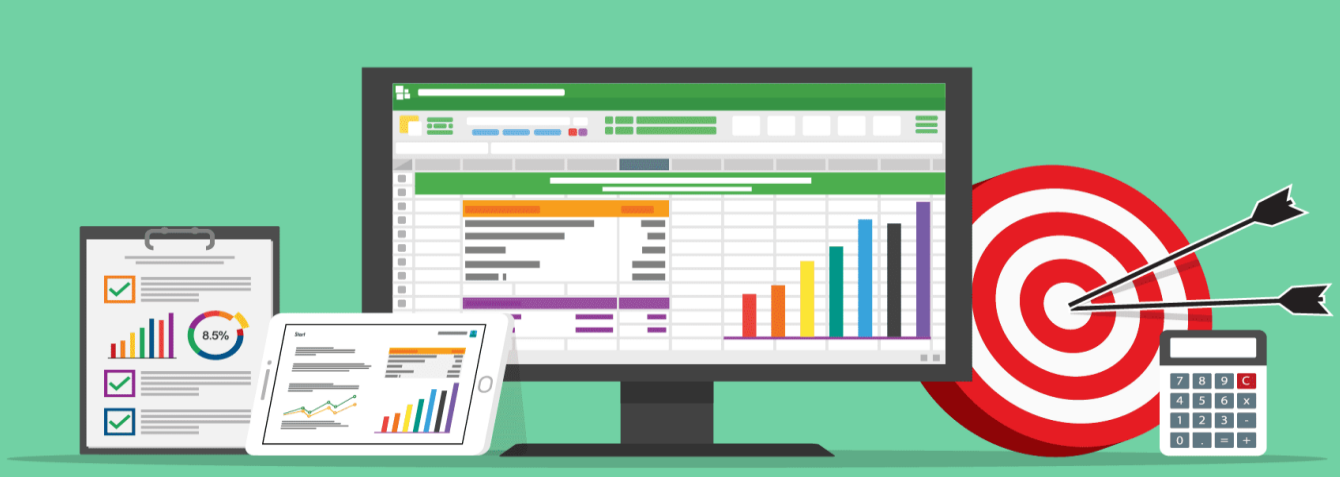


CHAPTER – 15

COMPUTERISED ACCOUNTING SYSTEM (CAS)

Computerised Accounting System means mechanised processing of accounting transaction through the use of computer hardware and software which in turn produce accounting records and reports, *i.e.*, Cash Book, Journal and Ledger. The process is termed as Computerised Accounting System (CAS).

Computerised Accounting System takes accounting transactions as inputs that are processed through Accounting Software to generate the following reports:

1. Day books/Journals
2. Ledger
3. Trial Balance
4. Position Statement (Balance Sheet)
5. Income Statement (Profit & Loss Account or Statement of Profit & Loss)

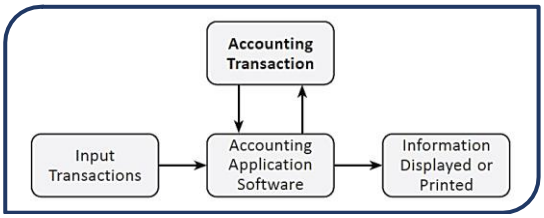
PROCESS OF COMPUTERISED ACCOUNTING SYSTEM (CAS)

The phases of Accounting System involve:

1. Business transactions are analysed.
2. The transactions are recorded in the Journal.
3. Journal entries are posted to the ledger accounts.
4. A trial balance is prepared from balances of accounts.
5. Accounts are reviewed and the necessary adjustments made.
6. Adjustments are posted in the ledger to prepare adjusted trial balance.
7. Adjusted trial balance is used to prepare the Balance Sheet and Profit & Loss Account (in case of Sole Proprietorship) or Statement of Profit & Loss (in case of Companies).
8. Financial Statements are prepared from the finally adjusted ledger and balancing the accounts.

The above accounting cycle can be processed through the use of computers.

Basic Flow of Accounting Transaction



SALIENT FEATURES OF COMPUTERISED ACCOUNTING SYSTEM (CAS)

Simple and Integrated

It is designed to automate and integrate all the business operations, such as sales, finance, purchase, inventory and manufacturing. It is integrated to provide accurate, up-to-date business information rapidly.

Transparency and Control

It provides sufficient time to plan, increases data accessibility and enhances user satisfaction.

Accuracy and Speed

It provides user-definable templates (data entry screens or forms) for fast, accurate data entry of the transactions. It also helps in generalising desired documents and reports.

Scalability

It enables in changing the volume of data processing in tune with the change in the size of the business. The software can be used for any size of the business and type of the organisation.

Reliability

It makes sure that the generalised critical financial information is accurate, controlled and secured.

TYPES OF COMPUTERISED ACCOUNTING SYSTEM (CAS)

The market offers a wide range of computer accounting systems which are equipped with unique features.

Therefore, it is essential to understand them and identify the one which meets the organisational needs.

Readymade or Ready-to-use Computerised Accounting System

It is developed not for any specific user but for the users in general. Therefore, it is not necessary that all the modules of the system are of use of every user. It involves low-cost installation and maintenance but also limits the number of users.

Designed Computerised Accounting System

A Designed Computerised Accounting System is more suitable for small to medium scale business entities. The installation and maintenance costs are relatively high as it provides additional features along with basic features for generating reports, data and graphs.

Customised or Tailored Computerised Accounting System

The term Tailored Computerised Accounting System refers to designing and developing user-specific accounting system. It is best suited for large entities and it caters to specific needs of the user.

COMPONENTS OF COMPUTERISED ACCOUNTING SYSTEM (CAS)

The Computerised Accounting System (CAS) has the following components:

Procedure: A logical sequence of actions to perform a task.
Data: The raw fact (as input) for any business application.
People: Users.
Hardware: Computer, associated peripherals, and their network.
Software: System software and Application software.

GROUPING OF ACCOUNTS

The increase in the number of transaction changes the volume and size of the business. Therefore, it becomes necessary to have proper classification of data. The basic classifications of different accounts embodied in a transaction are resorted through accounting equation.

Accounting Equation is based on double-entry system, which implies equality of assets and equities (liabilities and capital), *i.e.*,

$A = E$

Where, $E = L + C$, Now $A = L + C$ Where A = Assets
(E = Equities; C = Capital; L = Liabilities)
Thus, Assets = Liabilities + Capital
The above equation can be re-written as:
Assets = Liabilities + Capital + (Revenues – Expenses)
Each component of the above equation can be divided into groups of accounts.
There is a hierarchical relationship between the groups and its components. In order to maintain the hierarchical relationship between a group and its sub-groups, proper codification is required to ensure neatness of classification.

CODIFICATION OF ACCOUNTS

The term code means "a system of letter or figure with arbitrary meaning for brevity and for machine processing of information". Thus, code is an identification mark.

Method of Codification

The coding scheme of Account-heads should be such that it leads to grouping of accounts at various levels so as to generate Position Statement (Balance Sheet) and Statement of Profit & Loss (Profit & Loss Account).
For example, we may allot the codes for top-level grouping of accounts (forming the 1st digit of the Account Code) as follows:

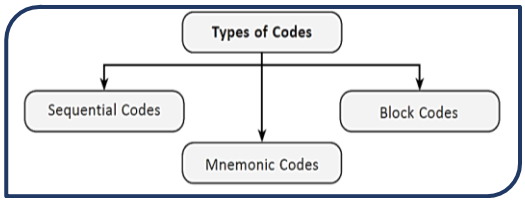
1. Equity and Liabilities
2. Assets
3. Revenues
4. Expenses

Under Equity: 11. Shareholders' Funds
Under Liabilities: 13. Non-Current Liabilities
14. Current Liabilities

Under Assets: 21. Non-Current Assets
23. Current Assets

The above codification scheme utilises the hierarchy present (used) in grouping of accounts.

