

Draft Study Material



# JUNIOR SOFTWARE DEVELOPER

(Job Role)

(Qualification Pack: Ref. Id. SSC/Q0508)

Sector: Information Technology-Information Technology Enable Services (IT-ITeS)

(Grade XII)



**PSS CENTRAL INSTITUTE OF VOCATIONAL EDUCATION**

(a constituent unit of NCERT, under Ministry of Education, Government of India)

Shyamla Hills, Bhopal - 462 002, M.P., India

© PSS Central Institute of Vocational Education, Bhopal 2024

No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without the prior permission of the publisher.

PSSCIVE Draft Study Material © Not to be Published

## Preface

Vocational Education is a dynamic and evolving field, and ensuring that every student has access to quality learning materials is of paramount importance. The journey of the PSS Central Institute of Vocational Education (PSSCIVE) toward producing comprehensive and inclusive study material is rigorous and time-consuming, requiring thorough research, expert consultation, and publication by the National Council of Educational Research and Training (NCERT). However, the absence of finalized study material should not impede the educational progress of our students. In response to this necessity, we present the draft study material, a provisional yet comprehensive guide, designed to bridge the gap between teaching and learning, until the official version of the study material is made available by the NCERT. The draft study material provides a structured and accessible set of materials for teachers and students to utilize in the interim period. The content is aligned with the prescribed curriculum to ensure that students remain on track with their learning objectives.

The contents of the modules are curated to provide continuity in education and maintain the momentum of teaching-learning in vocational education. It encompasses essential concepts and skills aligned with the curriculum and educational standards. We extend our gratitude to the academicians, vocational educators, subject matter experts, industry experts, academic consultants, and all other people who contributed their expertise and insights to the creation of the draft study material.

Teachers are encouraged to use the draft modules of the study material as a guide and supplement their teaching with additional resources and activities that cater to their students' unique learning styles and needs. Collaboration and feedback are vital; therefore, we welcome suggestions for improvement, especially by the teachers, in improving upon the content of the study material.

This material is copyrighted and should not be printed without the permission of the NCERT-PSSCIVE.

Deepak Paliwal  
(Joint Director)  
PSSCIVE, Bhopal

Date: 07 September, 2024

## STUDY MATERIAL DEVELOPMENT COMMITTEE

### Members

*Deepak D. Shudhalwar, Professor (CSE), Head, Department of Engineering and Technology, PSSCIVE, NCERT, Bhopal, Madhya Pradesh*

*Ganesh Kumar Dixit, Assistant Professor in IT-ITeS (Contractual), Department of Engineering and Technology, PSSCIVE, NCERT, Bhopal*

*Prakash Khanale, Professor and Head, Department of Computer Science, DSM College, Parbhani, Maharashtra*

*Rizwan Alam, Assistant Professor in IT-ITeS (Contractual), Department of Engineering and Technology, PSSCIVE, NCERT, Bhopal*

### Member Coordinator

*Deepak D. Shudhalwar, Professor (CSE), Head, Department of Engineering and Technology, PSSCIVE, NCERT, Bhopal, Madhya Pradesh*

PSSCIVE Draft Study Material © Not to be Published

## Table of Contents

S. No.	Title	Page No.
1	Module 1: RDBMS Concepts in MySQL	1
	Module Overview	1
	Learning Outcomes	1
	Module Structure	2
	Session 1: RDBMS Concepts	2
	Check Your Progress	12
	Session 2: Structured Query Language (SQL)	15
	Check Your Progress	45
	Session 3: Functions In SQL	50
	Check Your Progress	69
2	Module 2: Advanced Python Programming	72
	Module Overview	72
	Learning Outcomes	72
	Module Structure	72
	Session 1: Implementing Data Structure using Stack & Queue	73
	Check Your Progress	78
	Session 2: Exception Handling in Python	79
	Check Your Progress	88
	Session 3: File Handling in Python	89
	Check Your Progress	100
	Session 4: Numpy Array	102
	Check Your Progress	109
	Session 5: Pandas and Series in Python	110
	Check Your Progress	124
	Session 6: Graphical Representation using MatpotLib	125
	Check Your Progress	134
	Session 7: Database Connectivity with MySQL	134
	Check Your Progress	144
3	Module 3: Software Engineering	145
	Module Overview	145
	Learning Outcomes	145

	Module Structure	145
	Session 1: Software Engineering Concepts	146
	Check Your Progress	154
	Session 2: Software Development Process	155
	Check Your Progress	181
	Assignment	183
4	Module 4: Emerging Trends and Social Impact	193
	Module Overview	193
	Learning Outcomes	193
	Module Structure	193
	Session 1: Emerging Trends and Technologies	193
	Check Your Progress	208
	Session 2: Societal Impact	209
	Check Your Progress	218
5	Answer Key	220

**Module 1****RDBMS CONCEPTS  
IN MYSQL****Module Overview**

RDBMS stands for Relational Database Management System. It is a type of database management system (DBMS) that stores data in a row-based table structure which connects related data elements. It is called relational because the values within each table are related to each other. This makes it easy to locate and access specific values within the database.



In this unit, you will understand the various data models and various concepts associated with RDBMS. There are various database management software available in the market. Popular examples of RDBMSs include MySQL, Oracle, and SQL Server. The RDBMS concepts using MySQL is covered in this unit.

SQL stands for Structured Query Language, is a special-purpose programming language designed to manage data in a relational database management system (RDBMS) or stream processing in a relational data stream management system (RDSMS). SQL is used to search, store, modify records in database management system. SQL queries are used to retrieve the data needed for specific job functions. It is a standardized way to request information from relational databases. In this unit, you will be able to create database objects, insert data in database and use various types of commands to retrieve the required data from the database.

SQL function is used to perform particular tasks and it returns zero or more values as a result. Functions are useful while writing SQL queries. Functions can be applied to work on single or multiple records (rows) of a table. There are various readily available functions in SQL that can be used in queries. It includes single row functions, multiple row functions, group records based on some criteria. The use of these functions is illustrated in this unit.

**Learning Outcomes**

After completing this module, you will be able to:

- Describe the concepts of Relational Database Management System
- Describe the Structured Query Language (SQL)
- Execute the SQL commands in MySQL

- Execute the SQL functions in MySQL

## Module Structure

Session 1: RDBMS Concepts

Session 2: Structured Query Language (SQL)

Session 3: Functions In SQL

### Session 1: RDBMS Concepts

Kushaal was playing with his father's mobile phone. Accidentally all the contacts on the phone deleted by him. To recover these contacts his father consulted a technician. The technician recovered all the contacts available using the Google contacts as these contacts are linked with the Google account. Thus it is possible to save and manage the contacts in smartphone. This is possible with the use of database application available with Google. You can easily understand that the data can be saved and managed through database application.

*Fig. 1.1 Illustration*

In this session, you will understand the concept of database and how data is organised in the database. The various data models and the various concepts associated with database application such as constraints, primary key, foreign key are also discussed. There are various database management software available in the market. We will discuss about MySQL which is a popular relational database management system (RDBMS).

#### 1.1 INTRODUCTION TO DATABASE SYSTEMS

The word data is taken from "*Datum*", means raw facts. Datum is a single piece of factual information of interest to us. Data, the plural of datum, is a collection of information. Data is the name given to basic raw facts and entities such as names, numbers and quantity. Data can be defined as a collection of facts and records. Data, representing facts, figures, and ideas, are commonly used in everyday life. Data needs to be managed to use it effectively.

Data items are organised or processed to produce the information. It can be used for processing some useful information from it. The examples of data are weights, prices, costs, numbers of items sold, employee names, product names, addresses, tax codes, registration fees, obtained marks, reservation details, images, sounds, multimedia and animated data. Data can exist in form of text, graphics, sound, video that represents every kind of information. The data items can be stored manually in a diary. But it is difficult to retrieve and process the data items when there are large number of data items. Through the relational database systems, users can access a view of data called relations. With relational database management systems (RDBMS), programmers can perform database operations without knowing data storage details.

It is observed that the schools are maintaining the student data and that is stored in the register. After several years of leaving the school, the student can get the duplicate of school leaving certificate. For issuing such certificate the office staff check the student data from the registers which is maintained year-wise and class-wise. By checking the student record, the office staff can easily give the certificate to the student. This is how the schools are maintaining the student data in the register.

The office staff also manually maintain student details who are presently learning. Their Admission number, Name, Date of Birth, Address, Contact Number are stored in the school register.

There are two major types of databases – relational and non-relational. Relational databases are the most commonly used databases today. The following are several types of databases in use.

*Flat file databases* – Stores data in permanent files that mostly are in text form;

*Hierarchical databases* – Arranges data in a tree-like structure;



*Network databases* – Arranges data in network-like structure;

*Relational databases* – Contains a set of tables in which data are related;

*Object databases* – Represents information in the form of objects as used in object-oriented programming;

## **1.2 FILE SYSTEM**

Now you must have noticed that maintaining such type of records manually does not allow to correct, modify or delete the data in the register. Also searching the details of the student is difficult. To overcome the hassles faced in manual record keeping, this data can be stored in computer. The student details are stored in computer in the form of separate file.

In computer, any contents are stored in the form of file, which is opened and viewed in the respective software. In computer, file is a container to store data or information. These files are stored on the storage device of computer, such as hard disk drive or pen drive.

The student data can be stored in the document file or spreadsheet file. These files stored on computer can be accessed quickly. To process or manipulate this data, it is required to write the program in computer programming languages. The various operations can be performed through computer programming. It includes searching, sorting, computing the percentage of marks, number of days attendance, retrieving the data.

### **1.2.1 Limitations of a File System**

There are certain limitations to maintain and manipulate such type of data when there are several hundreds or thousands of students. It also becomes difficult to maintain the number of files as it increases the volume when data grows. There are certain limitations of file system to maintain such type of data. The limitations of file system are,

**Difficulty in Access** – Files themselves do not provide any mechanism to retrieve data. Data maintained in a file system are accessed through application programs. While writing such programs, the developer may not anticipate all the possible ways in which data may be accessed. So, sometimes it is difficult to access data in the required format and one has to write application program to access data.

**Data Redundancy** – Redundancy means same data are duplicated in different files. For example if we are maintaining students data for the various purpose then data such as student names are maintained in different files. The common data in all such files are required to be maintained number of times. This may cause the data redundancy which is difficult to avoid in a file system. Redundancy leads to excess storage use and may cause data inconsistency also.

**Data Inconsistency** – Data inconsistency occurs when same data maintained in different places do not match. If a student wants to get changed the spelling in name, it needs to be changed in the number of files where it appears. Likewise, if a student leaves school, the details need to be deleted from these files. As the files are being maintained by different people, the changes may not happen in one of the files. In that case, the student name will be different (inconsistent) in both the files.

**Data Isolation** – Although these files are maintained for the students of the same class, but there is no link or mapping between these files. The school will have to write separate programs to access these files. This is because data mapping is not supported in file system. In a more complex system where data files are generated by different person at different times, files being created in isolation may be of different formats. In such case, it is difficult to write new application programs to retrieve data from different files.

**Data Dependence** – Data are stored in a specific format or structure in a file. If the structure or format itself is changed, all the existing application programs accessing that file also need to be changed. Otherwise, the programs may not work correctly. This is data dependency. Hence, updating the structure of a data file requires modification in all the application programs accessing that file.

**Controlled Data Sharing** – There can be different category of users like teacher, office staff and parents. Ideally, not every user should be able to access all the data. It means different types of users should be given different types of access, such as read only. It is very difficult to enforce this kind of access control in a file system while accessing files through application programs.

### 1.3 DATABASE MANAGEMENT SYSTEM

Limitations faced in file system can be overcome by storing the data in a database where data are logically related. A database management systems (DBMSs) is used as an interface to manage databases.

A database is an *organized collection of data*, generally stored and accessed electronically from a computer system. It supports the storage and manipulation of data. In other words, databases are used by an organization as a method of storing, managing and retrieving information. It is possible to store and organise related data in a database so that it can be managed in an efficient and easy way.

A DBMS is a collection of software components designed to create and maintain databases and control all access to them. DBMS allows to create a database, store, manage, update/modify and retrieve data from that database by users or application programs. DBMS is used to provide an effective method of performing database operations, troubleshooting database issues, and restricting data access. Relational Database Management System (RDBMS), which is still popular today, is an advanced version of a DBMS system. Dr. E. F. Codd defined the criterias to determine whether a DBMS is a relational database management system or not. These criteria are known as twelve rules Codd's (E. F. Codd, 1985).

Some examples of open source and commercial DBMS include MySQL, Oracle, PostgreSQL, SQL Server, Microsoft Access, MongoDB as presented in Table 1.1.

**Table 1.1 Popular DBMS**

DBMS	Primary Database Model	License
Oracle	RDBMS	Commercial (restricted free version is available)
MySQL	RDBMS	Open Source
Microsoft SQL Server	RDBMS	Commercial (restricted free version is available)
PostgreSQL	RDBMS	Open Source
MongoDB	Document store	Open Source



**Fig: 1.2 Different type of DBMS/RDBMS available in market**

Some database management systems include a graphical user interface for users to create and manage databases. Other database systems use a command line interface that requires users to use programming commands to create and manage databases.

A database system hides certain details about how data are actually stored and maintained. Thus, it provides users with an abstract view of the data. A database system has a set of programs through which users or other programs can access, modify and retrieve the stored data.

The DBMS serves as an interface between the database and end users or application programs. Retrieving data from a database through special type of commands is called querying the database. In addition, users can modify the structure of the database itself through a DBMS.

Databases are widely used in various fields. Some applications are given in Table 1.2.

**Table 1.2 Use of Database in Real-life Applications**

Application	Database to maintain data about
Banking	customer information, account details, loan details, transaction details.
Crop Loan	kisan credit card data, farmer's personal data, land area and cultivation data, loan history, repayment data.
Inventory	Management product details, customer information, order details, delivery data.
Organisation Resource Management	employee records, salary details, information, branch locations.
Online Shopping	items description, user login details, users preferences details,

### 1.3.1 Limitations of DBMS

**Increased Complexity** – Use of DBMS increases the complexity of maintaining functionalities like security, consistency, sharing and integrity.

**Increased data vulnerability** – As data are stored centrally, it increases the chances of loss of data due to any failure of hardware or software. It can bring all operations to a halt for all the users.

### 1.3.2 Application of the DBMS system

Here, are few important applications of the DBMS system:

- Student Admission System, School Examination System, Library Management System
- Payroll, HR, Sales & Personnel Management System
- Accounting System, Hotel Reservation System and Airline Reservation System
- It is used in the Banking system for Customer information, account activities, Payments, deposits, loans etc.
- Insurance management system
- DBMS system also used by universities to keep all records
- Finance for storing information about stock, sales, and purchases of financial instruments like stocks and bonds.

### 1.3.2 Advantages of DBMS system

The advantages of DBMS system are:

- DBMS offers a variety of techniques to store & retrieve data
- Uniform administration procedures for data storage and retrieval
- Application programmers never exposed to details of data representation and Storage.
- A DBMS uses various powerful functions to store and retrieve data efficiently.
- Offers Data independence, Data Integrity and Data Security and reduce data redundancy.
- The DBMS implies integrity constraints to get a high level of protection against prohibited access to data.

