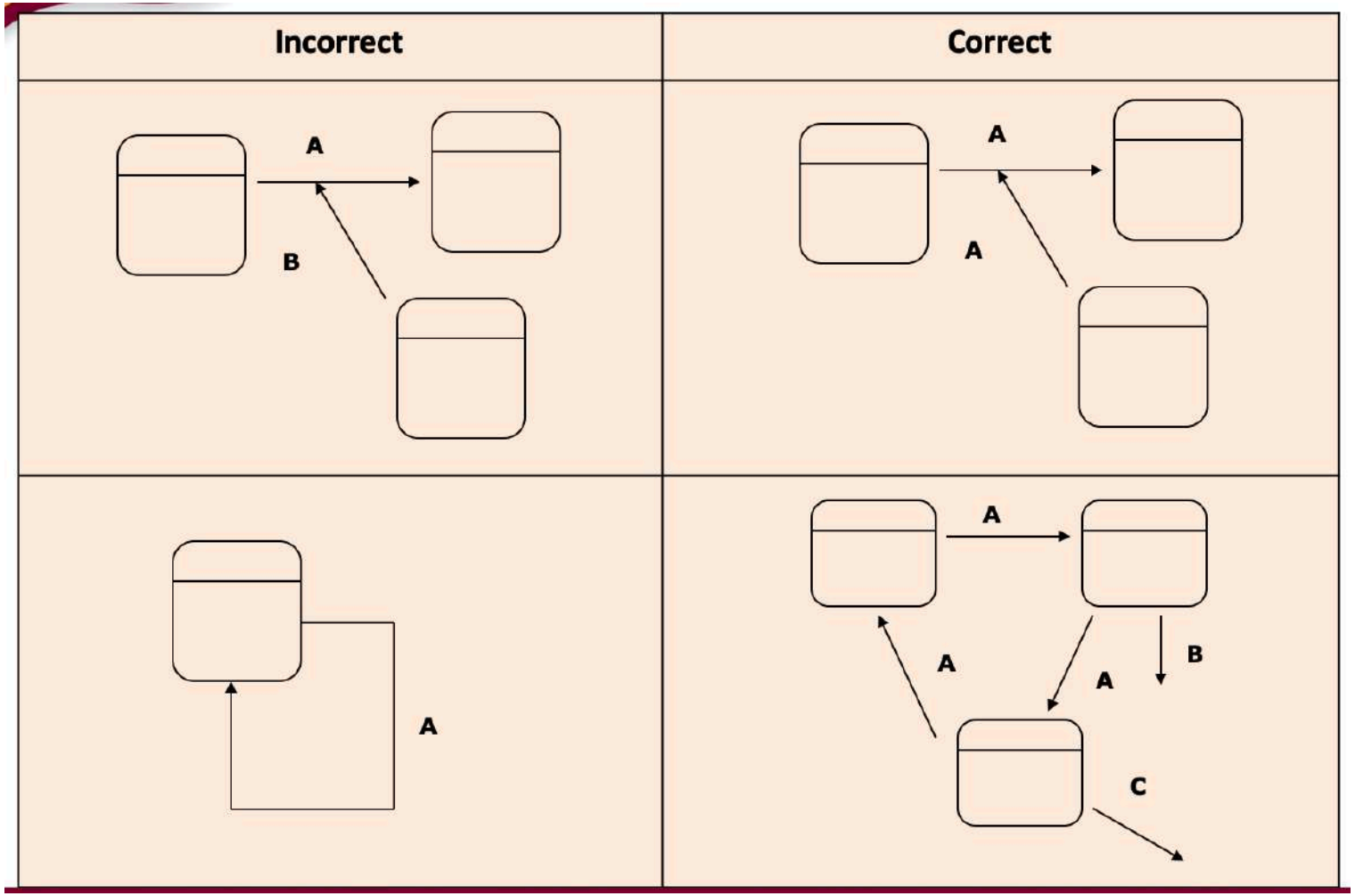
DATA FLOW DIAGRAM ERRORS



A data flow has only one direction between symbols. It may flow in both direction between process and a data store to show a READ before an UPDATE, thus two separate arrow is needed to show that the data flow occur at different time.

A **FORK** in a data flow means that exactly the same data goes from a common location to two or more different processes, data stores or sources. This indicate different copies of the same data goes to different places.

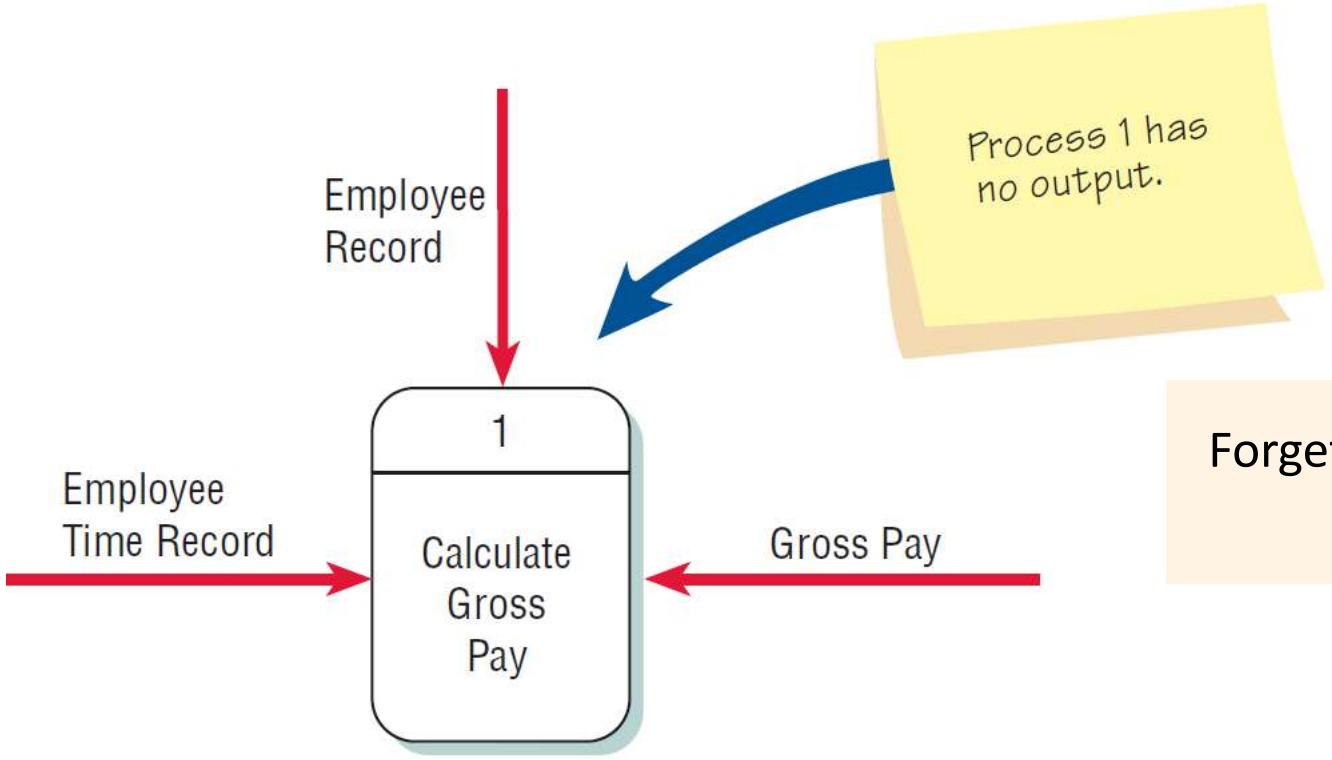
DATA FLOW DIAGRAM ERRORS



A **JOIN** in a data flow means that exactly the same data come from any of two or more different processes, data stores or sources.