Types of Codes



Sequential Codes

Under these codes, numbers and/or letters are assigned in consecutive order. These codes are applied primarily to source documents such as cheques, invoices, etc. A sequential code can facilitate document searches.

Examples:

CODES	ACCOUNTS
CL001	SAVLON LTD
CL002	DETTOL LTD
CL003	BSES YAMUNA POWER LIMITED

Block Codes

In a block code, a range of numbers is partitioned into a desired number of subranges and each sub-range is allotted to a specific group. In most of the uses of block codes, numbers within a sub-range follow sequential coding scheme, *i.e.*, the numbers increase consecutively. As an example, dealer codes for a trading firm could be as follows:

CODES	DEALER-TYPE
100 – 199	Small Pumps
200 – 299	Medium Pumps
300 – 399	Pipes
400 – 499	Motors

Mnemonic Codes

These codes consist of alphabets or abbreviations as symbols to codify a piece of information. SJ for "Sales Journals", HQ for "Head Quarters" are examples of mnemonic codes. Another common example is the use of alphabetic codes in Railways in identifying railway stations such as DLH for Delhi, NDLS for New Delhi, BRC for Baroda, etc.

ADVANTAGES OF COMPUTERSIED ACCOUNTING SYSTEM (CAS)

Following are the advantages of Computerised Accounting System (CAS):

1. Timely generation of reports and information in desired format.
2. Efficient record keeping.
3. Ensures effective control over the system.
4. Economy in the processing of accounting data.
5. Confidentiality of data is maintained.

LIMITATIONS OF COMPUTERSIED ACCOUNTING SYSTEM (CAS)

Following are the limitation of CAS software:

1. Faster obsolescence of technology necessitates investment in shorter period of time.
2. Data may be lost or corrupted due to power interruptions.
3. Data are prone to hacking.
4. Un-programmed and un-specified reports cannot be generated.

ACCOUNTING INFORMATION SYSTEM (AIS)

Accounting Information System (AIS) and its various sub-systems may be implemented through Computerised Accounting System. The subsystems of Accounting Information System (AIS) are as follows:**5**

Cash and Bank Sub-system

It deals with the receipt and payment of cash both physical cash and electronic fund transfer.

Sales and Accounts Receivable Sub-system

It deals with recording of sales, maintaining of sales ledger and receivables.

Inventory Sub-system

It deals with the recording of different items purchased and issued specifying the price, quantity and date.

Purchase and Accounts Payable Sub-system

It deals with the purchase and payments to creditors.

Payroll Accounting Sub-system

It deals with payment of wages and salary to employees.

Fixed Assets Accounting Sub-system

It deals with the recording of purchases, additions, deletions, usage of fixed assets such as land and buildings, machinery and equipment, etc.

Expense Accounting Sub-system

This sub-system records expenses under broad groups such as manufacturing, administrative, financial, selling and distributions and others.

Tax Accounting Sub-system

This sub-system deals with compliance requirement regarding Goods & Services Tax (GST), excise, customs and income tax.

Final Accounts Sub-system

This subsystem deals with the preparation of Trading and Profit & Loss Account (in case of sole proprietorship) or Statement of Profit & Loss (in case of Companies), Balance Sheet and Cash Flow Statements for reporting purposes.

Costing Sub-system

It deals with the ascertainment of cost of goods produced. It has linkages with other accounting sub-systems for obtaining the necessary information about cost of material, labour, and other expenses.

Budget Sub-system

It deals with the preparation of budget for the coming financial year as well as comparison with the current budget of the actual performances.

Management Information System

Management Information System (MIS) deals with generation and processing of reports that are vital for management decision-making.

SECURITY FEATURES OF CAS

Every accounting software ensures data security, safety and confidentiality.

Therefore every, software provides the following:

1. *Password Security:* Password is a mechanism, which enables a user to access a system including data.

2. *Data Audit:* It enables one to know as to who and what changes have been made in the original data thereby helping and fixing the responsibility of the person who has manipulated the data and also ensures data integrity.

3. *Data Vault:* It provides additional security through data encryption

DATABASE MANAGEMENT SYSTEM (DBMS)

The foremost need of Computerised Accounting System is the overwhelming quantity of data in organisations. The conventionally used paper filing system, text documents, and even spread-sheets may not suffice for the growing needs of tracking this voluminous and critical information. A solution to this situation is available in the form of a Database Management System (DBMS).

Database Management System (DBMS) provides a variety of software tools for organising, processing and querying data in a flexible manner. MS-Access, Oracle, SQL Server, IBM-DB2 are examples of DBMS software.

OBJECTS IN DATABASE MANAGEMENT SYSTEM (DBMS)

The database management a pre-requisite for understanding of this chapter, and hence, we will restrict ourselves to the simpler and easy to comprehend 'MS Access' program for developing some practical accounting applications.

Databases in Access are composed of four objects:

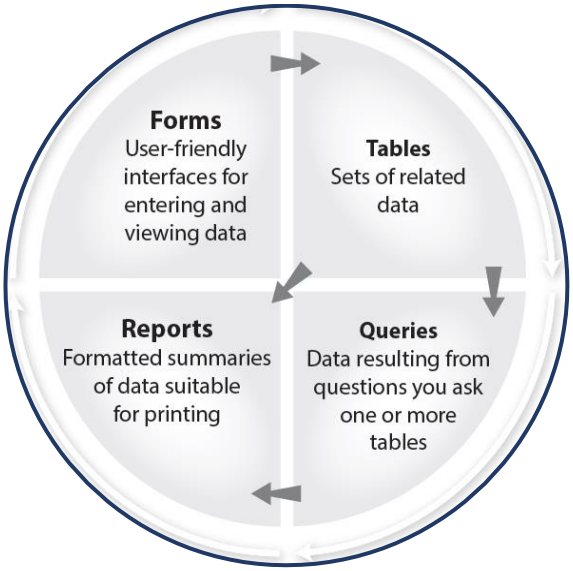
1. Tables,
2. Queries,
3. Forms, and
4. Reports.

UNDERSTANDING AND DEFINING THE DATABASE REQUIREMENTS

With the continuous improvements in computer's processing speed, storage capacity, networking techniques, operating systems, etc., the capabilities of computer applications have also undergone many changes. Various computer applications that are commercially available today not only provide fairly comprehensive tools for all conceivable needs but also have become extensively user friendly. So, when we look forward to putting into use

database applications such as 'Access', we really do not need much of programming skills. Nevertheless, any programming knowledge may improve our efficiency and effectiveness in handling such applications.

On the other hand, before we develop any database application, we ought to have a complete understanding of our requirements expected from the application. This is one area where application itself may not extend much help. Further, the correct understanding of our requirement also has a bearing on the choice of Database Management Systems (DBMS), *i.e.*, whether to go for 'Desktop Database' or to choose the 'Server Database.



Desktop Database

Desktop database is mostly designed to run on "desktop" or personal computers. It is much cheaper and simpler. Most of the desktop databases are very user friendly. A user does not need to understand SQL in order to use them, and contain a fairly graphical interface. It also many a times provides web solutions so that the user can publish data onto the web in various fashions. Microsoft Access is a very popular, well known, desktop database that many people already own.

Server Database

Server database is comparatively more complex and expensive than desktop database. This type of databases offer organizations the ability to manage large amounts of data and many users. In a server database, APIs are used for rapid development of custom applications providing users with a lot of flexibility. It can also expand very quickly if the money is put into investing the proper hardware into one. Unfortunately, this can get rather costlier. Common server database includes Microsoft SQL Server and Oracle.