- Reduced Application Development Time and occupy lesser space

#### 1.3.4 Disadvantages of the DBMS system

The disadvantages of DBMS system are:

- Cost of Hardware and Software of a DBMS is quite high, which increases the budget of your organization.
- Most database management systems are often complex systems, so the training for users to use the DBMS is required.
- The use of the same program at a time by many users sometimes lead to the loss of some data.
- DBMS can't perform sophisticated calculations
- Data-sets begins to grow large as it provides a more predictable query response time.
- It required a processor with the high speed of data processing.
- The database can fail because of power failure or the whole system stops.
- The cost of DBMS is depended on the environment, function, or recurrent annual maintenance cost.

#### 1.3.5 Comparison of Database Management System (DBMS) with File System

The comparative points of DBMS) with File System are given in Table 1.3.

**Table 1.3: Comparison of DBMS with File System**

File System	DBMS
A file system is a software that manages and organizes the files in a storage medium. It controls how data is stored and retrieved.	DBMS or Database Management System is a software application. It is used for accessing, creating, and managing databases.
The file system provides the details of data representation and storage of data.	DBMS gives an abstract view of data that hides the details
Storing and retrieving of data can't be done efficiently in a file system.	DBMS is efficient to use as there are a wide variety of methods to store and retrieve data.
It does not offer data recovery processes.	There is a backup recovery for data in DBMS.
The file system doesn't have a crash recovery mechanism.	DBMS provides a crash recovery mechanism
Protecting a file system is very difficult.	DBMS offers good protection mechanism.
In a file management system, the redundancy of data is greater.	The redundancy of data is low in the DBMS system.
Data inconsistency is higher in the file system.	Data inconsistency is low in a database management system.
The file system offers lesser security.	Database Management System offers high security.
File System allows you to stores the data as isolated data files and entities.	Database Management System stores data as well as defined constraints and interrelation.
Not provide support for complicated transactions.	Easy to implement complicated transactions.
The centralization process is hard in File Management System.	Centralization is easy to achieve in the DBMS system.
It doesn't offer backup and recovery of data if it is lost.	DBMS system provides backup and recovery of data even if it is lost.
There is no efficient query processing in the file system.	You can easily query data in a database using the SQL language.

These system doesn't offer concurrency.

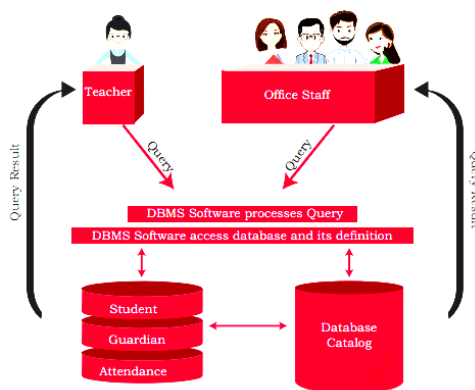
DBMS system provides a concurrency facility.

#### 1.4 Key Concepts in DBMS

It is important to understand the following concepts to efficiently manage data using a DBMS.

##### 1.4.1 Database schema

A database schema is a set of schema for a database's relations. It consists of table with all attributes with their data types and constraints if any. It also represents the relationships among the tables. It is also used to visualize the logical architecture of database and how the data are organized in a database. The schema of a relation may not change, but the relation, which is a variable, changes over time. (Figure 1.3)



**Fig. 1.3 Database schema for Student Attendance system**

##### 1.4.2 Data Constraint

Sometimes it is required to put certain restrictions or limitations on the type of data to be inserted in the columns of a table. This is done by specifying constraints on that column(s) while creating the tables. For example, the constraint that the column *mobile number* can only have non-negative integer values of exactly 10 digits. Since each student shall have one unique roll number, we can put the NOT NULL and UNIQUE constraints on the *RollNumber* column. Constraints are used to ensure accuracy and reliability of data in the database

##### 1.4.3 Meta-data or Data Dictionary

The database schema along with various constraints on the data is stored by DBMS in a database catalog or dictionary, called meta-data. A meta-data is data about the data.

##### 1.4.4 Database Instance

When we define database structure or schema, state of database is empty. After loading data, the state or snapshot of the database at any given time is the database instance. We may then retrieve data through queries or manipulate data through updation, modification or deletion. Thus, the state of database can change, and thus a database schema can have many instances at different times.

##### 1.4.5 Query

A query is a request to a database for obtaining information in a desired way. Query can be made to get data from one table or from a combination of tables. For example, "find names of all those students present today" is a query to the database. To retrieve or manipulate data, the user needs to write query using a query language called Structured Query Language (SQL).

##### 1.4.6 Data Manipulation

Modification of database consists of three operations viz. Insertion, Deletion or Updation. Suppose Rivaan joins as a new student in the class then the student details need to be added in **StudentRecord** as well as in **ParentRecord** files of the **STUDENTATTENDANCE** database. This is called Insertion operation on the database. In case a student leaves the school, then student as well as parent data need to be removed from **StudentRecord**, **ParentRecord** and

**AttendanceRecord** tables, respectively. This is called Deletion operation on the database. Suppose Rivaan's Parent has changed his mobile number, his *Par\_Phone* should be updated in **ParentRecord** file. This is called Update operation on the database.

#### 1.4.7 Database Engine

Database engine is the underlying component or set of programs used by a DBMS to create database and handle various queries for data retrieval and manipulation.

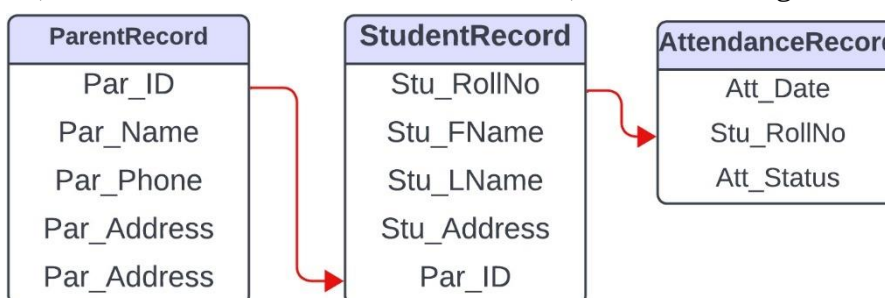
### 1.5 RELATIONAL DATA MODEL

A data model describes the structure of the database and represent data. It defines and represents relationships among relations. In database design, first the conceptual data model is designed for non-technical users. Then based on the conceptual data models, the logical data models are designed by the technical users. It represents how to store and retrieve data logically. Finally, the logical design models is converted into physical data models that show all table structures. Relational data model is the most commonly used data model. So here we will focus on relational data model.

#### 1.5.1 Key terms in Relational Data Model

In relational model, tables are called relations that store data for different entities. Each relation in a relational model represents a specific type of entity. An entity is an object and we store data about the object. In other words, a relation is a two-dimensional table used to store data.

Let us consider, the relational database **SCHOOLRECORD** along with the three relations (tables) **StudentRecord**, **AttendanceRecord** and **ParentRecord**, as shown in Figure 1.4.



**Fig. 1.4 Representing SchoolRecord database using Relational Data Model**

Observe that, a relation **AttendanceRecord** has attribute *Stu\_RollNo* which links it with corresponding student record in relation **StudentRecord**. Similarly, attribute *Par\_ID* is placed with **StudentRecord** table for extracting parent details of a particular student. If linking attributes are not there in appropriate relations, it will not be possible to keep the database in correct state and retrieve valid information from the database.

**Table 1.4 Relation schema along with its description of Student Attendance database**

Relation Schema	Description of attributes
<b>StudentRecord</b> ( <i>Stu_RollNo</i> , <i>Stu_FName</i> , <i>Stu_LName</i> , <i>Stu_DOB</i> , <i>Stu_Address</i> , <i>Par_ID</i> )	<i>Stu_RollNo</i> : unique id of the student <i>Stu_FName</i> : First name of the student <i>Stu_LName</i> : Last name of the student <i>Stu_DOB</i> : Student's date of birth <i>Stu_Address</i> : Home address of the student <i>Par_ID</i> : unique id of the parent of the student
<b>AttendanceRecord</b> ( <i>Att_Date</i> , <i>Stu_RollNo</i> , <i>Att_Status</i> )	<i>Att_Date</i> : date on which attendance is taken <i>Stu_RollNo</i> : roll number of the student <i>Att_Status</i> : Either P (for present) or A (for absent) <b>Note</b> : Combination of <i>Att_Date</i> and <i>Stu_RollNo</i> will be unique in each record of the table

<b>ParentRecord</b> (Par_ID, Par_Name, Par_Phone, Par_Address, Par_Email)	Par_ID : unique id of the parent Par_Name : Name of the parent Par_Phone : Contact number of the parent Par_Address : Address of the parent Par_Email : Email id of the parent
--	--

Each tuple (row) in a relation (table) corresponds to data of a real-world entity as in **StudentRecord**, **ParentRecord**, and **AttendanceRecord**. In the **ParentRecord** relation (Table 1.4), each row represents the facts about the parent and each column name in the **ParentRecord** table is used to interpret the meaning of data stored under that column. A database that is modeled on relational data model concept is called *Relational Database*. Figure 1.5 shows relation **ParentRecord** with some populated data.

Let us now understand the commonly used terminologies in relational data model using Figure 1.5.

**Relation ParentRecord with 5 Attribute/Columns**

Par_ID	Par_Name	Par_Phone	Par_Address	Par_Email
111122223333	Manu P Singh	9834567890	203, Khandari, Agra, UP	mpsingh@xyz.com
222233331111	Ashok K Sharma	9845678910	144 Gr Kailash, New Delhi	aksharmaji@abc.com
333311112222	Ashutosh Gaur	9856789120	JP Greens, Noida, UP	ashutoshgaur@lat.com
112233445566	Sachin Agrawal	9812389120	Kanda, Bagheshwar, UK	sachinag@bby.com
223344556611	Chandra Roy	9891201238	Fortune Somya, Bhopal, MP	Ch.roy@pqr.com
334455661122	Dinesh Dixit		Lajpat Nagar, Mathura, UP	dinesh.dixit@hpq.com
445566112233	Rizwan Alam	9255614563	Deep Nagar, Sahrsa, Bihar	riz.alam@ppc.com
556611223344	Ashish Gupta	8544556978	T Nagar, Hyderabad, Telangana	ashish.gupta@drdd.com
661122334455	Gurmeet Singh	9635214789	Shahid Nagar, Amritsar, PB	gurmeet.007@ppc.com
991122334455	Michal DeSousa	8554658958	Guindy, Chennai, TN	michal.don@abc.com

#### Facts about Relation ParentRecord

1. Degree (Number of attributes ) = 5
2. Cardinality (Number of rows / tuple/ records ) = 10
3. Relation is a flat file having single value in each column and each record has same number of column.

Record or Tuple or Row

**Fig 1.5 Relation ParentRecord with its attributes (Columns) and tuples (Rows)**

**Attribute** – Characteristic or parameters for which data are to be stored in a relation. Simply stated, the columns of a relation are the attributes which are also referred as fields. For example, Par\_ID, Par\_Name, Par\_Phone and Par\_Address are attributes of relation **ParentRecord**.

**Tuple** – Each row of data in a relation (table) is called a tuple. In a table with n columns, a tuple is a relationship between the n related values.

**Domain** – It is a set of values from which an attribute can take a value in each row. Usually, a data type is used to specify domain for an attribute. For example, in **StudentRecord** relation, the attribute Stu\_RollNo takes integer values and hence its domain is a set of integer values. Similarly, the set of character strings constitutes the domain of the attribute Stu\_Fname.

**Degree** – The number of attributes in a relation is called the Degree of the relation. For example, relation **ParentRecord** with four attributes is a relation of degree 5.

**Cardinality** – The number of tuples in a relation is called the Cardinality of the relation. For example, the cardinality of relation **ParentRecord** is 10 as there are 10 tuples in the table.

### 1.5.2 Three Important Properties of a Relation

In relational data model, following three properties are observed with respect to a relation which makes a relation different from a data file or a simple table.

**Property 1:** imposes following rules on an attribute of the relation.

- Each attribute in a relation has a unique name.
- Sequence of attributes in a relation is immaterial.

**Property 2:** governs following rules on a tuple of a relation.

- Each tuple in a relation is distinct. For example, data values in no two tuples of relation **AttendanceRecord** can be identical for all the attributes. Thus, each tuple of a relation must be uniquely identified by its contents.
- Sequence of tuples in a relation is immaterial. The tuples are not considered to be ordered, even though they appear to be in tabular form.

**Property 3:** imposes following rules on the state of a relation.

- All data values in an attribute must be from the same domain (same data type).
- Each data value associated with an attribute must be atomic (cannot be further divisible into meaningful subparts). For example, Par\_Phone of relation **ParentRecord** has ten digits numbers which is indivisible.
- No attribute can have many data values in one tuple. For example, any Parent cannot specify multiple contact numbers under Par\_Phone attribute.
- A special value “NULL” is used to represent values that are unknown or non-applicable to certain attributes. For example, if a parent does not share his or her contact number with the school authorities, then Par\_Phone is set to NULL (data unknown).

## 1.6 KEYS IN A RELATIONAL DATABASE

The tuples within a relation must be distinct. It means no two tuples in a table should have same value for all attributes. That is, there should be at least one attribute in which data are distinct (unique) and not NULL. That way, we can uniquely distinguish each tuple of a relation. So, relational data model imposes some restrictions or constraints on the values of the attributes and how the contents of one relation be referred through another relation. These restrictions are specified at the time of defining the database through different types of keys as given below:

### 1.6.1 Candidate Key

A relation can have one or more attributes that takes distinct values. Any of these attributes can be used to uniquely identify the tuples in the relation. Such attributes are called candidate keys as each of them are candidates for the primary key.

As shown in Figure 1.5, the relation **ParentRecord** has five attributes out of which Par\_ID and Par\_Phone always take unique values. No two parents will have same phone number or same Par\_ID. Hence, these two attributes are the candidate keys as they both are candidates for primary key.

### 1.6.2 Primary Key

Out of one or more candidate keys, the attribute chosen by the database designer to uniquely identify the tuples in a relation is called the primary key of that relation. The remaining attributes in the list of candidate keys are called the alternate keys.

In the relation **ParentRecord**, suppose Par\_ID is chosen as primary key, then Par\_Phone will be called the alternate key.



