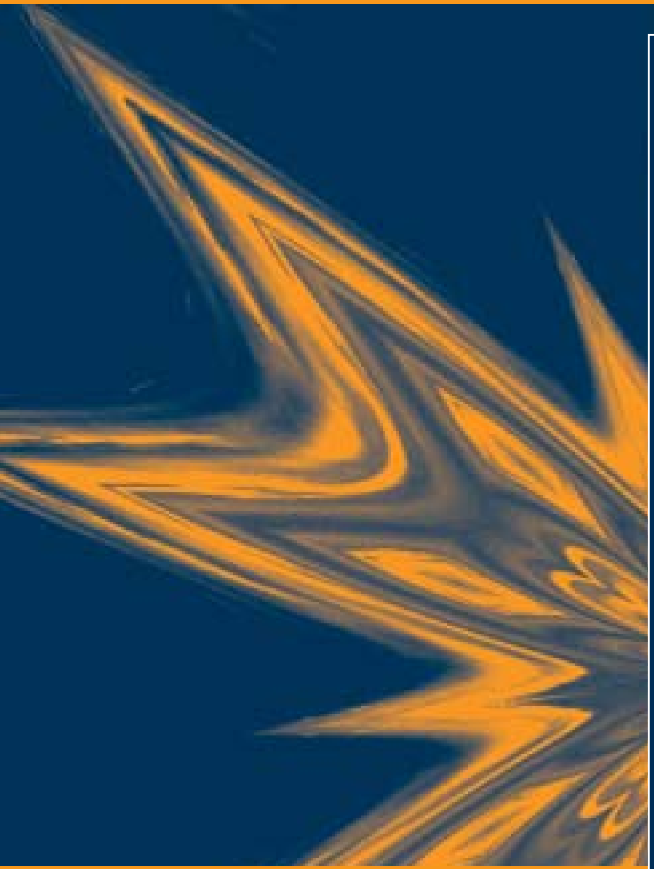
c h a p t e r

3 Documentation techniques and databases



Learning objectives

- To learn about data flow diagrams (DFDs) and the symbols used in them.
- To understand a context diagram and its decomposition.
- To understand the standard symbols used in flow charts.
- To describe and construct different types of flow charts.
- To understand the advantages of flow charts over written descriptions.
- To be able to compare and contrast DFDs and flow charts.
- To understand decision tables and their relevance to programming.
- To learn about the construction of E-R diagrams as a basis for databases.

the story

Having learned the workings of a good accounting system, David and Mary thought that it would be a good idea to document all the accounting processes and the various flows of data so that anyone, at any time, would know what tasks were carried out in the business, what data were used and what processes were used to convert data into information. David and Mary were quick to realise that proper documentation of these elements would help any new employees in particular to familiarise themselves with the business's procedures. They were convinced that anyone should be able to read the documentation and understand how the system works. David and Mary had heard about the existence of certain diagrams used for this type of documentation and they immediately embarked on a mission to explore these diagrams.

They perused online software reviews over the Internet and found that *ABC Flowcharter* was the most popular software for flow charting, and apparently the simplest to use. They purchased a copy and set to work on mapping the various processes within the business. It was a little time consuming and tedious at first, but they persevered until they were happy with the results. They discovered that there was much more involved in operating the business than just buying and selling furniture. Interestingly, they found that there were numerous, critical relationships that needed to be managed effectively with both internal and external stakeholders. Although the customer relationship philosophy was not new to David, he was particularly surprised at how much more to business there was than just the customers. Therefore, the process of mapping accounting and information flows helped them to identify potential weaknesses in their operations that could interrupt the smooth running of the retail side of their business.

During the process of charting data and information flows, David and Mary observed these important details:

1. Job design is very important to overall business performance.
2. Financial separation of duties helps to reduce the possibility of internal fraud.
3. Too much information is not helpful for effective decision making.
4. Employees need access to a reliable reporting mechanism so that the owners can keep abreast of developments in the business as they occur.
5. A good employee induction program, incorporating the relevant process flows, will help to retain new staff and ensure that they become operationally proficient in their roles more quickly.
6. Process modelling also helps to identify new income opportunities and future growth alternatives such as backward or forward vertical integration.

After completing the process modelling exercise, David and Mary were so happy with their results that they visited Roddney the CPA again to tell him what they had accomplished. He was very pleased to see that his friends had taken the initiative and completed such an important task. Roddney explained that many new small businesses fail within the first year or so primarily because they do not fully comprehend their own business models and processes. He also concluded that their bank manager would look more favourably on any future funding applications they made because he or she would be confident that they were competent business proprietors who were serious about their enterprise.

key terms

attribute	A named property or characteristic of an entity that is of interest to a business organisation
binary notation	A number system with only two digits—0 and 1
bit	A binary digit—0 or 1
byte	A configuration of eight bits used to represent a single character of information
context diagram	An overview of an organisational system showing system boundaries, external entities and major information flows
data dictionary	The repository of all data definitions for users and programmers
data flow	Data in motion, moving from one place in a system to another
data flow diagram	A picture of the movement of data between external entities, processes, and data stores within the system
data model	A documentation technique, such as an E-R diagram, for organising and documenting a system's data
data store	The repository where data at rest is stored
data structure	The relationship between data items
data warehouse	A database constructed for quick searching, queries, and retrieval
database	A physical repository for financial data
database management system	Software that permits utilisation of the database approach to data management
decision table	A matrix representation of the logic of a decision, which specifies the possible conditions for the decision and the resulting actions
decomposition of a data flow diagram	Breaking the description of a system down into more and more detail. This creates a set of diagrams in which one process in a given diagram is explained in greater detail on another diagram
documentation	A written or diagrammatical description of how a system works
entity	A resource, event or agent
E-R diagram	A documentation technique used to represent the relationship between entities
flow charts	Charts that show the flow of data, documents and so on in a business using standard symbols
flow line	A flow chart symbol used to represent the connecting path between flow chart symbols
foreign key	An attribute that appears as a non-key attribute in one relation and as a primary key attribute in another relation
lead time	The time lapse between ordering goods and receiving them
primary key	Characteristics that uniquely identify each record in a database table
relational database model	A data model that represents data in the form of tables or relations
relationship	An association, between the instances of one or more entity types, that is of interest to an organisation
schema	A logical description of the entire database
secondary key	Characteristics that identify groups of records with a common attribute in a database table
subschema	A logical description of a portion of a database

introduction

If we need to implement a new accounting information system, or improve an existing accounting information system, we should carefully plan this operation. In the planning process, the system needs to be documented. This **documentation** should be a detailed report of the input, processes and output of the system. The system needs to be decomposed into the lowest level of subsystems for the documentation process to be successful.

It may be difficult for a reader of these descriptive documents to comprehend the processes and flows referred to. Therefore, the documentation is done through a series of drawings or diagrams that make the processes and flows easier to understand. However, a series of drawings on their own may not fully explain the situation. It is best to add descriptive words to the diagrams. The number of words must be restricted to a minimum so as to maintain the flow of information one receives from the drawings. Further, the drawings use standard symbols so that a particular process, document or flow will be clearly expressed without any ambiguity.

There are various angles from which a system is viewed depending on who is doing the viewing. A holistic picture is needed to understand the system's input, processing and output as well as its relationship to the environment. However, a system needs to be broken down into its subsystems in order to design and implement it. In this process the basic concepts, such as internal control, need to be taken care of. To begin the process, the logical flow of data should be decided for the planned system and documented on a data flow diagram. In the case of a project to improve an existing system, it is also better to start with a series of data flow documentations.

Data flow diagrams

In any organisation, various data are collected and processed to obtain useful information on an item of interest. For example, data on hours worked by an employee is collected and processed in the payroll section by applying a rate of pay, making appropriate deductions for tax payable, superannuation contributions and so on, to derive net pay information. The data on hours worked comes in the form of timecards from the employee's department, and other required data, such as rate of pay, comes from the human resources department. Therefore, these departments become data sources. The process transforms the data. In this case, the process is the calculation and production of payroll cheques, and other cheques to the Australian Taxation Office and a superannuation fund. The cheques are then sent to the bank, the Australian Taxation Office and the superannuation fund. These recipients are data destinations. Data may need to be temporarily or permanently stored. The repository for such storage is referred to as a **data store**.

It is advantageous to use simple diagrams to represent how various data move between sources and destinations in an organisation. These diagrams are called **data flow diagrams**. A data flow diagram describes graphically the flow of data within an organisation and is composed of four basic elements which are represented by standard symbols as follows:

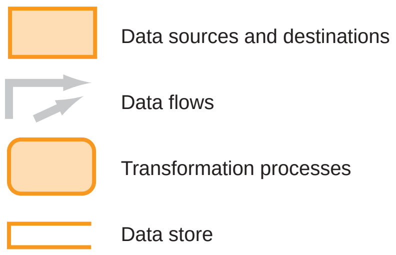


Figure 3.1 depicts the relationship between the data source (which is an organisation or part of an organisation such as a department or people that send data to a system); the data destination (which can be entities similar to data sources); the process (which transforms data from inputs to meaningful data output); **data flows** (the flow of data into and out of a process) and the data store (which stores data temporarily or permanently for future use, reference or processing).

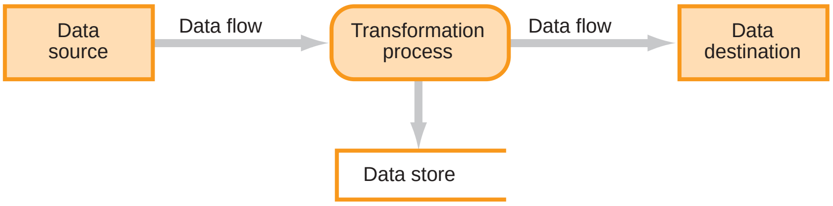


Figure 3.1 A basic data flow diagram

The planned sales system of P & S Furniture Mart

Let us look into a section of the planned sales system of P & S Furniture Mart. It will stock mainly the sample floor stock items of furniture for display purposes, which will be used to attract customer orders. Generally, there will not be further items in stock unless they are low in cost and fast moving. If the manufacturer of a high-cost item has extra stock and is willing to supply that extra stock to P & S Furniture Mart on the condition that stock items will be paid for only after sale, then they are accepted.