IDENTIFICATION OF DATA TO BE STORED IN TABLES

Tables allow us to create the framework for storing information in the database. Each column (also called 'field') of the table corresponds to a specific characteristic (or 'attribute' in database terms) of the stored information. Each row (also called 'record') corresponds to a particular instance of the information. A set of tables often with well established relationships between them constitutes the database covering total spectrum of stored information. The term 'database design' can be used to describe the structure of different parts of the overall database.

We may get a better understanding of database design by referring to practical example of an accounting problem. We may take up the case of payroll accounting.

Let us begin by identifying various attributes of information that are required to be stored in our Payroll database. We may have a set of attributes pertaining to employee's personal details such as: 'Employee ID', 'Name', 'Designation', and 'Location'. On the other hand, we may also deal with such attributes pertaining to employee's pay as: 'Basic Pay', 'Dearness Allowance (DA)', 'House Rent Allowance (HRA)', 'Transport Allowance (TA)', 'Provident Fund (PF) Deduction', etc. We may also want to know the attributes of 'Gross Salary' and 'Net Salary' which is obtained by subtracting 'PF Deductions' from 'Gross Salary'. However, 'Gross Salary' and 'Net Salary' attributes may not require being stored in the database as they are merely computational outcomes from other attributes. We may also require some attributes concerning pay formulations such as '% Rate of DA' which may fluctuate month to month, '% Rate of HRA' varying with the location of employee, and 'TA Slabs' varying with the designation of employee.

Now the main question is how to store these attributes in our database tables. Shall we make all of above attributes as part of one table, or shall we opt for multiple tables. The 'Access' as such will not put any constraint on the number of tables we opt for, or the type of data we chose to put in any table. It will have to be entirely our decision, based on logical structuring of data that we seek to apply. You may also appreciate the fact that the purpose of a database is not so much for the storage of information, as for its quick retrieval. Hence, you ought to structure your database in such a manner that it can be queried quickly and efficiently.

CREATING DATABASE TABLES IN MICROSOFT ACCESS

Before we take up the task of database design using Access, we will have to first start up the Microsoft Access Application. For this, click on the **Start** button on the Windows Taskbar. Now point to **All Programs**; then point to **Microsoft Office**; and then click **Microsoft Office Access 2007**.

In Simplified manner it is as follows:

Start > All Programs > Microsoft Office > Microsoft Access 2007

Click on **Blank Database** under central section. This will open up a dialogue box on the right section asking for database file name. This dialogue box could have also been opened by Clicking on the **Office button** located at the upper left corner of the screen and then Clicking **New** at the dropdown menu.

Enter the file name **'PayRollApplication'** and then click on the **Create** button.

Instead of the default location for your database file, you can also identify your own location by clicking on the Browse icon given on the right side of file name. In New File dialogue box, you can also accept the suggested name (in default

folder My Documents) and later on rename the file and move it to any other folder of your choice.

Clicking **Create** button on the New file dialogue box takes you directly to the new database window with the **Tab** for **DataSheet** opened up.

In the left side **Navigation Pane** you will see that a default table has already been created and the same is also opened up in the working area of the window.

We can straight away start working on this table assuming it to be our first table. All we have to do is to type an employee name (say 'Ram Kishore') in the second column and click **enter** (or use **Tab**) to move to third column; type a designation ID (say '1') followed by enter; and finally move to fourth column and type a location ID (say '1').

In this case, Access will automatically set a data type for each field based on the type of data entered into each column. You will see that headings of these columns are named as 'Field1', 'Field2' and 'Field3'.

Move the mouse pointer over the column heading 'Field1' and then double-click to select the column heading.Type 'EmpName' and press enter to change the column name from 'Field1' to 'EmpName'. Similary, change the names of columns 'Field2' and 'Field3' to 'DesgID' and 'LocationID'.

You might have noticed by now that the first column of the table has the name 'ID' which was created automatically. This is an AutoNumber field in which the field value is assigned automatically by Access as we enter a new record.

Move the mouse pointer over 'ID' heading; double click to select this heading, and change its value to 'EmpID'.

The structure of the 'Table' in the Datasheet View is already complete. At this stage, you can choose to enter the values of the other records as well. If you are not contended with the width of any column in your table then with the insertion point positioned in any record of this column click

Home Tab > More (under Records Group) > Column Width.

Click on the **Save** button on the **Quick Access** toolbar on left hand top corner of window. A **Save As** dialogue box will appear with the default table name. Replace the name with **'TabEmpDetails'** and press **enter** or click **OK** button to save the table with the intended name.

Now let us look at this table in **Design View**. Click on the **View** button placed on the left end of the ribbon (in View group) under the Datasheet Tab. In Design View, you can tell Access which all fields will go into the table by entering desired attributes in the column titled 'Field Name'. The type of data corresponding to each attribute can be identified in the column 'Data Type'.

Having fully completed the structure and data of our first table and also having understood the various concepts related to its design, we can now move on and create rest of the four tables. Click on **Create Tab > Table Button**.

You can choose this view or the Design View to create the rest of the tables and save them under intended names. The field names and data types of all five tables are summarrised as under:

S. No.	Field Name	Data Type	Description
1.	**TabEmpID** EmpID EmpName DesgID LocationID	Auto Number Text Number Number	Unique Employee ID—Primary Key.
2.	**TabDesignations** DesgID Designation TA	AutoNumber Text Number	Unique Designation ID—Primary Key. Transportation Allowance—Fixed as per designation.
3.	**TabLocations** LocationID Location RateofHRA	AutoNumber Text Number	Unique Location ID—Primary Key. % Rate of House Rent Allowance.
4.	**TabDARates** MonthID SalMonth RateofDA	AutoNumber Text Number	Unique Month ID—Primary Key. With Values as 'April-2022', 'May-2022',etc. % Rate of Dearness Allowance.
5.	**Tab Monthly Salary** SalaryID MonthID EmpID Basic DedForPF	AutoNumber Text Number Number Number	Basic pay based on level & attendance of employee. Deduction for PF opted by employee & as per rules.

Having completed the designs of all data tables, we will now move on to the task of establishing relationships between different tables. Click on the **Database Tools** Tab and then **Relationships** button under **Show/Hide** group. In the working area, a **Show Table** dialogue box will appear. In case you do not see this dialogue box click on **Design > Show Table**. In the **Show Table** dialogue box, select a table and click **Add** button to add it in the relationship window. Add all

the five tables in this manner. Close the Show Table dialogue box by clicking on **Close** button.

In the working area you will see all five table objects, each detailing the fields within them. You can reposition these table objects anywhere within the relationship window. To do this, point the mouse pointer in the caption area of the selected table object, hold down the left mouse button and then drag the mouse to shift table to its new location.

Now to create a relationship between TabMonthlySalary and TabDARates, position the mouse pointer over MonthID in the TabMonthlySalary table object, hold down the left mouse button, drag the pointer right to MonthID in the TabDARates, and then release the mouse button. A **Edit Relationships** dialogue box will appear as soon as you release the mouse button. You will notice that the dialogue box shows relationship type as **One-To-Many**.

Access has determined the relationship type based on the fields selected for joining the two tables. Click on the **Create** button. A black line will appear joining the two tables at the common field. This line, also referred to as **Join Line**, establishes the relationship between TabMonthlySalary and TabDARates using the common field of MonthID. Remember, MonthID forms Primary Key in TabDARates and Foreign Key in TabMonthlySalary and hence is the relationship One-To-Many type. You may now repeat above steps to establish other relationship between the tables. The overall relationship between different tables are summarised as under:

Table	Related Table	Common Field	Relationship Type
TabDARates	TabMonthlySalary	MonthID	One-To-Many
TabEmpDetails	TabMonthlySalary	EmpID	One-To-Many
TabDesignations	TabEmpDetails	DesgnID	One-To-Many
TabLocations	TabEmpDetails	LocationID	One-To-Many

CREATION OF QUERY IN MICROSOFT ACCESS

The Queries provide the real power to a database in terms of its capabilities to answer more complex requests (or queries). In case of Access, Queries provide the capability of combining data from multiple tables and placing specific conditions for the retrieval of data. In its simplest form, you may consider a Query to be another tabular view of your data showing information from one or more tables.