### 1.6.3 Composite Primary Key

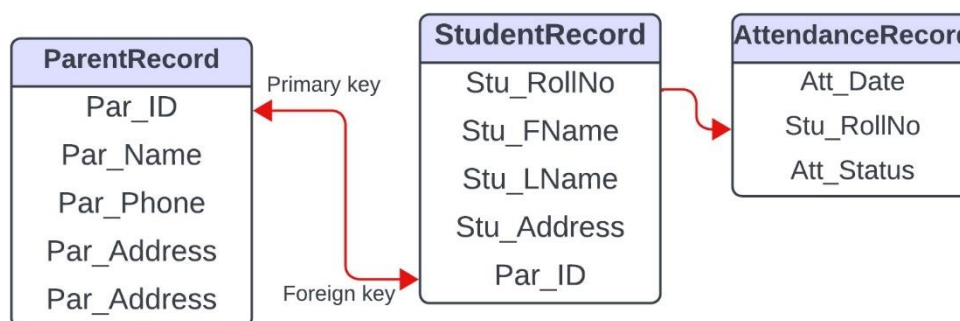
If no single attribute in a relation is able to uniquely distinguish the tuples, then more than one attributes are taken together as primary key. Such primary key consisting of more than one attribute is called **Composite Primary key**. In relation **AttendanceRecord**, Roll Number cannot be used as primary key as roll number of same students will appear in another row for a different date. Similarly, in relation **AttendanceRecord**, Att\_Date cannot be used as primary key because same date is repeated for each roll number.

However, combination of these two attributes Stu\_RollNo and Att\_Date together would always have unique value in **AttendanceRecord** table as on any working day, of a student would be marked attendance only once. Hence {Stu\_RollNo, Att\_Date} will combine to make the of **AttendanceRecord** relation composite primary key.

### 1.6.4 Foreign Key

A foreign key is used to represent the relationship between two relations. A foreign key is an attribute whose value is derived from the primary key of another relation. This means that any attribute of a relation (referencing), which is used to refer contents from another (referenced) relation, becomes foreign key if it refers to the primary key of referenced relation. The referencing relation is called Foreign Relation. In some cases, foreign key can take NULL value if it is not the part of primary key of the foreign table.

The relation in which the referenced primary key is defined is called primary relation or master relation. In Figure 1.6, two foreign keys in **STUDENTATTENDANCE** database are shown using schema diagram where the foreign key is displayed as a directed arc (arrow) originating from it and ending at the corresponding attribute of the primary key of the referenced table. The underlined attributes make the primary key of that table.



**Fig. 1.6 StudentAttendance database with the Primary and Foreign keys**

### Summary

- A file in a file system is a container to store data in a computer.
- File system suffers from Data Redundancy, Data Inconsistency, Data Isolation, Data Dependence and Controlled Data sharing.
- Database Management System (DBMS) is a software to create and manage databases. A database is a collection of tables.
- Database schema is the design of a database
- A database constraint is a restriction on the type of data that that can be inserted into the table.
- Database schema and database constraints are stored in database Catalog. Whereas the snapshot of the database at any given time is the database instance.
- A query is a request to a database for information retrieval and data manipulation (insertion, deletion or update). It is written in Structured Query Language (SQL).

- Relational DBMS (RDBMS) is used to store data in related tables. Rows and columns of a table are called tuples and attributed respectively. A table is referred to as a relation.
- Restrictions on data stored in a RDBMS is applied by use of keys such as Candidate Key, Primary Key, Composite Primary Key, Foreign Key.
- Primary key in a relation is used for unique identification of tuples.
- Foreign key is used to relate two tables or relations.
- Each column in a table represents a feature (attribute) of a record. Table stores the information for an entity whereas a row represents a record.
- Each row in a table represents a record. A tuple is a collection of attribute values that makes a record unique.
- A tuple is a unique entity whereas attribute values can be duplicate in the table.

## Check Your Progress

### A. Multiple choice questions

1. A database is a (a) organized collection of information that cannot be accessed, updated, and managed (b) collection of data or information without organizing (c) organized collection of data or information that can be accessed, updated, and managed (d) organized collection of data that cannot be updated
2. Which of the following is not a valid SQL type? (a) float (c) numeric (c) decimal (d) character
3. In DBMS, table is known as \_\_\_\_\_ and row is known as \_\_\_\_\_. (a) relation, tuple (b) tuple, tuple (c) tuple, relation (d) relation, relation
4. In any table, the data types describe the kind of \_\_\_\_\_ that it can contain. (a) table (b) data (c) number (d) column
5. The SQL statement used to select data items from the database is (a) SELECT (b) USE (c) ALTER (d) CREATE
6. The database can be renamed using \_\_\_\_\_ SQL statement (a) CREATE DATABASE (b) RENAME DATABASE (c) DROP DATABASE (d) SELECT DATABASE
7. The syntax used to show all databases is (a) USE DATABASES (b) SELECT DATABASES (c) SHOW DATABASES (d) DISPLAY DATABASE
8. In a database table the field which uniquely identifies each row in the table is known as \_\_\_\_\_ (a) primary key (b) unique key (c) composite key (d) foreign key
9. Foreign key is a \_\_\_\_\_ key in another table. (a) primary (b) unique (c) composite (d) candidate key
10. The multiple columns that are used as primary key is known as (a) unique key (b) composite key (c) foreign key (d) candidate key
11. Which of the following key is used to link between two tables (a) primary (b) foreign (c) composite (d) unique
12. A primary key cannot be (a) Zero (b) foreign key (c) duplicate (d) NULL

### B. Fill in the blanks

1. In DBMS, table is known as \_\_\_\_\_ and row is known as \_\_\_\_\_.
2. Organized collection of data or information for accessing, updating and management is known as \_\_\_\_\_.
3. A relational database consists of a collection of \_\_\_\_\_.
4. To see all available databases in MySQL; \_\_\_\_\_ command is used.
5. Data Definition language is the language which is used to defining the \_\_\_\_\_ of relation.
6. In order to build a link between two tables, \_\_\_\_\_ is used.

7. In order to make multiple columns as a Primary Key, \_\_\_\_\_ can be used.
8. Foreign key is a field in a table that is \_\_\_\_\_ in another table.
9. A Key which uniquely identifies each row in the table is known as \_\_\_\_\_.
10. A foreign key can be \_\_\_\_\_ or \_\_\_\_\_. (null, duplicate)

### C. State whether True or False

1. DBMS is an interface between Database application and database.
2. Using the SQL statement RENAME DATABASE; a database can be renamed.
3. To see all existing databases; SHOW DATABASES; syntax is used.
4. A Primary Key is basically a Column or Columns.
5. To make a link between two tables, We can use foreign key constraints.
6. A Primary Key can be NULL
7. A Foreign Key can not be Duplicate.
8. If multiple columns are used as Primary Key, it is known as Composite Key.
9. There could be two Primary keys constraints in a single table.
10. A Foreign Key can not have NULL value

### D. Short answer questions.

1. What is file system? Write down limitations of file system.
2. Why foreign keys are allowed to have NULL values? Explain with an example.
3. What are the limitations of file system and how that are overcome by DBMS?
4. What is database schema?
5. What is data redundancy and its associated problems?
6. How data redundancy problem is solved in DBMS?
7. What is MYSQL and its features?
8. What are various data types available in MYSQL?
9. Differentiate between: (a) Database state and database schema (b) Primary key and foreign key (c) Degree and cardinality of a relation
10. Explain the terms (a) Relation (b) Domain (c) Tuple (d) Attribute (e) Degree (f) Cardinality (g) Primary Key (h) Foreign Key
11. Describe the various integrity constraints?

### E. Practical Exercises

1. Considering the following three tables Student, Teacher and Subject, answer the following questions.

**Table Name: Student**

Field Name	Description
Reg_No	Student Register Number
First_Name	Name of the Student
Sur_Name	Surname of the Student
Address	Address of the Student
City	City of Student
Pincode	Pincode of city
Birthdate	Date of Birth
Gender	Male or Female
Standard	Studying in which standard
Join_Date	Date of Joining School
Leaving_Date	Date of Leaving School

**Table Name: Teacher**

Field Name	Description
------------	-------------

Teacher_No	Teacher Number
First_Name	Name of the Teacher
Sur_Name	Surname of the Teacher
Address	Address of the Teacher
City	City of teacher
Pincode	Pin code of the city
Phone_no	Phone number of teacher
Email_id	E-mail id of teacher
Mobile_No	Mobile number of teacher

**Table Name: Subject**

Field Name	Description
Sub_Name	Name of the Subject
Details	Description of the subject

- Write data type for each field in each table.
- Write Primary Key and other key constraints if any in each table.
- Whether it is possible to relate any two tables. If yes, justify your answer.
- What is the degree of each relation.

2. The medical shop wants to maintain a database “Medicos” to keep track of all medicines in the shop. Design a database by answering the following questions.

- Create a relation “medicine” to the medicine details such as medicine name, company, price, batch no, mfd date, qty available and expiry date.
- Assign the appropriate attribute names with their data type.
- Store the medicine and its price at once.
- Assign the constraints if required.

3. Canteen Store Department wants to create a database CSD\_Customer to maintain details of all working as well as their dependent family member details.

WORKING\_EMPLOYEE(Emp\_No, Emp\_Name, Address, Aadhar\_Number, Dept, DOJ)

DEPENDENT (SNo, Dep\_Name, Relationship, Emp\_No, Valid\_Date)

- Name the attributes of WORKING\_EMPLOYEE, which can be used as primary key and candidate keys.
- The CSD wants to retrieve details of dependent of any specific employee. Name the tables and the key which are required to retrieve this information.
- What is the degree and cardinality of WORKING\_EMPLOYEE and DEPENDENT relation?

4. Considering the following three tables Student, Project\_Assigned Teacher and Project, answer the following questions.

**Table: STUDENT**

Roll_No	Name	Class	Section	Regi_ID
11	Anshika	XII	B	CS-101-10
12	Hiba	XII	A	CS-103-14
21	Kushaal	XI	B	IP-104-15
22	Manmeet	XII	B	CS-101-14
23	Vibhanshu	XI	A	IP-101-15

**Table: PROJECT\_ASSIGNED**

Regi_ID	Project_No
IP-101-15	101
IP-104-15	102
CS-103-14	103
CS-101-14	104
CS-101-10	105

**Table: PROJECT**

Proj_No	Project_Name	Sub_Date
101	Airline Reservation System	12-01-22
102	Library Automation System	12-01-22
103	Employee Management System	15-01-22
104	Student Management System	12-01-22
105	Inventory Management System	15-01-22
106	Railway Reservation System	15-01-22

Answer the following questions:

- Write the name of primary key of each table.
- Write the name of foreign key(s) in table PROJECT\_ASSIGNED.
- Is there any alternate key in table STUDENT? Give justification for your answer.
- Can a user assign duplicate value to the field Roll\_No of STUDENT table? Justify.

5. Consider the database STUDENT\_PROJECT given above and answer the following questions with justification.

- Can you insert a new student record with missing roll number.
- Can you insert a new student record with missing registration id value.
- Can you insert a new project detail without Sub\_date.
- Can you insert a new project detail without Proj\_no.
- Can you insert a new record with Regi\_ID as IP-101-19 and Project\_No 206 in table PROJECT\_ASSIGNED.

## Session 2: Structured Query Language (SQL)

Once the result date is declared, Shyam was eager to see the result on website. (Figure 2.1) He opened the website to enter his Roll number to see the result. After entering Roll number, he pressed the OK button. Immediately score card of Shyam got displayed on the screen and passed with first division marks. Shyam was very happy and also surprised, how a computer searches the Roll number so fast among approximately 5 lacs students records. Later on, Shyam understand that it was possible because of the database query language which is also known as Structured Query Language (SQL). SQL is used to search, store, modify records in data base management system. In this chapter, you will understand to create database objects, insert data in database and various types of commands used to retrieve the required data from the database.



**Fig. 2.1 Checking result online**

## 2.1 Structured Query Language (SQL)

In file system it is required to write programs to access data. However in DBMS there exists a Structured Query Language (SQL), is a special kind of query language used to access and manipulate data from the database. SQL is the most popular query language used by major relational database management systems (RDBMS), such as MySQL, Oracle, Informix, PostgreSQL, SQL server, MS Access, and Sybase.

SQL is easy to learn as the statements comprise of descriptive English words. It is possible to interact with a database using SQL very easily. It is simply required to specify what is to be retrieved rather than how to retrieve data from the database. SQL provides statements for defining the structure of data, manipulating data in the database, declaring constraints and retrieving data from the database in various ways, depending on requirement.

### 2.1.1 Installing MySQL

MySQL is an open source RDBMS software which can be easily downloaded from its official website <https://dev.mysql.com/downloads>. After installing MySQL, start MySQL service. The appearance of mysql> prompt as shown below. MySQL is ready to accept SQL statements on this prompt. (Figure 2.2)

```

dds@dds-HP-240-G7-Notebook-PC: ~
dds@dds-HP-240-G7-Notebook-PC:~$ sudo mysql
[sudo] password for dds:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.27-0ubuntu0.21.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement
.
mysql> █

```

**Fig. 2.2: MySQL Shell**

Following are some important points to be kept in mind while using SQL.

- SQL is not case insensitive. For example, the column names 'salary' and 'SALARY' are the same for SQL.
- SQL statements terminates with a semicolon (;). In multi-line SQL statements, the ";" is not required after the first line. Just press the Enter key to continue on the next line. The prompt mysql> then changes to "->", indicating that statement is continued to the next line. Only at the end of SQL statement, put ";" and press Enter.

## 2.2 Data Types and Constraints in MySQL

We know that a database consists of one or more relations and each relation (table) is made up of attributes (column). Each attribute has a data type. It is also possible to specify constraints for each attribute of a relation.

### 2.2.1 Data type of Attribute

Data type of an attribute indicates the type of data value that an attribute can have. It also decides the operations that can be performed on the data of that attribute. For example, arithmetic operations can be performed on numeric data but not on character data. Commonly used data types in MySQL are numeric types, date and time types, and string types as shown in Table 2.1.

**Table 2.1 Commonly used data types in MySQL**

Data Type	Description
<b>CHAR (n)</b>	Specifies character type data of length n where n could be any value from 0 to 255. CHAR is of fixed length, means, declaring CHAR (10) implies to reserve spaces for 10 characters. If data does not have 10 characters (for example, 'city' has four characters), MySQL fills the remaining 6 characters with spaces padded on the right.
<b>VARCHAR (n)</b>	Specifies character type data of length 'n' where n could be any value from 0 to 65535. But unlike CHAR, VARCHAR is a variable-length data type. That is, declaring VARCHAR (30) means a maximum of 30 characters can be stored but the actual allocated bytes will depend on the length of entered string. So 'city' in VARCHAR (30) will occupy the space needed to store 4 characters only.
<b>INT</b>	INT specifies an integer value. Each INT value occupies 4 bytes of storage. The range of values allowed in integer type are -2147483648 to 2147483647. For values larger than that, we have to use BIGINT, which occupies 8 bytes.
<b>FLOAT</b>	Holds numbers with decimal points. Each FLOAT value occupies 4 bytes.
<b>DATE</b>	The DATE type is used for dates in 'YYYY-MM-DD' format. YYYY is the 4 digits year, MM is the 2 digits month and DD is the 2 digits date. The supported range is '1000-01-01' to '9999-12-31'.