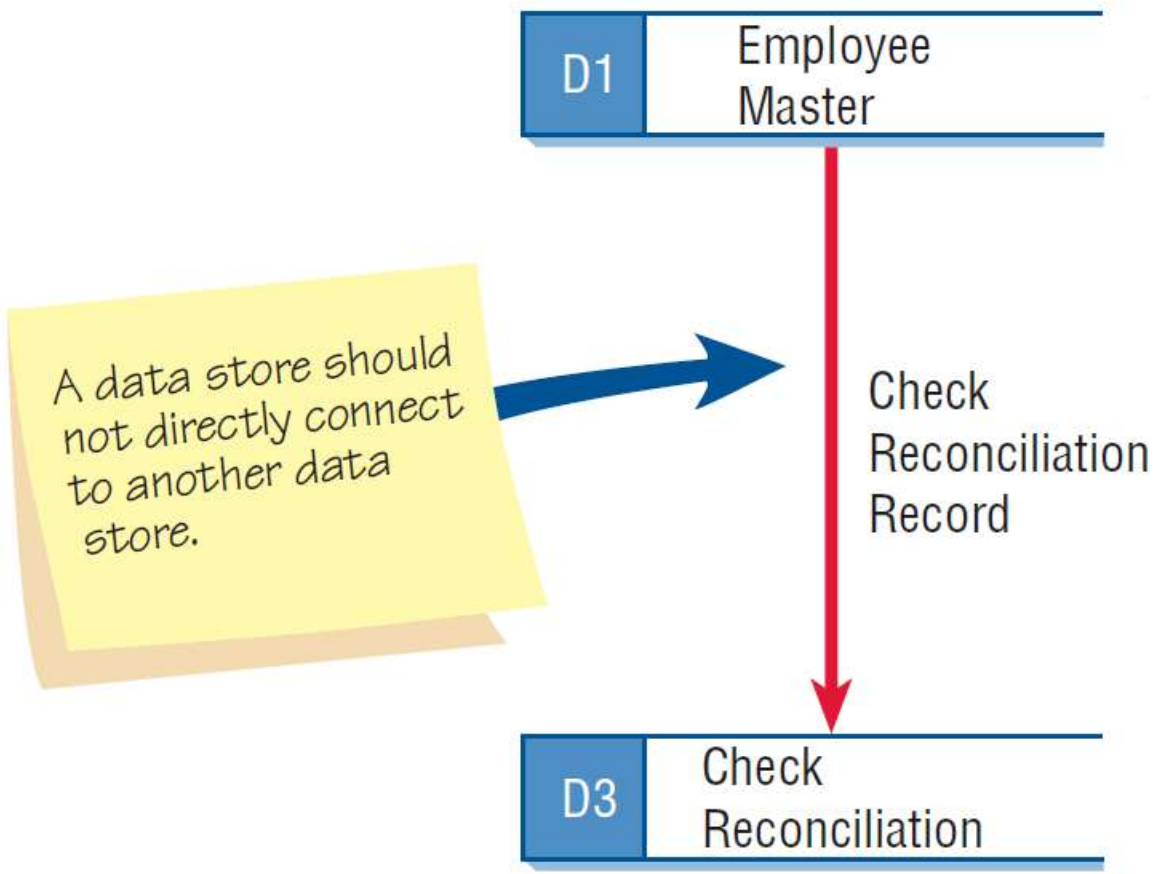
A data flow CANNOT go directly back to the same process it leaves. There MUST be at least one other process that handle the data flow, produces some other data flow and returns original data flow to the beginning process.

CHECKING THE DIAGRAMS FOR ERRORS – Common Mistakes



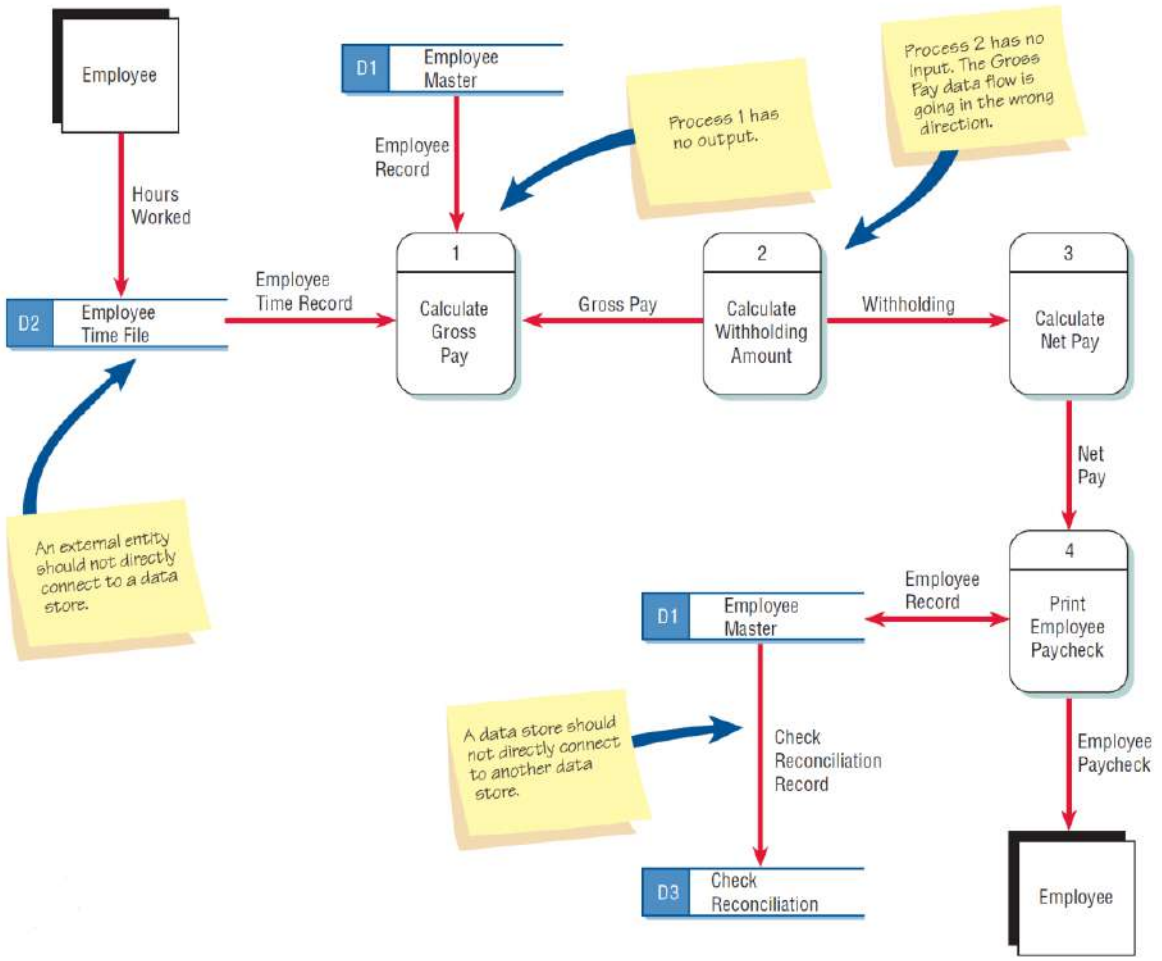
Forgetting to include a data flow or pointing an arrow in the wrong direction

CHECKING THE DIAGRAMS FOR ERRORS – Common Mistakes



Connecting data stores and external entities directly to each other

CHECKING THE DIAGRAMS FOR ERRORS – Common Mistakes



LET'S TRY

Checkpoint 2: DFD Error Questions (Midterm Test 2016/2017)

a) Consider the DFD in Figure 1. List and justify THREE (3) errors (rule violations) on this DFD.

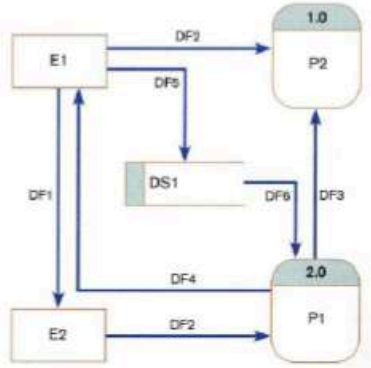


Figure 1: DFD for Question 8a

b) Consider the three DFDs in Figure 2. List and justify THREE (3) errors (rule violations) on these DFDs.

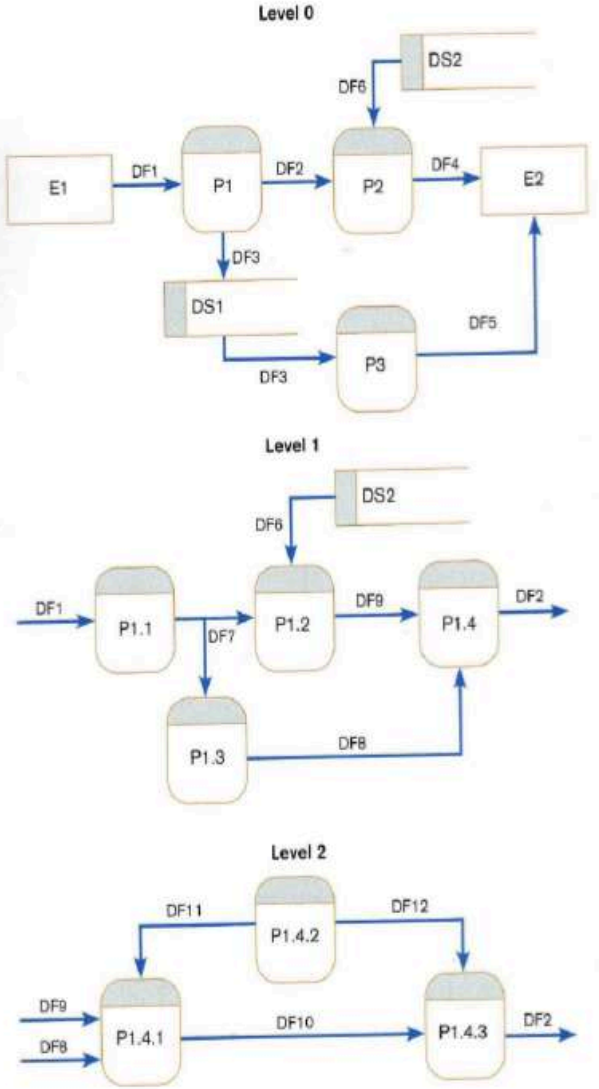


Figure 2: DFD for Question 8b

MAJOR TOPICS

LOGICAL & PHYSICAL DFD

- From logical to physical DFD

TYPES OF DFDs

- **Current** - how data flows now
- **Proposed** - how we'd like it to flow
- **Logical** - the "essence" of a process
- **Physical** - the implementation of a process
- **Partitioned physical** - system architecture or high-level design