Click on **Create > Query Design**. A **Show Table** dialogue box will appear with a Query Table in the background.

In case you do not see this dialogue box click on **Design > Show Table**. In the **Show Table** dialogue box, select a table and click **Add** button to add it in the relationship window. Add all the five tables in this manner. Close the Show Table dialogue box by clicking on **Close** button. In working area above the Query Table, you will see all five table objects (with complete list of their fields) along with the one-to-one relationship that has been established between tables earlier. You can reposition these table objects in the manner discussed earlier. In the portion below Table objects, you will see blank columns that represent columns in the Query results datasheet. These columns are also referred to as the **Design Grid**.

In the first column of the design grid, click on the Table row. A dropdown button will appear on this cell. Click on the button to get the dropdown list of tables and click on 'TabMonthlySal'. Now in the same column, click on Field row and get the dropdown list of all fields in 'TabMonthlySal'. Click on the 'SalaryID' field.

In this manner the SalaryID (from TabMonthlySal) has been selected for view in the Query. The above operation could have also been done simply by double clicking the 'SalaryID' in the list box of TabMonthlySal object.
You shall take care to fill different fields from Table object into the Design Grid in the same order in which you want to display these fields in the Query Results data sheet. Now select any of the described methods to fill other columns of the design grid in the specified order with fields of MonthID (TabMonthlySalary), SalMonth (TabDARates), EmpName (TabEmpDetails), and Basic (TabMonthlySalary).

At this stage, if you click on the Run button under Results group of Design Tab, you will see the result of Query displayed with the fields of SalaryID, SalMonth, EmpName, and Basic. Clicking of 'Run' actually tells the Access to execute all instructions stored by you in the Query, and to display the results. You may adjust the column widths by any of the methods described earlier for Tables. Click on the Save button on the Quick Access toolbar. At the **Save** As disalogue box, type QueryMonthlySalary and click **OK** to save our Query.

As we are not seeing the computational fields and the field of Ded For PF in the **DataSheet View**. Go back to the Design Grid view by clicking on Design View under Home Tab. In the Design Grid, Click on the Field row of the first blank column (after Basic) and type '**DA: [Basic]* [TabDARates. RateOfDA]/100'**. In this expression, the word 'DA' coming before the column(:) will be treated as the name of the computational field which will appear as heading of the column. The rest of the typed part will form the actual expression which will be evaluated and its value displayed on clicking of **Run** command.

In our expression for 'DA', we have made use of 'Basic' field which is already forming a part of Query by taking it within the square parenthesis ([ ]). We have also made use of 'RateOFDA' field which is not appearing in the Query columns; and for this we recorded the corresponding Table name as well as Field name separated by a period (.) within the square parenthesis. It may be easily comprehendable that the operator '*' and '/' stand for multiplication and division respectively. At this stage if you click on the **'Run'** command, you will notice the last field for 'DA' with its computed values for different employees as per the % DA Rates for different months. Come back to the design view (by Clicking **Design View** button under **Views** Group) and

repeat above listed procedure to fill up the next field of Query for computing HRA using the expression **'HRA: [Basic]* [Tab Locations. RateOfHRA]/100'**. The next column in the Query is for the **TA** field, which is to be included as such from the Table **'TabDesignations'** using any of the methods enumerated earlier. Next, create the field for 'Gross Salary' by typing the expression **'GrossSalary**: **[Basic]+[DA]+[HRA]+[TA]'**.

Now we can also include the field of **'DedForPF'** from the **'TabMonthlySalary'**. Finally, we shall include the field of Net Salary using the simple expression **'NetSalary: [GrossSalary]-[DedForPF]'**. At this stage, save your Query as instructed earlier and also run the **'Run'** command to look back the outcome of your effort in the Data Sheet View.

For a better presentation, you may also like to sort the records in ascending or descending order against a chosen field of the Query. For this, click on the **Sort** row below the Emp Name field. Now Click on the pull down button that appears on the end of the cell and select **Ascending**. Run the Query and notice that the records are alphabetically arranged as per names of the employees.

CREATION OF FORMS IN MICROSOFT ACCESS

The Access 2007 provides facility for creation of a **Simple Form** wherein you can enter information for one record at a time. Access also provides tools for creation of a **Split Form** which shows the underlying datasheet in one half of the section and a Form in other half for entering information in the record selected in the datasheet. The two views in this form are synchronised so that scrolling in one view causes the scrolling of other view to same location of the record.

The Microsoft Access provides two primary mechanisms for placing information into the tables as and when the need arises. The first method, which we are already familiar with, is to simply bring up the table in a window by double clicking on it and adding information to the new blank row at the bottom. We can also make changes in any other row as one would do in case of a spreadsheet. You can save the changes in the table by clicking on the Save button on the Quick Access toolbar.

As second mechanism for information handling (*i.e.*, for addition, modification and deletion), the Access provides a user-friendly Forms interface that allows users to enter information in a graphical form, and this information transparently passes to the underlying database. This method is more user-friendly for the data entry operator, though it may entail a little more work on the part of the database designer.

CREATION OF REPORTS IN MICROSOFT ACCESS

A **Report** is also used for the print purpose, though it allows for more flexibility in selecting the fields to print, and to have more control on the overall layout and format of the print output. The **Report** in Access is thus another object which is designed to print information from the database on to the screen, or to a file or directly to the printer.

Here also you can take the elaborate route of step-by-step design as was done in case of Tables, Query or Form. However in this case, you may find the Report Wizard becoming a more favorable tool as it guides the designer through a series of dialogue boxes to create the most suitable Report.

Evidently in our case, we will be interested in generating the Report for the **'QueryMonthlySalary'**. With this intent, Click on 'QueryMonthlySalary' in the Navigation Pane, then click on **Create** Tab, and then on **Report Wizard** under Reports Group. In the dialogue box that appears with QueryMonthlySalary Query already selected, move all fields from the **Available Fields** List Box to the Selected Fields List Box using appropriate **Arrow** button.

Click **Next** to see the second dialogue box asking for Grouping Levels. Here select 'SalMonth' and click on the Arrow button (else double click the 'SalMonth') to print data in groups of different salary months.

Click **Next** to reach the third dialogue box asking for sorting order of the records. Here you can click on the **Pull-down Arrow** next to the first text box and then select **'SalaryID'** from the drop down list so as to arrange the records in ascending order of the salary ID within each data group.

Click **Next** to see the dialogue box for layout of the Report. Retain the default **'Stepped'** layout but change the orientation of Report to **'Landscape'** so as to see all the columns of the Report in a single page view.

Again Click **Next** to see the dialogue box for selecting style of Report. You may choose any style and later on experiment with the others as well. One more click of **Next** will take you to the dialogue box seeking the name of Report object. Type **ReportMonthlySalary** and click on the Finish button to see the Report appearing in working area of window in its **Print Preview** form.

You may notice that the extent of data in above designed Report is limited as per the **Criteria** laid down in our Query **'QueryMonthlySalary'**. By changing the Query suitably, we can modify the Report for generating monthly as well as annual Pay Rolls. We can also change the Criteria of our Query to get individual Report for each month (*i.e.*, Pay Slip) or for the whole of year (*i.e.*, Annual Statement).