If a customer orders an item, the availability of the item will first be checked from stock records and, if the item is in stock, a delivery date will be advised to the customer. If the item is not available, an estimated delivery time will be advised to the customer, after considering the supplier's **lead time**. Sometimes the floor stock may be sold. This is dependent on certain conditions. In all circumstances, to confirm the sale, an advance will be collected from the customer and, apart from certain preapproved customers, they will be expected to pay for the furniture on delivery. The money collected will be banked on the same day to prevent large sums of money being left in the premises overnight.

Figure 3.2 shows the data flow diagram for informing a customer about delivery of their furniture. Note that both the data source and the data destination are the customer. The data stores are stock records and recorded information on lead times for various items from different suppliers. This data flow diagram easily explains the flow of data and the processes involved. It is possible that there could also be data flows after deciding on the delivery date of the items ordered by the customer, delivery dates to the customer, the prices of items to various data destinations such as the accounts department (to collect the required advance from the customer) and the purchasing department (to order the required furniture if not in stock). These are not shown in this diagram for the sake of simplicity.

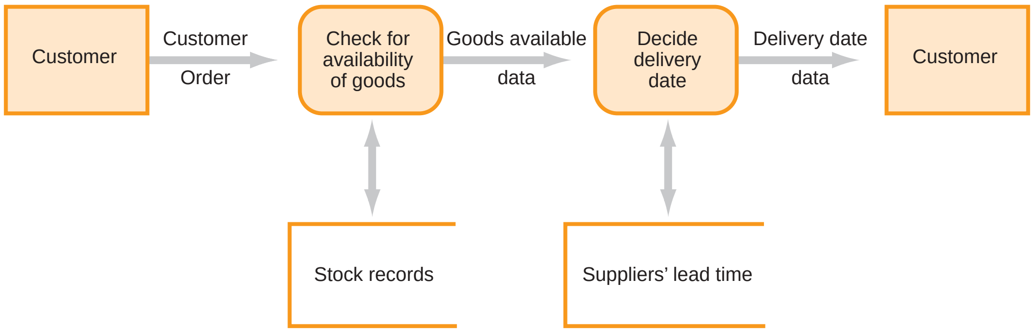


Figure 3.2 A data flow diagram for informing a customer about delivery dates

Having shown, on the data flow diagram, the process involved in informing a customer about a delivery date (which is only a subsystem of the planned sales system at P & S Furniture Mart), it would be beneficial to provide an outline of the whole system on a single diagram, called a 'context diagram'.

Context diagrams

A **context diagram** shows major data flows into and out of a system. Therefore, detailed data flows within the system are not shown on this type of diagram. A context diagram describes each subsystem as a process thus showing the interrelationship of those subsystems, and their relationship to the main system. In designing a system, the relationships should be clearly defined so as to smooth out the data flows without causing bottlenecks, and to maintain internal control. This also helps when designing suitable interfaces to absorb the fluctuations in the volume of data by creating temporary storage at the interfaces.

For example, using a number of order books in which the sales staff take down the details of customer orders alleviates the need for each salesperson to wait for the orders to be processed by an accounts clerk before attempting to collect the advance from the customer. The salesperson leaves the book on the accounts counter where the customer pays the advance and uses another book to take orders from other customers. Thus the salesperson's time is efficiently used in attending to the customer rather than waiting at the counter for the order books to be cleared.

The context diagram for the planned sales system of P & S Furniture Mart is shown in Figure 3.3. The customers are systems external to the planned sales system of P & S Furniture Mart. Customers will have their own process for deciding what to buy after considering their financial situation, their needs and their preferences. For our purposes, they are the originators of the data flow when they convey their decision to buy a particular piece(s) of furniture and then when they make payments. The customers are data destinations for receiving information on the delivery date and the invoice. Also, the decision makers, such as the sales manager and finance manager, become the data destination for information on sales. The selling function—including informing the customer about delivery dates and collecting money from customers—is the process.

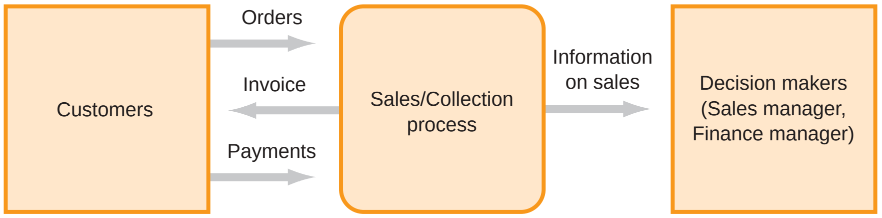


Figure 3.3 A context diagram for P & S Furniture Mart detailing the planned sales system

The context diagram depicts the logical flow of data in a system in a summary form. It should be noted that a context diagram is also basically a data flow diagram and is the starting point for studying any system. A context diagram depicts the system at the highest level. This level is normally referred to as a level zero data flow diagram when each of the processes involved is referred to on the diagram by the numbers 1.0, 2.0, 3.0 and so on. More details are introduced by decomposing the level zero diagram to a level one diagram. For example, when process 1.0 is decomposed, the next level of processes would be referred to as 1.1, 1.2, 1.3 and so on. When process 1.1 is further decomposed, it would be represented by 1.1.1, 1.1.2, and 1.1.3. Further levels of decomposition can occur depending on the degree of detail required. This action of expanding each process with a more detailed data flow diagram is called **decomposition of a data flow diagram**.

Figure 3.4 shows the level zero data flow diagram of the planned sales system for P & S Furniture Mart. Note that it differs from Figure 3.3 (a context diagram) in that

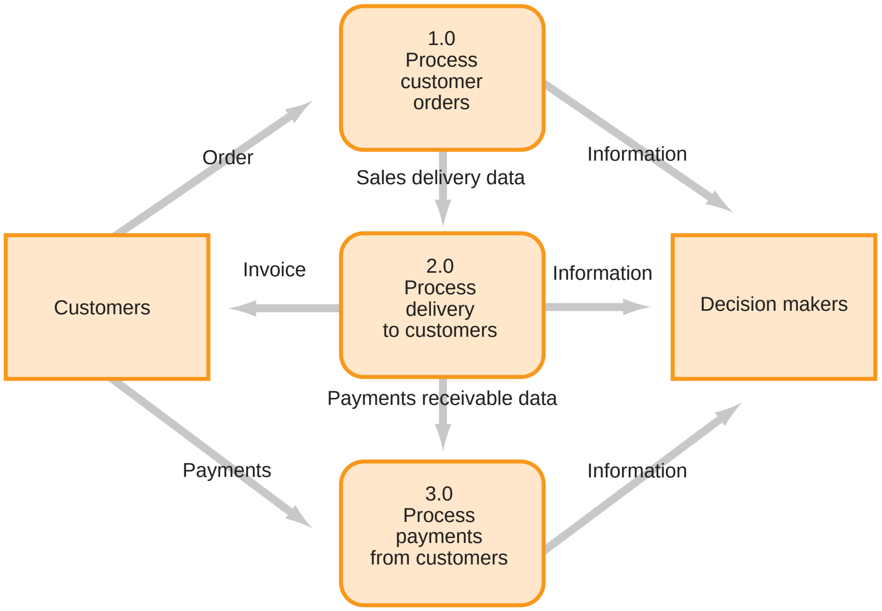


Figure 3.4 A level zero data flow diagram detailing the planned sales system of P & S Furniture Mart

the sales/collection process is broken down into three processes: customer orders, delivery to customers and payments from customers. These processes are referred to by the numbers 1.0, 2.0 and 3.0 respectively to indicate the level zero data flow diagram. The data flow diagram of customer order processing for the planned sales system of P & S Furniture Mart has two subprocesses represented by 1.1 and 1.2. This is shown in Figure 3.5

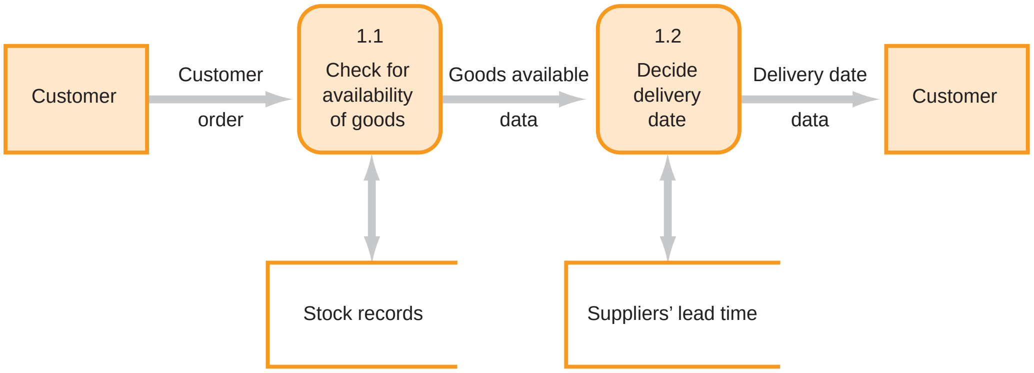


Figure 3.5 A data flow diagram showing subprocesses involved in customer order processing

Flow charts

Once the logical flow of data has been determined and documented using data flow diagrams, we need to determine the physical flows of various items in the system, such as data and documents. These physical flows, when documented in the form of a diagram, are referred to as a **flow chart**. This is a diagrammatical version of the physical operating processes of a system. Flow charts are constructed not only for developing new systems but also for analysing existing systems with a view to improving them. The diagrammatical representation of an existing system helps to identify the flaws, such as a lack of internal controls and bottlenecks, and remedy them.