LOGICAL AND PHYSICAL DFDs

WHAT = LOGICAL DFD

- Focuses on the business and how the business operates
- Not concerned with how the system will be constructed
- Describes the business events that take place and the data required and produced by each event

WHAT + HOW = PHYSICAL DFD

- Shows how the system will be implemented
- Depicts the system

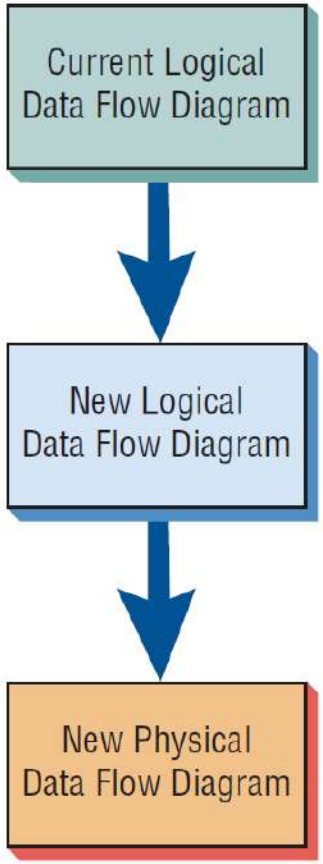
PHYSICAL DFDs

- Model the implementation of the system
- Start with a set of child diagrams or with Level 0 diagram
- Add implementation details
  - indicate manual vs. automated processes
  - describe form of data stores and data flows
  - extra processes for maintaining data

LOGICAL AND PHYSICAL DFDs – Common Features

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).

MODEL PROGRESSION FROM LOGICAL AND PHYSICAL DFDs

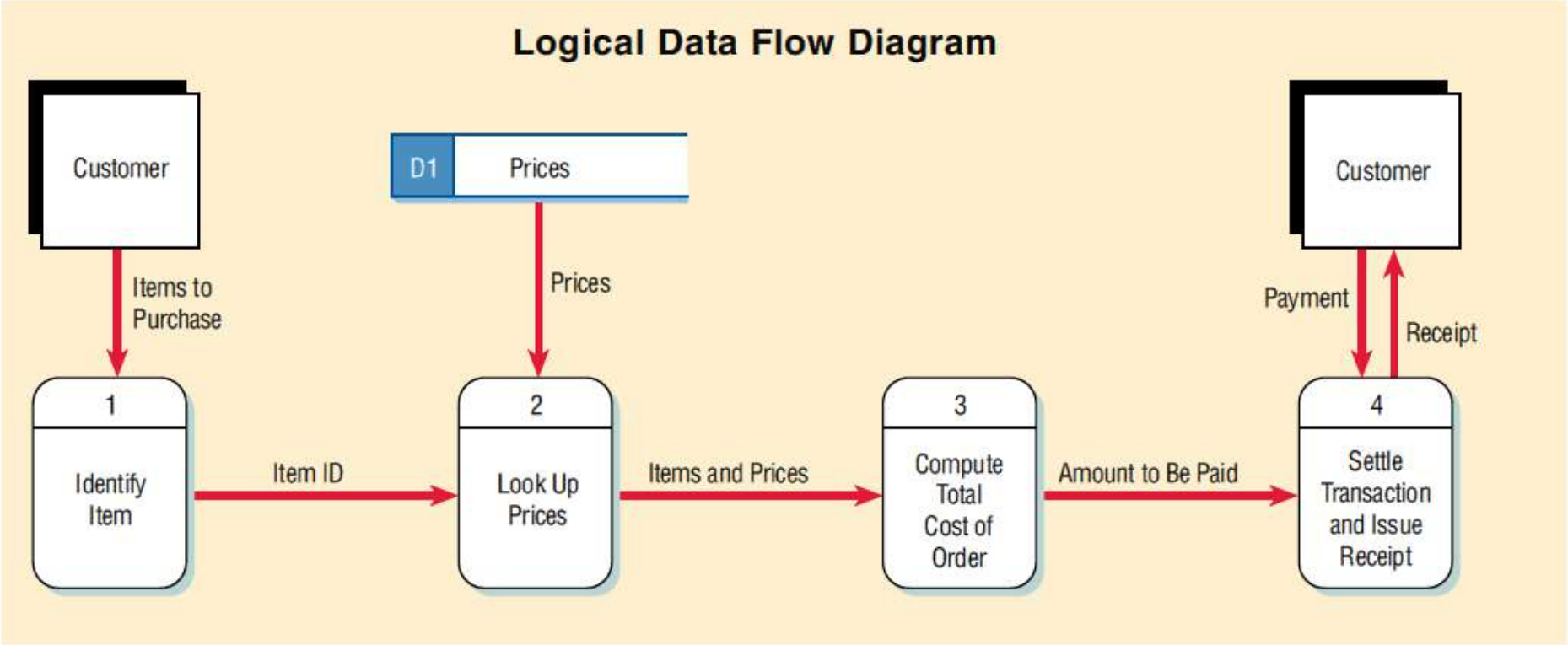


Derive the logical data flow diagram for the current system by examining the physical data flow diagram and isolating unique business activities.

Create the logical data flow diagram for the new system by adding the input, output, and processes required in the new system to the logical data flow diagram for the current system.

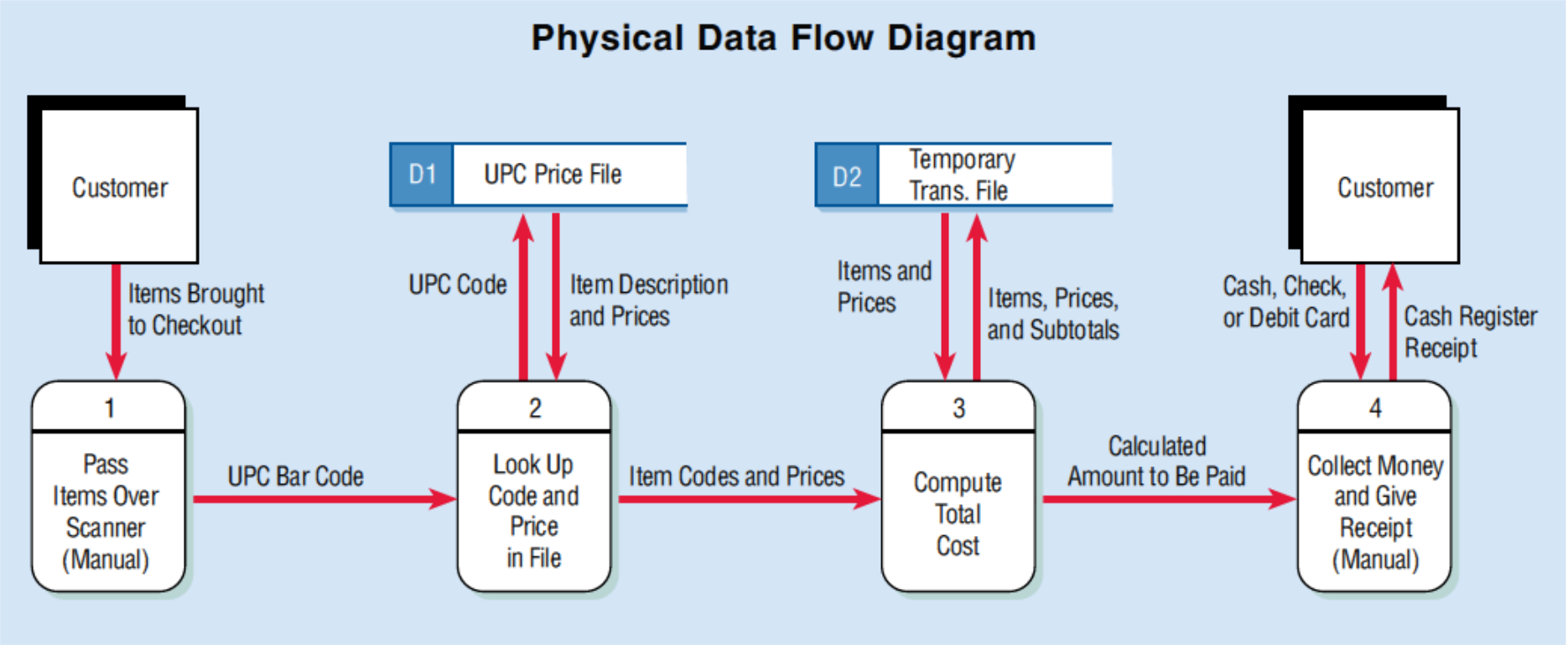
Derive the physical data flow diagram by examining processes on the new logical diagram. Determine where the user interfaces should exist, the nature of the processes, and necessary data stores.

