Data Science Joint Education Program

INFO 200
Information Systems Analysis

Chapter #8

Analysing Systems Using Data Dictionaries

Keywords and Phrases

base elements	physical data structure
binary format	repeating group
data dictionary	repeating item
data structure	repository
derived element	schema
document type definition	structured record
extensible markup language (XML)	system deliverables
ID	varchar
IDREF	zoned decimal
package decimal	

Chapter #8 Learning Objectives

- On completion of this chapter you will:
  - Understand how analysts use of data dictionaries for analyzing data-oriented systems.
  - Understand the concept of a repository for analysts' project information and the role of CASE tools in creating them.
  - Create data dictionary entries for:
    - data processes
    - Data stores
    - Data flows
    - Data structures
    - Logical and physical elements of the systems being studied based on DFDs
  - Recognize the functions of data dictionaries in helping users update and maintain information systems.

Cataloging

- Data flow diagrams can be used to catalog:
  - Data processes
  - Flows
  - Stores
  - Structures
  - Elements
- Cataloging takes place with the data dictionary

Major Topics Introduced

- The important topics addressed in Chapter #8 are:
  - The data dictionary
  - The data repository
  - Defining data flow
  - Defining data structures
  - Defining data elements
  - Defining data stores
  - Using the data dictionary
  - XML

The Data Dictionary

Data Dictionaries

- Many database systems have an automated data dictionary
  - These dictionaries can be simple or elaborate
  - Some computerized data dictionaries automatically catalog data items when coding is carried out
- Other database systems only provide a data dictionary template
  - Users are prompted to fill in the template manually
  - Completing the template can help to retain a common reference for every data entry

Data Dictionaries

- An important motivation for maintaining a data dictionary to keep clean data
  - Clean data must be consistent
  - For example:
    - Male gender stored as
      - A character: “M”
      - A string: “Male”
      - An integer number: “1”
    - This is not clean data
- Keeping a data dictionary will help to avoid such issues

Data Dictionaries

- A data dictionary can be viewed in terms of:
  - A reference work of data about data (metadata)
- A data dictionary:
  - Collects data terms
  - Coordinates data terms
  - Confirms what each term means to all users of a system in an organization
- Data dictionaries may have four general categories
  - Data flows
  - Data structures
  - Elements
  - Data stores

The Need for Understanding the Data Dictionary

Understanding Data Dictionaries

- An analyst must understand:
  - The data that create a data dictionary is essential
  - The conventions used in data dictionaries
  - How a data dictionary is developed
  - DFD's (see Chapter #7) are a good starting point for identifying data dictionary entries
- The following slides show:
  - The rationale and motivation for the contents of automated data dictionaries
  - The processes a data dictionary may be used for

Understanding Data Dictionaries

- The processes a data dictionary may be used for
  - Validate the data flow diagram for completeness and accuracy
  - Provide a starting point for developing screens and reports
  - Determine the contents of data stored in files
  - To develop the logic for DFD processes
  - Create XML
  - Provide documentation
  - Eliminate redundancy

The Data Repository

The Data Repository

- A data dictionary contains information about data and procedures
- A larger collection of project information is termed a *repository*
- One of the benefits of CASE tools to develop data dictionaries is:
  - The ability to develop a *repository*
  - A shared collection of projects
  - Team contributions
- The following slide sets out the potential contents a *repository* can hold

The Data Repository

- The potential contents a *repository* include:
  - Information about the data maintained by the system
  - Procedural logic and use cases
  - Screen and report design
  - Data relationships
  - Project requirements and final system deliverables
  - Project management information

The Data Repository

- A data dictionary is created by examining and describing the contents of the:
  - Data flows
  - Data stores
  - Processes
- Figure 8.1 shows how data dictionaries relate to DFD's:
  - Each datastore and data flow should be defined and then explained to include the details of the elements
  - The logic of each process should be described using data flows
  - Omissions and other design errors must be identified and resolved

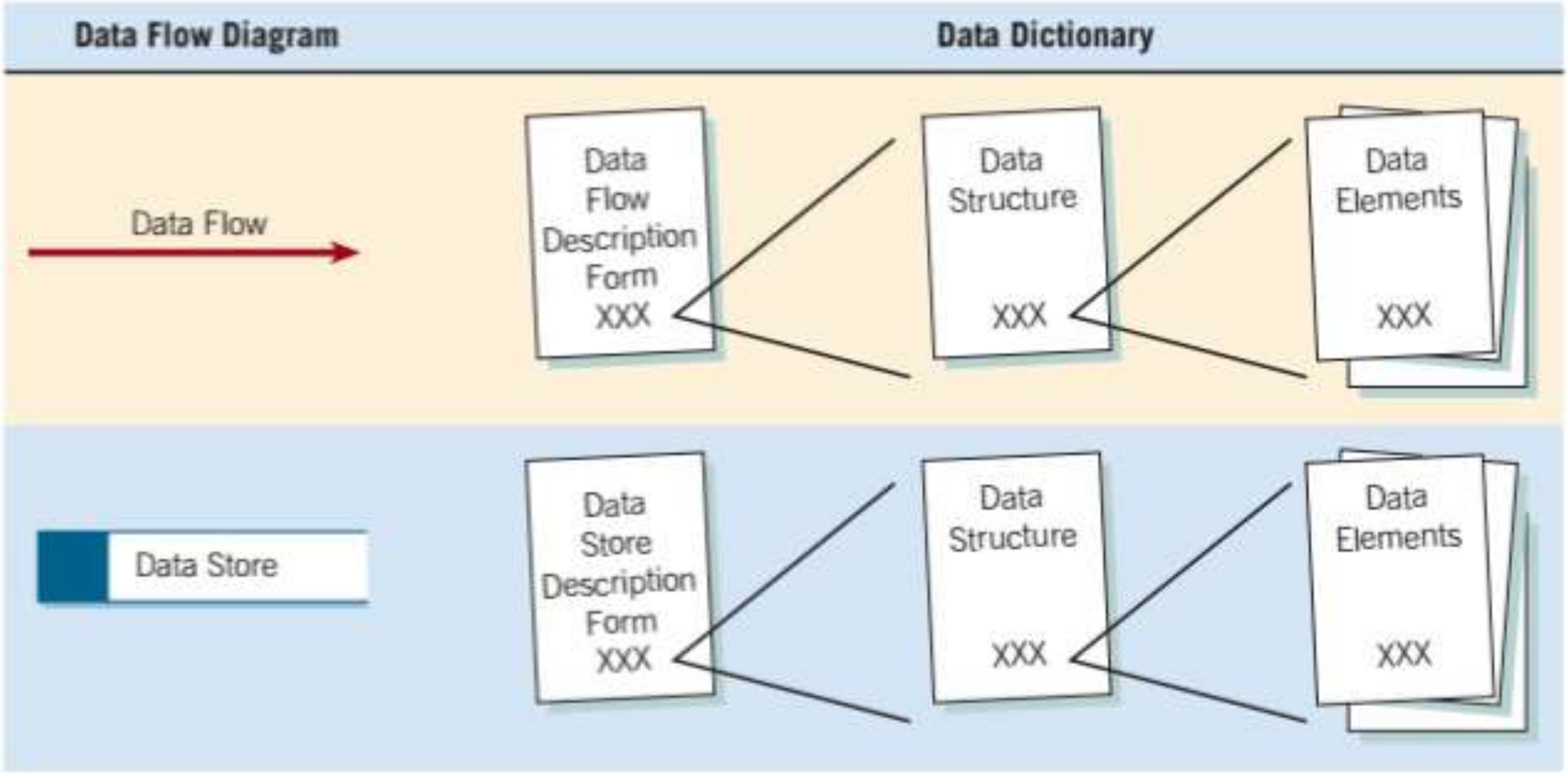


FIGURE 8.1

How data dictionaries relate to data flow diagrams.