To simplify understanding of flow charts most of the symbols are standardised. The commonly used symbols and the meaning they convey are shown in Figure 3.6. There may be other symbols, or the symbols shown here may have a different meaning in a particular organisation, but the meanings of the symbols will be intimated to the people who draw the flow charts and those who use them. Descriptive words (or narratives) are added to the symbols to make the charts more understandable. The sequential order of the flow of data and documents is well represented in flow charts. The style used for drawing flow charts differs from person to person. The style is not standardised like the symbols but the same style is used consistently within an organisation.

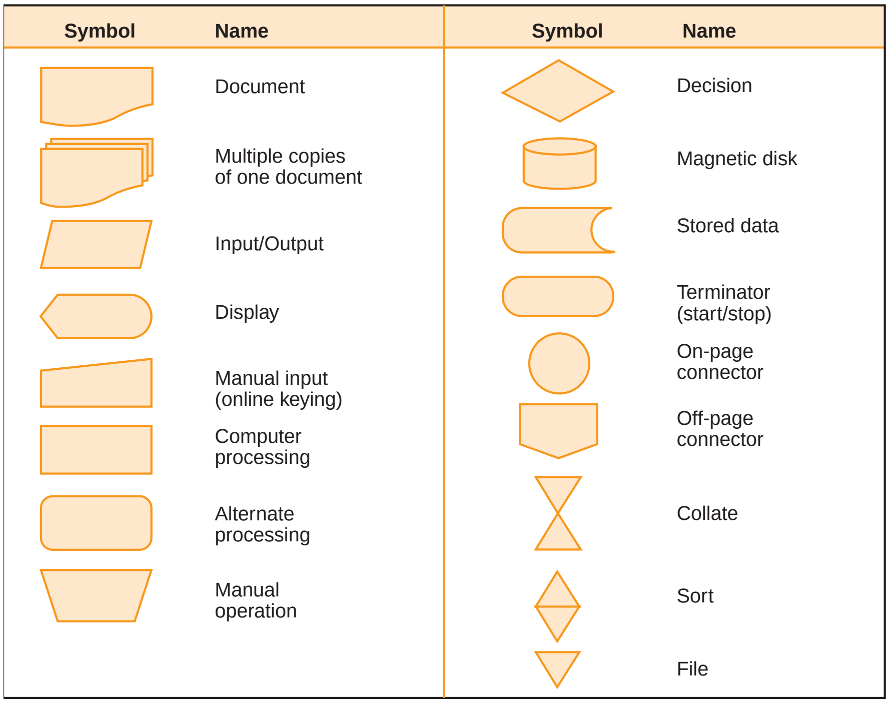


Figure 3.6 **Flow chart symbols**

Advantages and disadvantages of flow charts

The following are the advantages of representing a system as a flow chart:

- A flow chart enables any system, especially a complex system, to be represented in an easily understandable manner.
- The overall picture of the system is easily seen.
- The relationship among different parts of the system is highlighted.

The disadvantages in representing a system as a flow chart are:

- The creation of a flow chart can be time consuming.
- Numerous symbols can cause confusion.
- A process may not be completely representable using symbols.

The user of a flow chart will counterbalance the time spent in creating a flow chart by the saving of time in understanding the system. Confusion regarding the symbols can be avoided by giving a brief description of the process involved alongside the symbols, or by providing cross-references at the foot of the flow chart—these descriptions help in fully understanding the process involved.

Differences between data flow diagrams and flow charts

Though both data flow diagrams and flow charts depict the flow of data, documents and processes in a diagrammatic form, there are several differences between them as listed below:

1. While a data flow diagram shows the flow and processing of data in a system, a flow chart generally emphasises the flow of documents containing data, the processing of those data and the creation of new documents (when necessary) containing data.
2. The logical flow of data is the theme of data flow diagrams. This helps users of a system to conceptualise the relationships of data items and what happens to them. It shows the source (where the data originates), its destination (where it ends), the process the data undergoes between these two entities and the storage of data temporarily or permanently in the stages of the process. A flow chart shows the physical flow of data, the documents in which it is contained, the processing of data, and the creation of new documents (if any) containing data. It also details the physical aspects of the system, such as manual or computer processing, storage media and file organisation.
3. The sequence of data flow and processing is not emphasised in data flow diagrams but flow charts always emphasise this sequence.
4. When drawing data flow diagrams only four symbols are used. Flow charts use many symbols to pictorially represent more details.

System flow charts

The flow chart which shows the relationship between input, processing and output including data, documents and storage, is known as a system flow chart. A system flow chart pictorially represents the relationship between various processes. The **flow lines** are used to represent the inputs to and outputs from processes. The processes include computerised and manual operations. Therefore, system flow charts thoroughly illustrate the information system.

Figure 3.7 shows the system flow chart for the planned sales system of P & S Furniture Mart. For the sake of simplicity, the system depicted in the chart starts with the decision made by the customer and ends with the customer receiving an acknowledgment of the order, an invoice and a receipt for the payment made. It should be noted that the system flow chart shows the physical workings of the system in more detail than a data flow diagram.

The details shown in Figure 3.7 are rearranged in Figure 3.8. Note that three columns are introduced to add clarity to the flow chart by dividing the actions carried out by the three different groups, namely, customer, salesperson and cashier. The columns need not only refer to people but can also represent the tasks carried out by the departments. This kind of representation, which adds the dimension of activities or tasks carried out by different people or departments, helps in identifying the segregation of duties, which is the primary aim of internal control. Since the system flow chart is the only flow chart that describes a system thoroughly, later chapters of this book will refer to system flow charts simply as flow charts. Also, the flow charts will have the columns added to differentiate the activities or tasks carried out by different people or groups of people.

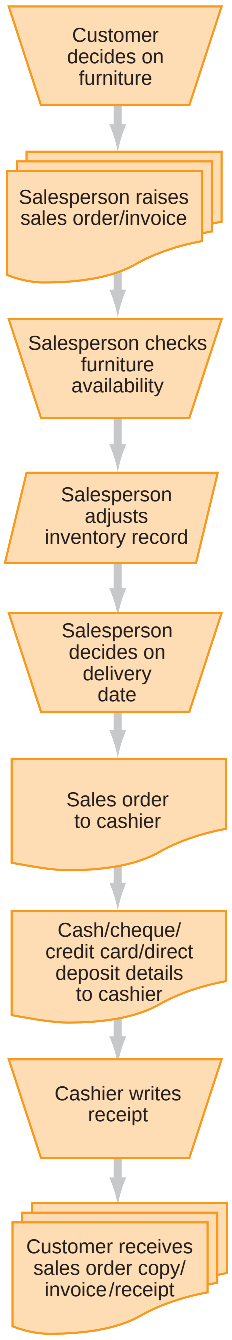

Customer decides on furniture
Salesperson raises sales order/invoice
Salesperson checks furniture availability
Salesperson adjusts inventory record
Salesperson decides on delivery date
Sales order to cashier
Cash/cheque/ credit card/direct deposit details to cashier
Cashier writes receipt
Customer receives sales order copy/ invoice/receipt

Figure 3.7 A system flow chart for the planned sales system of P & S Furniture Mart

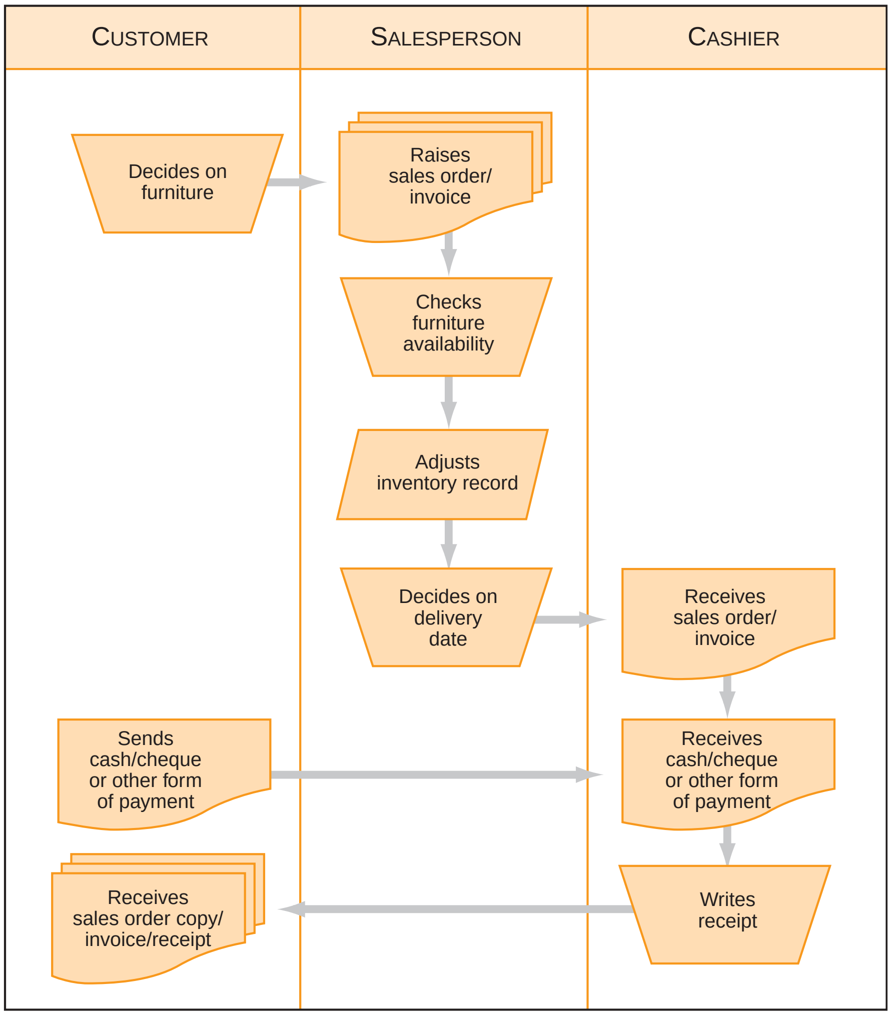

CUSTOMER
SALESPERSON
CASHIER
Decides on furniture
Raises sales order/ invoice
Checks furniture availability
Adjusts inventory record
Decides on delivery date
Receives sales order/ invoice
Sends cash/cheque or other form of payment
Receives cash/cheque or other form of payment
Writes receipt
Receives sales order copy/ invoice/receipt

Figure 3.8 A system flow chart (modified)

Document flow charts

A system flow chart, which includes not only the documents used by the system but also procedures and processes involved in the system, may not be necessary in many situations. The processing details may be a distraction when one needs to have only a general overview of the data flow. A system flow chart with all the procedures and processes removed is called a document flow chart. A document flow chart only emphasises the flow of documents between various people, groups of people and the departments of an organisation. More detail can be achieved by dividing the chart into columns to represent the different people, groups and departments.