We may like to improve the Report further by getting sum total of all the values under different columns for each group. For example, we may want to see the total Basic of all employees for April, 2022 group. For such modification, open the Report in Design View by Clicking on **Home** Tab and then clicking on **Design View** under Views Group. In the design view, select the field **Basic** by clicking it once. Now click on **Totals** under **'Grouping and Totals'** Group of **Design** Tab. In the pull down menu that appears, click on **'Sum'**. You will notice that an expression **'= Sum([Basic])'** will appear below the Basic Text Box. You can repeat above steps with other fields containing numerical (both stored as well as computed) values. Now if you click on **View** button (*i.e.*,click **Design > Report View**), you will see the sum total of each column (with in each group) appearing inthe report.

At this stage, close the Report by clicking on the **Cross** button immediately above the report. Save the application once again by clicking on the **Save** button under **Quick Access**

Bar. Our Pay Roll Application in Access has been completed and saved. You may now close your application by clicking on the **Close** button under **'Office Button'** or by simply clicking on the **Cross** button at right-hand top corner of the window.

SPREADSHEET

A spreadsheet is a configuration of rows and columns. Rows are horizontal vectors while columns are vertical vectors. A spreadsheet is also known as a worksheet.

A spreadsheet is used to record, calculate and compare numerical or financial data. Each value can either be an independent (*i.e.*, basic) value or it may be derived on the basis of values of other variables. The derived value is the outcome of an arithmetic expression and/or a function (*i.e.*, a formula). Spreadsheet application (sometimes referred to simply as spreadsheet) is a computer program that allows us to add (*i.e.*, enter) and process data.

BASIC CONCEPTS OF SPREADSHEET

The Spread Sheet has rows and columns. Rows are numbered numerically from top to bottom while Columns are numbered alphabetically. The intersection of a row and a column is called a cell. A cell is identified by a combination of a letter and a number corresponding to a particular location within the spreadsheet.

FEATURES OF ELECTRONIC SPREADSHEET

1. It is easy to understand and operate.
2. It is required especially for tabulation of data. It reduces manual work and provides high degree of accuracy in results.
3. It exhibits data in simpler and explainable form by presenting it in the forms of charts, graphs, pie diagrams, bar diagrams, etc.
4. Calculations can be made automatically through built-in programs or by placing mathematical formula in the Cell.
5. The data in it, can be sorted either in ascending or descending order without any hassle.

Cell Reference—A cell reference identifies the location of a cell or group of cells in the spreadsheet also referred as a cell address. Cell references are used in formulas, functions, charts, other Excel commands and also refer to a group or range of cells.

Ranges are identified by the cell references of the cells in the upper left (cell A1) and lower right (cell E2) corners. The ranges are identified using colon (:), *e.g.*, A1 : E2 which tells Excel to include all the cells between these start and end points.

By default cell reference is **relative**; which means that as a formula or function is copied and pasted to other cells, the cell references in the formula or function change to reflect the new location.

The other cell reference is **absolute** cell reference which consists of the column letter and row number surrounded by dollar ($) signs, *e.g.*, C4. An absolute cell reference is used when we want a cell reference to stay fixed on specific cell, which means that when a formula or function is copied and pasted to other cells, the cell references in the formula or function do not change.

Movement	**Key Stroke (Press Key)**
One cell down	Down arrow key (↓) or Enter key
One cell up	Up arrow key (↑)
One cell left	Left arrow key (↖)
One cell right	Right arrow key (↘) or **Tab** key
Top of Worksheet (cell A1)	**CTRL + HOME** (*i.e.*, Keep **CTRL** key pressed and then press **HOME** key)
The cell at the intersection of the last row and last column containing data	**CTRL + END** keys
Moving consecutively to the first and the last filled cells of clusters of filled cells in a row by successive pressing of **CTRL + Right arrow key (↘) or else END + Right arrow key (↘)**	**CTRL + Right arrow key** (↘) or else **END + Right arrow key** (↘)
Moving consecutively to the first and the last filled cells of clusters of filled cells in a row by successive pressing of **CTRL + Down arrow key (↙)or else END + Down arrow key (↙)**	**CTRL + Down arrow key** (↙) or else **END + Down arrow key** (↙)
Beginning of the Row Beginning of the Column	**HOME Key**

A simple example of a **spreadsheet application** is to calculate compound interest and maturity amount to be paid on fixed deposit.
The first step (*i.e.*, the Planning Step) is to define six cells with column headings:
1. Principal Amount (PA in column B)
2. Rate of Interest (r in column C)
3. Period in years (NY)
4. Period of Compounding (CP in column D)
5. Compound Interest (CI in column F)
6. Maturity Amount (MA in column E)

The formula for Maturity Amount (MA) and Compound Interest (CI) computations considering yearly compounding of interest are as follows:
MA = PA * (1 + R / (100 * CP)) ^ (R * CP)
CI = MA – PA

As this stage, layout of the worksheet for (compound) interest calculation can be designed.
In case any basic values are modified, the derived values as a result are revised accordingly. This feature of Spreadsheets enables us to study various **what-if scenarios**.

A **what-if scenario** is used to generate a number of alternatives to examine the cause (if) and effect (what). Thus, it helps in analysing the impact of changes due to variations in one or more input values.

Taking the above example, if all the other values are kept same, one can see how different rates of interest and different periods of compounding would affect the Compound Interest and the Maturity Amount to be received.

Basic Terminologies and Features of the Spreadsheet

LABELS

A text or especial character will be treated as labels for rows or columns or descriptive information. Labels cannot be treated mathematically multiplied, subtracted, etc. Labels include any cell contents beginning with A-Z. Principal Amount, Rate of Interest, Maturity amount, etc. will be taken as labels.

FORMULAS

The formula means a mathematical calculation on a set of cells. Formulas must start with an = sign (equalto sign).
When a cell contains a formula, it often contains references to other cells. Such a cell reference is a type of variable. Its value is the value of the referenced cell or some derivation of it. If that cell in turn references other cells, the value depends on the values of those.**3**

Order of Mathematical Operations (expressions)

Computer math uses the rules of Algebra. Any operation(s) contained in brackets will be carried out first followed by any exponents.

After that, Excel considers division or multiplication operations to be of equal importance, and carries out these operations in the order they occur left to right in the equation.

The same goes for the next two operations – addition and subtraction. They are considered equal in the order of operations. Whichever one appears first in an equation, either addition or subtraction is the operation carried out first.
Three easy ways to remember the order of operations is to use the acronym:

GEMS PEMDAS BEMDAS

( ) Grouping Please – ( ) parenthesis ( ) Brackets
^ Exponents Excuse – ^ exponents ^ Exponents
* Multiplication : My – * multiply * Multiplication
/ or Division : Dear – / divide / Division
– Subtraction : Aunt – + add + Addition
+ Addition : Sally – – subtract – Subtraction
A function is a special key word which can be entered into a cell in order to perform and process the data which is appended within brackets.

There is a function button on the formula toolbar (fx); when we click with the mouse on it; a function offers assistance and useful prompts into a spreadsheet cell. Alternatively, we can enter the function directly into the formula bar. A function involves four main issues:

1. Name of the function.
2. The purpose of the function.
3. The function needs what argument(s) in order to carry its assignment.
4. The result of the function.

A function is a built-in set of formulas which starts with an = "equal to sign" such as = **Function Name (Data)**.

SUM (), AVERAGE () and COUNT () are common functions and relatively easy to understand. They each apply to a range of cells containing numbers (or blank but not text) and return either the arithmetic total of the numbers, the average mean value or the quantity of values in the range.