LOGICAL DFD EXAMPLE



The logical DFD illustrates the processes involved without going into detail about the physical implementation of activities.

PHYSICAL DFD EXAMPLE



The physical DFD shows that processes involved with going into detail about the physical aspects of the activities

DEVELOPING LOGICAL AND PHYSICAL DFDs

LOGICAL DFD

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

PHYSICAL DFD

- Clarifying which processes are performed by humans and which are automated
- Describing processes in more detail
- Sequencing processes that have to be done in a particular order
- Identifying temporary data stores
- Specifying actual names of files and printouts
- Adding controls to ensure the processes are done properly

DEVELOPING LOGICAL AND PHYSICAL DFDs

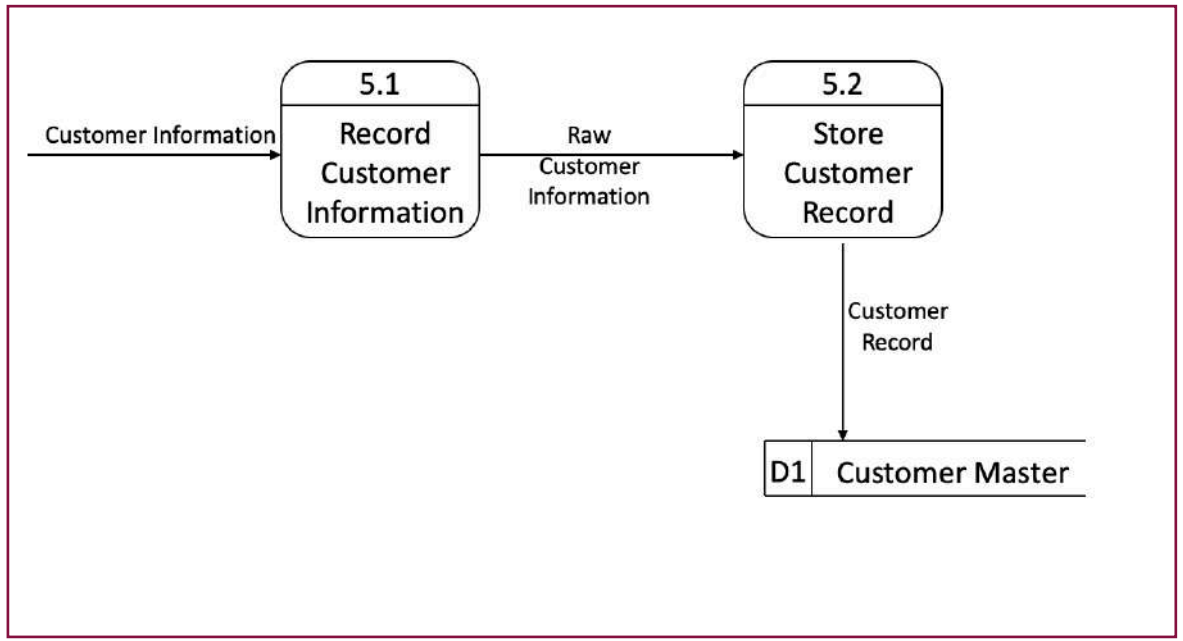
Physical DFD Contain Many Items Not Found in Logical DFD

Contents of Physical Data Flow Diagrams

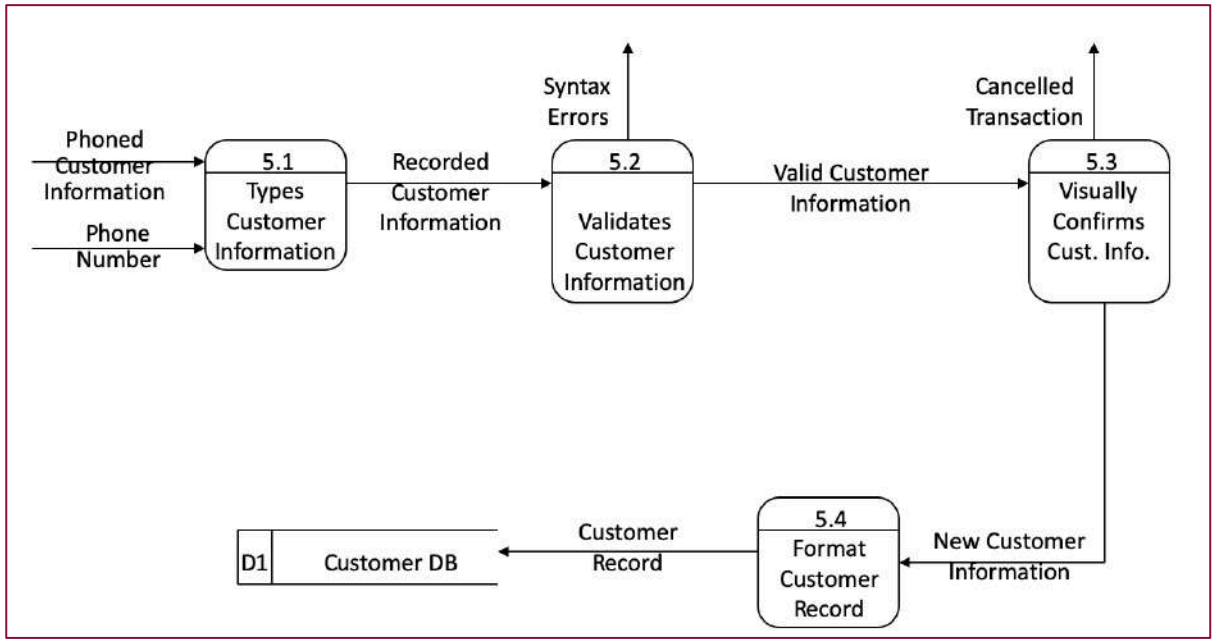
- Manual processes
- Processes for adding, deleting, changing, and updating records
- Data entry and verifying processes
- Validation processes for ensuring accurate data input
- Sequencing processes to rearrange the order of records
- Processes to produce every unique system output
- Intermediate data stores
- Actual file names used to store data
- Controls to signify completion of tasks or error conditions