The document flow chart in Figure 3.9 represents the flow of documents shown in the system flow chart in Figure 3.8. This document flow chart provides a good overview of information processing by avoiding a lot of unnecessary details. The reader, by using experience, and applying common sense to the information provided by the document flow chart, may be able to better understand what occurs at each stage of the system.

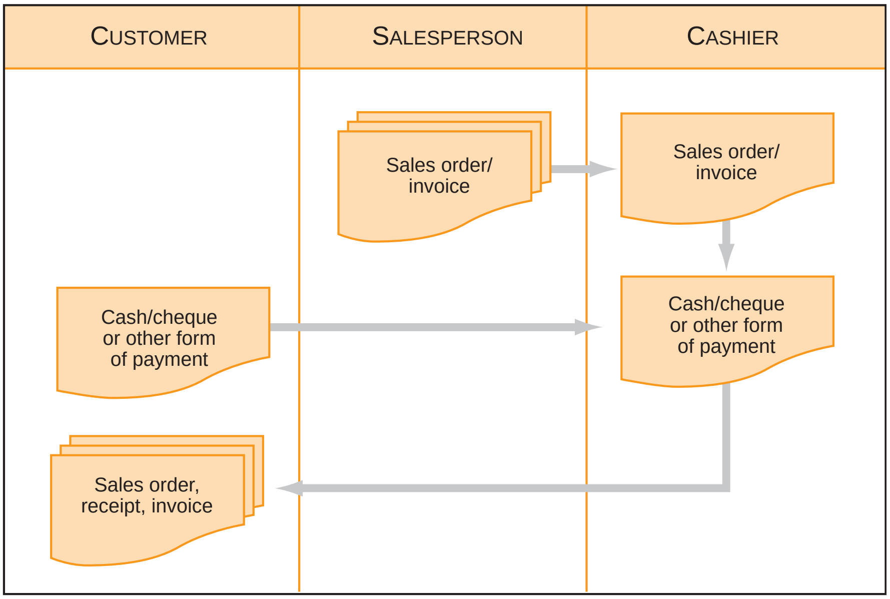

CUSTOMER
SALESPERSON
CASHIER
Sales order/ invoice
Sales order/ invoice
Cash/cheque or other form of payment
Cash/cheque or other form of payment
Sales order, receipt, invoice

Figure 3.9 A document flow chart

Program flow charts

Although system flow charts describe a system in a high level of detail they do not show how individual computer programs, which form part of the system, work. To overcome this, the detailed steps of a computer program are shown in another chart called a program flow chart. Program flow charts typically show the logic and processing steps that may be used to develop computer programs. These program flow charts are examined when evaluating the internal controls of a computer-based accounting system. Figure 3.10 shows a program flow chart. This flow chart uses the planned sales system of P & S Furniture Mart but only details the decision-making process regarding whether full sale value or only an advance is to be collected from the customer.

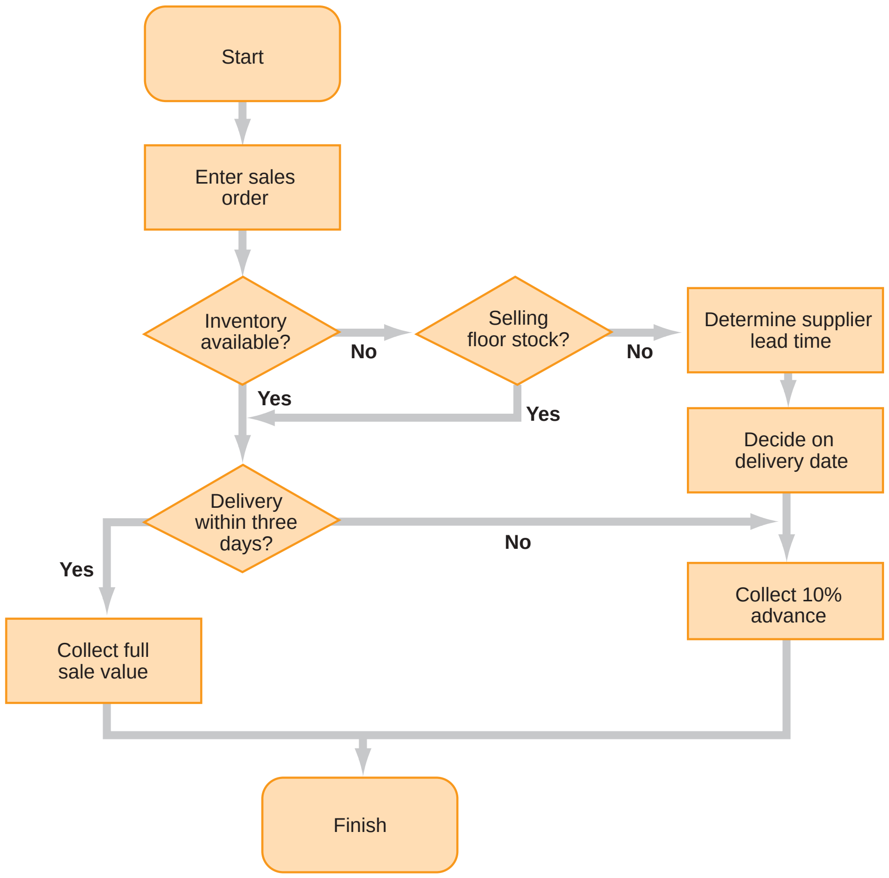


Figure 3.10 **A program flow chart**

Decision tables

The program flow chart may not be suitable to describe a complex decision process in a summary format. An alternative method used under these circumstances is a **decision table**. A decision table lists in tabular form the decision logic in a program flow chart. A decision table is constructed by creating two main columns and two rows. The first main row is further divided into a number of rows of conditions. The second main row is further divided into a number of rows of actions. The second main column is further divided into a number of columns—each representing a decision rule. The decision rule is a combination of conditions and the corresponding action taken. The conditions are either stated as 'Yes', representing that the condition is satisfied, or 'No', representing that the condition is not satisfied. An 'X' is put against the action to be taken. A decision table should list all the possible combinations of conditions for making a decision and the corresponding action outcome. It should be noted that the sequence of events, as portrayed in a program flow chart, is sacrificed in the decision table to simplify the way the decision process is expressed.

	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5
Conditions					
Stock available?	No	Yes	Yes	No	No
Selling floor stock?	No	—	—	Yes	Yes
Delivery within three days?	—	No	Yes	Yes	No
Actions					
Collect full sale value			X	X	
Collect 10% advance	X	X			X

Figure 3.11 A decision table

Figure 3.11 shows a decision table of the program flow chart in Figure 3.10. The decision table tells us the following:

1. If stock is not available and floor stock is not being sold, collect 10% advance.
2. If stock is available but delivery is not within three days, collect 10% advance.
3. If stock is available and delivery is within three days, collect full value of sale.
4. If stock is not available but floor stock is being sold within three days, collect full value of sale.
5. If stock is not available but floor stock is being sold after three days, collect 10% advance.

Computerised data processing

Learning to flow chart the activities of an organisation leads us into learning about capturing, storing and using data. Most organisations extensively use computer systems as tools for processing their data. The data is represented in a computer by electrical pulses that have two states, either magnetised (energised) or not magnetised. The data is transferred through communication links by these electronic pulses. These pulses are represented by **binary notation** which uses the digit '1' to represent the existence of a pulse and the digit '0' to represent the non-existence of a pulse. This pulse is termed a **bit**. When computer hardware is designed, eight bits are used to represent a human identifiable character called a **byte**.

A computer word is a group of bits or bytes that may be manipulated and stored as a unit. The internal circuitry of a computer is designed around word size. For example, the 32-bit word machine can work comparatively faster than an 8-bit word machine, as the word travels as a unit and thus carries more volume of bytes.

For data processing purposes, a number of bytes are grouped together into a field that stores a single piece of data. The code that refers to a particular piece of furniture in P & S Furniture Mart is an example of a field. A group of fields form a record in a computer. The record is equivalent to a document in a manual system. An example of a record in the computer system is an inventory record that will have the furniture code, its description, cost price, selling price and stock availability. A number of records of a similar nature form a file.