The SUM or AutoSum (Σ) function is the most basic and one of the common user functions. It is used to get the addition of various numbers or the contents of various cells. The AutoSum function also includes other series-based functions such as AVERAGE, MIN, MAX and COUNT.

Naming Ranges – IF Functions – Nested IF Functions

Naming Cells and Ranges

Naming ranges in Excel will save time for writing complex formulas. The name can be used in place of cell range whenever reference it, *e.g.*, in D3 we have = SUM (B1 : F1)

The cell referenced in the function B1 : F1 can be replaced with a descriptive name say Numbers (name range) which is easier to remember and in D3 it will be = SUM (Numbers).

Now we will use a summation of numbers using condition in the cell D3. Type the formula = SUMIF (Numbers," ≤ 6) and answer will be for the numbers less than or equal to 6 in the named range B1 : F1.

IF Function

In continuation to our need of what-if scenario now we will learn about an important logical function IF Function. This function returns one value if a specified condition evaluated to TRUE and another value if it evaluatesto FALSE.
An IF function has the following format:

IF (logical_test, value_if_true, value_if_false)

Where,
Logical_test: The value or expression that is determined to be true or false; this requires the usage of a logical operator. A logical operator is one used to perform a comparison between two values and produce a result of true or false (there is no middle result: something is not half true or half false or "Don't Know"; either it is true or it is false).

Value_if_true: The value returned if the test is determined to be true. This value can be a value, text, or expression, formula, etc. or it can be return the value of another cell.

Value_if_false: The value returned, if the test is determined to be false. This value can be a value, text, or expression, formula, etc. or it can return the value of another cell.

OTHER USEFUL FUNCTIONS

In business applications the input of data usually contains dates (date of invoice preparation, date of payment, payment

received date, or due date etc.), rate of interest, tax percentage and output information may require age calculation, duration, delays in payment, accumulated interest, depreciation, future value, net presentvalue, etc.
On the ribbon of MS Excel, the formula tab contains categorised function libraries.

(*a*) Date and Time Function

(*i*) TODAY () is the function for today's date in the blank worksheet.

(*ii*) NOW () is similar function but it includes the current time also.

(*iii*) DAY (Serial_number) function returns the day of a date as an integer ranging from 1 to 31.

(*iv*) DATEVALUE (date_text) converts a date in the form of text to a serial number.

(*b*) Mathematical Function.

(*i*) **SUMIF** is the function which adds the cells as per given specified criteria.

SUMIF (**range**, **criteria**, sum range) where

Range it is the range of cells to evaluate.

Criteria it is the criteria in the form of a number, expression, or text that defines which cells will be added.

Sum range are the actual cells to sum.

(*ii*) **ROUND** is the function to rounds a number to specified number of digits.
ROUND (**number**, num_digits) where
Number Is the number to round (preferably fractional number).
Num_digits specifies the number of digits to round the Number.
There are several ways to round a number other than ROUND:
ROUNDUP (number, num_digits) which rounds a number up, away from 0 (zero).
ROUNDDOWN (number, num_digits) which rounds a number down, toward zero.

(*iii*) **COUNT** is used to get the number of the entries in a number field (including date also), *i.e.*, in a range or array of numbers.

COUNTA function will be count logical values, text, or error values.
There are other functions also such as **ROWS** and **COLUMNS** are used. The syntax is as follows:

ROWS (array) The function returns the number of rows in a reference or array; where an Array is an array, an array formula or a reference to a range of cells for which we want the number of rows.

COLUMNS (array) This function returns the number of columns in an array or named range reference; where an Array is an array or array formula or a reference to a range of cells for which we want the number of columns.

Array: Used to build single formulas that produce multiple results or that operate on a group of arguments that are arranged in rows and columns. An array range shares a common formula; an array constant is a group of constants used as an argument.

Array formula: A formula that performs multiple calculations on one or more sets of values, and then returns either a single result or multiple results. Array formulas are enclosed between braces { } and are entered by pressing **CTRL + SHIFT + ENTER**.

Criteria are the form of a number, expression, cell reference, or text that defines which cells will be counted.

(c) Text Manipulation function.

(*i*) **TEXT** This function converts a numeric value to text in a specific number format.
TEXT (value, format_text) where

Value is a numeric value, a formula that evaluates to a numeric value, or a reference to a cell containing a numeric value.

Format_text is a numeric format as a text string enclosed in quotation marks. We can see various numeric formats by clicking the **Number, Date, Time, Currency, or Custom** in the Category box of the Number tab in the Format Cells dialog box, and then viewing the formats displayed.

(*ii*) CONCATENATE This function joins two or more text strings into one text string.

CONCATENATE (text1, text2, ...) where
text1, **text2**, …. are 2 to 255 text items to be joined into a single text item. The text items can be text strings, numbers, or single-cell references.

(*d*) Logical Function (Other than IF).

A **logical value** (true or false) outcome is the comparison of data values or results of arithmetic expressions compared with another data values or results of another arithmetical expressions using logical operator.

(*i*) **AND** function gives only a **TRUE** or **FALSE** answer
To determine whether the output will be TRUE or FALSE, the AND function evaluates at least one mathematical expression located in another cell in the spreadsheet.
= AND (logical-1, logical-2, ... logical-255) where
logical-1, **logical-2**, ... – refers to the cell reference that is being checked. Up to 255 logical values can be entered into the function. Returns **TRUE** if all its arguments evaluate to **TRUE**; returns **FALSE** if one or more arguments evaluate to **FALSE**

(*ii*) **OR** function is like other logical functions, the OR function gives only a **TRUE** or
FALSE answer. To determine whether the output will be **TRUE** or **FALSE**, the OR functions evaluates at least one mathematical expression located in another cell in the spreadsheet. This function returns TRUE if any argument is **TRUE**; returns **FALSE** if all arguments are **FALSE**.
= OR (**logical-1**, **logical-2**, ... logical-255) where
Logical-1, logical-2 ... – refers to the cell references that are being checked.
Up to 255 logical values can be entered into the function.

(*e*) Lookup and References Function.

The **LOOKUP** function returns a value either from a one-row or one-column range or from an array.The **LOOKUP** function has two syntax forms: **vector** and **array**.

The **vector form** of **LOOKUP** looks in a one-row or one-column range (known as a vector) for a value, and then returns a value from the same position in a second one-row or one-column range.

The **array form** of **LOOKUP** looks in the first row or column of an array for the specified value, and then returns a value from the same position in the last row or column of the array.

(*i*) **LOOKUP (Vector Form)**

LOOKUP (lookup_value, lookup_vector, result_vector)

Lookup_value is a value that LOOKUP searches for in the first vector.

Lookup value can be a number, text, a logical value, or a name or reference that refers to a value.

Lookup_vector is a range that contains only one row or one column. The values in lookup vector can be text, numbers, or logical values.

Result_vector is a range that contains only one row or column. It must be the same size as lookup vector.

If LOOKUP cannot find the lookup value, it matches the largest value in lookup vector that is less than or equal to lookup_value. If lookup_value is smaller than the smallest value in lookup_vector, LOOKUP gives

(ii) **LOOKUP (Array Form)**

LOOKUP (lookup_value, array)

Lookup_value is a value that LOOKUP searches for in an array. Lookup_value can be a number, text, a logical value, or a name or reference that refers to a value.

Array is a range of cells that contains text, numbers, or logical values that we want to compare with lookup_value.

If LOOKUP cannot find the lookup_value, it uses the largest value in the array that is less than or equal to lookup_value.

If lookup_value is smaller than the smallest value in the first row or column (depending on the array dimensions), LOOKUP returns the # N/A error value.

If array covers an area that is wider than it is tall (more columns than rows), LOOKUP searches for lookup_value in the first row.