### 2.2.2 Constraints

Constraints are the certain types of restrictions on the data values that an attribute can have. Table 2.2 lists some of the commonly used constraints in SQL. They are used to ensure correctness of data. However, it is not mandatory to define constraints for each attribute of a table.

**Table 2.2 Commonly used SQL Constraints**

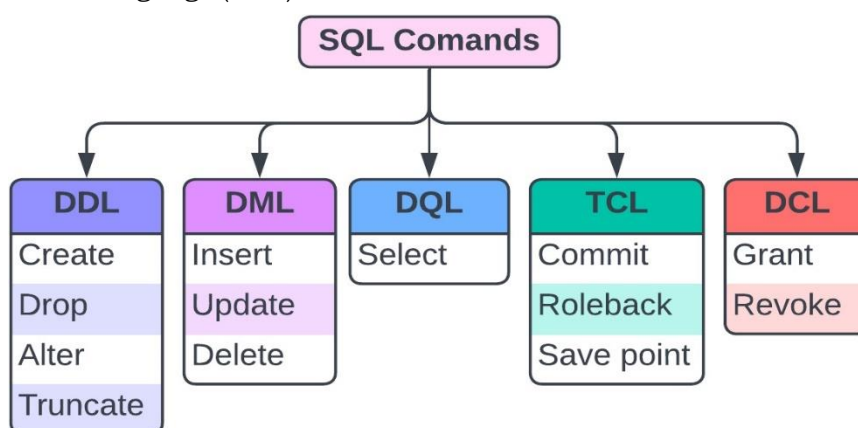
Constraint	Description
NOT NULL	Ensures that a column cannot have NULL values where NULL means missing/unknown/not applicable value.
UNIQUE	Ensures that all the values in a column are distinct/unique.
DEFAULT	A default value specified for the column if no value is provided.
PRIMARY KEY	The column which can uniquely identify each row or record in a table.

FOREIGN KEY	The column which refers to value of an attribute defined as primary key in another table.
-------------	---

### 2.2.3 Types of Structured Query Language (SQL)

SQL is a standardized language used for making communication with relational databases and performing various operations on it. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems. SQL statements are used to perform tasks such as insert, update delete data in any database. On the basis of different types of operation, SQL commands are divided into five categories as shown in Figure 2.3.

1. Data Definition Language (DDL)
2. Data Manipulation Language (DML)
3. Data Query Language (DQL)
4. Transaction Control Language (TCL)
5. Data Control Language (DCL)



**Fig. 2.3: Types of SQL command**

### 2.3 SQL – Data Definition Language (DDL)

It is first necessary to define the relation schema to store data in database. Defining a schema includes creating a relation and giving name to a relation, identifying the attributes in a relation, deciding upon the datatype for each attribute and also specify the constraints as per the requirements. Sometimes, it may require to make changes to the relation schema also. SQL provides commands for defining the relation schema, modifying relation schema and deleting relations. These are called as Data Definition Language (DDL).

As you know that the data are stored in relations or tables in a database. Database is a collection of database object such as tables, queries and views. The CREATE statement is used to create a database and its tables (relations). Before creating a database, it should be clear about the number of tables in the database, the columns (attributes) in each table along with the data type of each column. This is how we decide the relation schema. This category of SQL provides a set of commands to create the database structure or schema.

#### 2.3.1 CREATE Database

This SQL command is used to create various database objects. The syntax and example for creating database is given below.

**Syntax:**

```
CREATE DATABASE databasename;
```

*Example 2.1:* The following command is used to create a database with the name “SchoolRecord”.

```
mysql> CREATE DATABASE SchoolRecord;
```



After successful execution of the command a message “Query OK” is displayed on the sql prompt. It is also possible to see the newly created database by using the “show” command. The show command displays the newly created database along with some default databases of MySQL as shown in Figure 2.4.

```
Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create database SchoolRecord;
Query OK, 1 row affected (0.23 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| SchoolRecord |
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
5 rows in set (0.00 sec)

mysql> █
```

**Note:** In any RDBMS, it is possible to manage multiple databases on a single computer. USE command is used to select the specific database. After selecting the database, it is possible to create tables or querying data from this database.

To select the database *SchoolRecord*, issue the “USE” command followed by database name.

```
mysql> use SchoolRecord;
Database changed
mysql> █
```

**Note/Tip:** In LINUX OS environment, names for database and tables are case-sensitive whereas in WINDOWS OS, there is no such differentiation. However, as a good practice, it is suggested to write database or table name in the same letter cases that were used at the time of their creation.

### 2.3.2 CREATE Table

After creating database *SchoolRecord*, it is required to define relations (create tables) in this database. In each relation specify attribute (column name) for each attribute with their required data types. The syntax for CREATE TABLE statement is as follows.

*Syntax:*

```
CREATE TABLE tablename (
    Col_name1 datatype constraint,
    Col_name2 datatype constraint,
    :
    Col_nameN datatype constraint );
```

Let us understand how to choose attribute names and their respected data types. First identify data types of the attributes in table “*StudentRecord*” along with their constraint, if any. Let us assume that there are total 100 students in a class and values of Roll number are in a sequence from 1 to 100. Since the data values of attribute “*Stu\_RollNo*” is stored in digits, the data type integer (INT) is appropriate for this attribute. In the same way total number of characters in student First name and Last name can be upto 20 characters. Since the number of characters can vary for different students, the data type VARCHAR is used for these columns. In the same

the data type VARCHAR is used for student address upto 50 characters in length. The specific data type DATE is used for specifying any type of date. So DATE data type is used for attribute “Date of Birth”. For student's parent id, Aadhaar number is used which is a 12 digit number. Since Aadhaar number is of fixed length and it is not required to perform any mathematical operation, the character data type with fixed length of 12 character, CHAR (12) is used for this attribute.

**Table 2.3 Data types and constraints for the attributes of relation StudentRecord**

Attribute	Data expected to be stored	Data type	Constraint
Stu_RollNo	Numeric value consisting of maximum 3 digits	Int	Primary Key
Stu_FName	Variable length string of maximum 20 characters	Varchar (20)	Not Null
Stu_LName	Variable length string of maximum 20 characters	Varchar (20)	Not Null
Stu_DOB	Date value	Date	Not Null
Stu_Address	Variable length string of maximum 50 characters	Varchar (50)	Not Null
Par_ID	Fixed length string of 12 digits for Aadhaar Number	Char (12)	Foreign Key

**Table 2.4 Data types and constraints for the attributes of relation ParentRecord**

Attribute	Data expected to be stored	Data type	Constraint
Par_ID	Fixed length string of 12 digits Aadhaar number	Char (12)	Primary Key
Par_Name	Variable length string of maximum 20 characters	Varchar (20)	Not Null
Par_Phone	Numeric value consisting of 10 digits	Char (10)	Null Unique
Par_Address	Variable length string of size 30 characters	Varchar (30)	Not Null
Par_Email	Variable length string of size 30 characters	Varchar (30)	

**Table 2.5 Data types and constraints for the attributes of relation AttendanceRecord**

Attribute	Data expected to be stored	Data type	Constraint
Att_Date	Date value	Date	Primary Key*
Stu_RollNo	Numeric value consisting of maximum 3 digits	Int	Primary Key* Foreign Key
Att_Status	‘P’ for present and ‘A’ for absent	Char(1)	Not Null

Table 2.3, 2.4 and 2.5 show the chosen data type and constraint for each attribute of the relations *StudentRecord*, *ParentRecord* and *AttendanceRecord* respectively.

**Example 2.2:** The following command is used to create table **StudentRecord**. To create the table in SchoolRecord database, first open the database with USE SchoolRecord command. Then create the table under StudentRecord database by using the CREATE TABLE command.

```
mysql> USE SchoolRecord;
Database changed
mysql> CREATE TABLE StudentRecord (
-> Stu_RollNo INT,
-> Stu_FName VARCHAR(20),
-> Stu_LName VARCHAR(20),
-> Stu_DOB DATE,
-> Stu_Address VARCHAR(50),
-> Par_ID CHAR(12),
-> PRIMARY KEY (Stu_RollNo) );
Query OK, 0 rows affected (3.17 sec)

mysql> █
```

**Note:** “,” is used to separate two attributes and each statement terminates with a semi-colon (;). The arrow (->) is an interactive continuation prompt. If we enter an unfinished statement, the SQL shell will wait for us to enter the rest of the statement.

**Example 2.3:** The following command is used to Create table **ParentRecord**.

```
mysql> CREATE TABLE ParentRecord (
-> Par_ID CHAR(12),
-> Par_Name VARCHAR(20),
-> Par_Phone CHAR(10),
-> Par_Address VARCHAR(50),
-> Par_Email VARCHAR(30) );
Query OK, 0 rows affected (2.57 sec)

mysql> █
```

**Example 2.4:** The following command is used to Create table **AttendanceRecord**.

```
mysql> CREATE TABLE AttendanceRecord (
-> Att_Date DATE,
-> Stu_RollNo INT,
-> Att_Status CHAR(1) );
Query OK, 0 rows affected (2.81 sec)

mysql>
```

### 2.3.3 DESCRIBE Table

DESCRIBE or DESC command is used to view the structure of an already created table.

*Syntax:* DESCRIBE tablename;

**Example 2.5:** The following SQL command is used to show the structure of *StudentRecord* table.

```
mysql> DESCRIBE StudentRecord;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Stu_RollNo | int           | NO   | PRI | NULL    |       |
| Stu_FName  | varchar(20)   | YES  |     | NULL    |       |
| Stu_LName  | varchar(20)   | YES  |     | NULL    |       |
| Stu_DOB    | date          | YES  |     | NULL    |       |
| Stu_Address | varchar(50)   | YES  |     | NULL    |       |
| Par_ID     | char(12)      | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.01 sec)

mysql>
```

The SHOW TABLES statement is used to display all the table in database. We have created three tables in the database SchoolRecord.

**Example 2.6:** The following SQL command is used to display the tables created in the database SchoolRecord. It shows all the three tables created so far.

```
mysql> show tables;
+-----+
| Tables_in_SchoolRecord |
+-----+
| AttendanceRecord      |
| ParentRecord          |
| StudentRecord         |
+-----+
3 rows in set (0.01 sec)
mysql>
```

### 2.3.4 ALTER Table

After creating a table, it is possible to add or remove an attribute or modify the datatype of existing attribute or to add constraint in attribute. ALTER statement is used to change or alter the structure of the table.

*Syntax:* ALTER TABLE tablename ADD/Modify/DROP attribute1, attribute2,..

#### (a) Add primary key to a relation

**Example 2.7:** The following SQL command is used to add a primary key to the relation "ParentRecord"

```
mysql> ALTER TABLE ParentRecord ADD PRIMARY KEY (Par_ID);
Query OK, 0 rows affected (3.91 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql>
```

A composite primary key is made up of two attributes. The primary key to the "AttendanceRecord" relation will be composite primary key of two attributes. "AttendanceDate" and "Stu\_RollNo".

**Example 2.8:** The following SQL command is used to add the composite primary key to the relation "AttendanceRecord".

```
mysql> ALTER TABLE AttendanceRecord ADD PRIMARY KEY (
-> Att_Date, Stu_RollNo);
Query OK, 0 rows affected (4.34 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql>
```

#### (b) Add foreign key to a relation

It is also possible to add foreign keys to the relation, if any. A relation may have multiple foreign keys and each foreign key is defined on a single attribute. Note the following points while adding foreign key to a relation.

- The referenced relation must be already created.
- The referenced attribute must be a part of primary key of the referenced relation.
- Data types and size of referenced and referencing attributes must be same.

#### **Syntax:**

```
ALTER TABLE table_name ADD FOREIGN KEY (attribute name)
REFERENCES referenced_table_name (attribute name);
```

Let us now add foreign key to the table StudentRecord.

In table "StudentRecord", the attribute Par\_ID (the referencing attribute) is a foreign key and it refers to attribute Par\_ID (the referenced attribute) of table "ParentRecord". Hence, "StudentRecord" is the referencing table and "ParentRecord" is the referenced table.

**Example 2.9:** The following SQL command is used to add the foreign key. The ALTER statement change the table StudentRecord.

```
mysql> ALTER TABLE StudentRecord ADD FOREIGN KEY(Par_ID)
-> REFERENCES ParentRecord(Par_ID);
Query OK, 0 rows affected (5.16 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql>
```

### (c) Add constraint unique to an existing attribute

In “ParentRecord” table, attribute “Par\_Phone” has a constraint **UNIQUE**, means no two values in that column should be same.

#### Syntax:

```
ALTER TABLE table_name ADD UNIQUE (attributename);
```

**Example 2.10:** The following SQL command is used to add the constraint UNIQUE with attribute “Par\_Phone” of the table “ParentRecord”.

```
mysql> ALTER TABLE ParentRecord ADD UNIQUE(Par_Phone);
Query OK, 0 rows affected (1.56 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql>
```

### (d) Add an attribute to an existing table

Sometimes, it is required to add an additional attribute in a table. The syntax for this is.

*Syntax:* ALTER TABLE table\_name ADD attribute\_name DATATYPE;

Suppose the Principal of the school has decided to award scholarship to some needy students for which income of the parents must be known. But school has not maintained income attribute with table “ParentRecord” so far.

**Example 2.11:** The following command is used to add a new attribute income of data type INT in the table “ParentRecord”.

```
mysql> ALTER TABLE ParentRecord ADD Par_Income INT;
Query OK, 0 rows affected (1.38 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql>
```

The newly added attribute “income” with data type INT in the table “ParentRecord” can be viewed using DESC command as follows.

```
mysql> desc ParentRecord;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Par_ID     | char(12)  | NO   | PRI | NULL    |      |
| Par_Name   | varchar(20)| YES  |     | NULL    |      |
| Par_Phone  | char(10)  | YES  | UNI | NULL    |      |
| Par_Address| varchar(60)| YES  |     | NULL    |      |
| Par_Email  | varchar(30)| YES  |     | NULL    |      |
| Par_Income | int       | YES  |     | NULL    |      |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)

mysql>
```

### (e) Modify datatype of an attribute

It is possible to modify the data types of the existing attributes of a table using the following statement.

*Syntax:*

```
ALTER TABLE table_name MODIFY attribute DATATYPE;
```

Suppose, to change the size of attribute “Par\_Address” from VARCHAR (30) to VARCHAR (40) of the “ParentRecord” table.

**Example 2.12:** The following command is used to change the size of attribute “Par\_Address” in “ParentRecord” table.

```
mysql> ALTER TABLE ParentRecord MODIFY Par_Address VARCHAR(60);
Query OK, 0 rows affected (0.51 sec)
Records: 0 Duplicates: 0 Warnings: 0
mysql>
```

#### **(f) Modify constraint of an attribute**

When creating a table, by default each attribute takes null value except for the attribute defined as primary key. It is possible to change an attribute’s constraint from NULL to NOT NULL using ALTER statement.