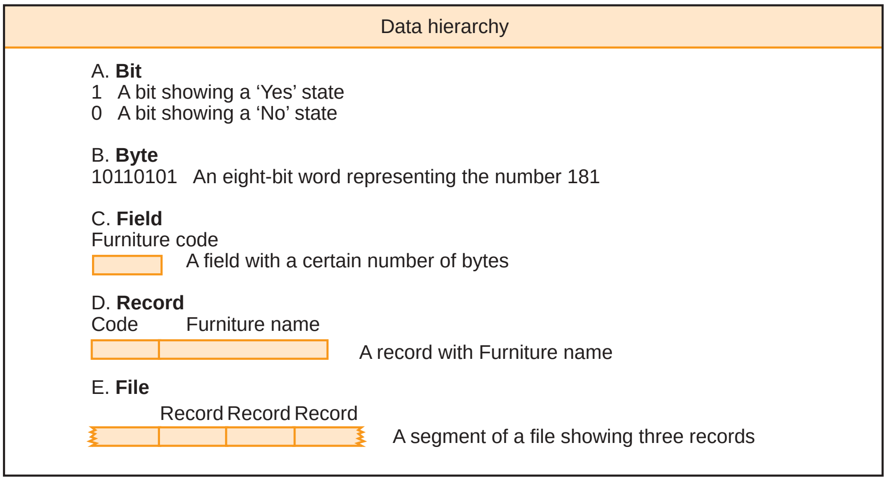

Data hierarchy
A. Bit
1 A bit showing a 'Yes' state
0 A bit showing a 'No' state
B. Byte
10110101 An eight-bit word representing the number 181
C. Field
Furniture code
A field with a certain number of bytes
D. Record
Code
Furniture name
A record with Furniture name
E. File
Record Record Record
A segment of a file showing three records

Figure 3.12 The relationship of a bit to a file

The application programs act as the manipulator of data for the needs of the computer system user. Each application program carries out a certain set of functions. They act as intermediaries for processing and transferring the data input to a storage location. They also perform the function of retrieving the data from storage and processing them for output as required. For example, an application program collects the information on the item of furniture sold and the quantity sold and stores that in an inventory file. The same application program is used to extract information on the number of items of a particular piece of furniture available in stock at a given point in time. There can be many files in a particular computer system which store different data. The application programs interact with different files depending on the particular situation. Figure 3.13 shows the interaction of an application program and data files.

Maintaining the different files may result in the same information being repeatedly stored in different files. This will lead to resources being wasted in maintaining those files, and the processing time will be comparatively high. A second problem arises in circumstances where only one file is accessed and updated by one user at a time, which precludes the same information from being updated in the other relevant files containing that information. For example, while the file maintained by the accounts department of P & S Furniture Mart records the quantity of a particular item of furniture sold, the stores clerk may inadvertently fail to process the same information in the inventory file.

Again, a difficulty arises when there is a necessity to interrogate a number of files to get common information needed by a decision maker. These problems can be overcome by using an integrated collection of records and files, referred to as a database.

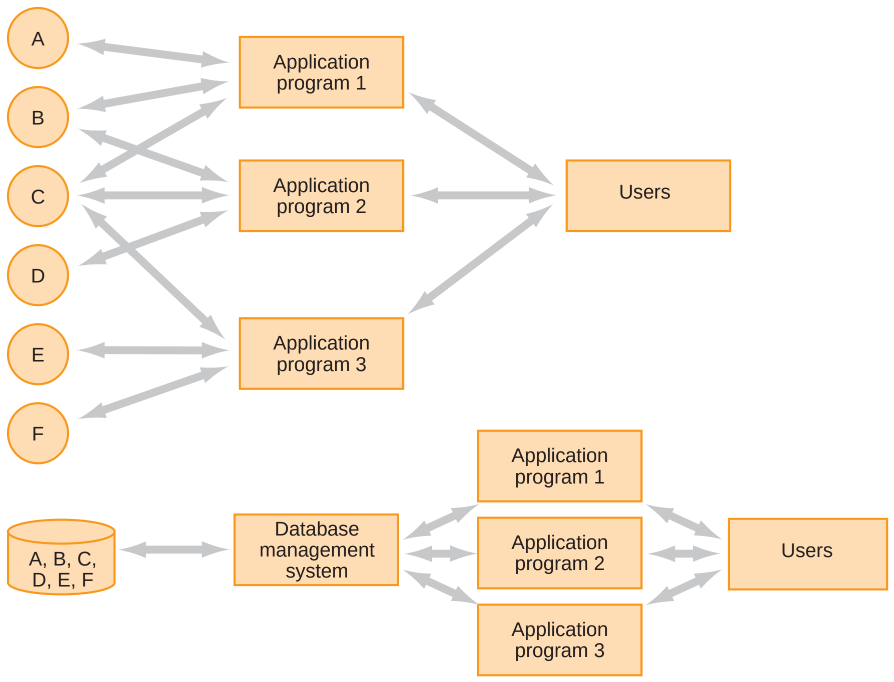


Figure 3.13 Organisation of data files and database management

Databases

A **database** is the actual physical repository for an organisation's data. A special computer program that only enables authorised users to create and update files, select and retrieve data, and generate various outputs is called a **database management system** (DBMS). The database management system acts as an interface between the database and the application programs.

The term **data structure** is used to describe the technique of physically arranging records in a database. A database constructed for ease of use, and to answer ad hoc queries, is called a **data warehouse**. When constructing a database, it is usual to have a document that describes each data element in the database. This document is called the **data dictionary**. The use of a data dictionary by programmers reduces the risk of having one item stored as two different items.

Developing a database

A database management system segregates the view of the users of the data from the data actually stored on the storage device. One way of describing a database management system is from the point of view of programmers and users. The other way is from the point of view of how data is physically stored. Data management software determines the physical arrangement of data after consideration has been given to quantifying the efficiency and effectiveness with which the data is extracted from the database by the database management system for the use of the associated application programs.

The application program needs to prepare the data in a prearranged structure in order for it to manipulate the data by additions, deletions, and so on, and for reporting purposes. The logical structuring of the data for the purposes of an application program is called a **subschema**. The subschema is generally different for each application program. The structure of the relationship between all the record types of a database is called a **schema**. For security reasons, the schema is not divulged to the person who develops an application program. As opposed to schema, subschema show only some of the records and their relationship in a database. While there may be a number of subschema depending on the needs of the organisation, there will be only one schema for a database. Examples of schema and subschema are given in Figure 3.14.

Schema for inventory records for P & S Furniture Mart

Furniture code
Description
Dimensions
Colour
Supplier
Quantity purchased
Quantity sold
Quantity in stock
Quantity on order
Cost price
Selling price

Subschema for furniture availability inquiry

Furniture code
Description
Dimensions
Colour
Selling price
Quantity in stock

Subschema for furniture sales analysis

Furniture code
Description
Colour
Cost price
Selling price
Quantity sold

Figure 3.14 **Examples of schema and subschema**

Types of schema or database models

The schema, or logical data structure, is how the database management system searches through the database to identify and retrieve a particular record. Different database management systems use different structures depending on the needs of the particular system. The model of the structure can be hierarchical, network or relational. The selection of the model is based on the content of the database and the general relationship of the data items.

The hierarchical (or tree) model occurs when the data is organised in a top-down structure in a one-to-many relationship. When this structure is represented from the bottom up, it resembles a tree and, therefore, it is also called the tree model. Figure 3.15 shows a hierarchical model. In this figure, information about a supplier forms the beginning of the hierarchical database model. Two furniture items, 'Furniture A' and 'Furniture B', are supplied by this supplier and, in turn, two particular colours are used in these furniture items. Note that in this model each data element at a data level is linked to only one element at the next data level up. But, they may be linked to more than one element at the next data level down. This relationship is referred to as the parents and children relationship—because a parent can have any number of children but not the other way around. The database management system searches for data through a single record using this relationship to extract information for a subschema.

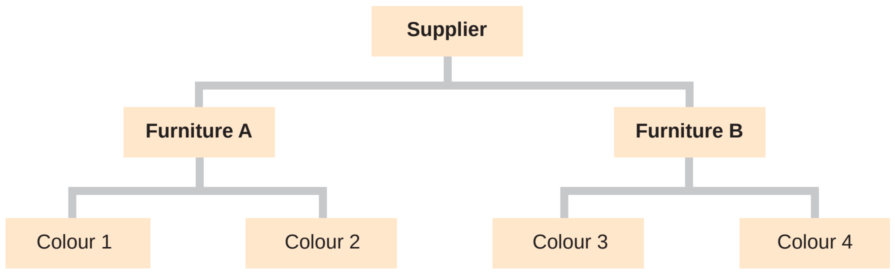


Figure 3.15 A hierarchical database model

The network model is an extension of the hierarchical model but instead of having various levels of one-to-many relationships, this model takes the form of a many-to-many relationship. Figure 3.16 shows a network model where the two suppliers have a supply relationship with many furniture items. As opposed to a hierarchical model, the search for data elements is multipronged and comparatively less time consuming. This structure may be suitable for a relatively complicated data relationship.

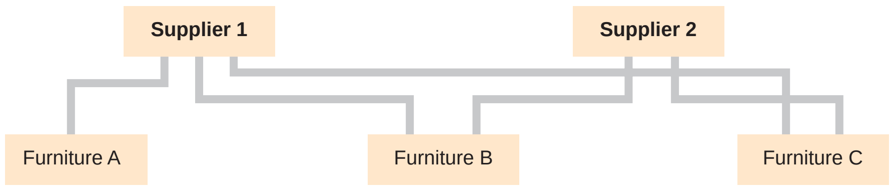


Figure 3.16 A network database model

An inefficient database may involve:

- entering the same data several times when the database is being created
- having to delete data from several areas in the database when a single change (update) needs to be made

- having to insert data in several areas in the database when a single piece of information is added
- having to access many different areas of a database to get a single piece of information.

The cost of these inefficiencies is usually related to processing time (the computer takes a long time to produce organised information) and maintenance time (the user has to spend hours entering data when minor changes need to be made).