A Data Repository Example

- To show the creation of data dictionary entries are created the "Worlds Trend" Catalog Division is used as an example
- Figure 8.2 shows the "Worlds Trend" order form which provides:
- The steps are:
  1. Capture and store the *name*, *address*, and *telephone number* of the customer
  2. Identify the order details including the *item description*, *size*, *price*, *colour*, etc.
- Following these steps the data may be stored for later use

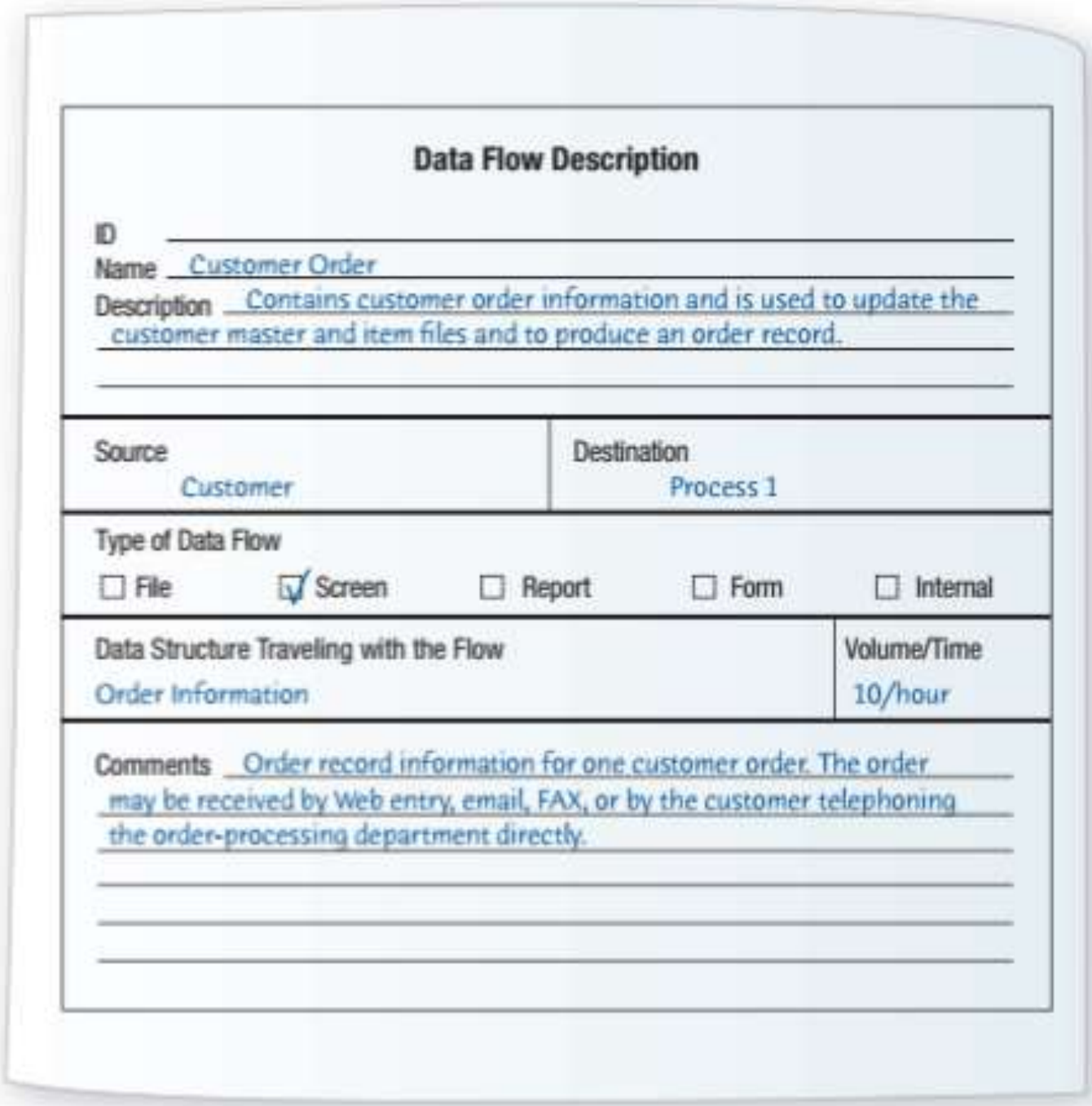
FIGURE 8.2

An online order form from World's Trend Catalog Division.

Defining the Data Flows

Defining Data Flows

- A systems analysis determines system inputs and outputs using interactive and unobtrusive methods (see Chapters #4 and #5)
- The “Worlds Trend” Catalog Division (see Chapter #7) illustrates the completed form
- Figure 8.3 is an example of the data flow description representing:
  - The screen used to add a new CUSTOMER ORDER
  - Updating of the customer and item files
- The following slides show a summary of the information captured for each data flow using a form



Data Flow Description

ID

Name Customer Order

Description Contains customer order information and is used to update the customer master and item files and to produce an order record.

Source	Destination
Customer	Process 1

Type of Data Flow

☐ File ☑ Screen ☐ Report ☐ Form ☐ Internal

Data Structure Traveling with the Flow	Volume/Time
Order Information	10/hour

Comments Order record information for one customer order. The order may be received by Web entry, email, FAX, or by the customer telephoning the order-processing department directly.

FIGURE 8.3

An example of a data flow description from World's Trend Catalog Division.

Data Flow Information

- An optional identification number. Sometimes the ID is coded using a scheme to identify the system and the application in the system
- A unique descriptive name for the data flow. This name is the text that should appear on the diagram and be referenced in all descriptions using the data flow
- A general description of the data flow
- The source of the data flow (the source could be an external entity, a process, or a data flow coming from a data store)

Defining Data Flows

- The destination of the type data flow (same items listed under the source)
- An indication of whether the data flow is a record entering or leaving a file or a record containing a report, form, or screen. If the data flow contains data that are used between processes, it is designated as internal
- The name of the data structure describing the elements found in this data flow. For a simple data flow, it could be one or several elements
- The volume per unit of time. The data could be records per day or any other unit of time
- An area for further comments and notations about the data flow

Defining Data Flows

- Data flows for all inputs and outputs should be described first
  - They usually represent the human interface
- Then identify the intermediate data flows with data flows to and from data stores
- The details of data flow is described using:
  - Elements (or fields)
  - A data structure (or)
  - A group of elements
- A simple data flow may be described using a single element such as:
  - A customer number (used to find matching customer record)

Describing Data Structures

Data Structures

- Data structures are made up of smaller structures and elements
- An algebraic notation is used to describe data structures
- Algebraic Notation
  - Equal sign means “is composed of”
  - Plus sign means “and”
  - Braces {} mean repetitive elements
  - Brackets [] for an either/or situation
    - Either one element or another maybe present but not both
    - The elements listed between brackets are mutually exclusive
  - Parentheses () for an optional element
    - Optional elements may be left blank on entry screens and may contain spaces or zeros for numeric file structures

Data Structures

- Figure 8.4 is an example of a data structure for adding a customer at the “Worlds Trend” Catalog Division
- Each structural structural record must be further defined until the entire set is broken down into its component elements
- Structural records and elements used in multiple systems are given non-specific names such as:
  - Street
  - City
  - Zip code
- Non-specific names allow the definition of records once with reuse in other applications

Customer Order = Customer Number +
Customer Name +
Address +
Telephone +
Catalog Number +
Order Date +
{Available Order Items} +
Merchandise Total +
(Tax) +
Shipping and Handling +
Order Total +
Method of Payment +
(Credit Card Type) +
(Credit Card Number) +
(Expiration Date)

Customer Name = First Name +
(Middle Initial) +
Last Name

Address = Street +
(Apartment) +
City +
State +
Zip +
(Zip Expansion) +
(Country)

Telephone = Area Code +
Local Number

Available Order Items = Quantity Ordered +
Item Number +
Item Description +
Size +
Color +
Price +
Item Total

Method of Payment = [Check ¦ Charge ¦ Money Order]

Credit Card Type = [World's Trend ¦ American Express ¦ MasterCard ¦ Visa]

FIGURE 8.4

Data structure example for adding a customer order at World's Trend Catalog Division.