*Syntax:*

```
ALTER TABLE table_name MODIFY attribute DATATYPE NOT NULL;
```

**Note:** It is required to specify the data type of the attribute along with constraint NOT NULL while using MODIFY.

**Example 2.13:** The following command is used to associate NOT NULL constraint with attribute “Stu\_FName” of table “StudentRecord”.

```
mysql> ALTER TABLE StudentRecord MODIFY Stu_FName
-> VARCHAR(20) NOT NULL;
Query OK, 0 rows affected (4.23 sec)
Records: 0 Duplicates: 0 Warnings: 0
mysql>
```

#### **(g) Add default value to an attribute**

The syntax to specify the default value for an attribute is,

*Syntax:*

```
ALTER TABLE table_name MODIFY attribute
DATATYPE DEFAULT default_value;
```

To set default value of “Stu\_DOB” of “StudentRecord” to 15th May 2000, write the following statement.

```
mysql> Alter Table StudentRecord MODIFY Stu_DOB DATE
-> DEFAULT '2000-05-15';
Query OK, 0 rows affected (0.61 sec)
Records: 0 Duplicates: 0 Warnings: 0
mysql>
```

**Note:** It is required to specify the data type of the attribute along with DEFAULT while using MODIFY.

#### **(h) Remove an attribute**

It is possible to remove attributes from a table using ALTER.

*Syntax:*

```
ALTER TABLE table_name DROP attribute;
```

**Example 2.14:** The following command is used to remove the attribute income from the table “ParentRecord”.

```
mysql> ALTER TABLE ParentRecord DROP Par_Income;
Query OK, 0 rows affected (3.91 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql>
```

#### (i) Remove primary key from the table

Sometimes it may be required to remove the primary key constraint from the table. In such case, the syntax for ALTER TABLE command is.

*Syntax:*

```
ALTER TABLE table_name DROP PRIMARY KEY;
```

**Example 2.15:** The following command is used to remove primary key of table “ParentRecord”

```
mysql> ALTER TABLE StudentRecord DROP PRIMARY KEY;
Query OK, 0 rows affected (5.61 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql>
```

**Note:** The primary key is dropped from StudentRecord table, but each table should have a primary key to maintain uniqueness. Hence, to use ADD command to specify primary key for the StudentRecord table as shown in earlier examples

### 2.3.5 DROP TABLE Command

Sometimes it may require to remove a table in a database or the database itself. DROP statement is used to remove a database or a table permanently from the system. Since this command will delete the table or database permanently, you have to be cautious while using this statement as it cannot be undone. Let us assume that you have created a table with name “ParantRecord” instead of “ParentRecord”. DROP command can be used to delete the table created with wrong name.

*Syntax:*

```
DROP TABLE table_name;
```

It is also possible to drop the entire database.

*Syntax:*

```
DROP DATABASE database_name;
```

**Example 2.16:** The following command is used to delete the table name “ParantRecord” from the current database.

#### **Cautions:**

- Using the Drop statement to remove a database will ultimately remove all the tables within it.
- DROP statement will remove the tables or database created by you. Hence you may apply DROP statement at the end of the chapter.

### 2.3.6 TRUNCATE TABLE Command

It is possible to remove all records form a table using TRUNCATE command. Later on, you can insert new records in the same table. This command will delete all records from the table but table structure will exist in database. While using DROP command, all the records with table structure will be deleted from the database. So care should be taken while using both TRUNCATE and DROP command in SQL.

*Syntax:*

Truncate Table Table\_Name;

**Example 2.17:** The following command is used to Truncate the table “StudentRecordBackup”.

```
mysql> Select * from STUDENTBACKUP;
+-----+-----+-----+-----+-----+
| Stu_RollNo | Stu_FName | Stu_LName | Stu_DOB   | Stu_Address |
+-----+-----+-----+-----+-----+
|          1 | Rajvardhan | Singh     | 2003-05-15 | 203, Khandari, Agra UP |
|          2 | Trilok     | Sharma    | 2004-08-15 | 144 Gr Kailash, New Delhi |
|          3 | Aditi      | Gaur      | 2005-06-04 | JP Greens, Noida, UP |
|          4 | Anshika    | Agrawal   | 2003-05-17 | Kanda, Bagheshwar, UK |
|          6 | Pawani     | Dixit     | 2004-12-11 | Lajpat Nagar, Mathura, UP |
|          7 | Hiba       | Rizwan    | 2006-12-03 | Deep Nagar, Sahrssa, Bihar |
|          8 | Riddhi     | Gupta     | 2005-11-01 | TNagar, Hyderabad, Telangana |
|         10 | John       | DeSousa   | 2005-08-17 | Guindy, Chennai, TN |
+-----+-----+-----+-----+-----+
8 rows in set (0.00 sec)

mysql> TRUNCATE STUDENTBACKUP;
Query OK, 0 rows affected (3.49 sec)

mysql> Select * from STUDENTBACKUP;
Empty set (0.01 sec)

mysql>
```

### 2.3.7 CREATE TABLE From Existing Table

If you want to create a new table from existing table with partial or additional fields, then you can use the CREATE table command with SELECT statement. The new table is created with the result of SELECT statement with results provided by it.

*Syntax:*

```
mysql> Create table NewTableName AS
      (Select Field 1, Field 2, Field 3, ...Field N from Old_Table_Name)
```

**Example 2.18:** The following command is used to Create table “NewStudentRecord” from the existing table “StudentRecord”.

```
mysql> Create table NewStudentRecord AS (Select Stu_RollNo, Stu_FName, Stu_LName,
Stu_DOB, Stu_Address from StudentRecord);
Query OK, 9 rows affected (2.40 sec)
Records: 9 Duplicates: 0 Warnings: 0

mysql>
```

It will create a new table named as “NewStudentRecord” with only 5 attributes and all the records which are inserted in this table earlier.

```
mysql> Show tables;
+-----+
| Tables_in_SchoolRecord |
+-----+
| AttendanceRecord       |
| NewStudentRecord       |
| ParentRecord           |
| StudentRecord          |
+-----+
4 rows in set (0.00 sec)

mysql>
```

It is possible to create a new table with all attributes and all records available in the existing table.



**Example 2.19:** The following command is used to create a new table “*StudentRecord1*” with all attributes and all records available in the existing table “*StudentRecord*”.

```
mysql> CREATE TABLE StudentRecord1 AS(Select * from StudentRecord);
Query OK, 9 rows affected (3.39 sec)
Records: 9 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM StudentRecord1;
+-----+-----+-----+-----+-----+-----+
| Stu_RollNo | Stu_FName | Stu_LName | Stu_DOB | Stu_Address | Par_ID |
+-----+-----+-----+-----+-----+-----+
| 1 | Rajvardhan | Singh | 2003-05-15 | 203, Khandari, Agra UP | 452695874564 |
| 2 | Trilok | Sharma | 2004-08-15 | 144 Gr Kailash, New Delhi | 252154687451 |
| 3 | Aditi | Gaur | 2005-06-04 | JP Greens, Noida, UP | 362115264625 |
| 4 | Anshika | Agrawal | 2003-05-17 | Kanda, Bagheshwar, UK | 602125125261 |
| 6 | Pawani | Dixit | 2004-12-11 | Lajpat Nagar, Mathura, UP | 268953264578 |
| 7 | Hiba | Rizwan | 2006-12-03 | Deep Nagar, Sahrssa, Bihar | 485466192343 |
| 8 | Riddhi | Gupta | 2005-11-01 | TNagar, Hyderabad, Telangana | 521556651761 |
| 10 | John | DeSousa | 2005-08-17 | Guindy, Chennai, TN | 954891122475 |
| 5 | Nandini | Roy | 2003-12-19 | Fortune Somya, Bhopal, MP | 225423344657 |
+-----+-----+-----+-----+-----+-----+
9 rows in set (0.00 sec)

mysql>
```

### 2.3.8 RENAME TABLE command

Sometime it may be required to change the name of existing table. It is possible to do so by using RENAME or ALTER command.

*Syntax:*

```
RENAME TABLE old_table_name TO new_table_name;
```

**Example 2.20:** The following command is used to rename the table “*NewStudentRecord*” to “*StudentRecord1*”

```
mysql> RENAME TABLE NewStudentRecord TO StudentRecord1;
```

It is possible to rename multiple tables using single command as under.

*Syntax:*

```
RENAME TABLE Old_tableA TO New_tableA, Old_tableB TO New_tableB,
Old_tableC TO New_tableC;
```

It is also possible to use ALTER command to rename the table as given below.

*Syntax:*

```
ALTER TABLE Old_table_name RENAME TO new_table_name;
```

**Example 2.21:** The following is the command to alter the table “*StudentRecord1*” to “*StudentRecord2*”

```
mysql> ALTER TABLE StudentRecord1 RENAME TO StudentRecord2;
```

### 2.3.9 CREATE VIEW command

Like table, view is another database objects. It is a special kind of virtual table. It does not hold its own data. A view can has rows and columns just like in table. It is possible to create a view using CREATE VIEW command, by selecting fields from one or more tables present in the database. A View can either have all the rows of a table (s) or specific rows based on certain criteria. The syntax to create a view is as under.

*Syntax:*

```
CREATE VIEW view_name AS
SELECT column1, column2 ..... columnN
FROM table_name WHERE condition;
```

**Example 2.22:** The following is the command to create a view from single table.

```
mysql> CREATE VIEW EMP_VIEW AS Select * FROM emp where SAL>10000;
Query OK, 0 rows affected (0.54 sec)

mysql>
```

Now the view named EMP\_VIEW will be created with only those employee records who have salary more than 10000. You can use this view similar to Employee table to see all records using SELECT command. To see all records from EMP\_VIEW, use the SELECT command as under.

```
mysql> CREATE VIEW EMP_VIEW AS Select * FROM emp where SAL>10000;
Query OK, 0 rows affected (0.54 sec)

mysql> Select * from EMP_VIEW;
+-----+-----+-----+-----+-----+-----+-----+-----+
| empno | ename      | job      | mgr  | hiredate | sal    | comm  | deptno |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7216  | Jawahar   | Manager  | 7489 | 1995-03-30 | 10975 | NULL  | 20     |
| 7348  | Balwinder | Manager  | 7489 | 1995-04-28 | 10850 | NULL  | 30     |
| 7432  | Chetana   | Manager  | 7489 | 1995-06-06 | 10450 | NULL  | 10     |
| 7438  | Sachin    | Analyst  | 7216 | 1996-12-05 | 11000 | NULL  | 20     |
| 7489  | Kushaal   | President | NULL | 1995-11-14 | 13000 | NULL  | 10     |
| 7552  | Farhan    | Analyst  | 7216 | 1995-10-27 | 11000 | NULL  | 20     |
+-----+-----+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)

mysql>
```

## Activity 1

**Practical Activity 2.1 – Create the table “Employee” and “Department” in MySQL with the following attributes specification.**

### Employee Table

Attribute	Data expected to be stored	Data type	Constraint
empno	Numeric value consisting of 4 digits	Int	Primary Key
ename	Variable length string of max 30 characters	Varchar (30)	Not Null
job	Variable length string of max 15 characters	Varchar (15)	Not Null
mgr	Numeric value consisting of 4 digits	Int	Not Null
hiredate	Date of joining the company	Date	Not Null
sal	Numeric value consisting of 6 digits	Int	Not Null
comm	Numeric value consisting of 4 digits	Int	Not Null
Deptno	Dept no which is Numeric type consisting of maximum 2 digits	Int	

### Department Table

Attribute	Data expected to be stored	Data type	Constraint
deptno	Numeric value consisting of 4 digits	Int	Primary Key
dname	Variant length string of max 20 characters	Varchar (20)	Not Null
loc	Variant length string of max 25 characters	Varchar (25)	Not Null

## 2.4 SQL FOR DATA MANIPULATION LANGUAGE (DML)

In the previous section, we created the database **SchoolRecord** with three relations (or tables) i.e. *StudentRecord*, *ParentRecord* and *AttendanceRecord*. Creating a table, creates its structure only. It is required to manipulate the data in the table by entering, deleting and updating the data records. The commands or statements used to insert, delete and update the records come under SQL Data Manipulation Language (DML).

Data Manipulation means either retrieval (access) of existing data, insertion of new data, removal of existing data or modification of existing data in the database. Updation and deletion of data records are also important in SQL. These data manipulation methods are discussed in the following section.

### 2.4.1 INSERTION of Records

INSERT INTO statement is used to insert new records in any table or relation.

*Syntax:*

```
INSERT INTO tablename VALUES (value 1, value 2,...);
```

Here, value 1 corresponds to attribute 1, value 2 corresponds to attribute 2 and so on. It is required to specify attribute names in INSERT statement if there are exactly same number of values in the INSERT statement as the total number of attributes in the table.

**Caution:** While populating records in a table with foreign key, ensure that records in referenced tables are already populated.

Let us insert some records in the **SchoolRecord** database. First insert the records in *ParentRecord* table first as it does not have any foreign key. A set of sample records for *ParentRecord* table is shown in Table 2.6.

**Table 2.6 Records to be inserted into the ParentRecord Table**

Par_ID	Par_Name	Par_Phone	Par_Address	Par_Email
452695874564	Manu P Singh	9834567890	203, Khandari, Agra, UP	mpsingh@gmail.com
252154687451	Ashok K Sharma	9845678910	144 Gr Kailash, New Delhi	aksharmaji@mail.com
362115264625	Ashutosh Gaur	9856789120	JP Greens, Noida, UP	ashutoshgaur@gmail.com
602125125261	Sachin Agrawal	9812389120	Kanda, Bagheshwar, UK	sachinag@gmail.com
225423344657	Chandra Roy	9891201238	Fortune Somya, Bhopal, MP	Ch.roy@rediff.com
268953264578	Dinesh Dixit		Lajpat Nagar, Mathura, UP	dinesh.dixit@hp.com
485466192343	Rizwan Alam	9255614563	Deep Nagar, Sahrsa, Bihar	riz.alam@gmail.com
521556651761	Ashish Gupta	8544556978	T Nagar, Hyderabad, Telangana	ashish.gupta@hotmail.com
686113652987	Gurmeet Singh	9635214789	Shahid Nagar, Amritsar, PB	gurmeet.007@ymail.com
954891122475	Michal DeSousa	8554658958	Guindy, Chennai, TN	michal.don@gmail.com

**Example 2.23:** The following command is used to insert the record in the “*ParentRecord*” table.

```
mysql> INSERT INTO ParentRecord VALUES (45269587456, 'Manu P Singh',
-> 9834567890, '203, Khandari, Agra, UP', 'mpsingh@gmail.com');
Query OK, 1 row affected (0.25 sec)

mysql>
```

We can use the SQL statement “***SELECT \* from table\_name;***” to view the inserted record after any statement to see the current changes in table.

```
mysql> select * from ParentRecord;
+-----+-----+-----+-----+-----+
| Par_ID      | Par_Name    | Par_Phone  | Par_Address          | Par_Email      |
+-----+-----+-----+-----+-----+
| 45269587456 | Manu P Singh | 9834567890 | 203, Khandari, Agra, UP | mpsingh@gmail.com |
+-----+-----+-----+-----+-----+
1 row in set (0.01 sec)

mysql>
```

It is also possible to provide values only for some of the attributes in a table by just specifying the attribute name alongside each data value as per the following syntax.