DEVELOPING LOGICAL AND PHYSICAL DFDs: Example

Current Logical Child Diagram

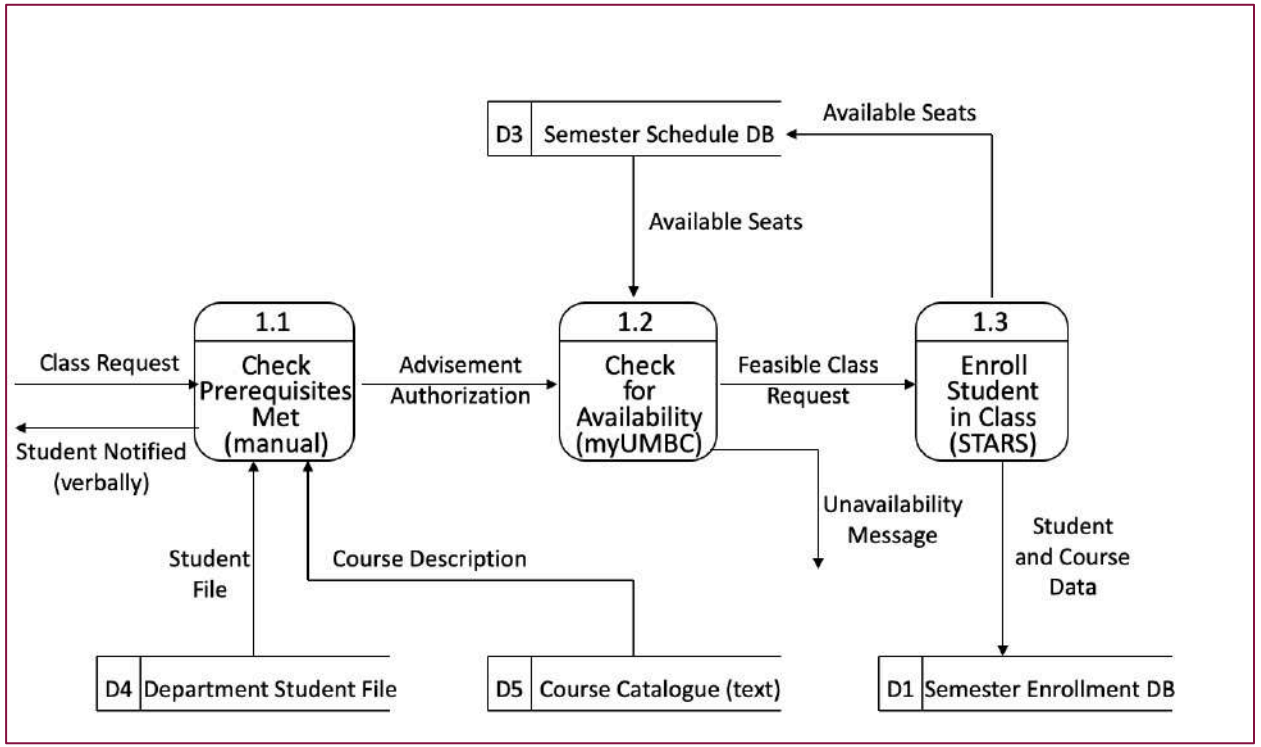


Physical Child Diagram

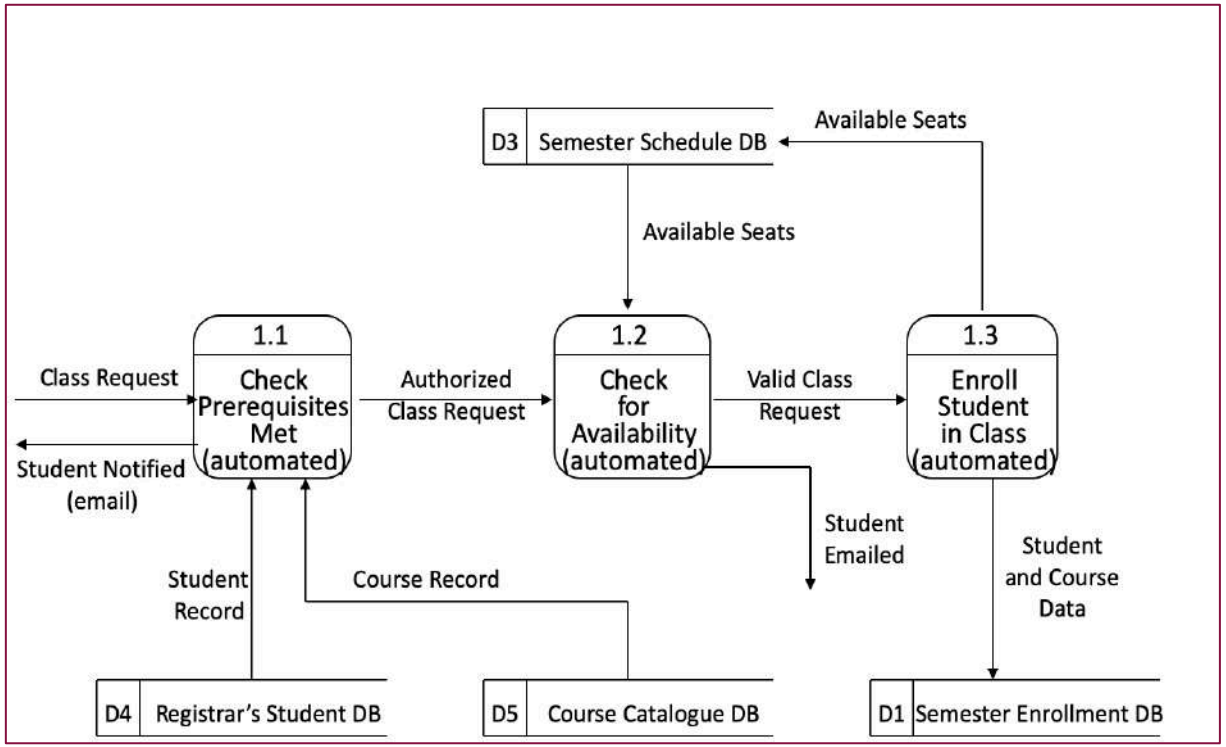


COMPARING CURRENT AND PROPOSED PHYSICAL DFDs: Example

Current Physical Child Diagram



Proposed Physical Child Diagram



MAJOR TOPICS

PARTITIONING

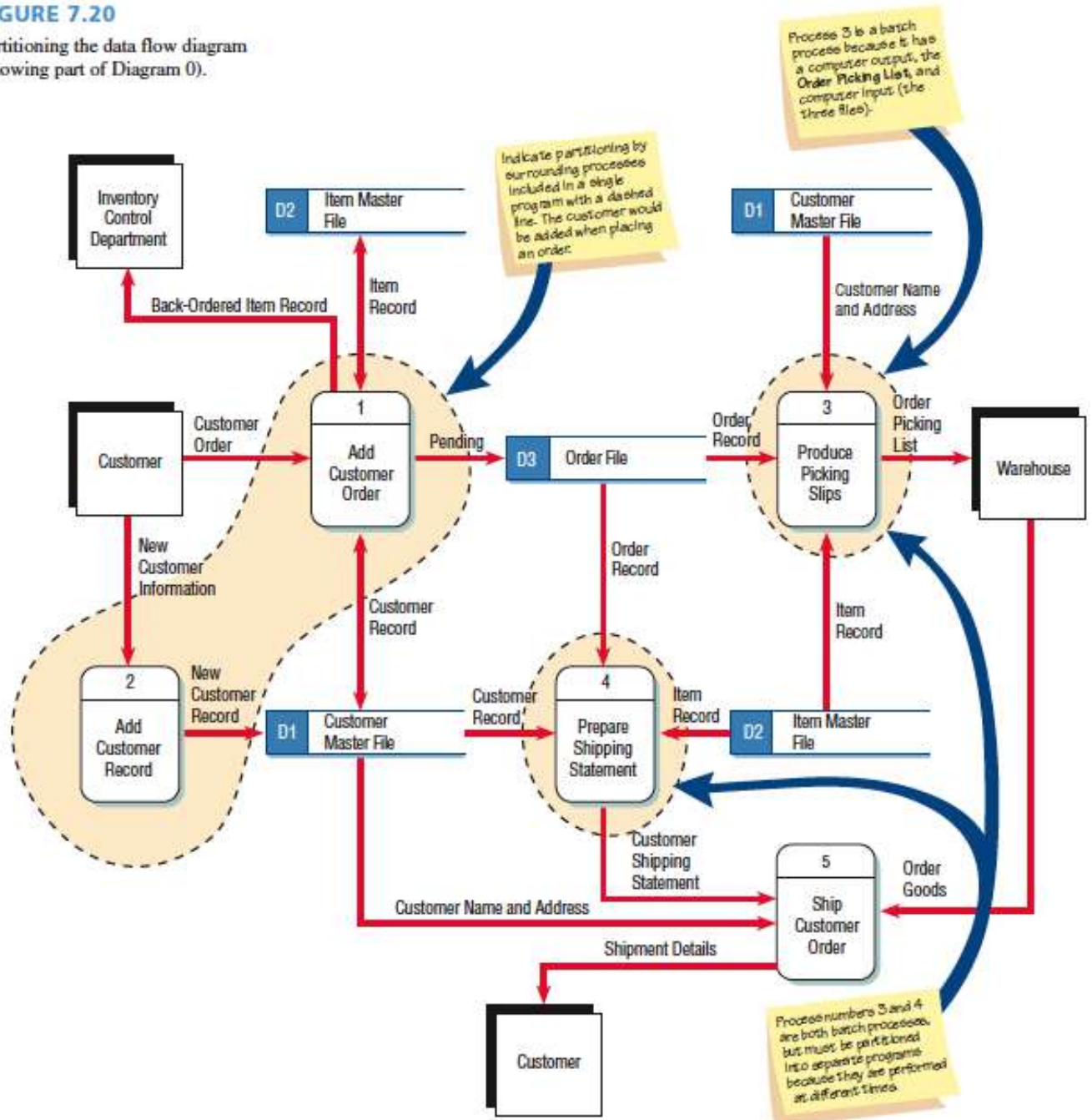
- Partitioning DFD
- CRUD matrix

PARTITIONING DATA FLOW DIAGRAM

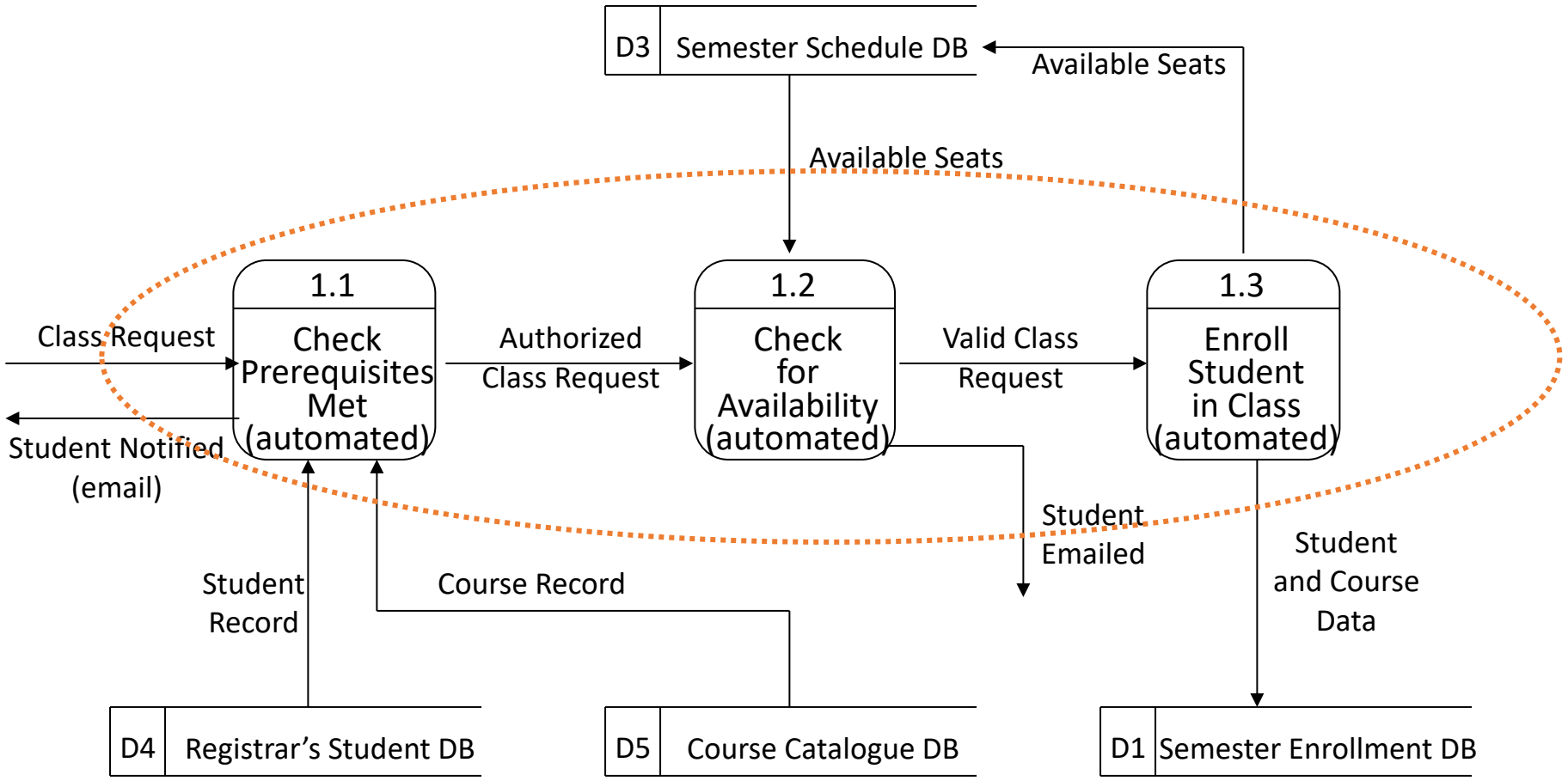
- Partitioning is the process of examining a data flow diagram and determining how it should be divided into collections of manual procedures and computer programs