Customer Billing Statement = Current Date +
Customer Number +
Customer Name +
Address +
$_{1}^{5}${Order Line} +
(Previous Payment Amount) +
Total Amount Owed +
(Comment)

Order Line = Order Number +
Order Date +
Order Total

FIGURE 8.5

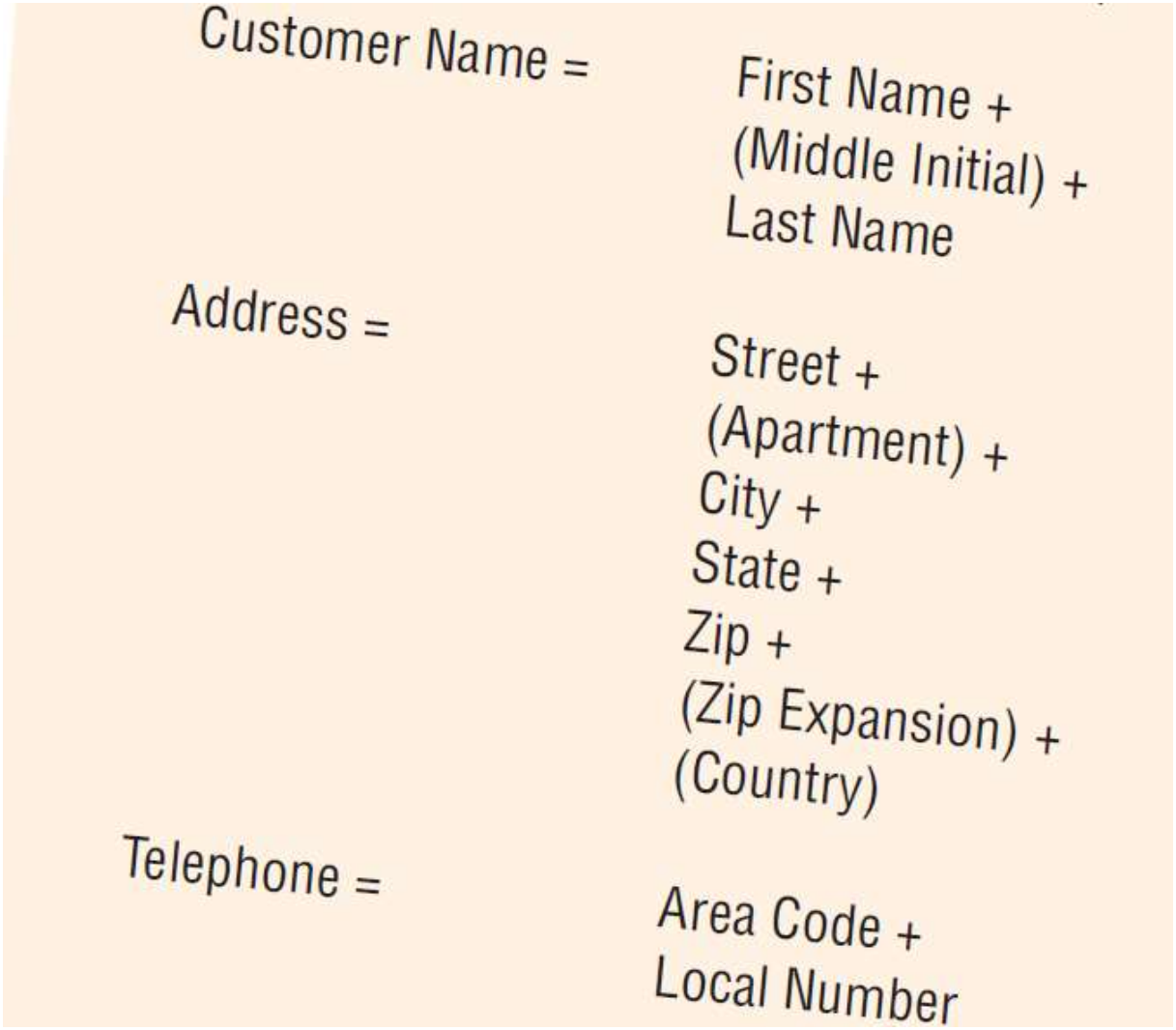
Physical elements added to a data structure.

Data Structures

- Figure 8.4 is an example of a data structure for adding a customer at the “Worlds Trend” Catalog Division
- Each structural structural record must be further defined until the entire set is broken down into its component elements
- Structural records and elements used in multiple systems are given non-specific names such as:
  - Street
  - City
  - Zip code
- Non-specific names allow the definition of records once with reuse in other applications

Structural Record Example

- A structural record example
- We can see:
  - The structural elements
  - The non-specific names
  - The names promote reuse



Logical and Physical Structures

Logical and Physical Data Structures

- Logical:
  - Show what data the business needs for its day-to-day operations
- Physical:
  - Include additional elements necessary for implementing the system
- When data structures are first defined:
  - only the data elements the user would see, such as a name, address, and balance due, are included
  - This stage is the logical design, showing what data the business needs for its day-to-day operations
- Considering HCI it is important that logical design reflects a user's mental model of how a system is viewed

Examples of Physical Design Elements

- Using the *logical* design as a basis:
  - The analysis must design the *physical* data structures which include additional elements required to implement the system
- Examples of physical design elements
  - Key fields used to locate records in a database table
  - Codes to identify the status of master records
  - Transaction codes to identify types of records when a file contains different record types
  - Repeating group entries (counting how many items are in a group
  - Limits on the number of items in a repeating group
  - A password used by a customer accessing a secure website

Examples of Physical Design Elements

- Figure 8.5 is an example of the data structure for:
  - A CUSTOMER BILLING STATEMENT
  - This shows that the ORDER LINE is both a repeating item and a structural record
- Repeating group notation may have several other formats:
  - If the group repeats a fixed number of times the number is placed next to the opening brace as in 12 (Monthly Sales) indicating that the group repeats once for each month of the year
  - If no number is indicated the group repeats indefinitely
  - An example is the Customer Master Table = {Customer Records}

Customer Order = Customer Number +
Customer Name +
Address +
Telephone +
Catalog Number +
Order Date +
{Available Order Items} +
Merchandise Total +
(Tax) +
Shipping and Handling +
Order Total +
Method of Payment +
(Credit Card Type) +
(Credit Card Number) +
(Expiration Date)

Customer Name = First Name +
(Middle Initial) +
Last Name

Address = Street +
(Apartment) +
City +
State +
Zip +
(Zip Expansion) +
(Country)

Telephone = Area Code +
Local Number

Available Order Items = Quantity Ordered +
Item Number +
Item Description +
Size +
Color +
Price +
Item Total

Method of Payment = [Check | Charge | Money Order]

Credit Card Type = [World's Trend | American Express | MasterCard | Visa]

FIGURE 8.4

Data structure example for adding a customer order at World's Trend Catalog Division.

Customer Billing Statement = Current Date +
Customer Number +
Customer Name +
Address +
$_{1}^{5}${Order Line} +
(Previous Payment Amount) +
Total Amount Owed +
(Comment)

Order Line = Order Number +
Order Date +
Order Total

FIGURE 8.5

Physical elements added to a data structure.

Examples of Physical Design Elements

- The number of entries in repeating groups may depend on a condition:
  - For example: an entry on the Customer Master Record for each item ordered
- This condition could be stored in the data dictionary as {Items Purchased}:
  - Where 5 is the number of items