*Syntax:*

```
INSERT INTO tablename (column1, column2, ...)
VALUES (value1, value2, ...);
```

Suppose to insert the sixth record in “*ParentRecord*” table (Table 2.6) keeping the value of “*Par\_Phone*” to NULL. Then it is required to insert the values for other four fields. In this case, specify the names of attributes in which the values are to be inserted. The values must be given in the same order in which attributes are written in INSERT command.

**Example 2.24:** The following command is used to insert the record in “*ParentRecord*” table by specifying the field name and corresponding values.

```
mysql> INSERT INTO ParentRecord(Par_ID, Par_Name, Par_Address, Par_Email)
-> VALUES(268953264578, 'Dinesh Dixit', 'Lajpat Nagar, Mathura, UP',
-> 'dinesh.dixit@hp.com');
Query OK, 1 row affected (0.25 sec)

mysql>
```

Now observe that all the four values has been inserted in the table *ParentRecord* except “*Par\_Phone*” which is being set to NULL at the time of creating a table.

```
mysql> SELECT * FROM ParentRecord;
+-----+-----+-----+-----+-----+
| Par_ID      | Par_Name    | Par_Phone  | Par_Address          | Par_Email      |
+-----+-----+-----+-----+-----+
| 268953264578 | Dinesh Dixit | NULL       | Lajpat Nagar, Mathura, UP | dinesh.dixit@hp.com |
| 452695874564 | Manu P Singh | 9834567890 | 203, Khandari, Agra, UP | mpsingh@gmail.com |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

**Note:** Text and date values must be enclosed in ‘ ’ (single quotes).

## Activities

**Practical Activity 2.2** – Insert the records in the *ParentRecord* table using INSERT command and check the records inserted in *ParentRecord* as below.

```
mysql> Select * from ParentRecord;
+-----+-----+-----+-----+-----+
| Par_ID      | Par_Name    | Par_Phone  | Par_Address          | Par_Email      |
+-----+-----+-----+-----+-----+
| 225423344657 | Chandra Roy | 9891201238 | Fortune Somya, Bhopal, MP | Ch.roy@rediff.com |
| 252154687451 | Ashok K Sharma | 9845678910 | 144 G Kailsh, New Delhi | aksharmaji@mail.com |
| 268953264578 | Dinesh Dixit | NULL       | Lajpat Nagar, Mathura, UP | dinesh.dixit@hp.com |
| 362115264625 | Ashutosh Gaur | 9856789120 | JP Greens, Noida, UP | ashutoshgaur@gmail.com |
| 452695874564 | Manu P Singh | 9834567890 | 203, Khandari, Agra, UP | mpsingh@gmail.com |
| 485466192343 | Rizwan Alam | 9255614563 | Deep Nagar, Sahrsa, Bihar | riz.alam@gmail.com |
| 521556651761 | Ashish Gupta | 8544556978 | T Nagar, Hyderabad, Telangana | ashish.gupta@hotmail.com |
| 602125125261 | Sachin Agrawal | 9812389120 | Kanda, Bagheshwar, UK | sachinag@gmail.com |
| 686113652987 | Gurmeet Singh | 9635214789 | Shahid Nagar, Amritsar, PB | gurmeet.007@ymail.com |
| 954891122475 | Michal DeSousa | 8554658958 | Guindy, Chennai, TN | michal.don@gmail.com |
+-----+-----+-----+-----+-----+
10 rows in set (0.00 sec)

mysql>
```

**Practical Activity 2.3** – Insert the records in StudentRecord table (Table 2.7).

**Table 2.7** Records to be inserted into the StudentRecord table

Stu_RollNo	Stu_FName	Stu_LName	Stu_DOB	Stu_Address	Par_ID
1	Rajvardhan	Singh	5/15/2003	203, Khandari, Agra UP	452695874564
2	Trilok	Sharma	8/15/2004	144 Gr Kailash, New Delhi	252154687451
3	Aditi	Gaur	4/6/2005	JP Greens, Noida, UP	362115264625
4	Anshika	Agrawal	5/17/2003	Kanda, Bagheshwar, UK	602125125261
5	Nandini	Roy	12/29/2003	Fortune Somya, Bhopal, MP	225423344657
6	Pawani	Dixit	11/12/2004	Lajpat Nagar, Mathura, UP	268953264578
7	Hiba	Rizwan	12/3/2006	Deep Nagar, Sahrsa, Bihar	485466192343
8	Riddhi	Gupta	1/11/2005	T Nagar, Hyderabad, Telangana	521556651761
9	Manpreet	Singh	9/8/2005	Shahid Nagar, Amritsar, Punjab	686113652987
10	John	DeSousa	8/17/2005	Guindy, Chennai, TN	954891122475

**Example 2.25:** The following command is used to insert the first record in table “StudentRecord”.

```
mysql> INSERT INTO StudentRecord VALUES(1, 'Rajvardhan', 'Singh', '2003-05-15', '203, Khandari, Agra UP', 452695874564);
Query OK, 1 row affected (0.22 sec)

mysql> select * from StudentRecord;
+-----+-----+-----+-----+-----+-----+
| Stu_RollNo | Stu_FName | Stu_LName | Stu_DOB   | Stu_Address          | Par_ID   |
+-----+-----+-----+-----+-----+-----+
|          1 | Rajvardhan | Singh    | 2003-05-15 | 203, Khandari, Agra UP | 452695874564 |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

When column names are not mentioned in the INSERT command, then it is necessary to mention the values for all the columns. So if there is no “ParentID” for Trilok, then mention the NULL value for the “Par\_ID”.

**Example 2.26:** The following command inserts the second record with “Par\_ID” value as NULL.

```
mysql>INSERT INTO StudentRecord VALUES (2,'Trilok','Sharma','8/15/2004', '144 Gr Kailash','New Delhi' NULL);
```

```
mysql> select * from studentrecord;
+-----+-----+-----+-----+-----+-----+
| Stu_RollNo | Stu_FName | Stu_LName | Stu_DOB   | Stu_Address          | Par_ID   |
+-----+-----+-----+-----+-----+-----+
|          1 | Rajvardhan | Singh    | 2003-05-15 | 203, Khandari, Agra UP | 452695874564 |
|          2 | Trilok    | Sharma   | 2004-08-15 | 144 Gr Kailash, New Delhi | NULL      |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

**Note/Tip:** Please be careful while entering date in INSERT command. Use the ‘YYYY-MM-DD’ format to write date.

#### Practical Activity 2.4 – Use INSERT command

Insert the records in employee table using INSERT command and display it after inserting all record using SELECT statement.

```
mysql> select * from emp;
```

empno	ename	job	mgr	hiredate	sal	comm	deptno
7019	Smita	Clerk	7552	1994-12-14	8800	NULL	20
7049	Alam	Salesman	7348	1995-02-17	9600	1800	30
7171	Wasim	Salesman	7348	1995-02-19	9250	2000	30
7216	Jawahar	Manager	7489	1995-03-30	10975	NULL	20
7304	Manoj	Salesman	7348	1995-09-25	9250	2900	30
7348	Balwinder	Manager	7489	1995-04-28	10850	NULL	30
7432	Chetana	Manager	7489	1995-06-06	10450	NULL	10
7438	Sachin	Analyst	7216	1996-12-05	11000	NULL	20
7489	Kushaal	President	NULL	1995-11-14	13000	NULL	10
7494	Tarun	Salesman	7348	1995-09-05	9500	0	30
7526	Amar	Clerk	7438	1997-01-08	9100	NULL	20
7550	Jyoti	Clerk	7348	1995-11-30	8950	NULL	30
7552	Farhan	Analyst	7216	1995-10-27	11000	NULL	20
7584	Mohan	Clerk	7432	1996-01-20	9300	NULL	10
7984	Lalitha	Clerk	7432	1998-05-23	10300	NULL	10

```
15 rows in set (0.00 sec)
```

```
mysql>
```

Insert the records in Department table using INSERT command and display it after inserting all record using SELECT statement.

```
mysql> select * from dept;
```

deptno	dname	loc
10	Accounting	New Delhi
20	Research	Banglore
30	Sales	Mumbai
40	Operation	Hyderabad
50	HR	Noida

```
5 rows in set (0.00 sec)
```

```
mysql>
```

#### 2.4.2 UPDATION of Records using UPDATE and DELETE Command

UPDATE and DELETE are also the part of SQL Data Manipulation Language (DML).

UPDATE command is used to make changes in existing data value(s) of one or more columns of existing records in a table. For example, we may require some changes in address, phone number or spelling of name.

##### Syntax:

```
UPDATE table_name
SET attribute1 = value1, attribute2 = value2, ...
WHERE condition;
```

In ParentRecord table, Phone number is not available for Parent Name Dinesh Dixit. So, it is required to update Phone number of Dinesh Dixit, update the table *ParentRecord* use the command.

**Example 2.27:** The following command is used to update the Phone number of Dinesh Dixit in *ParentRecord* table.

```
mysql> UPDATE ParentRecord SET Par_Address = 'WZ-68, Azad Avenue, Boriwali, Mumbai',
-> Par_Phone = 9988776644 WHERE Par_ID = 485466192343;
Query OK, 1 row affected (0.61 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

```
mysql>
```

The updated data can be verified using the statement.

```
SELECT * FROM ParentRecord.
```

It is also possible to update values for more than one column using the UPDATE statement.

Suppose, the ParentRecord with Par\_ID 485466192343 has requested to change Address to 'WZ - 68, Azad Avenue, Boriwali, Mumbai' and Phone number to '9988776644'.

**Example 2.28:** The following SQL statement will update this record.

```
mysql> UPDATE ParentRecord SET Par_Address = 'WZ - 68, Azad Avenue, Boriwali, Mumbai',
Par_Phone = 9988776644 WHERE Par_ID = 485466192343;
```

The changes affected can be verified by using the SELECT statement as below.

```
mysql> Select * from ParentRecord;
```

Par_ID	Par_Name	Par_Phone	Par_Address	Par_Email
225423344657	Chandra Roy	9891201238	Fortune Somya, Bhopal, MP	Ch.roy@rediff.com
252154687451	Ashok K Sharma	9845678910	144 Gr Kailsh, New Delhi	aksharmaji@mail.com
268953264578	Dinesh Dixit	9956895689	Lajpat Nagar, Mathura, UP	dinesh.dixit@hp.com
362115264625	Ashutosh Gaur	9856789120	JP Greens, Noida, UP	ashutoshgaur@gmail.com
452695874564	Manu P Singh	9834567890	203, Khandari, Agra, UP	mpsingh@gmail.com
485466192343	Rizwan Alam	9988776644	WZ-68, Azad Avenue, Boriwali, Mumbai	riz.alam@gmail.com
521556651761	Ashish Gupta	8544556978	T Nagar, Hyderabad, Telangana	ashish.gupta@hotmail.com
602125125261	Sachin Agrawal	9812389120	Kanda, Bagheshwar, UK	sachinag@gmail.com
686113652987	Gurmeet Singh	9635214789	Shahid Nagar, Amritsar, PB	gurmeet.007@ymail.com
954891122475	Michal DeSousa	8554658958	Guindy, Chennai, TN	michal.don@gmail.com

```
10 rows in set (0.00 sec)

mysql>
```

### 2.4.3 DELETION of Records using DELETE

DELETE statement is used to delete or remove one or more records from a table.

*Syntax:*

```
DELETE FROM table_name WHERE condition;
```

Suppose the student with roll number 2 has left the school.

**Example 2.29:** The following SQL statement is used to delete that record from the STUDENT table.

The changes affected can be verified by using the SELECT statement as below.

```
mysql> SELECT * from StudentRecord;
```

Stu_RollNo	Stu_FName	Stu_LName	Stu_DOB	Stu_Address	Par_ID
1	Rajvardhan	Singh	2003-05-15	203, Khandari, Agra UP	452695874564
2	Trilok	Sharma	2004-08-15	144 Gr Kailash, New Delhi	252154687451
3	Aditi	Gaur	2005-06-04	JP Greens, Noida, UP	362115264625
4	Anshika	Agrawal	2003-05-17	Kanda, Bagheshwar, UK	602125125261
6	Pawani	Dixit	2004-12-11	Lajpat Nagar, Mathura, UP	268953264578
7	Hiba	Rizwan	2006-12-03	Deep Nagar, Sahrsa, Bihar	485466192343
8	Riddhi	Gupta	2005-11-01	TNagar, Hyderabad, Telangana	521556651761
10	John	DeSousa	2005-08-17	Guindy, Chennai, TN	954891122475
5	Nandini	Roy	2003-12-19	Fortune Somya, Bhopal, MP	225423344657

```
9 rows in set (0.00 sec)

mysql>
```

**Caution:** The WHERE clause should be used in the UPDATE and DELETE statement, otherwise it will apply on all the records.

### 2.5 SQL FOR DATA QUERY LANGUAGE (DQL)

So far we have learned to create a database, store and manipulate data in the database tables. The data stored in a database can be retrieved using a mechanism called as *Query*. SQL provides efficient mechanisms to retrieve data stored in multiple tables in MySQL database (or any other RDBMS). The SQL statement SELECT is used to retrieve data from the tables in a database and is also called a query statement. One of the most commonly used DQL is SELECT statement.

#### 2.5.1 SELECT Statement

In SQL, the SELECT statement is used to retrieve data from tables in a database and output is displayed as per the specified parameter on successful execution of statement.

*Syntax:*

```
SELECT attribute1, attribute2,... attribute N
FROM table_name
WHERE condition;
```

Here, attribute1, attribute2, ... attributeN are the names of columns of the table table\_name from which data is to be retrieved. The FROM clause is always written with SELECT clause as it specifies the name of the table from which data has to be retrieved. The WHERE clause is optional and is used to retrieve data to meet any specified condition(s).

To select all the columns and rows available in a table, use the following select statement.

```
SELECT * FROM table_name;
```

Here \* is used to retrieve all columns/attributes available in the table.

Let us use SELECT statement to retrieve names of the student whose name are starting with alphabet “D”.

**Example 2.30:** The following SQL query statement is used to retrieve the name and date of birth of student whose roll number is 1.

```
mysql> SELECT Stu_RollNo, Stu_DOB FROM studentrecord WHERE Stu_RollNo = 1;
+-----+-----+
| Stu_RollNo | Stu_DOB |
+-----+-----+
|          1 | 2003-05-15 |
+-----+-----+
1 row in set (0.00 sec)

mysql>
```

In the above query, observe that the Student Roll Number and Date of birth of the of the student whose roll number is 1 is retrieved using WHERE clause.

### 2.5.2 Querying using database OFFICE

Let us consider an EMP table of employee database with the following fields. The “empno” is a primary key and “deptno” as foreign key. Table 3.1 shows the data entered in the Emp table.