The hierarchy and network models may display the inefficiencies described above. A **relational database model** may be the solution that overcomes these inefficiencies. The relational model is a logical data structure that views the database as a collection of two-dimensional tables called database tables. The columns of the table represent the fields and the rows represent the records. The column headings are called **attributes**. Each column in a table is uniquely named. However, different tables may contain columns with the same name. Figure 3.17 represents a table of a furniture database where details of furniture items are stored.

Furniture code	Furniture description	Supplier code	Cost price
001–38	SOFA–2 SEATER	D 723	$372
001–53	SOFA–3 SEATER	D 723	$502
001–79	SOFA BED	N 054	$400

Figure 3.17 A table within a furniture database

The furniture code, a unique identification code of a furniture item, is an attribute of the table above. This attribute, being unique for furniture, can be used as the **primary key** field of this record. The primary key is used to identify a record in a particular file. A **secondary key** field is used to access or retrieve the desired records. The secondary key value need not be unique. In the above table, the supplier code can serve as a secondary key used for extraction of lists of furniture items supplied by a particular supplier. The supplier details can be found in another table for which the supplier code may be a primary key. If any of the attributes, other than the primary key (i.e. the furniture code), in the above table is to appear as a primary key in another table it is called a **foreign key** in this table. The supplier code is a typical example.

Because the data models in accounting information systems are so complicated, it may be wise to use a relational model to build databases in an accounting information system. Hence, the design of a relational database is detailed below.

Relational database design

Consider a company that has several departments within its structure. Each department has a name and a unique number. Each department is controlled by one

Identifying the relationships between data is the first step in database design

manager, and a particular manager can only control one department within the company. The company would like to keep a record of when the manager of a particular department joined, as well as a list of all the employees and the departments they work in. Each employee can only work in one department. Departments carry out projects, and details of which department works on which project are to be kept in the database. Projects are identified by unique project numbers, as well as by descriptions, and one department can only handle a particular project at a time.

Step 1 In order to organise related data into tables in a relational database it is first necessary to identify the relations between pieces of data. The easiest way to do this is to write elementary sentences that describe these relations:

(a) A project with a *project number* can be handled by only one *department*.
(b) A *department* can handle many different projects (with different *project numbers*).
(c) Each project with a *project number* is given a *description*.
(d) A project *description* can apply to many projects with a *project number* (e.g. a department may be running an advanced and a beginner 'Product Planning' seminar that is given the same description—product planning—but may involve different staff and different agendas).
(e) A *department* is given one unique *department number*.
(f) A *department number* can belong to only one *department*.
(g) A department with a given *department number* can only be managed by one *manager*.
(h) A *manager* can only manage one department with a given *department number*.
(i) A *manager* can manage many *employees*.
(j) An *employee* can only be managed by one *manager*.
(k) A *manager* can only have started managing a department in a particular *year*.
(l) A particular *year* may have been the starting year for many *managers*.
(m) An *employee* can have only one *first name*.
(n) A *first name* can belong to many *employees*.
(o) An *employee* can only work in one department with a given *department number*.
(p) A *department* with a given department number can employ many *employees*.

These elementary sentences may seem trivial and some sentences may seem to be saying the same thing. Each sentence, however, states a **relationship** between two entities within the company. When information about the company is later organised into tables, we will find that the column headings of the tables will be the entities identified here. The entities identified in this example are: project number, project description, department name, department number, manager, employee, year, and employee first name.

Step 2 Although the elementary sentences in Step 1 outline the relationship between pairs of entities, it is useful to explicitly state the nature of these relationships in a more formal way. It is also important to state any constraints (rules) that we want enforced in our database.

We do this by taking two elementary sentences at a time as follows:

(a) A project with a *project number* can be handled by only one *department*.

(b) A *department* can handle many different projects (with different *project numbers*).

→ The relationship between project number and department is a one-to-many relationship. Constraints: If a project number is recorded in the database, we must record the department that is handling the project. If a department is recorded in the database, however, we do not need to record any projects because the department may not have any projects when it first starts.

(c) Each project with a *project number* is given a *description*.

(d) A project *description* can apply to many projects with a *project number* (e.g. a department may be running an advanced and a beginner 'Product Planning' seminar that is given the same description—product planning—but may involve different staff and different agendas).

→ The relationship between project number and project description is a one-to-many relationship. Constraints: If a project number is recorded in the database, we must record a project description to describe it. If a description is to be entered on the database then the project number it applies to has to be stored in the database.

(e) A *department* is given one unique *department number*.

(f) A *department number* can belong to only one *department*.

→ The relationship between department number and department name is a one-to-one relationship. Constraints: If we record a department number, then a department name must also be recorded and vice versa.

(g) A department with a given *department number* can only be managed by one *manager*.

(h) A *manager* can only manage one department with a given *department number*.

→ The relationship between department number and manager is a one-to-one relationship. Constraints: If we record a department number, then a manager must also be recorded, and vice versa.

(i) A *manager* can manage many *employees*.

(j) An *employee* can only be managed by one *manager*.

→ The relationship between manager and employee is a many-to-one relationship. Constraints: If we record a manager in the database, we do not necessarily need to record employees under that manager (the manager alone may look after a department). If we record an employee, we do not necessarily have to record a manager, since the employee may be the manager.

(k) A *manager* can only have started managing a department in a particular *year*.

(l) A particular *year* may have been the starting year for many *managers*.

→ The relationship between manager and year is a one-to-many relationship. Constraints: If we record a manager in the database, we must record the year the manager started. If a year is recorded, then the name of a manager must also be recorded.

(m) An *employee* can have only one *first name*.

(n) A *first name* can belong to many *employees*.

→ The relationship between employee and first name is a one-to-many relationship. Constraints: If an employee is recorded on the database (using a surname), then a first name must also be recorded, and vice versa.

(o) An *employee* can only work in one department with a given *department number*.

(p) A *department* with a given department number can employ many *employees*.

→ The relationship between employee and department is a one-to-many relationship. Constraints: If an employee is recorded in the database, then the department they work in must be recorded. If a department is recorded, however, we do not necessarily need to record any employees, as the department may have just started.

It should be noted that the constraints outlined above are subjective. In fact, subjective constraints implemented into the design of a database can be the cause of many problems later if someone else comes along who has different ideas about what constraints should apply. This is one reason why databases often need to be upgraded or even changed completely over time.

Step 3 With the information supplied from Steps 1 and 2, it is now possible to identify how many tables are needed in the database, and what should go in each table. Even in such a simple example, as shown here, the information can seem confusing and unclear. To help clarify and summarise the relationships and constraints in Steps 1 and 2, a diagram called an **E-R diagram** (entity-relationship diagram) is often used. An E-R diagram simply shows the same information in a condensed form. The construction of E-R diagrams involves four steps:

1. Identifying the entities
2. Determining the nature of the relationship between the entities
3. Analysing the nature of interaction between the entities
4. Drawing the E-R diagram using the correct symbols.

An E-R diagram serves as a convenient method for introducing **data models**. A data model is a concept for presenting entities or objects about which data are collected and the interrelationships (including constraints) among data.

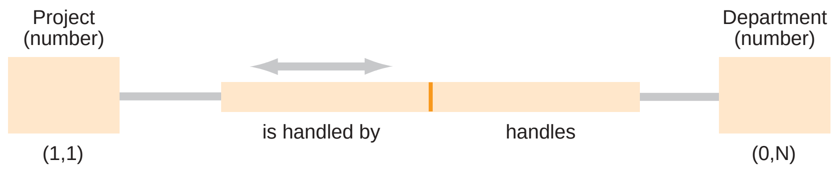


Figure 3.18a **One-to-many relationship (elementary sentences (a) & (b))**

Explanation: ‘Project’ and ‘Department’ are called **entities**. The numbers in brackets are called cardinalities. Cardinalities are a property of database relationships that indicate the number of instances of one entity that may be associated with a single instance of another entity. The number to the left within a bracket is the minimum cardinality and that on the right is the maximum cardinality. The cardinality (1,1) is interpreted as: for one instance of project number the minimum number of departments associated with that number is 1 and the maximum number is also 1. Conversely, for one instance of a department, the minimum number of project numbers is 0 but the maximum number of project numbers can be many—represented by the letter ‘N’. The two numbers ‘1’ and ‘N’ respectively, on the right side within the two brackets, indicate that the relationship between project number and department is one-to-many.

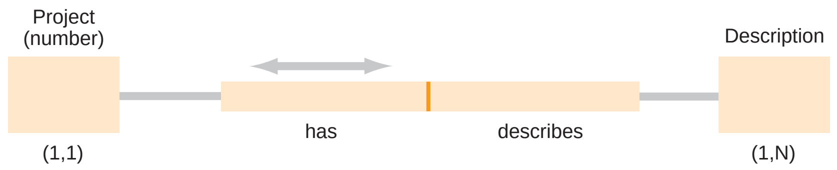


Figure 3.18b **One-to-many relationship (elementary sentences (c) & (d))**

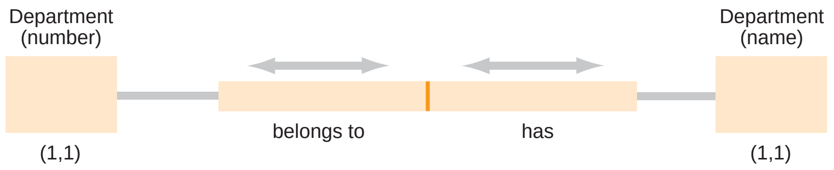


Figure 3.18c **One-to-one relationship (elementary sentences (e) & (f))**

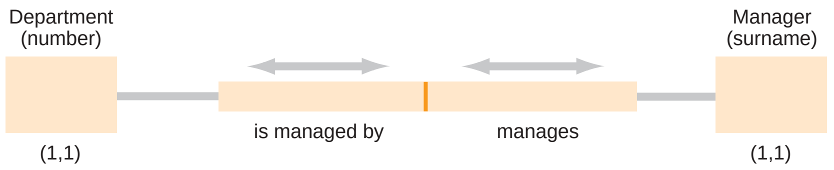


Figure 3.18d **One-to-one relationship (elementary sentences (g) & (h))**

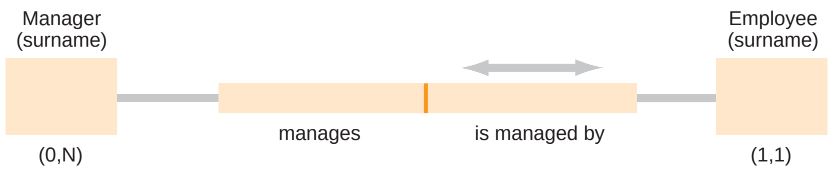


Figure 3.18e Many-to-one relationship (elementary sentences (i) & (j))