- Decide what processes should be grouped together in the system components
- A dashed line is drawn around a process or group of processes that should be placed in a single computer program
- Reasons for partitioning
  - Different user groups
  - Timing
  - Similar tasks
  - Efficiency
  - Consistency of data
  - Security

FIGURE 7.20

Partitioning the data flow diagram (showing part of Diagram 0).



PARTITIONING DFD (EXAMPLE)



Course Registration: Physical diagram (partitioned)

PARTITIONING WEBSITES

- Improves the way humans use the site
- Improves speed of processing
- Ease of maintaining the site
- Keep the transaction secure

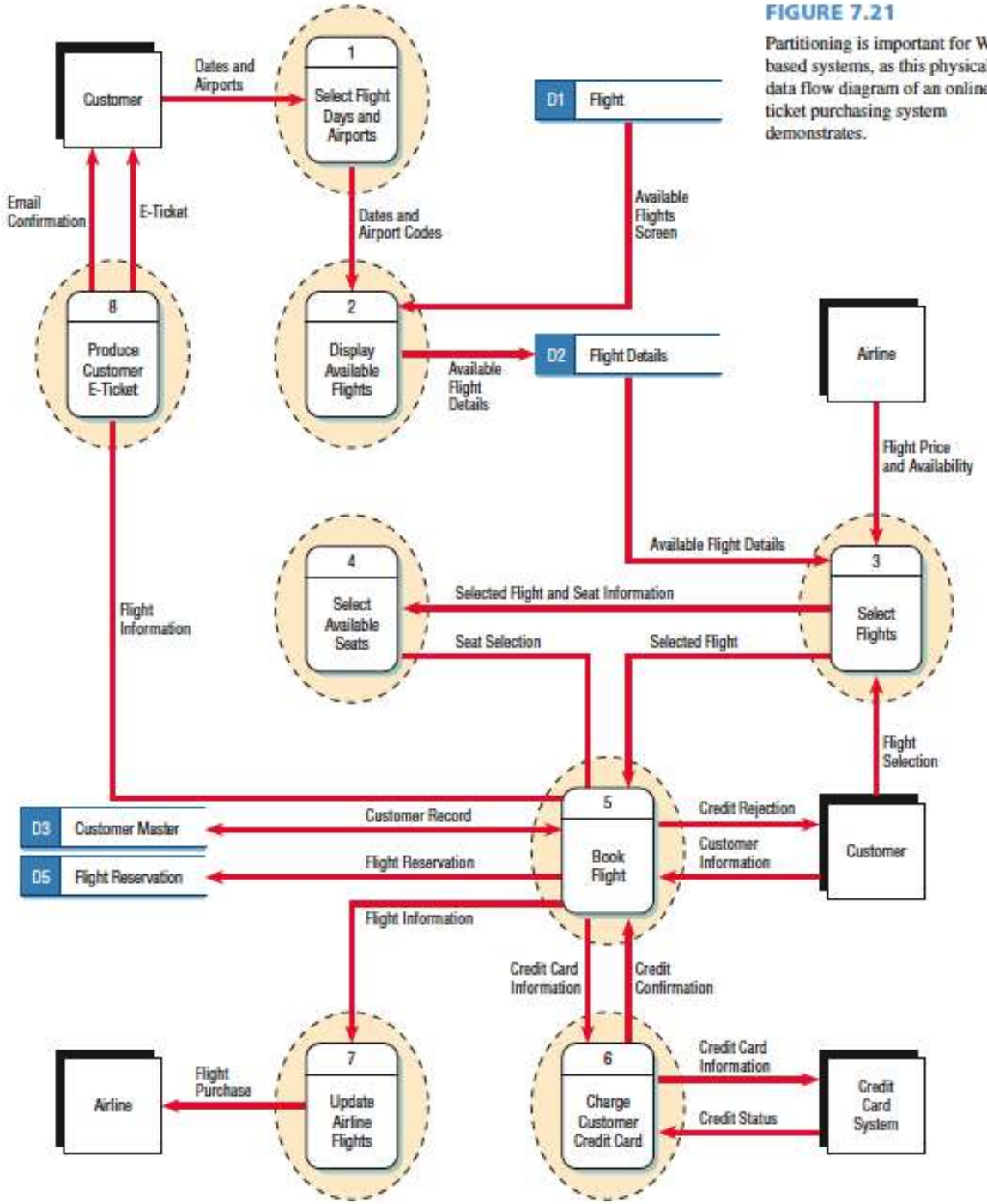
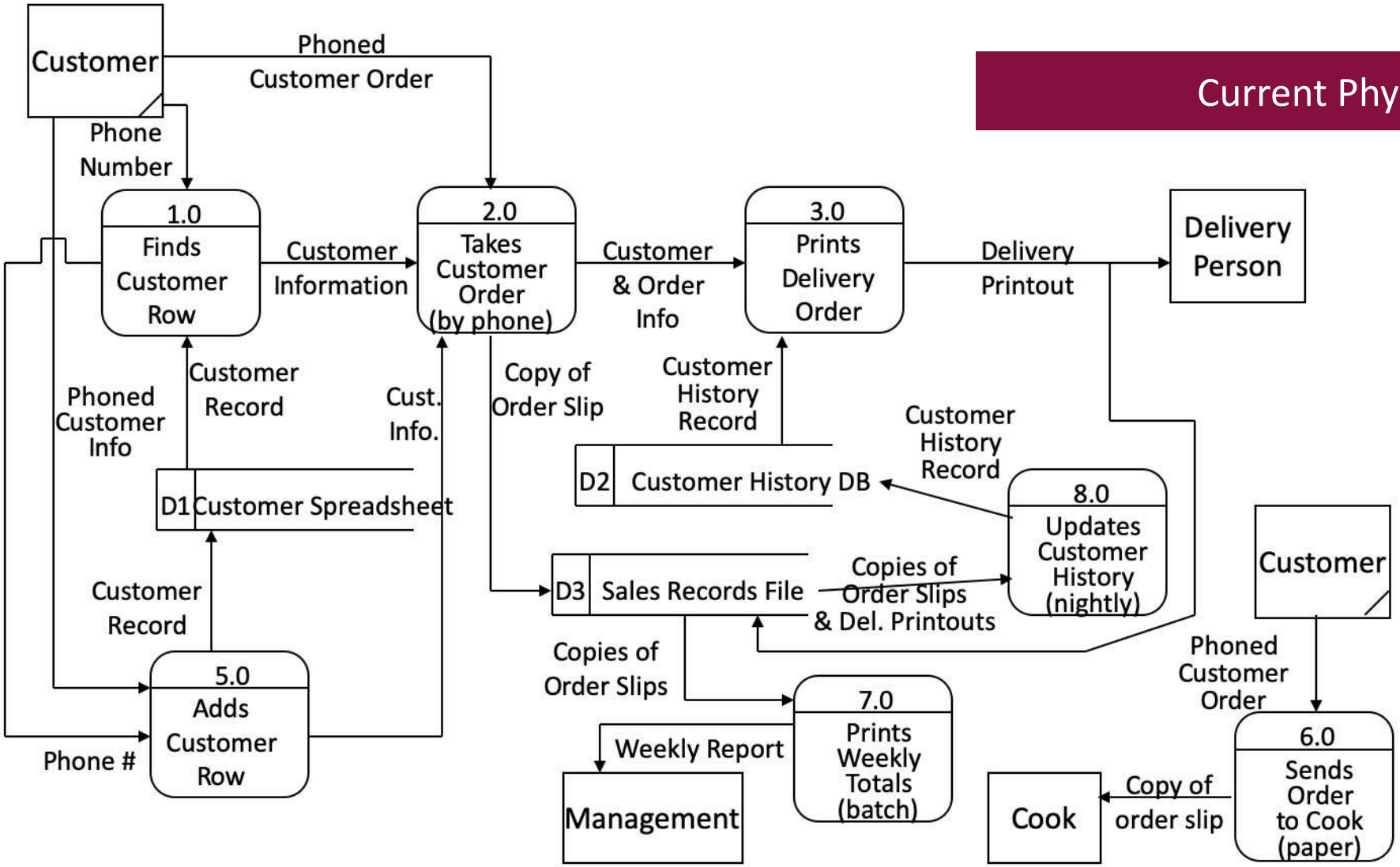