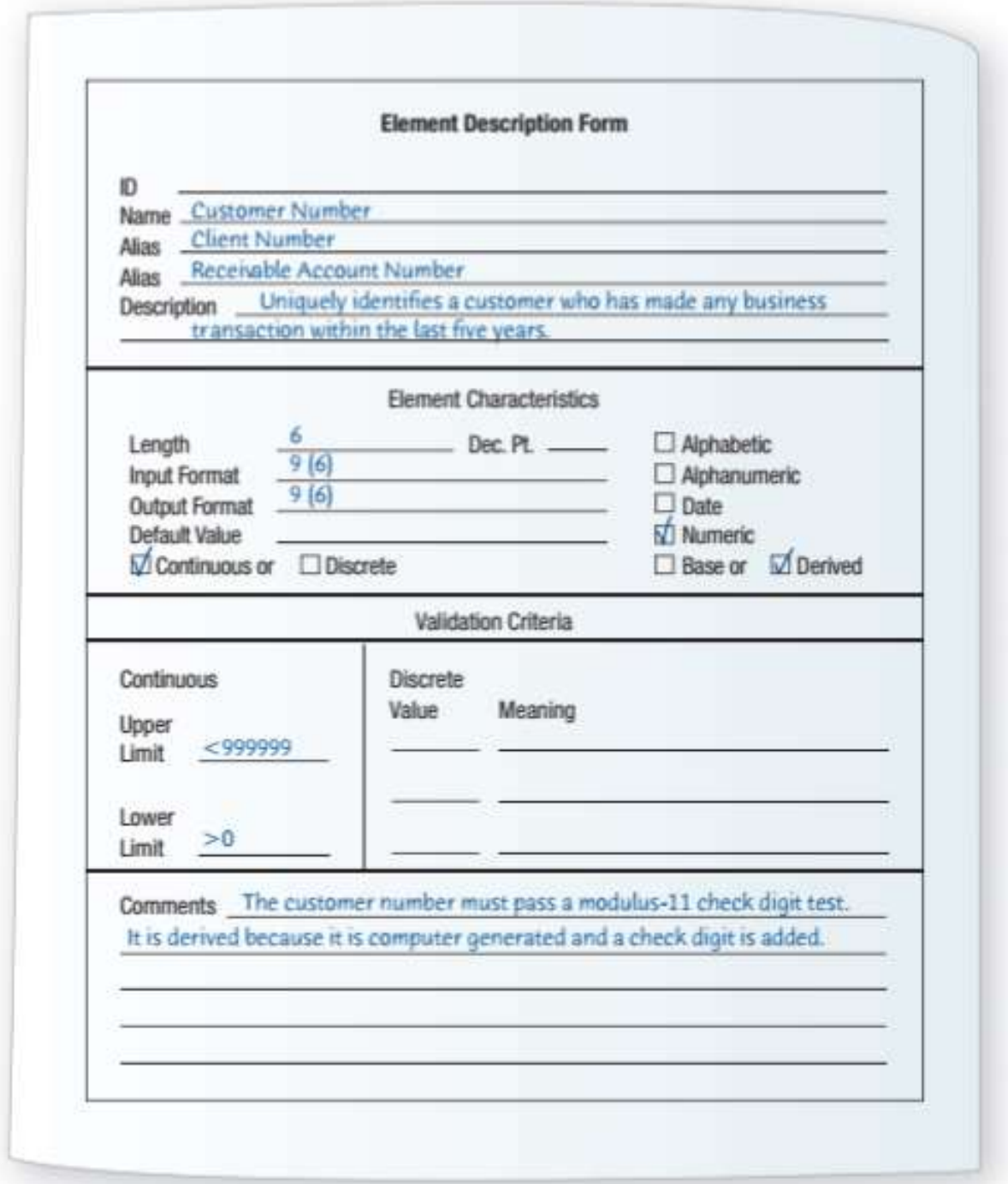
Data Elements

Data Elements

- Data elements
  - Are defined once in the data dictionary
  - May also be entered on an element description form (see Figure 8.6)
- Characteristics commonly included on the element description form include:
  - Element ID
    - an optional entry – allows the building of automated data dictionary entries)
  - The name of the element (unique / descriptive / meaningful)
    - Based on what the element is commonly called in most programs or by the major user of the element

Data Elements

- Figure 8.6
- An Element Description Form Example from the “World’s Trend” Catalog Division example



Element Description Form

ID
Name Customer Number
Alias Client Number
Alias Receivable Account Number
Description Uniquely identifies a customer who has made any business transaction within the last five years.

Element Characteristics

Length 6 Dec. Pt. ____
Input Format 9 (6)
Output Format 9 (6)
Default Value ____
☑ Continuous or ☐ Discrete

☐ Alphabetic
☐ Alphanumeric
☐ Date
☑ Numeric
☐ Base or ☑ Derived

Validation Criteria

Continuous
Upper Limit <999999
Lower Limit >0

Discrete
Value Meaning

Comments The customer number must pass a modulus-11 check digit test.
It is derived because it is computer generated and a check digit is added.

Data Elements (continued)

- Aliases
  - Synonyms or other names for the element)
  - Names used by different users in different systems
  - A CUSTOMER NUMBER may also be called a RECEIVABLE ACCOUNT NUMBER or a CLIENT NUMBER
- A short description of the element
  - An example might be: uniquely identifies a customer who has made any business transactions within the last five years
- Whether an element is base or derived
  - A base element is initially keyed into the system
  - Derived elements are created by processes resulting from a calculation or a series of decision-making statements

Data Elements (continued)

- Type of data
  - Alphanumeric or text data
  - Formats include:
    - Mainframe: packed, binary, display
    - Microcomputer (PC) formats
    - PC formats, such as Currency, Number, or Scientific, depend on how the data will be used
- Input and output formats
  - These should be included
  - It is helpful to use special coding symbols to indicate how the data should be presented
  - These symbols and their uses are illustrated in Figure 8.8

Data Elements (continued)

- Figure 8.8
- Format Character Codes

Formatting Character	Meaning
X	May enter or display/print any character
9	Enter or display only numbers
Z	Display leading zeros as spaces
,	Insert commas into a numeric display
.	Insert a period into a numeric display
/	Insert slashes into a numeric display
-	Insert a hyphen into a numeric display
V	Indicate a decimal position (when the decimal point is not included)

Data Elements (continued)

- Validation criteria
  - Ensure that accurate data are captured by the system
  - Elements are either:
    - Discrete (meaning they have fixed values)
    - Continuous (with a smooth range of values)
- Default value
  - Include any default value the element may have
  - The default value is displayed on entry screens
  - Reduces the amount of keying
  - Default values on GUI screens
    - Initially display in drop-down lists
    - Are selected when a group of radio buttons are used

Data Elements (continued)

- Default value
  - Include any default value the element may have
  - The default value is displayed on entry screens
  - Reduces the amount of keying
    - Default values on GUI screens
      - Initially display in drop-down lists
      - Are selected when a group of radio buttons are used
- An additional comment or remark area
  - This might be used to:
    - Indicate the format of the date
    - Identify special validation that is required
    - Check-digit method used
    - Etc.

Data Elements (continued)

- The length of an element (see Figure 8.7):
  - Lengths may vary and can be domain and case specific
  - The final length will be based on the following considerations:
    - Numeric lengths (some elements have standard lengths, state abbreviations, zip codes, or telephone numbers)
    - Name and address length may be based on the following table

Field	*Length*	*Percentage of Data That Will Fit (U.S.)*
Last Name	11	98
First Name	18	95
Company Name	20	95
Street	18	90
City	17	99

Data Formats

- Figure 8.7
- Examples of Data Formats Used in PC Systems

Data Type	Meaning
Bit	A value of 1 or 0, a true/false value
Char, varchar, text	Any alphanumeric character
Datetime, smalldatetime	Alphanumeric data, several formats
Decimal, numeric	Numeric data that are accurate to the least significant digit; can contain a whole and decimal portion
Float, real	Floating-point values that contain an approximate decimal value
Int, smallint, tinyint	Only integer (whole digit) data
Currency, money, smallmoney	Monetary numbers accurate to four decimal places
Binary, varbinary, image	Binary strings (sound, pictures, video)
Cursor, timestamp, uniqueidentifier	A value that is always unique within a database
Autonumber	A number that is always incremented by one when a record is added to a database table

Data Elements (continued)

- Data truncation
- For other fields:
  - It is often useful to examine (or) sample historical data found in the organization to determine a suitable field length
  - If the element is too small:
    - The data will be truncated
  - The analyst must decide how this will affect the system outputs
  - If a last name is truncated
    - Traditional postal mail will usually still be delivered
  - A truncated email address or web address
    - Will not meet the required template specification and will fail (as it is not usable)

Data Elements

- Data element descriptions such as CUSTOMER NUMBER may be called CLIENT NUMBER elsewhere in the system
  - Perhaps old code written with this alias needs to be updated
- Another type of element is an alphabetic element
  - In the “Worlds Trend” example codes are used to describe colours – for example:
    - BL (blue)
    - WH (white)
    - GR (green)
  - When implemented this element will need a ‘look-up’ table for users to read the code meanings (codes are addressed in Chapter #15)