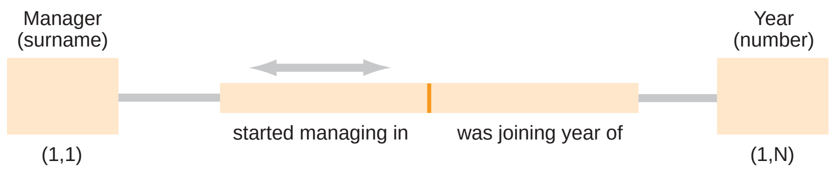


Figure 3.18f One-to-many relationship (elementary sentences (k) & (l))

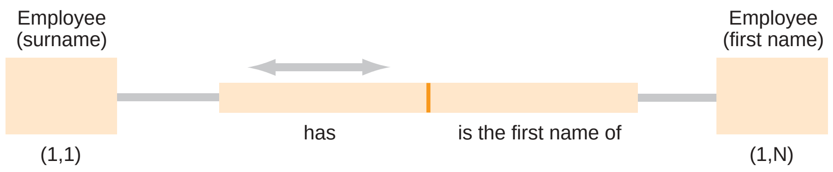


Figure 3.18g One-to-many relationship (elementary sentences (m) & (n))

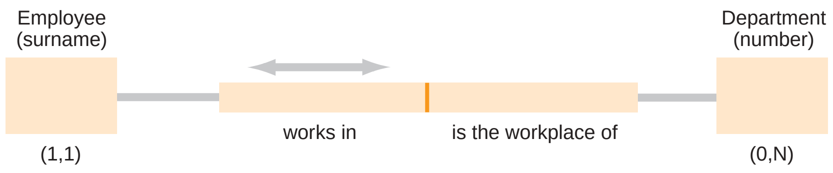


Figure 3.18h One-to-many relationship (elementary sentences (o) & (p))

The E-R diagrams shown above detail the types of relationships between different entities, and the cardinalities related to these, in order to increase understanding of the concept. There are other styles of E-R diagrams. Figure 3.18h, showing the one-to-many relationship, is redrawn below using an alternative style.

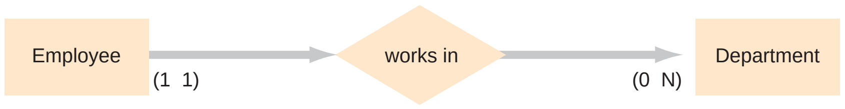


Figure 3.18i An alternative E-R diagram to Figure 3.18h

Step 4 The E-R diagrams make it easy to now group entities into tables. The following rules can be applied when carry out this grouping:

- Look at one entity at a time.
- If this entity is involved in a one-to-many or one-to-one relationship with another entity, both entities will go in the same table.
- If this entity is involved in a many-to-many relationship with another entity, both entities go in a separate table.
- Any remaining entities are grouped in a separate table.

1. First look at the entity 'Project number'
   'Project number', 'Department number' and 'Project description' will all go in the one table. ('Project number' is in a one-to-many relationship with both 'Department number' and 'Description'.)
2. Now look at the entity 'Department number'
   'Department number', 'Department name' and 'Manager' will all go in a separate table. ('Department number' is in a one-to-one relationship with both 'Department name' and 'Manager'.)
3. Now look at the entity 'Manager'
   'Manager' and 'Year' will go in a separate table. (They are in a one-to-many relationship.)
4. Now look at the entity 'Employee'
   'Employee surname', 'Employee first name', 'Department number' and 'Manager' will all go in a separate table. ('Employee' is in a one-to-many relationship with 'Employee first name', 'Department number' and 'Manager'.)

Table 1

Project number	Project description	Department number

Table 2

Department number	Department name	Manager surname

Table 3

Manager surname	Year number

Table 4

Employee surname	Employee first name	Manager surname	Department number

Step 5 At this stage, the tables we have decided on should be checked to see if they make sense in the way they are arranged.

Table 1

For every project recorded, it makes sense to record its description and the department handling the project.

You might suggest that the manager of each project should be included in this table. This information, however, can be derived. In Table 1, if we look up a project number in the table, we can also scan across the row and find the department number. We can then look up Table 2, find that department number and scan across the row to find the manager. In a computer database, the database software can do this 'looking up' in the blink of an eye.

You might suggest that for a given project number, we should record all the people working on that project in Table 1. This information, however, can also be derived. Once again, using the department number in Table 1 that corresponds to the project of interest, we can look up Table 4 and find all the employees in that department. At this point, we can conclude that Table 1 is a sensibly constructed table.

Table 2

For every department number, it makes sense to record its actual name and the manager of the department.

If you want to know the projects this department is working on you can look up the department number in Table 1 and find all the projects of interest. If you want to know the employees in a department, you can look up the department number in Table 4. Therefore, Table 2 is also a sensible table.

Table 3

For every manager, the company would like to know the year they started managing a department.

How do we find out what departments are managed by managers who started in a given year? We find the managers who started in that year in Table 3, and we then look up their names in Table 2. Again, it is apparent that Table 3 is a sensible table.

Table 4

For every employee in the company, we should have a record of their first name, the number of the department they work in and their manager. This table may seem to be necessary, but it contains data that can be derived using other investigative methods.

If we remove the department number column, we can still work out the department an employee belongs to by looking up the name of their manager in Table 2 and reading across the row to find their department. If we want to work out which project an employee is working on, we can look up Table 2 to find their department and then Table 1 to find their project.

Another option would be to remove the manager column. We can still work out the manager of an employee by looking up the department they work in using Table 2. This seems to be a more reasonable option than the previous one. So the manager column in Table 4 can be removed, and if we want to find an employee's manager in the database later, we can look it up in Table 2.

Table 4 (modified)

Employee surname	Employee first name	Department number

Step 6 Populate the tables with some sample data. Let us call this Design 1.

Design 1
Table 1 Project

Project number	Project description	Department number
1	ABM Training	10
2	Product Planning	11
3	Excel Manual	12

Table 2 Department

Department number	Department name	Manager surname
10	Management	Wong
11	Accounting	Cheers
12	Marketing	Kanasa

Table 3 Manager

Manager surname	Year number
Cheers	1990
Kanasa	1996
Wong	2000

Table 4 Employees

Employee surname	Employee first name	Department number
Carter	James	11
Cheers	Ryan	11
Kanasa	Harry	12
Simpson	Nigel	10
Stewart	Robyn	12
Wong	Tsae	10

Step 7 Try to find an example in each table where contradictory information can be entered. If this is possible, then the database design (Steps 1–5) has neglected something.

Testing Table 1

Project number	Project description	Department number
1	ABM Training	10
1	ABM Training	12

→ An impossible situation since our constraints (Step 2), which will be enforced by the database, tells us that a project can only be handled by one department.

Project number	Project description	Department number
1	ABM Training	10
1	Excel Manual	10

→ An impossible situation since our constraints (Step 2), which will be enforced by the database, tells us that a project can only have one description.

Project number	Project description	Department number
1	ABM Training	10
2	ABM Training	10

→ A *possible* situation that does not violate any rules. It can be seen that the project number makes the row unique. We call an entity that makes a row unique a primary key. In a database, entities that are labelled as primary keys form a column in which no value of that entity can be repeated. In Table 1, 'Project number' is a primary key, and the database (once we have told it that 'Project number' is a primary key) will not allow duplicate values to be entered in that column. Therefore, the following situation will be impossible:

Project number	Project description	Department number
1	ABM Training	10
2	ABM Training	10
1	ABM Training	10

→ If we try to enter the data for row 3, as soon as we tell the database the new project is project number 1, the database will give an error message. Carrying out a similar analysis for Tables 2 to 4, we would find the primary keys would be 'Department number' for Table 2, 'Manager surname' for Table 3, and 'Employee surname' and 'Employee first name' for Table 4.

What if we didn't bother with all these steps in the design of a database? What if we designed the tables intuitively? Let us compare the database (consisting of four tables) we have just designed with another that has been intuitively designed.

Design 2
Table 1 Project

Project number	Project description	Department number
1	ABM Training	10
2	Product Planning	11
3	Excel Manual	12

Table 2 Department

Department number	Department name	Manager surname
10	Management	Wong
11	Accounting	Cheers
12	Marketing	Kanasa

Table 3 Employees

Manager	Employee surname	Employee first name	Year joined (manager)	Department number
Cheers	Carter	James	1990	11
Kanasa	Stewart	Robyn	1996	12
Wong	Simpson	Nigel	2000	10

On the surface, this design seems better. It contains only three tables instead of four. Table 3 in this design is much more compact than the Employees table in the first design. So why not use this design?

Imagine that a new department is formed (13—Finance), and a new manager is appointed (Matt Crameri) in 2001, along with 100 new employees. What do we need to do to update our database?