FIGURE 7.21

Partitioning is important for Web-based systems, as this physical data flow diagram of an online ticket purchasing system demonstrates.

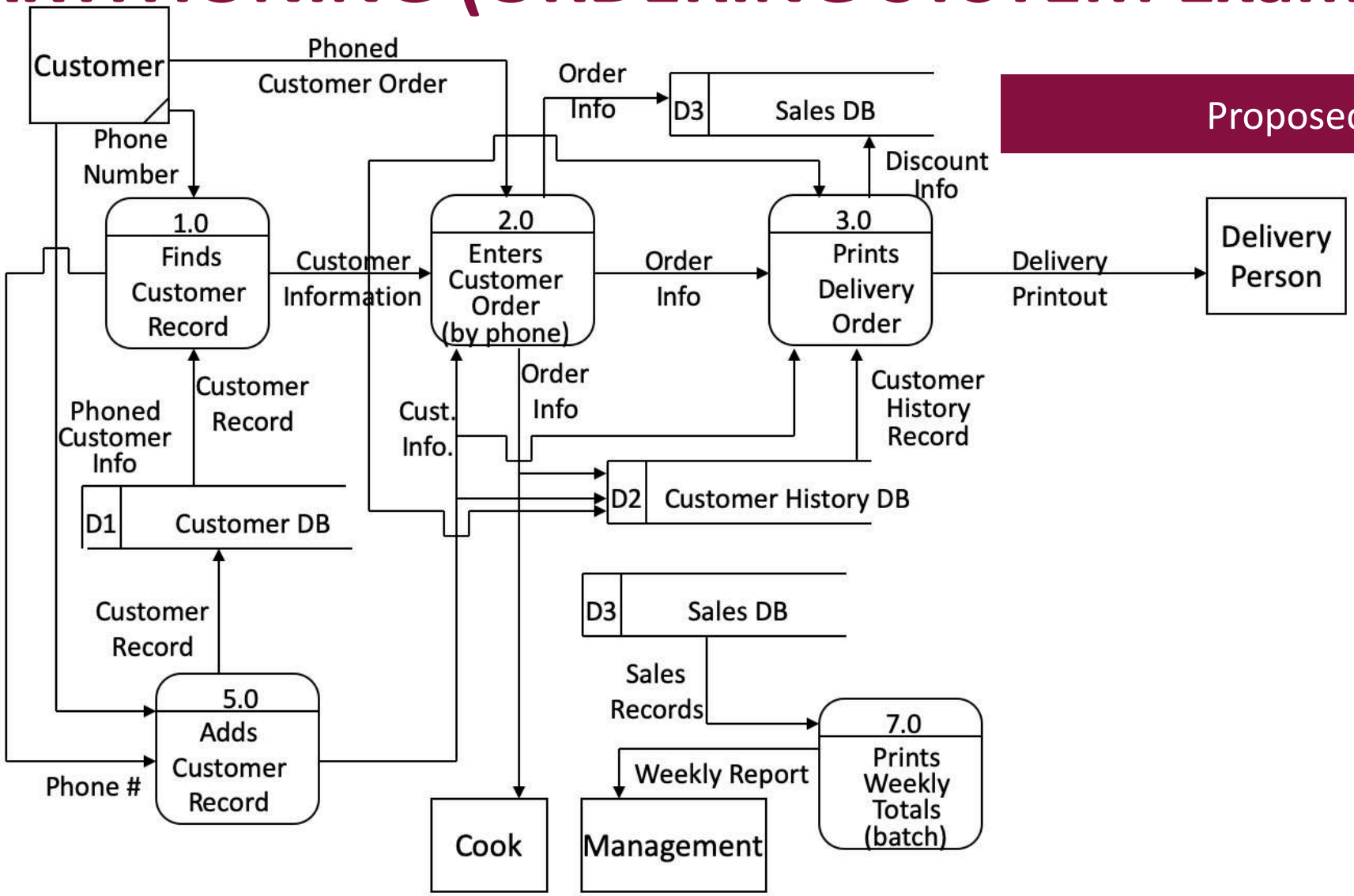
PARTITIONING (ORDERING SYSTEM Example)

Current Physical Diagram 0



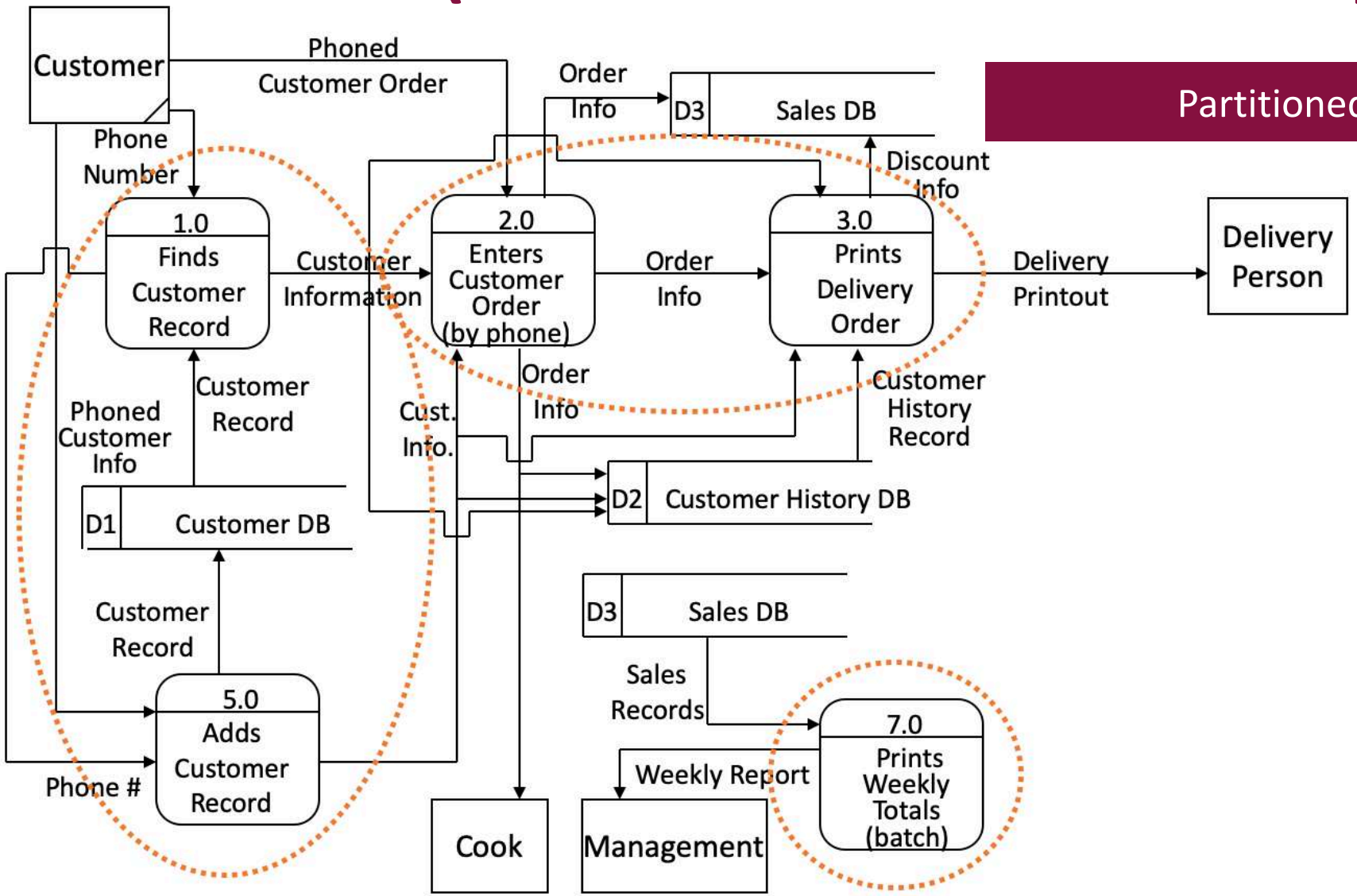
PARTITIONING (ORDERING SYSTEM Example)