**Table 2.8 Records available in EMP table**

empno	ename	job	mgr	hiredate	sal	comm	deptno
7019	Smita	Clerk	7552	12/14/1994	8800	NULL	20
7049	Alam	Salesman	7348	02/17/1995	9600	1800	30
7171	Wasim	Salesman	7348	02/19/1995	9250	2000	30
7216	Jawahar	Manager	7489	03/30/1995	10975	NULL	20
7304	Manoj	Salesman	7348	09/25/1995	9250	2900	30
7348	Balwinder	Manager	7489	04/28/1995	10850	NULL	30
7432	Chetana	Manager	7489	06/06/1995	10450	NULL	10
7438	Sachin	Analyst	7216	12/05/1996	11000	NULL	20
7489	Kushaal	President	NULL	11/14/1995	13000	NULL	10
7494	Tarun	Salesman	7348	09/05/1995	9500	0	30
7526	Amar	Clerk	7438	01/08/1997	9100	NULL	20
7550	Jyoti	Clerk	7348	11/30/1995	8950	NULL	30
7552	Farhan	Analyst	7216	10/27/1995	11000	NULL	20
7584	Mohan	Clerk	7432	01/20/1996	9300	NULL	10
7984	Lalitha	Clerk	7432	05/23/1998	10300	NULL	10

Now if you wish to retrieve the desired data from the table, let us see how to apply the SELECT clause to retrieve the data.

(a) **Retrieve selected columns** – It is possible to retrieve the data of one column of table.



```
mysql> SELECT empno FROM emp;
+-----+
| empno |
+-----+
| 7019  |
| 7049  |
| 7171  |
| 7216  |
| 7304  |
| 7348  |
| 7432  |
| 7438  |
| 7489  |
| 7494  |
| 7526  |
| 7550  |
| 7552  |
| 7584  |
| 7984  |
+-----+
15 rows in set (0.00 sec)
mysql>
```

**Example 2.31:** The following SQL query statement is used to retrieve employee number of all employees in the table.

Observe that the above query retrieve *empno* of all the employee from *Emp* table as only one column is specified to retrieve.

Let us see another query that select two columns such as *emp no* and corresponding *employee name*. Modify the same query by specifying two fields of table as “*empno*” and “*ename*”. and observe the desired output as below.

```
mysql> SELECT empno, ename FROM emp;
+-----+-----+
| empno | ename |
+-----+-----+
| 7019  | Smita |
| 7049  | Alam  |
| 7171  | Wasim |
| 7216  | Jawahar |
| 7304  | Manoj |
| 7348  | Balwinder |
| 7432  | Chetana |
| 7438  | Sachin |
| 7489  | Kushaal |
| 7494  | Tarun |
| 7526  | Amar  |
| 7550  | Jyoti |
| 7552  | Farhan |
| 7584  | Mohan |
| 7984  | Lalitha |
+-----+-----+
15 rows in set (0.00 sec)
mysql>
```

**Example 2.32:** The following SQL query statement will retrieve the data of employee number and name in two columns.

**(b) Renaming of columns** – There is specific naming conventions of the fields in table. It is possible to rename any column while displaying the output by using the alias ‘AS’.

**Example 2.33:** The following SQL query statement selects *Employee name* as “Name” in the output for all the employees.

```
mysql> SELECT ename as Name FROM emp;
+-----+
| Name |
+-----+
| Smita |
| Alam  |
| Wasim |
| Jawahar |
| Manoj |
| Balwinder |
| Chetana |
| Sachin |
| Kushaal |
| Tarun |
| Amar  |
| Jyoti |
| Farhan |
| Mohan |
| Lalitha |
+-----+
15 rows in set (0.00 sec)
mysql>
```

**Example 2.34:** The following SQL query statement will calculate and to display the annual salary of employee. Annual salary is calculated as “sal\*12”.

```
mysql> SELECT ename as Name, sal*12 FROM emp;
+-----+-----+
| Name      | sal*12 |
+-----+-----+
| Smita     | 105600 |
| Alam     | 115200 |
| Wasim    | 111000 |
| Jawahar  | 131700 |
| Manoj    | 111000 |
| Balwinder | 130200 |
| Chetana  | 125400 |
| Sachin   | 132000 |
| Kushaal  | 156000 |
| Tarun    | 114000 |
| Amar     | 109200 |
| Jyoti    | 107400 |
| Farhan   | 132000 |
| Mohan    | 111600 |
| Lalitha  | 123600 |
+-----+-----+
15 rows in set (0.00 sec)

mysql>
```

Now it doesn't look nice to display the caption as “sal\*12” in the table. It is possible to display it with new caption as “Annual Salary” for “sal\*12”. The revised query and its output is given below.

```
mysql> SELECT ename as Name, sal*12 AS 'Annual Income' FROM emp;
+-----+-----+
| Name      | Annual Income |
+-----+-----+
| Smita     | 105600 |
| Alam     | 115200 |
| Wasim    | 111000 |
| Jawahar  | 131700 |
| Manoj    | 111000 |
| Balwinder | 130200 |
| Chetana  | 125400 |
| Sachin   | 132000 |
| Kushaal  | 156000 |
| Tarun    | 114000 |
| Amar     | 109200 |
| Jyoti    | 107400 |
| Farhan   | 132000 |
| Mohan    | 111600 |
| Lalitha  | 123600 |
+-----+-----+
15 rows in set (0.00 sec)

mysql>
```

Observe that “ename” is shown with the caption as “Name” and “sal\*12” is shown with the caption as “Annual Income”.

**Note** – Annual Income is just the caption to display. It will not add as a new column in the database table. It is just for displaying the output of the query. If an aliased column name has space as in the case of *Annual Income*, it should be enclosed in quotes as ‘**Annual Income**’.

(c) **Distinct Clause** – The SELECT clause retrieves all the data through query as output. There may be a chance of duplicate values such as 2 persons with the same name working in the department. The DISTINCT clause has provision to retrieve the unique records by omitting the duplicate records. The DISTINCT clause is used for this purpose.

**Example 2.35:** The following SQL query statement shows the different departments available in the “emp” table.

```
mysql> SELECT DISTINCT deptno FROM emp;
+-----+
| deptno |
+-----+
|      20 |
|      30 |
|      10 |
+-----+
3 rows in set (0.01 sec)

mysql>
```

Let us understand, how to retrieve different types of jobs available using DISTINCT clause in the following example.

**Example 2.36:** The following SQL query statement will use DISTINCT clause to retrieve different types of jobs available in the “emp” table.

```
mysql> SELECT DISTINCT(job) FROM emp;
+-----+
| job      |
+-----+
| Clerk    |
| Salesman |
| Manager  |
| Analyst  |
| President|
+-----+
5 rows in set (0.00 sec)

mysql>
```

Observe that there are 5 different job titles although more number of records exists.

(d) **WHERE Clause** – It retrieves data that meet some specified conditions. In our OFFICE database, more than one employee can have the same salary.

**Example 2.37:** The following SQL query statement will gives distinct salaries of the employees working in the department number 10.

```
mysql> SELECT DISTINCT sal FROM emp WHERE deptno=10;
+-----+
| sal      |
+-----+
| 10450    |
| 13000    |
| 9300     |
| 10300    |
+-----+
4 rows in set (0.01 sec)

mysql>
```

Observe in the output that all the records of employee working in dept no. 10 and having the distinct salary are retrieved.

In the above example, = operator is used in the WHERE clause. Other relational operators like (<, <=, >, >=, !=) can also be used to specify conditions as per your requirement. The logical operators AND, OR, and NOT are used to combine multiple conditions.

Let us see, how to compare columns/fields value/s to specific required records or columns.

**Example 2.38:** The following SQL query statement will display all the details of those employees of 30 department who earn more than 5000.

```
mysql> SELECT * FROM emp WHERE sal > 5000 AND deptno = 30;
+-----+-----+-----+-----+-----+-----+-----+
| empno | ename   | job      | mgr   | hiredate | sal   | comm  | deptno |
+-----+-----+-----+-----+-----+-----+-----+
| 7049  | Alam    | Salesman | 7348  | 1995-02-17 | 9600 | 1800  | 30     |
| 7171  | Wasim   | Salesman | 7348  | 1995-02-19 | 9250 | 2000  | 30     |
| 7304  | Manoj   | Salesman | 7348  | 1995-09-25 | 9250 | 2900  | 30     |
| 7348  | Balwinder | Manager | 7489  | 1995-04-28 | 10850 | NULL  | 30     |
| 7494  | Tarun   | Salesman | 7348  | 1995-09-05 | 9500 | 0     | 30     |
| 7550  | Jyoti   | Clerk    | 7348  | 1995-11-30 | 8950 | NULL  | 30     |
+-----+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)

mysql>
```

**Note:** Observe the output, two different conditions are being tested separately. First condition tested for Salary is greater than 5000 and second condition is for department number is 10. AND operator used to join both conditions.

Let us make a comparison of salary like who is getting more than 8000 and less than 11000.

**Example 2.39:** The following SQL query statement will selects the name and department number of all those employees who are earning salary between 8000 and 11000 inclusive of both values.

```
mysql> SELECT ename, deptno FROM emp WHERE sal >= 8000 AND sal <= 11000;
+-----+-----+
| ename | deptno |
+-----+-----+
| Smita | 20     |
| Alam  | 30     |
| Wasim | 30     |
| Jawahar | 20    |
| Manoj | 30     |
| Balwinder | 30    |
| Chetana | 10    |
| Sachin | 20    |
| Tarun | 30     |
| Amar  | 20     |
| Jyoti | 30     |
| Farhan | 20    |
| Mohan | 10     |
| Lalitha | 10    |
+-----+-----+
14 rows in set (0.01 sec)

mysql>
```

The query in example 2.39 defines a range of salary between 8000 and 11000 that can also be achieved using a comparison operator **BETWEEN**, in the query as below. The output of this query will be same as above.

```
mysql> SELECT ename, deptno FROM emp WHERE sal BETWEEN 8000 AND 11000;
+-----+-----+
| ename | deptno |
+-----+-----+
| Smita | 20     |
| Alam  | 30     |
| Wasim | 30     |
| Jawahar | 20    |
| Manoj | 30     |
| Balwinder | 30    |
| Chetana | 10    |
| Sachin | 20    |
| Tarun | 30     |
| Amar  | 20     |
| Jyoti | 30     |
| Farhan | 20    |
| Mohan | 10     |
| Lalitha | 10    |
+-----+-----+
14 rows in set (0.00 sec)

mysql>
```

**Note:** The BETWEEN operator defines the range of values in which the column value must fall into, to make the condition true.

**Example 2.40:** The following SQL query statement will select details of all the employees who work in any of the department number 10, 20, or 40.

```
mysql> SELECT * FROM emp WHERE deptno = 10 OR deptno = 20 OR deptno = 40;
```

empno	ename	job	mgr	hiredate	sal	comm	deptno
7019	Smita	Clerk	7552	1994-12-14	8800	NULL	20
7216	Jawahar	Manager	7489	1995-03-30	10975	NULL	20
7432	Chetana	Manager	7489	1995-06-06	10450	NULL	10
7438	Sachin	Analyst	7216	1996-12-05	11000	NULL	20
7489	Kushaal	President	NULL	1995-11-14	13000	NULL	10
7526	Amar	Clerk	7438	1997-01-08	9100	NULL	20
7552	Farhan	Analyst	7216	1995-10-27	11000	NULL	20
7584	Mohan	Clerk	7432	1996-01-20	9300	NULL	10
7984	Lalitha	Clerk	7432	1998-05-23	10300	NULL	10

```
9 rows in set (0.00 sec)

mysql>
```

### (E) Membership operator IN

The IN operator compares a value with a set of values and returns true if the value belongs to that set. The query given in *Example 2.40* can be rewritten using IN operator as below.

```
mysql> SELECT * FROM emp WHERE deptno IN (10, 20, 40);
```

It will give the same output as above.

**Example 2.41:** The following SQL query statement selects details of all the employees except those working in department number 10 or 20.

```
mysql> SELECT * FROM emp WHERE deptno IN (10, 20, 40);
```

empno	ename	job	mgr	hiredate	sal	comm	deptno
7019	Smita	Clerk	7552	1994-12-14	8800	NULL	20
7216	Jawahar	Manager	7489	1995-03-30	10975	NULL	20
7432	Chetana	Manager	7489	1995-06-06	10450	NULL	10
7438	Sachin	Analyst	7216	1996-12-05	11000	NULL	20
7489	Kushaal	President	NULL	1995-11-14	13000	NULL	10
7526	Amar	Clerk	7438	1997-01-08	9100	NULL	20
7552	Farhan	Analyst	7216	1995-10-27	11000	NULL	20
7584	Mohan	Clerk	7432	1996-01-20	9300	NULL	10
7984	Lalitha	Clerk	7432	1998-05-23	10300	NULL	10

```
9 rows in set (0.00 sec)

mysql>
```

**Note:** Here NOT operator is used in combination with IN to retrieve all records except with deptno 10 and 20.

**(F) ORDER BY Clause** – It is used to display data in an ordered form with respect to a specified column. By default, ORDER BY displays records in ascending order of the specified column values. The DESC keyword is used to display the records in descending.

Let us arrange the records in ascending or descending order using the ORDER BY clause with DESC clause example 2.42.

**Example 2.42:** The following SQL query statement selects details of all the employees in ascending order of their salaries.

```
mysql> SELECT * FROM emp ORDER BY sal;
+-----+-----+-----+-----+-----+-----+-----+-----+
| empno | ename   | job      | mgr   | hiredate | sal   | comm  | deptno |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7019 | Smita   | Clerk    | 7552  | 1994-12-14 | 8800 | NULL  | 20     |
| 7550 | Jyoti   | Clerk    | 7348  | 1995-11-30 | 8950 | NULL  | 30     |
| 7526 | Amar    | Clerk    | 7438  | 1997-01-08 | 9100 | NULL  | 20     |
| 7171 | Wasim   | Salesman | 7348  | 1995-02-19 | 9250 | 2000  | 30     |
| 7304 | Manoj   | Salesman | 7348  | 1995-09-25 | 9250 | 2900  | 30     |
| 7584 | Mohan   | Clerk    | 7432  | 1996-01-20 | 9300 | NULL  | 10     |
| 7494 | Tarun   | Salesman | 7348  | 1995-09-05 | 9500 | 0     | 30     |
| 7049 | Alam    | Salesman | 7348  | 1995-02-17 | 9600 | 1800  | 30     |
| 7984 | Lalitha | Clerk    | 7432  | 1998-05-23 | 10300 | NULL  | 10     |
| 7432 | Chetana | Manager  | 7489  | 1995-06-06 | 10450 | NULL  | 10     |
| 7348 | Balwinder | Manager | 7489  | 1995-04-28 | 10850 | NULL  | 30     |
| 7216 | Jawahar | Manager  | 7489  | 1995-03-30 | 10975 | NULL  | 20     |
| 7438 | Sachin  | Analyst  | 7216  | 1996-12-05 | 11000 | NULL  | 20     |
| 7552 | Farhan  | Analyst  | 7216  | 1995-10-27 | 11000 | NULL  | 20     |
| 7489 | Kushaal | President | NULL  | 1995-11-14 | 13000 | NULL  | 10     |
+-----+-----+-----+-----+-----+-----+-----+-----+
15 rows in set (0.01 sec)

mysql>
```

Observe that the records are displayed in ascending order of Salary of each employee. To arrange records in descending order, use DESC clause with ORDER BY as in example 2.43.