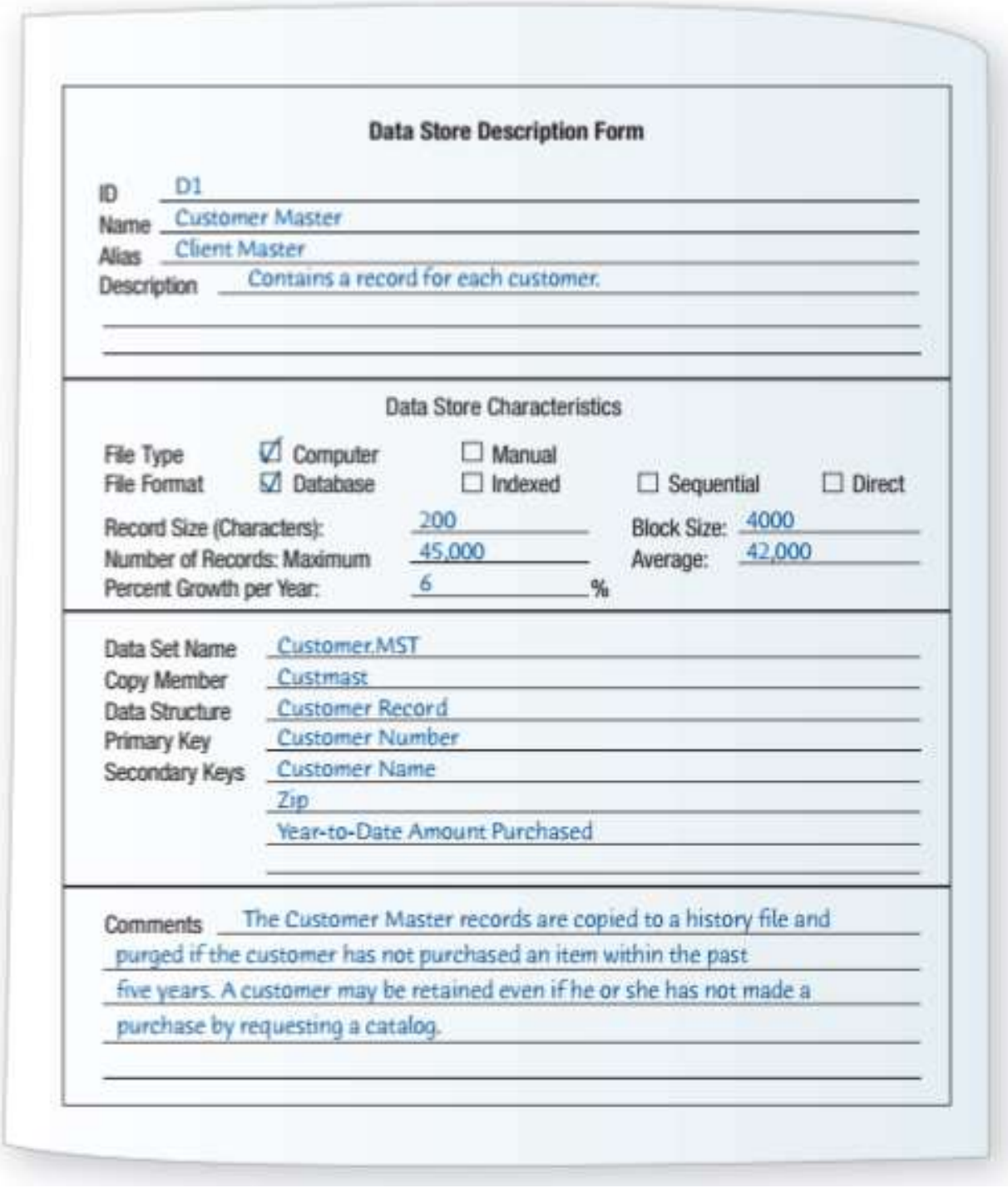
Data Stores

Data Stores

- All base elements and derived elements must be stored in the system
- Data stores are created for each different data entry being stored
- When data flow base elements are grouped together to form a structural record
  - A data store is created for each unique structural record
- Because a given data flow may only show part of the collective data that a structural record contains:
  - Many different data flow structures may need to be examined to arrive at a complete data store description
- Figure 8.9 shows a typical form used to describe a data store

Data Store Form

- Figure 8.9
- An example of a Data Store Form for “World’s Trend” Catalog Division example application



Data Store Description Form

ID: D1
Name: Customer Master
Alias: Client Master
Description: Contains a record for each customer.

Data Store Characteristics

File Type: ☑ Computer ☐ Manual
File Format: ☑ Database ☐ Indexed ☐ Sequential ☐ Direct
Record Size (Characters): 200 Block Size: 4000
Number of Records: Maximum 45,000 Average: 42,000
Percent Growth per Year: 6 %

Data Set Name: Customer.MST
Copy Member: Custmast
Data Structure: Customer Record
Primary Key: Customer Number
Secondary Keys: Customer Name
Zip
Year-to-Date Amount Purchased

Comments: The Customer Master records are copied to a history file and purged if the customer has not purchased an item within the past five years. A customer may be retained even if he or she has not made a purchase by requesting a catalog.

Data Store Form Information

- Figure 8.9 shows a typical form used to describe a data stors
- The information included on the form is as follows:
  - The data store ID
    - The ID is often a mandatory entry to prevent the storage of redundant information
    - An example is D1 for the CUSTOMER MASTER
  - The data store name
    - It should be descriptive and unique
  - An alias for the table
    - CLIENT MASTER can be an alias for the CUSTOMER MASTER
  - A short description of the data store
  - The file type
    - Either computer or manual files

Data Store Form Information (continued)

- The file format:
  - Format designates if the file is a database table (or) a simple flat file
  - File formats are discussed in Chapter #13
- The maximum and average number of records on the file as well as the growth per year:
  - This is used to try to predict the disk space required
  - Used for application and required for hardware planning
- The file or data set name:
  - Specifies the file name
  - If not known this item may be left blank

Data Store Form Information (continued)

- The data structure name:
  - Should use a name found in the data dictionary
  - A link to the elements in the data store should be provided
- Primary and secondary keys:
  - Are elements (or a combination of elements) found in the data structure
  - In the example the CUSTOMER NUMBER is the primary key and should be unique
  - The CUSTOMER NAME / ZIP / YEAR-TO-DATE AMOUNT PURCHASED are secondary keys (used to control record sequencing on reports and to locate records directly)
  - Keys are discussed in Chapter #13

Data Store Form Information (continued)

- Comments:
- Are used for information that does not fit into any of the previous categories
- Comments may include:
  - update (or) backup timing
  - Security
  - Other considerations.

Creating a Data Dictionary

Creating a Dara Dictionary

- Data dictionaries are:
  - Created following after the DFD has been completed (or)
  - They may be constructed as the DFD id under development
- Algebraic notation and structural records allows:
  - Development of the data dictionary and DFDs' using a top-down approach
- Figure 8.10 shows:
  - Part of two DFD levels
  - Corresponding data dictionary entries for producing an employee paycheck