Proposed Physical Diagram 0



PARTITIONING (ORDERING SYSTEM Example)

Partitioned Physical Diagram 0



CRUD MATRIX

- The acronym CRUD is often used for
  - Create
  - Read
  - Update
  - Delete
- These are the activities that must be present in a system for each master file
- A CRUD matrix is a tool to represent where each of these processes occurs in a system

CRUD MATRIX

Activity	Customer	Item	Order	Order Detail
Customer Logon	R			
Item Inquiry		R		
Item Selection		R	C	C
Order Checkout	U	U	U	R
Add Account	C			
Add Item		C		
Close Customer Account	D			
Remove Obsolete Item		D		
Change Customer Demographics	RU			
Change Customer Order	RU	RU	RU	CRUD
Order Inquiry	R	R	R	R

EVENT MODELING AND DFD

- An input flow from an external entity is sometimes called a trigger because it starts the activities of a process
- Events cause the system to do something and act as a trigger to the system
- An approach to creating physical data flow diagrams is to create a data flow diagram fragment for each unique system event

EVENT RESPONSE TABLE

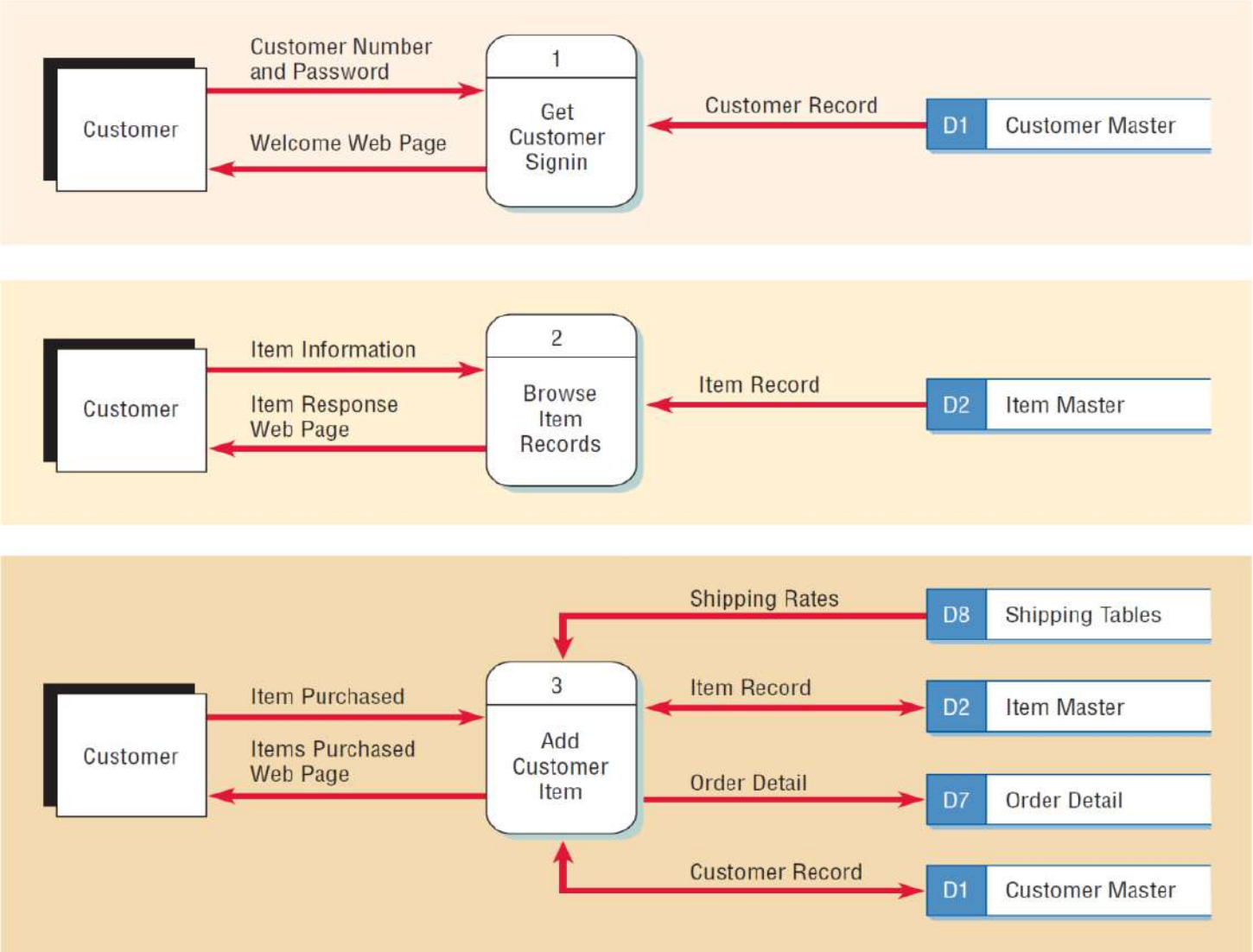
- An event table is used to create a data flow diagram by analyzing each event and the data used and produced by the event
- Every row in an event table represents a data flow diagram fragment and is used to create a single process on a data flow diagram

EVENT RESPONSE TABLE

Event	Source	Trigger	Activity	Response	Destination
Customer logs on	Customer	Customer number and password	Find customer record and verify password. Send Welcome Web page.	Welcome Web page	Customer
Customer browses items at Web storefront	Customer	Item information	Find item price and quantity available. Send Item Response Web page.	Item Response Web page	Customer
Customer places item into shopping basket at Web storefront	Customer	Item purchase (item number and quantity)	Store data on Order Detail Record. Calculate shipping cost using shipping tables. Update customer total. Update item quantity on hand.	Items Purchased Web page	Customer
Customer checks out	Customer	Clicks "Check Out" button on Web page	Display Customer Order Web page.	Verification Web page	
Obtain customer payment	Customer	Credit card information	Verify credit card amount with credit card company. Send.	Credit card data Customer feedback	Credit card company Customer
Send customer email		Temporal, hourly	Send customer an email confirming shipment.		Customer

Example Internet Storefront Event Response Table

EVENT RESPONSE TABLE → DFD



DFD of the Internet Storefront Event Response Table (from previous slide)

MAJOR TOPICS

COMMUNICATING DFD

TIPS IN COMMUNICATING USING DFD

- Use unexploded data flow diagrams early when ascertaining information requirements
- Meaningful labels for all data components

SUMMARY

DATA FLOW DIAGRAM

- Structured analysis and design tools that allow the analyst to comprehend the system and subsystems visually as a set of interrelated data flows

CREATING DFD

- Rounded rectangle (PROCESS)
- Double square (ENTITY)
- An arrow (DATA FLOW)
- Open-ended rectangle (DATA STORE)

LOGICAL & PHYSICAL DFD

1. Creating the logical DFD
   - Context-level data flow diagram
   - Level 0 logical data flow diagram
   - Child diagrams
2. Creating the physical DFD
   - Create from the logical data flow diagram
   - Partitioned to facilitate programming

PARTITIONING

- Whether processes are performed by different user groups
- Processes execute at the same time
- Processes perform similar tasks
- Batch processes can be combined for efficiency of data
- Processes may be partitioned into different programs for security reasons

COMMUNICATING DFD

UTM
UNIVERSITI TEKNOLOGI MALAYSIA

univteknologimalaysia utm_my utmofficial

Thank You

update: August 2019 (sharinhh)

www.utm.my

innovative • entrepreneurial • global