If array is square or is taller than it is the wide (more rows than columns), LOOKUP searches in the first column.

(*iii*) **VLOOKUP** The VLOOKUP function, which stands for vertical lookup, helps us to find specific information in large data tables such as an inventory list of parts or a large employee contact list. The VLOOKUP function searches and matches first the required value from the column of a range of cells, and then returns a value from any cell on the same row of the range.

VLOOKUP (lookup_value, table_array, col_index_num, range_lookup) where

Lookup_value – The value to search in the first column of the table. Lookup_value can be a value or a reference. If look_up value is smaller than the smallest value in the first column of table array, **VLOOKUP** returns the # N/A error value.

Table_array – Two or more columns of data. Use a reference to a range or a range name. The values in the first column of table_array are the values searched by lookup_value. These values can be text, numbers, or logical values. Uppercase and lowercase texts are equivalent.

Col_index_num – The column number in table array from which the matching value must be returned. A col_index_num of 1 returns the value in the first column in table_array; a col_index_num of 2 returns the value in the second column in table_array, and so on.

Range_lookup – A logical value that specifies whether we want VLOOKUP to find an exact match or an approximate match:

If TRUE or omitted, an exact or approximate match is returned. If an exact match is not found, the next largest value that is less than lookup_value is returned. The values in the first column of table_array must be placed in ascending sort order; otherwise, VLOOKUP may not give the correct value.

If FALSE, VLOOKUP will only find an exact match. In this case, the values in the first column of table_array do not need to be sorted. If there are two or more values in the first column of table_array that match the lookup_value, the first value found is used. If an exact match is not found, the error value # N/A is returned.

(*f*) Financial Function.

(*i*) **ACCRINT** This function returns the accrued interest for a security that pays periodic interest.

ACCRINT (issue, first_interest, settlement, rate, par, frequency, basis, calc_method)

Issue is the security's issue date.

First_interest is the security's first interest date.

Settlement is the security's settlement date. The security settlement date is the date after the issue date when the security is traded to the buyer.

Rate is the security's annual coupon rate.

Par is the security's par value. By default Par is 1000.

Frequency is the number of coupon payments per year. For annual payments, frequency = 1; for Semiannual, frequency = 2; for quarterly, frequency = 4.

Basis is the type of day count basis to use.

(*ii*) **CUMIPMT** This function returns the cumulative interest paid between two periods.

CUMIPMT (rate, nper, pv, start_period, end_period, type)

Rate is the interest rate.

Nper is the total number of payment periods.

Pv is the present value.

Start_period is the first period in the calculation. Payment periods are numbered beginning with 1.

End_period is the last period in the calculation.
Type is the timing of the payment (which may be either 0 or 1) 0 (zero) means Payment at the end of the period 1 means Payment at the beginning of the period.

(*iii*) **PV** This function returns the present value of an investment. The present value is the total amount that a series of future payments is worth now.
PV (rate, nper, pmt, fv, type) where
Rate is the interest rate per period. For example, for an automobile loan at a 10% annual interest rate and installments are made the monthly payments, then the interest rate per month is 10%/12, or 0.83%. The value for rate into the function will be 10%/12, or 0.83%, or 0.0083.
Nper is the total number of payment periods in an annuity. For example, if this loan is a four-yearcar loan and makes monthly payments, then loan has 4*12 (or 48) periods. The value for nper will be 48.
Pmt is the payment made each period and cannot be change over the life of the annuity.
Fv is the future value, or a cash balance to attain after the last payment is made.
Type is the number 0 or 1 and indicates when payments are due.
An **annuity** is a series of constant cash payments made over a continuous period. For example, a car loan or a mortgage is an annuity.

(*iv*) **FV** This function returns the future value of an investment based on periodic, constant payment and a constant interest rate.
FV (**rate, nper, pmt**, **pv, type**) where
Rate is the interest rate per period.
Nper is the total number of payment periods in an annuity.
Pmt is the payment made each period; it cannot change over the life of the annuity. Typically, pmt contains principal and interest but no other fees or taxes. If pmt is omitted, then include the pv value in the argument.
Pv is the present value, or the lump-sum amount that a series of future payments is worth right now. If pv is omitted, it is assumed to be 0 (zero), and then include the pmt value in the argument.
Type is the number 0 or 1 and indicates when payments are due. If type is omitted, it is assumedto be 0.

(*v*) **PMT** The function calculates the periodic payment for an annuity, assuming equal payments and a constant rate of interest.
= PMT(rate, nper, pv, [fv], [type]) where
rate is the interest rate per period,
nper is the number of periods,
pv is the present value or the amount the future payments are worth presently,
fv is the future value or cash balance that after the last payment is made (a future value of zero when we omit this optional argument)
type is the value 0 for payments made at the end of the period or the value 1 for payments made at the beginning of the period.
The PMT function is often used to calculate the payment for mortgage loans that have a fixed rate of interest.

(*vi*) **RATE** This function returns the interest rate per period of an annuity.
RATE (nper, pmt, pv, fv, type, guess) where
Nper is the total number of payment periods in an annuity.
Pmt is the payment made each period and cannot change over the life of the annuity. Typically, pmt includes principal and interest but no other fees or taxes. If pmt is omitted, then include the fv as argument.
Pv is the present value — the total amount that a series of future payments is worth now.
Fv is the future value, or a cash balance attain after the last payment is made. If fv is omitted, it is assumed to be 0 (the future value of a loan, for example, is 0). **Type** is the number 0 or 1 and indicates when payments are due. 0 or omitted means payment is due at the end of the period 1 means payment is due at the beginning of the period.
Guess is the guess for what the rate will be. If omitted, it is assumed to be 10 per-cent.

(*vii*) **NPV** This function calculates the net present value of an investment by using a discount rate and a series of future payments (negative values) and income (positive values).
NPV (rate, value1, value2, ...) where
Rate is the rate of discount over the length of one period.
Value1, value2, ... are 1 to 254 arguments representing the payments and income. Value1,value2, ... must be equally spaced in time and occur at the end of each period. NPV uses the order of value1, value2 ...., to interpret the order of cash flows. It is essential that entry for payment and income values are in the correct sequence.

DATA ENTRY, TEXT MANAGEMENT AND CELL FORMATTING

DATA ENTRY

Excel also facilitates fast data entry; and automatically repeats data or can fill data in different cells (column wise or row wise.) For example, if we repeatedly type the days of the week in different cells instead of that we could use the built-in data fill options to fill the different cells with the days automatically.
Methods of Data Entry:
(*a*) The data fill options
(*b*) Import/Copy Data from other sources

DATA VALIDATION

Data validation is a feature to define restrictions on type of data entered into a cell. We can configure data validation rules for cells data that will not allow users to enter invalid data, There may be warning messages when users tries to type wrong data in the cell. The messages also guide users to

what input is expected for the cell, and instructions to correct any errors.

DATA VALIDATION FORM

A form, whether printed or online, is a document designed with a standard structure and format that makes it easier to capture, organise, and edit information. A data form is a dialog box that displays one complete record at a time. Data forms can be used to add, change, locate, and delete records. To input data into a spreadsheet, often we type the data into cells directly. That's where data validation comes in handy. Instead of typing the same thing again and again, we can enter data into cells using drop-down lists or using data input form. Using a data form can make data entry easier than moving from column to column when we have more columns of data than can be viewed on the screen. To create input data form it is necessary that all the data names must be entered in the first row of the worksheet, because the input form refers these data names.

DATA FORMATTING

Formatting of spreadsheets makes easier to read and understand he important information (*e.g.*, conditional formatting, number formatting, text and general spreadsheet formatting, etc.). On the Ribbon there are several tools and shortcuts to format spreadsheets effectively.