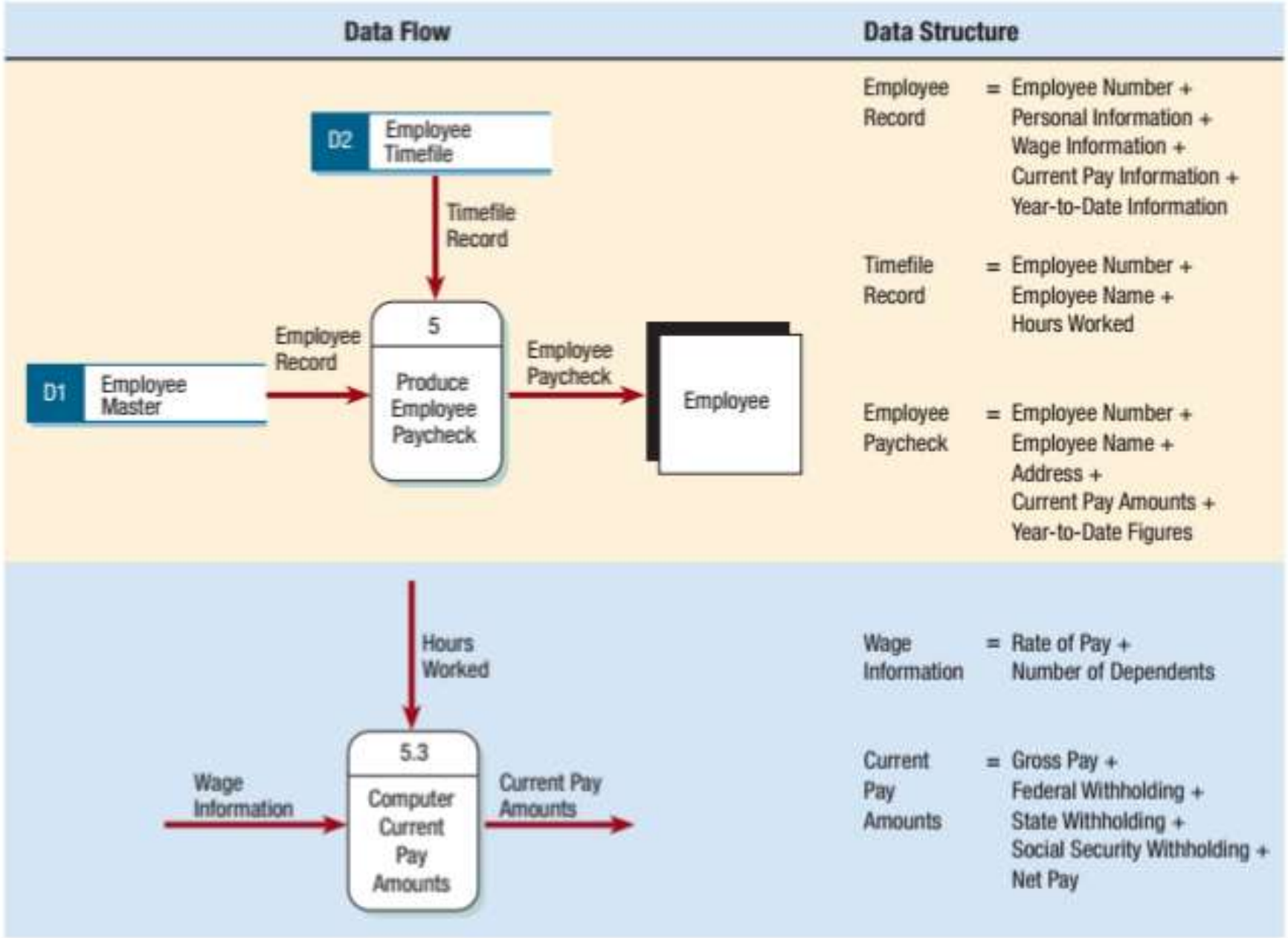


FIGURE 8.10

Two data flow diagrams and corresponding data dictionary entries for producing an employee paycheck.

Analysing Input and Output

Analysing Input and Output

- Input and output analysis forms commonly include the following fields:
  - A descriptive name for the input or output
  - The user contact responsible
  - Whether the data is input or output
  - The format of the data flow
  - Elements indicating the sequence of the data on a report or screen
  - A list of elements
- On completion of the form each element should be analysed to:
  - Check for repeating (or) optional elements
  - Check for mutually exclusive elements (or) if an element falls into a group (or) elements that regularly combine in many structures should be placed in a structural record

Input and Output

- Figure 8.11
- An Example of an Input and Output Analysis Form for “World’s Trend” Catalog Division

Input and Output Analysis Form

Input/Output Name: Customer Billing Statement
User Contact: Susan Han

File Type: ☑ Output ☐ Input
File Format: ☑ Report ☐ Screen ☐ Undetermined

Sequencing Element(s): Zip Code (Page Sequence), Order Number

Element Name	Length	B/D	Edit Criteria
Current Date	6	B	(System Supplied)
Customer Number	6	D	(Includes Check Digit)
Customer First Name	20	B	Not Spaces
Customer Last Name	15	B	Not Spaces
Customer Middle Initial	1	B	A through Z or Space
Street	20	B	Not Spaces
Apartment	20	B	Not Spaces
City	20	B	Not Spaces
State	2	B	Valid State Abbr.
Zip	9	B	Numeric, Last 4 Opt.
Order Number	6	D	> 0
Order Date	8	B	MM/DD/YYYY
Order Total	9	D	Format: 9 (7) V99
Previous Payment Amount	5	D	Format: 9 (7) V99
Total Amount Owed	9	D	Format; 9 (7) V99
Comment	60	B	

Comments: Print one page for each customer. If there are more items than will fit on a page, continue on a second page.

FIGURE 8.11

An example of an input–output analysis form for World’s Trend Catalog Division.

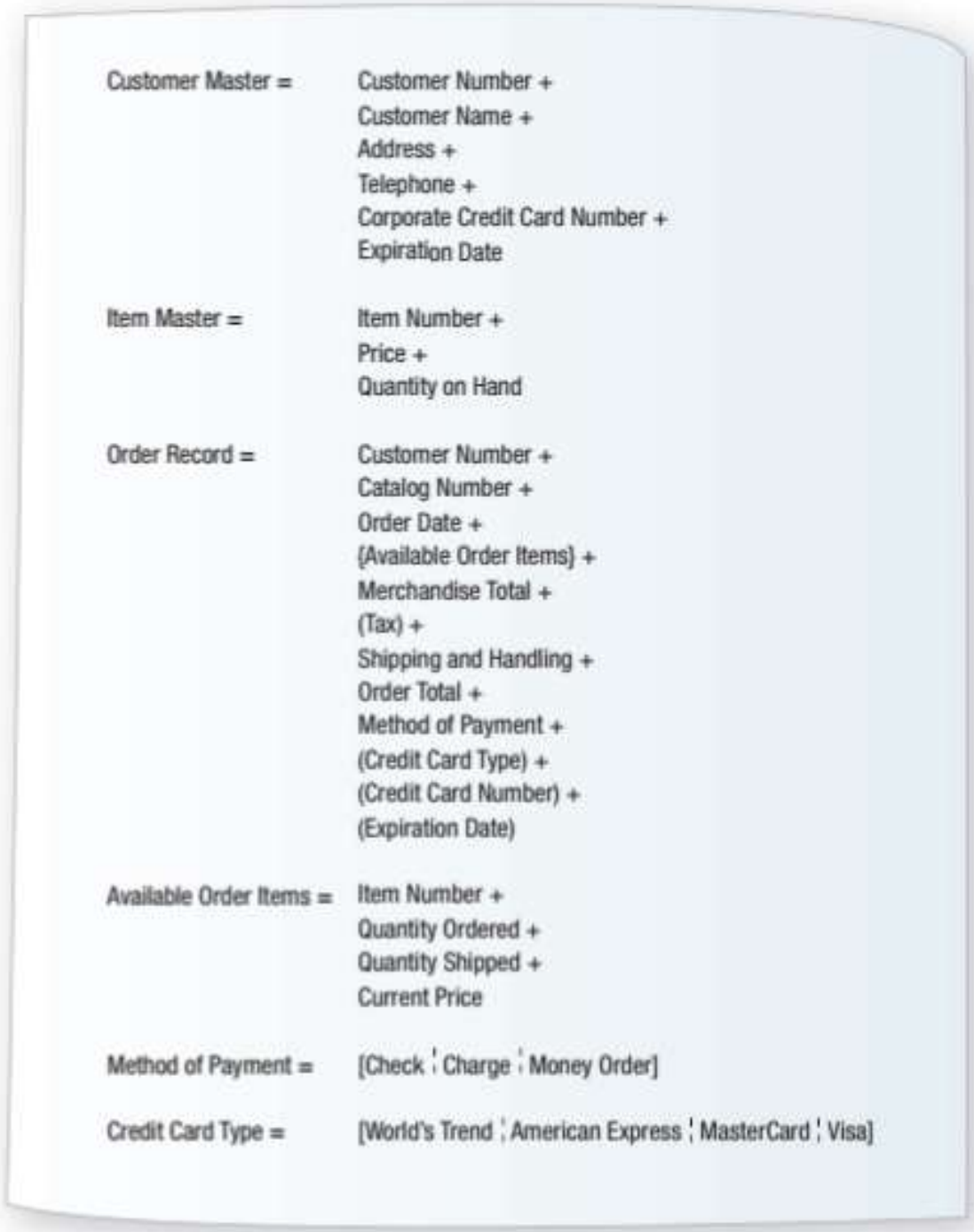
Developing Data Stores

Developing Data Stores

- An activity in creating a data dictionary is the development of data stores
- We have determined the data flows (from one process to another)
  - This information is described in data structures
  - This information may however be stored in multiple places and each place the data store may be different
  - Whereas data flows represent data in motion (and) data stores represent data at rest
  - When data stores are created for only one report or screen we refer to them as “user views”
- For example when an order arrives at “Worlds Trend” (see Figure 8.12)
    - It holds mostly semi-permanent (temporary) data (or) information stored permanently

Data Stores

- FIGURE 8.12
- Data stores derived from a pending order at “World’s Trend” Catalog Division



Customer Master = Customer Number + Customer Name + Address + Telephone + Corporate Credit Card Number + Expiration Date

Item Master = Item Number + Price + Quantity on Hand

Order Record = Customer Number + Catalog Number + Order Date + {Available Order Items} + Merchandise Total + (Tax) + Shipping and Handling + Order Total + Method of Payment + (Credit Card Type) + (Credit Card Number) + (Expiration Date)

Available Order Items = Item Number + Quantity Ordered + Quantity Shipped + Current Price

Method of Payment = [Check ¦ Charge ¦ Money Order]

Credit Card Type = [World’s Trend ¦ American Express ¦ MasterCard ¦ Visa]

FIGURE 8.12

Data stores derived from a pending order at World’s Trend Catalog Division.