In Design 2, we will need to enter three pieces of information in Table 2 ('13', 'Finance', 'Crameri'). In Table 3, we will need to insert 100 new rows. Within these new rows we will have to enter the name 'Crameri' 100 times, we will have to enter 100 surnames and 100 first names, 100 years joined ('2001'), and finally, we will have to enter the department number '13' 100 times. That is a total of 503 pieces of data to be entered.

In Design 1 (see Step 6, page 86), we would need to enter three pieces of information in Table 2 ('13', 'Finance', 'Crameri'). In Table 3, we would need to enter two pieces of data ('Crameri', '2001') and in Table 4, we would need to insert 100 new rows involving entering 100 surnames, 100 first names and the department number '13' 100 times. That is a total of 305 pieces of data that will need to be entered.

This is just one example of why database efficiency makes Design 1 better. This is not to say, however, that Design 1 is perfect. What if five years down the track the company changes policy so that more than one department can work on a project? What if the company decides that it wants to keep a record of every employee for the life of the company, so that even if Mr Kanasa resigns now, they can still retain his details in case he joins the company again at a later date? Events such as policy changes can have a drastic effect on the validity and usefulness of a database, to the point where a whole new database must be designed.

summary

An accounting information system within an organisation should be documented. For operational efficiency, employees and managers must get an insight into the flow of data within an organisation and the movement of various documents. Data flow diagrams will indicate the logical flow of data and document flow charts will show the physical flow of the various documents. Employees will gain a better understanding of the various processes within the organisation, and the information needs of the organisation, as they become familiar with these diagrams and with data modelling and databases.

This chapter has also explored the development of databases and revealed how E-R diagrams are constructed and used for this purpose. A thorough understanding of the issues discussed in this chapter will be a tremendous help to managers when re-engineering their business processes.

Checkpoint

1. What is a data flow diagram?
2. What is a document flow chart?
3. What general design principles are common to both data flow diagrams and flow charts?
4. What is a context diagram?
5. What is a database model?
6. Differentiate between a system flow chart and a program flow chart.
7. What is a decision table?
8. What is a database?
9. What is a relational database?
10. What is your understanding of an E-R diagram (entity-relationship diagram)?
11. What is a data dictionary used for?
12. Explain the meaning of the terms 'schema' and 'subschema'.

✔ Multiple-choice questions

Circle the letter corresponding to the correct answer.

1 A data flow diagram represents:
- **(a)** a flow of physical data
- **(b)** a logical view of data flow
- **(c)** databases
- **(d)** a basis for decision making

2 Which of the following statements is true?
- **(a)** data flow diagrams make use of only three symbols; flow charts make use of many symbols
- **(b)** a data flow diagram emphasises the flow of data while a flow chart emphasises the flow of documents or records containing data
- **(c)** flow charts and data flow diagrams convey the timing of events
- **(d)** both a flow chart and a data flow diagram show the sequence of processes

3 A rectangle in a flow chart represents:
- **(a)** computer processing
- **(b)** a manual operation
- **(c)** input or output
- **(d)** a decision

4 The drum symbol used in a flow chart represents:
- **(a)** stored data
- **(b)** terminator
- **(c)** magnetic disk
- **(d)** alternate process

5 Which of the following is true in relation to a decision table?
- **(a)** it is a technique for organising documents
- **(b)** it shows the combination of conditions and the corresponding action taken
- **(c)** it reveals how data is flowing
- **(d)** none of the above

6 Which of the following is a difference between a flow chart and a DFD?
- **(a)** a flow chart shows the logical flow of data while a DFD shows the physical flow of data
- **(b)** a flow chart shows the sequence of processes; a DFD does not show the sequence of processes
- **(c)** a flow chart shows the physical flow of data but a DFD shows both the physical and logical flow of data
- **(d)** none of the above are true

7 For the purposes of E-R diagrams an entity is defined as 'a fundamental thing of relevance to the organisation about which it needs to record data'. Which of the following is not an entity?
- **(a)** inventory
- **(b)** customer
- **(c)** accountant
- **(d)** purchase order

8 The meaning of the cardinality pair: 'Purchases' (0, N)—(0, 1) 'Suppliers' is:
- **(a)** for one instance of a purchase the minimum number of suppliers is one
- **(b)** for one instance of a purchase there can be many suppliers
- **(c)** for a given supplier the maximum number of purchases is zero
- **(d)** for a given supplier the minimum number of purchases is one

9 Which of the following is not a model for structuring databases?
- **(a)** a relational data model
- **(b)** a bottom-up organisational model
- **(c)** a hierarchical data model
- **(d)** a network data model

10 The term that describes the entire database, its record types and their relationship to each other is called:
- **(a)** an E-R diagram
- **(b)** a flow line
- **(c)** a schema
- **(d)** a DBMS

✔ Activities

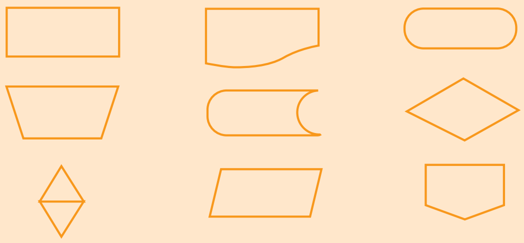
1 Explain what you understand by the following terms:
- **(a)** Data flow diagram
- **(b)** Context diagram
- **(c)** Flow chart.

2 What is meant by decomposition of a data flow diagram?

3 How does a system flow chart differ from a program flow chart?

4 What is a decision table? What is it used for?

5 **What do each of the following flow chart symbols represent?**



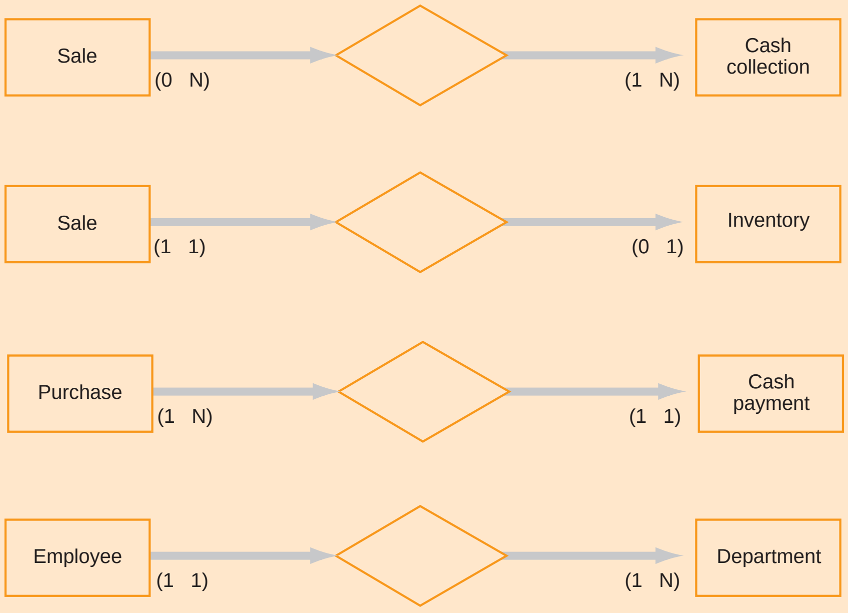
6 **Why is it important to understand database systems? How do users access data and information in a database?**

7 **What three characteristics must be considered in the design of a database?**

8 **What are the advantages and disadvantages of a database system?**

9 **Comment on the statement: 'An E-R diagram can be used as a design tool to reveal how the database is to be installed and used throughout the company'.**

10 **Interpret the following E-R diagrams:**



11 **A customer purchases a few items from a local shop. John, a sales clerk, enters the transaction in the cash register and takes the customer's money. At the end of the day John gives the cash and the register tape to his manager. Identify the data flow diagram elements.**

12 **Prepare a decision table for the following operation:**
Before selling a product, your company makes a credit check on credit customers. If a customer is not a credit risk the order is accepted. If the order is for an amount between $0 and $200, no discount is given. If the value of the order is between $201 and $500, the customer is eligible for a 5% discount. If the order is for $501 or more, a 10% discount is given.

13 **At Smith and Company the receptionist receives incoming mail. The receptionist opens and records the mail in a register and sends the lot to the general manager. After perusal, the general manager decides on the distribution of the mail and writes the initials of the recipient on each item. The mail is then returned to the receptionist for distribution. On a particular day there were seven letters, of which, four were directed to the accountant, one to the production manager and two to the sales manager.**
Show the above scenario in a flow chart.

14 **A company uses a relational database management system to maintain its accounting records. The following four tables are from its accounting database. Using the information in the tables answer the questions that follow.**

Customer table

Customer number	Customer name	Address
21	Arnold	Area 1
51	Benson	Area 2
32	Clement	Area 3
82	Damian	Area 4
63	Edward	Area 5

Product table

Product number	Description	Unit price $
321	A	6.00
654	B	20.00
987	C	14.00
234	D	8.00
567	E	10.00

Invoice table

Invoice number	Customer number
A1111	21
A2222	51
B3333	63
D4444	21
E5555	82

Line item table			
Invoice number	**Line item number**	**Product number**	**Quantity**
A1111	1	567	20
A2222	1	654	4
A2222	2	567	10
B3333	1	321	14
D4444	1	321	2
D4444	2	987	6

Required:

(a) Name the customer and the products billed on invoice number B3333.

(b) What is the total amount due from invoice number A2222?

(c) Which invoices represent sales to customer Benson in Area 2?

(d) Assuming that invoices in the Invoice table are all unpaid, how much is owed to the company by customer Arnold in Area 1?

Case study

P & S Furniture Mart has three principal events in its sales procedure:

- collecting orders from customers
- delivering goods to customers
- receiving payments from customers.

Mary wishes to trace the performance of each salesperson within the company, and the delivery contractors. The invoices are paid on delivery, with the exception of several preapproved customers.

Develop and draw an E-R diagram for this sales activity. Also develop a relational database table specifying each table's primary key and other attributes.