PREPARATION OF REPORTS USING PIVOT TABLE

A Pivot Table is way to present information in a report format. A Pivot Table report often provides enhanced layout, attractive and formatted report with improved readability. This report is prepared from the spreadsheet once we add the fields with appropriate level of details, calculations and group the data as per required information. The Pivot Table uses a List Data Table (Database).

A data table is a range of cells that shows the results of substituting different values in one or more formulas. There are two types of data table: One-variable and two variable.

One-variable data table. Formula used in a one-variable data table must refer to an input cell. The input cell is a cell used by Excel in which each input value from a data table is substituted (column-oriented, *i.e.*, input cell down one column or row-oriented, *i.e.*, across one-row).

PIVOT TABLES

The Pivot Table feature allows us to create a cross tabulation summary of data in which heading can subsequently moved to give different views of the data.

Advantages of Pivot Table (Report)

A Pivot Table report is designed for:

1. Querying large amounts of data in user-friendly ways.
2. Expanding and collapsing levels of data to focus on results, and providing from details to the summary of data for areas of interest.
3. Presenting concise, attractive, and annotated online or printed reports.
4. Filtering, sorting, grouping, and conditionally formatting the most useful and the interesting subset of data to enable us to focus on the information that we want.
5. Moving rows to column or columns to rows (or "pivoting") to see different summaries of the source data.
6. The use of a Pivot Table report is to analyse related totals, when we have a long list of figures to sum and to compare several facts about each figure

QUESTIONS FOR PRACTICE

MCQ

1. Which of the following is not a component of Computerised Accounting System?
(a) Hardware (b) Software
(c) Reports (d) Procedure

2. The step comprising of 'listing of accounts' in the process of installation of CAS is
(a) Grouping of Accounts.
(b) Chart of Accounts.
(c) Analysis of Transactions.
(d) Generations of Reports.

3. An effective Computerised Accounting System should be:
(a) Inaccessible, Complex, Fast.
(b) Simple, Integrated, Fast, Accurate, Reliable.
(c) Able to automatically assign codes to accounts.
(d) Obsolete, Reliable, Fast, Accurate.

4. A firm's payment to a supplier for goods (inventory) is recorded in
(a) Sales Accounting Sub-system.
(b) Costing Sub-system.
(c) Cash Sub-system.
(c) Budget Sub-system.

5. Hardware, Software and Data are some of the components of CAS. Identify the missing components:
(a) Procedure, People. (b) Timely access.
(c) Network. (d) Raw facts.

6. Which type of software package is suitable for an organisation where the volume of accounting transactions is very low and adaptability is very high?
(a) Specific (b) Generic
(c) Tailored (d) (1) and (3) both.

7. Which type of codes are usually assigned to source documents for facilitating document search?
(a) Range Codes (b) Mnemonic Codes
(c) Block Codes (d) Sequential Codes

8. Which of the following is a step in installation of CAS?
(a) Selection of Hardware and Software
(b) Planning
(c) Generation of Reports
(d) All of the above.

9. Match List I with List II:

List I	List II
A. Record of inflow and outflow of resources.	I. Accounting equation.
B. Equality of assets and liabilities.	II. Encryption.
C. Codification.	III. Hierarchy.
D. Difficult interpretation of information.	IV. Transaction.

Choose the correct option:
(a) A-III, B-I, C-IV, and D-II
(b) A-IV, B-I, C-II, and D-III
(c) A-I, B-IV, C-III, and D-II
(d) A-IV, B-I, C-III, and D-II

10. Match List I with List II:

List I	List II
A. Generation of reports vital for decision-making by management.	I. Expense Accounting Sub-system.
B. Preparation and analysis of budgets.	II. Management Information System
C. Generation of changes in cost of production.	III. Budget Sub-system.
D. Recording of expenses under various heads.	IV. Costing Sub-system.

Choose the correct option:
(a) A-III, B-I, C-IV, and D-II
(b) A-II, B-III, C-IV, and D-I
(c) A-II, B-III, C-I, and D-IV
(d) A-I, B-II, C-III, and D-IV

11. DBMS does not serve as a tool for which of the following:
(a) Organising data. (b) Creating data.
(c) Processing data. (d) Retrieving data.

12. _____ is the interactive program of a database application.
(a) Front-end (b) Back-end
(c) Report (d) Server

13. The fundamental building blocks of a database are
(a) Forms. (b) Queries.
(c) Cells. (d) Tables.

14. The common fields used in a relationship between tables are called
(a) Key fields. (b) Primary key.
(c) Attribute. (d) Record.

15. Employee name appears in a table only once. Such field in the said table is called
(a) Key field. (b) Primary key.
(c) Foreign key. (d) Record.

16. Duplication of information in a database is avoided by the process of
(a) Joining. (b) Saving.
(c) Normalisation. (d) Requirement analysis.

17. SQL stands for
(a) Simple Query Language.
(b) Singular Quantity Loading.
(c) Structured Query Language.
(d) Structured Quantity Load.

18. The maximum number of characters that can be used to write a field name in Access is
(a) 32. (b) 64.
(c) 108. (d) Unlimited.

19. The punctuation character that can be used in a field name is
(a) Exclamation mark. (b) Brackets.
(c) period. (d) Underscore.

20. _____ may be used to obtain a well formatted printable data from Access database.
(a) Report (b) Form
(c) Query (d) Table

21. 'Record' in a database table is a
(a) Size of the table.
(b) Horizontal row of table.
(c) Vertical column of table.
(d) Name of the table.

22. In MS Access, task-oriented tabs comprising of commands and features are placed in a standard area, which is called
(a) File. (b) Print preview.
(c) Join. (d) Ribbon.

23. The tool that guides the user through creation of database and queries and forms is called
(a) Spreadsheet. (b) DBMS.
(c) Access Wizard. (d) SQL.

24. The width of a column in a database table can be set to accommodate the longest entry by using
(a) View. (b) Create Tab.
(c) Query. (d) Best Fit.

25. A user can retrieve data from multiple tables in Access by placing specific conditions with the help of
(a) Design View. (b) Report.
(c) Query. (d) Joining.

26. In the formula "Compound interest = Maturity Value – Principle Amount", Maturity Value is the _____ value.
(a) Basic (b) Formula
(c) Derived (d) Cell

27. The key stroke for reaching the top of worksheet (cell A1) is
(a) CTRL + END. (b) CTRL + HOME.
(c) HOME key. (d) Up arrow key(t).

28. Every formula must start with
(a) An = sign (b) A * sign.
(c) A Label. (d) A Cell Reference.

29. 'What-if' scenarios are also called _____
(a) Labels.
(b) Data entry.
(c) Nested conditional operations.
(d) References.

30. Behind every computed number in an Excel-sheet, Excel is hiding
(a) Labels. (b) Worksheet names.
(c) Rows and columns. (d) Cell References.

SOLUTION FOR PRACTICE QUESTIONS

SOLUTION FOR MCQ QUESTIONS

1.	**(c)**	**11.**	**(b)**	**21.**	**(b)**
2.	**(b)**	**12.**	**(a)**	**22.**	**(d)**
3.	**(b)**	**13.**	**(d)**	**23.**	**(c)**
4.	**(c)**	**14.**	**(a)**	**24.**	**(d)**
5.	**(a)**	**15.**	**(b)**	**25.**	**(c)**
6.	**(b)**	**16.**	**(c)**	**26.**	**(a)**
7.	**(d)**	**17.**	**(c)**	**27.**	**(b)**
8.	**(d)**	**18.**	**(b)**	**28.**	**(a)**
9.	**(d)**	**19.**	**(d)**	**29.**	**(c)**
10.	**(b)**	**20.**	**(a)**	**30.**	**(d)**