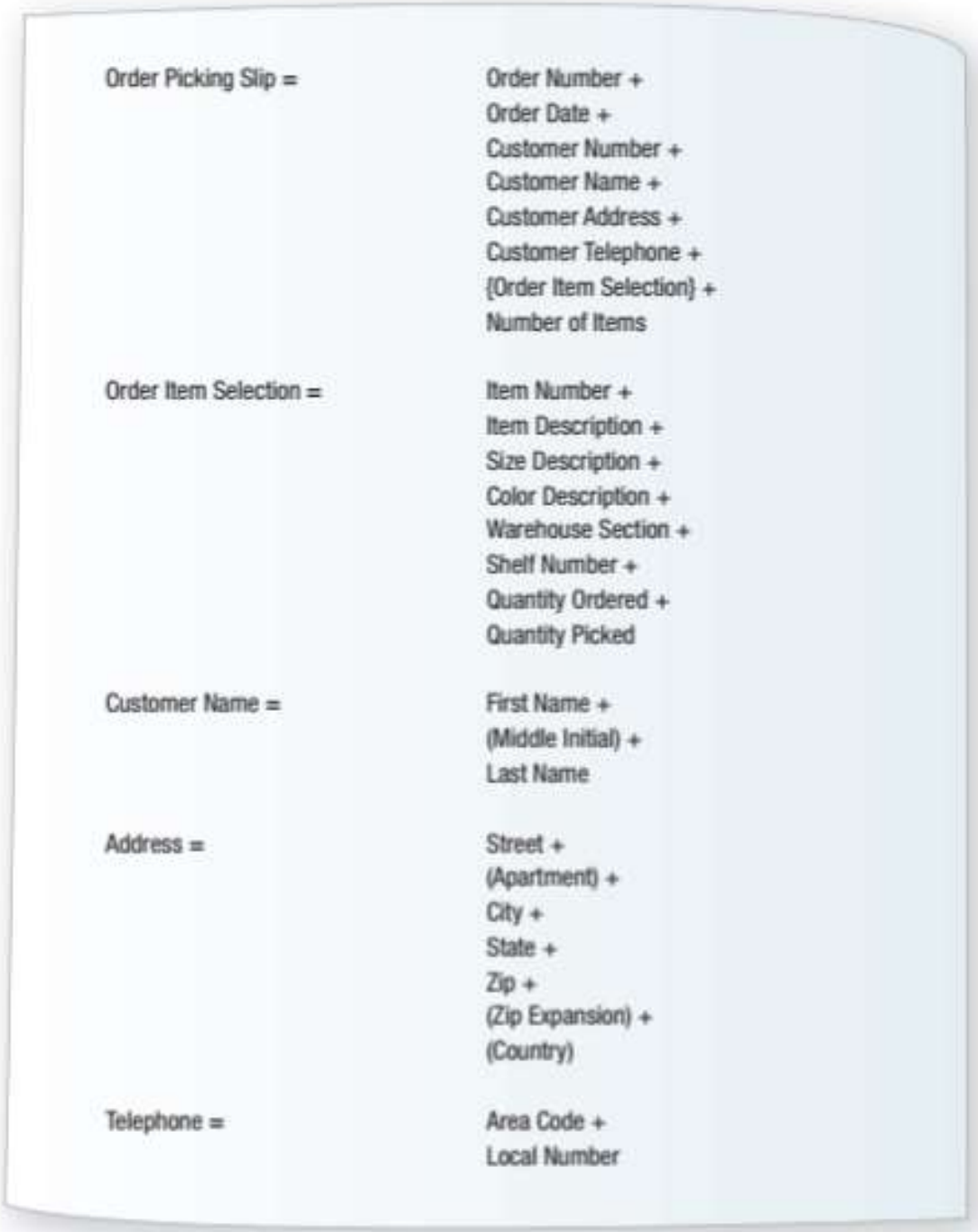
Using a Data Dictionary

Using Data Dictionaries

- A data dictionary should be automated, interactive, and evolutionary
- A data dictionary is not an end in itself but should be viewed as an activity that parallels systems analysis and design
- To have maximum power a data dictionary should be tied into a number of systems programs
- Data dictionaries may be used to
  - Create screens, reports, and forms
  - Generate computer language source code
  - Analyze the system design, detecting flaws and areas that need clarification

Using Data Dictionaries

- Figure 8.13 shows:
  - The data structure for the “Worlds Trend” ORDER PICKING SLIP
- To create physical documents
  - Arrange the elements in a functional way using design guidelines and common sense
  - Use the element definition and their lengths
  - Repeating groups are columns on the screen, report (see Figure 8.14), or forms
  - Structural groups are grouped together on screen, report, or form



Order Picking Slip =	Order Number + Order Date + Customer Number + Customer Name + Customer Address + Customer Telephone + {Order Item Selection} + Number of Items
Order Item Selection =	Item Number + Item Description + Size Description + Color Description + Warehouse Section + Shelf Number + Quantity Ordered + Quantity Picked
Customer Name =	First Name + (Middle Initial) + Last Name
Address =	Street + (Apartment) + City + State + Zip + (Zip Expansion) + (Country)
Telephone =	Area Code + Local Number

FIGURE 8.14

Order picking slip created from the data dictionary.

World's Trend
Order Picking Slip

Order Number: 999999   Order Date Z9/99/9999
Customer Number: 999999

Name: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Street: XXXXXXXXXXXXXXXXXXXXXXXXXXX
Apartment: XXXXXXXX
City, State, Zip XXXXXXXXXXXXXXXXXXXXXXXXXXX, XX 99999-ZZZZ
Country: XXXXXXXXXXXXXXXXXXXXXXXXXXX
Telephone: (999) 999-9999

---- Quantity ----			Shelf	Item			
Picked	Ordered	Section	Number	Number	Item Description	Size	Color
_____	ZZZZ9	XXXXX	99999	999999	XXXXXXXXXXXXXXXXX	XXXXXXXXX	XXXXXX
_____	ZZZZ9	XXXXX	99999	999999	XXXXXXXXXXXXXXXXX	XXXXXXXXX	XXXXXX
_____	ZZZZ9	XXXXX	99999	999999	XXXXXXXXXXXXXXXXX	XXXXXXXXX	XXXXXX
_____	ZZZZ9	XXXXX	99999	999999	XXXXXXXXXXXXXXXXX	XXXXXXXXX	XXXXXX
_____	ZZZZ9	XXXXX	99999	999999	XXXXXXXXXXXXXXXXX	XXXXXXXXX	XXXXXX
_____	ZZZZ9	XXXXX	99999	999999	XXXXXXXXXXXXXXXXX	XXXXXXXXX	XXXXXX
_____	ZZZZ9	XXXXX	99999	999999	XXXXXXXXXXXXXXXXX	XXXXXXXXX	XXXXXX
_____	ZZZZ9	XXXXX	99999	999999	XXXXXXXXXXXXXXXXX	XXXXXXXXX	XXXXXX

Number of Items: Z9

Using Data Dictionaries

- The data structure and elements of the data store:
  - Are commonly used to generate corresponding computer language source code used in computer programs
- The data dictionary may be used in conjunction with a DFD to analyze the system design, detect Flaws and areas that need clarification – some considerations are:
  - All base elements on an output data flow must be present on an input data flow to the process producing the output
  - A derived element should be created by a process and should be output from at least one process into which it is not input
  - The elements that are present in a data flow coming into or going out of a data store must be contained in the data store

Using Data Dictionaries

- A data dictionary:
  - If started early in the analysis and design process can save many hors of work
  - Is the one common source in organisations for answering questions and settling disputes related to any aspect of data description
- An up-to-date data dictionary can serve as a good reference for maintenance of unfamiliar systems
- Automated data dictionaries serve as a good reference point for both people and computer programs

Using a Data Dictionary to Create XML

Using Data Dictionaries to Create XML

- XML can be used to exchange data between businesses or between systems within a business
- XML addresses the problem of sharing data when users have different computer systems and software or different database management systems
- XML documents may be transformed into different output formats
- XML is a way to define, sort, filter, and translate data into a universal data language that can be used by anyone
- XML may be created from:
  - Databases
  - A form
  - Software programs
  - keyed directly into a document, text editor, or XML entry program

Using Data Dictionaries to Create XML

- XML can be used to exchange data between businesses or between systems within a business
- XML addresses the problem of sharing data when users have different computer systems and software or different database management systems
- XML documents may be transformed into different output formats
- XML is a way to define, sort, filter, and translate data into a universal data language that can be used by anyone
- XML may be created from:
  - Databases
  - A form
  - Software programs
  - keyed directly into a document, text editor, or XML entry program

Using Data Dictionaries to Create XML

- The data dictionary is an ideal starting point for developing XML content
- A standard definition of the data is created using a set of tags that are included before and after each data element or structure
- XML elements may also include attributes
- The XML document tends to mirror the data dictionary structure
- Figure 8.15 illustrates a data dictionary
  - The overall collection of customers is included in what is called the root element: *customers*

Creating XML

- Figure 8.15
- This figure illustrates a data dictionary containing:
  - Customer
  - Order
  - Payment information
- Using a data dictionary entry to develop XML content
- The XML document mirrors the data dictionary structure

Data Dictionary

Customer =	Name + Address + Current Balance + {Order Information} + Payment
Name =	Last Name + First Name + (Middle Initial)
Address =	Street + (Apartment) + City + State + Zip + Country
Order Information =	Order Number + Order Date + Ship Date + Total
Payment =	[Check	Credit Card] + Payment Date + Payment Amt
Check =	Check Number
Credit Card =	Credit Card Number + Expiration Date

XML

```
<?xml version="1.0">
<customers>
<customer number="C15008">
  <name type="I">
    <lastname>Stadler</lastname>
    <firstname>Karen</firstname>
    <middle_initial>L</middle_initial>
  </name>
  <address>
    <street>123 Oak Street</street>
    <apartment>Suite 16</apartment>
    <city>Madison</city>
    <state>WI</state>
    <zip>43704</zip>
    <country>United States</country>
  </address>
  <current_balance>123.45</current_balance>
  <order customer_number="C15008">
    <order_number>00123</order_number>
    <order_date format="yyyymmdd">2008-06-23</order_date>
    <ship_date format="yyyymmdd">2008-06-25</ship_date>
    <total>1345.89</total>
  </order>
  <order customer_number="C15008">
    <order_number>00127</order_number>
    <order_date format="yyyymmdd">2008-09-18</order_date>
    <ship_date format="yyyymmdd">2008-09-26</ship_date>
    <total>240.00</total>
  </order>
  <payment>
    <check>
      <check_number>7234</check_number>
    </check>
    <payment_date format="yyyymmdd">2008-09-30</payment_date>
    <payment_amt>1585.89</payment_amt>
  </payment>
</customer>
</customers>
```

XML Document Type Definitions

XML Document Type Definitions

- The element structure of XML content is defined using a document type definition (DTD)
  - DTD is used to determine whether the XML document content is valid (it conforms to the order and type of data that must be present in the document)
  - The DTD is easier to create if a data dictionary has been completed because the analyst has worked with users and made decisions on the structure of the data
- Figure 8.16 illustrates the document type definition for the Customer XML document

```
<!DOCTYPE   customers   [
<!ELEMENT   customers                 (customer)+>
<!ELEMENT   customer                  (name, address, current_balance, order*)>
<!ATTLIST   customer number           ID #REQUIRED>
<!ELEMENT   name                      (lastname, firstname, middle_initial?)>
<!ATTLIST   name type                 (I|C) #REQUIRED>
<!ELEMENT   lastname                  (#PCDATA)>
<!ELEMENT   firstname                 (#PCDATA)>
<!ELEMENT   middle_initial            (#PCDATA)>
<!ELEMENT   address                   (street, apartment?, city, state, zip, country)>
<!ELEMENT   street                    (#PCDATA)>
<!ELEMENT   apartment                 (#PCDATA)>
<!ELEMENT   city                      (#PCDATA)>
<!ELEMENT   state                     (#PCDATA)>
<!ELEMENT   zip                       (#PCDATA)>
<!ELEMENT   country                   (#PCDATA)>
<!ELEMENT   current_balance           (#PCDATA)>
<!ELEMENT   order                     (order_number, order_date, ship_date, total)>
<!ATTLIST   order customer_number     IDREF #REQUIRED>
<!ELEMENT   order_number              (#PCDATA)>
<!ELEMENT   order_date                (#PCDATA)>
<!ATTLIST   order_date format         (mmddyyyy|yyyymmdd|ddmmyyyy) #REQUIRED>
<!ELEMENT   payment                   (check|credit_card)>
<!ELEMENT   check                     (check_number)>
<!ELEMENT   credit_card               (credit_card_number, expiration_date)>
<!ATTLIST   credit_card type          (M|V|A|D|O) #REQUIRED>
<!ELEMENT   credit_card_number        (#PCDATA)>
<!ELEMENT   expiration_date           (#PCDATA)>
<!ELEMENT   payment_date              (#PCDATA)>
<!ATTLIST   payment_date format       (mmddyyyy|yyyymmdd|ddmmyyyy) #REQUIRED>
<!ELEMENT   payment_amt               (#PCDATA)>
    ]>
```

FIGURE 8.16

A document type definition for the customer XML document.

- Figure 8.16
- A Document Type Definition
- For the Customer XML Document

XML Schemas

A Schema

- In computer programming:
  - A *schema* (pronounced SKEE-mah) is the organization or structure for a database
  - The activity of data modelling leads to a schema
- In another usage derived from mathematics:
  - A *schema* is a formal expression of an inference rule for artificial intelligence (AI) computing

XML Schemas

- An *XML schema*:
  - is a description of a type of XML document
  - Typically expressed in terms of constraints on the structure and content of documents of that type above and beyond the basic syntactical constraints imposed by XML itself
- An *XML schema* is:
  - A more precise way to define the content of an XML document
  - Includes exact number of times an element may occur
  - Includes type of data within elements

Summary

Chapter #8 Summary

- In this chapter we have considered:
  - Creating the data dictionary
  - A data dictionary is a specialised application of dictionaries used in everyday life
  - A data dictionary is a reference work of data about data (*metadata*) and may be used to:
    1. Validate the DFD for completeness and accuracy
    2. Provide a starting point for developing screens and reports
    3. Determine the contents of data stored in files
    4. Develop the logic for data flow diagram processes
    5. Create XML

Chapter #8 Summary

- Data dictionaries are useful:
  - At all stages of systems analysis and design
  - Documentation
- Data dictionaries is the authoritative (definitive) source of how a data element is used and defined by system users
- Many large systems feature computerised data dictionaries that cross-reference all programs in the database using a particular data element
- The data dictionary is used to create XML enabling organisations with different systems, software, or database management systems to exchange data

Chapter #8 Summary

- Automated data dictionaries (part of the *CASE* tools):
  - Enable data items to be *cross-referenced*
  - Allowing necessary program changes to all programs that share a common element
- Automated data dictionaries are a vital component in large systems that produce many thousands of data elements requiring cataloguing and cross referencing
- A large collection of project information is termed a *repository*
  - CASE tools enable the creation and sharing of information including:
    - Data flows / stores / record structures / elements / procedural logic / screen / report design / data relationships
  - A repository may contain project requirements information and final system deliverables