**Example 2.43:** The following SQL query statement selects details of all the employees in descending order of their salaries.

```
mysql> SELECT * FROM emp ORDER BY sal DESC;
+-----+-----+-----+-----+-----+-----+-----+-----+
| empno | ename   | job      | mgr   | hiredate | sal   | comm  | deptno |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7489 | Kushaal | President | NULL  | 1995-11-14 | 13000 | NULL  | 10     |
| 7438 | Sachin  | Analyst  | 7216  | 1996-12-05 | 11000 | NULL  | 20     |
| 7552 | Farhan  | Analyst  | 7216  | 1995-10-27 | 11000 | NULL  | 20     |
| 7216 | Jawahar | Manager  | 7489  | 1995-03-30 | 10975 | NULL  | 20     |
| 7348 | Balwinder | Manager  | 7489  | 1995-04-28 | 10850 | NULL  | 30     |
| 7432 | Chetana | Manager  | 7489  | 1995-06-06 | 10450 | NULL  | 10     |
| 7984 | Lalitha | Clerk    | 7432  | 1998-05-23 | 10300 | NULL  | 10     |
| 7049 | Alam    | Salesman | 7348  | 1995-02-17 | 9600  | 1800  | 30     |
| 7494 | Tarun   | Salesman | 7348  | 1995-09-05 | 9500  | 0     | 30     |
| 7584 | Mohan   | Clerk    | 7432  | 1996-01-20 | 9300  | NULL  | 10     |
| 7171 | Wasim   | Salesman | 7348  | 1995-02-19 | 9250  | 2000  | 30     |
| 7304 | Manoj   | Salesman | 7348  | 1995-09-25 | 9250  | 2900  | 30     |
| 7526 | Amar    | Clerk    | 7438  | 1997-01-08 | 9100  | NULL  | 20     |
| 7550 | Jyoti   | Clerk    | 7348  | 1995-11-30 | 8950  | NULL  | 30     |
| 7019 | Smita   | Clerk    | 7552  | 1994-12-14 | 8800  | NULL  | 20     |
+-----+-----+-----+-----+-----+-----+-----+-----+
15 rows in set (0.00 sec)

mysql>
```

**Note:** DESC clause used after the column name on which the records to be displayed in descending order.

**(G) Handling NULL Values** – SQL supports a special value called NULL to represent a missing or unknown value. For example, the “*Par\_Phone*” column in the table “*ParentRecord*” can have missing value for certain records. Hence, NULL is used to represent such unknown values. It is important to note that NULL is different from value 0 (zero). Also, any arithmetic operation performed with NULL value gives NULL. For example,  $5 + \text{NULL} = \text{NULL}$  because NULL is unknown hence the result is also unknown. In order to check for NULL value in a column, use **IS NULL** operator in particular statement. Example 2.44 illustrates the use of NULL clause.

**Example 2.44:** The following SQL query statement selects details of all employees who have not been given a bonus. This implies that the bonus column will be blank.

```
mysql> SELECT * FROM emp WHERE comm IS NULL;
```

empno	ename	job	mgr	hiredate	sal	comm	deptno
7019	Smita	Clerk	7552	1994-12-14	8800	NULL	20
7216	Jawahar	Manager	7489	1995-03-30	10975	NULL	20
7348	Balwinder	Manager	7489	1995-04-28	10850	NULL	30
7432	Chetana	Manager	7489	1995-06-06	10450	NULL	10
7438	Sachin	Analyst	7216	1996-12-05	11000	NULL	20
7489	Kushaal	President	NULL	1995-11-14	13000	NULL	10
7526	Amar	Clerk	7438	1997-01-08	9100	NULL	20
7550	Jyoti	Clerk	7348	1995-11-30	8950	NULL	30
7552	Farhan	Analyst	7216	1995-10-27	11000	NULL	20
7584	Mohan	Clerk	7432	1996-01-20	9300	NULL	10
7984	Lalitha	Clerk	7432	1998-05-23	10300	NULL	10

```
11 rows in set (0.00 sec)

mysql>
```

Observe the output and see column mgr and comm where NULL is present.

It is also possible to join NULL statement with any other condition. Example 3.11 shows how to use it in statement.

**Example 2.45:** The following SQL query statement selects emp number, employee names and job of all those employees who have been given a comm (i.e., comm is not null) and works in the department 30.

```
mysql> Select empno, ename, job
-> FROM emp WHERE comm IS NOT NULL
-> AND deptno=30;
```

empno	ename	job
7049	Alam	Salesman
7171	Wasim	Salesman
7304	Manoj	Salesman
7494	Tarun	Salesman

```
4 rows in set (0.02 sec)

mysql>
```

**(H) Having clause** – It is used in SELECT statement to make group with certain condition in result of query.

*Syntax:*

```
SELECT expression1, expression2, ... expression_n,
       aggregate_function (expression)
FROM tables
   [WHERE conditions]
GROUP BY expression1, expression2, ... expression_n
HAVING condition;
```

Example 2.45 shows how to use Group by and Having clause jointly. The HAVING clause must follow the GROUP BY clause in any SELECT query and must also preceded by ORDER BY clause if used.

**Example 2.45:** The following SQL query statement selects jobs, number of employees in that job, their total salary and department number wise list where minimum 3 employee of same type of job are working.

```
mysql> SELECT job, COUNT(*), SUM(sal)
-> FROM emp GROUP BY job
-> HAVING COUNT(*)>2;
+-----+-----+-----+
| job      | COUNT(*) | SUM(sal) |
+-----+-----+-----+
| Clerk    |         5 |    46450 |
| Salesman |         4 |    37600 |
| Manager  |         3 |    32275 |
+-----+-----+-----+
3 rows in set (0.00 sec)

mysql>
```

**(I) Substring pattern matching** – Many times it may require that the query should not retrieve that exact text or value, rather it should retrieve the matching of few characters or values. For example, to find out names starting with “M” or to find out pin codes starting with “11”, is called substring pattern matching. Such patterns cannot match using = operator. SQL provides a LIKE operator that can be used with the WHERE clause to search for specified pattern in a column.

The LIKE operator makes use of the following two wild card characters - (%) and (\_). The percent (%) is used to represent zero, one, or multiple characters. The underscore (\_) is used to represent exactly a single character.

There are several situations when we search data records for some pattern matching. A very common situation when you search any contacts in your smart phone, you just start typing first few characters of the name, then immediately list appears with these characters and you tap on the required name to call. Example 3.46 to 3.51 demonstrates such situations to search some patterns in text values of records using LIKE clause.

**Example 2.46:** The following SQL query statement selects details of all those employees whose name starts with 'K'.

```
mysql> SELECT * FROM emp WHERE ename like 'K%';
+-----+-----+-----+-----+-----+-----+-----+
| empno | ename  | job      | mgr  | hiredate | sal   | comm  | deptno |
+-----+-----+-----+-----+-----+-----+-----+
| 7489  | Kushaal | President | NULL | 1995-11-14 | 13000 | NULL  | 10     |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

**Example 2.47:** The following SQL query statement selects details of all those employees whose name whose name ends with 'a', and gets a salary more than 8500.

```
mysql> SELECT * FROM emp WHERE ename like '%a' AND sal > 8500;
+-----+-----+-----+-----+-----+-----+-----+
| empno | ename  | job      | mgr  | hiredate | sal   | comm  | deptno |
+-----+-----+-----+-----+-----+-----+-----+
| 7019  | Smita  | Clerk    | 7552 | 1994-12-14 | 8800  | NULL  | 20     |
| 7432  | Chetana | Manager  | 7489 | 1995-06-06 | 10450 | NULL  | 10     |
| 7984  | Lalitha | Clerk    | 7432 | 1998-05-23 | 10300 | NULL  | 10     |
+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql>
```

**Example 2.48:** The following SQL query statement selects details of all those employees whose name consists of exactly 5 letters and starts with any letter but has 'mita' after that.



```
mysql> SELECT * FROM emp WHERE ename like '_mita';
+-----+-----+-----+-----+-----+-----+-----+-----+
| empno | ename | job   | mgr | hiredate | sal  | comm | deptno |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7019 | Smita | Clerk | 7552 | 1994-12-14 | 8800 | NULL | 20 |
+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

You can also match a particular character or string in between the text simply by using wild card character as shown in example 2.49.

**Example 2.49:** The following SQL query statement selects all columns of all employees containing 'ma' as a substring in name.

```
mysql> SELECT * FROM emp WHERE ename like '%ma%';
+-----+-----+-----+-----+-----+-----+-----+-----+
| empno | ename | job   | mgr | hiredate | sal  | comm | deptno |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7304 | Manoj | Salesman | 7348 | 1995-09-25 | 9250 | 2900 | 30 |
| 7526 | Amar  | Clerk   | 7438 | 1997-01-08 | 9100 | NULL  | 20 |
+-----+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

**Example 2.50:** The following SQL query statement selects all columns of employees containing 'a' as the second character in their names.

```
mysql> SELECT * FROM emp WHERE ename like '_a%';
+-----+-----+-----+-----+-----+-----+-----+-----+
| empno | ename   | job   | mgr | hiredate | sal  | comm | deptno |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7171 | Wasim  | Salesman | 7348 | 1995-02-19 | 9250 | 2000 | 30 |
| 7216 | Jawahar | Manager | 7489 | 1995-03-30 | 10975 | NULL | 20 |
| 7304 | Manoj  | Salesman | 7348 | 1995-09-25 | 9250 | 2900 | 30 |
| 7348 | Balwinder | Manager | 7489 | 1995-04-28 | 10850 | NULL | 30 |
| 7438 | Sachin | Analyst  | 7216 | 1996-12-05 | 11000 | NULL | 20 |
| 7494 | Tarun  | Salesman | 7348 | 1995-09-05 | 9500  | 0    | 30 |
| 7552 | Farhan | Analyst  | 7216 | 1995-10-27 | 11000 | NULL | 20 |
| 7984 | Lalitha | Clerk   | 7432 | 1998-05-23 | 10300 | NULL | 10 |
+-----+-----+-----+-----+-----+-----+-----+-----+
8 rows in set (0.00 sec)

mysql>
```

**Example 2.51:** The following SQL query statement selects records of all the employees except Alam.

## 2.6 SQL FOR DATA CONTROL LANGUAGE (DCL)

```
mysql> SELECT * FROM emp WHERE NOT ename= 'Alam';
+-----+-----+-----+-----+-----+-----+-----+-----+
| empno | ename   | job   | mgr | hiredate | sal  | comm | deptno |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7019 | Smita   | Clerk | 7552 | 1994-12-14 | 8800 | NULL | 20 |
| 7171 | Wasim  | Salesman | 7348 | 1995-02-19 | 9250 | 2000 | 30 |
| 7216 | Jawahar | Manager | 7489 | 1995-03-30 | 10975 | NULL | 20 |
| 7304 | Manoj  | Salesman | 7348 | 1995-09-25 | 9250 | 2900 | 30 |
| 7348 | Balwinder | Manager | 7489 | 1995-04-28 | 10850 | NULL | 30 |
| 7432 | Chetana | Manager | 7489 | 1995-06-06 | 10450 | NULL | 10 |
| 7438 | Sachin | Analyst  | 7216 | 1996-12-05 | 11000 | NULL | 20 |
| 7489 | Kushaal | President | NULL | 1995-11-14 | 13000 | NULL | 10 |
| 7494 | Tarun  | Salesman | 7348 | 1995-09-05 | 9500  | 0    | 30 |
| 7526 | Amar   | Clerk   | 7438 | 1997-01-08 | 9100  | NULL | 20 |
| 7550 | Jyoti  | Clerk   | 7348 | 1995-11-30 | 8950  | NULL | 30 |
| 7552 | Farhan | Analyst  | 7216 | 1995-10-27 | 11000 | NULL | 20 |
| 7584 | Mohan  | Clerk   | 7432 | 1996-01-20 | 9300  | NULL | 10 |
| 7984 | Lalitha | Clerk   | 7432 | 1998-05-23 | 10300 | NULL | 10 |
+-----+-----+-----+-----+-----+-----+-----+-----+
14 rows in set (0.01 sec)

mysql>
```

Data Control Language is the part of SQL, which have commands to manage users for their work permission. The user will be able to work as per the permissions granted to them by DBA (Database Administrator). DCL includes the commands GRANT and REVOKE, which are used to provide rights & permissions to user.

**GRANT statement** – The GRANT statement is used to give access privileges to a specific user to work with any selected database only.

*Syntax:*

```
GRANT SELECT, UPDATE ON Test_Table TO NewUser1, NewUser2;
```

*Example:*

```
GRANT SELECT, UPDATE, DELETE ON carshowroom TO 'WebUser';
```

Here the user 'WebUser' will be able to use only three SELECT, UPDATE and DELETE SQL statements when working on carshowroom database.

**REVOKE statement** – The REVOKE statement is used to withdraw privileges from a specific user so that specific user could not use specific statement on selected database. In other words it is useful to take back the given permission/s from the user.

*Syntax:*

```
REVOKE Privilege_Name ON Object_Name FROM User_Name.
```

*Example:*

```
REVOKE DELETE ON carshowroom FROM WebUser;
```

## 2.7 SQL FOR TRANSACTION CONTROL LANGUAGE (TCL)

Transaction control language (TCL) is the part of SQL commands that allows to permanently change the databases or undo the databases transactions. It is similar to save the database or undo the current changes. The COMMIT, ROLLBACK and SAVEPOINT statements comes under this category.

**COMMIT** – Commit command is used to save all the transactions to the database. After completing any operation or SQL statement, you can simply write COMMIT as the next statement to permanently save data in the database.

*Syntax:*

```
Commit;
```

*Example:* `DELETE FROM ClassStudents WHERE RollNo =25;`

```
Commit;
```

Here, after DELETE statement, the COMMIT statement is used. It means the student record whose RollNo is 25 is permanently deleted. Now after COMMIT statement, it is not possible to rollback the record of that student.

**ROLLBACK** – ROLLBACK command allows to undo transactions that have not already been saved to the database. This statement is useful to restore the database to the state where last commit statement was used. Rollback statement is also used with SAVEPOINT statement to jump to specific Savepoint in the database transactions.

*Syntax:*

```
ROLLBACK;
```

**SAVEPOINT** – This command helps to sets a Savepoint within a transaction. Basically SAVEPOINT statement is used to save a transaction temporarily so that user can rollback to that point as and when required.

*Syntax:* SAVEPOINT Savepoint\_Name;

## SUMMARY

- SQL is a domain-specific language that is used to manage relational databases.
- Currently almost all RDBMS such as MySQL, Oracle, Informix, SQL server, MS Access, and Sybase uses SQL as their standard database language.
- SQL is easy to learn as the statements comprise of descriptive English words.

- SQL is a open source, interactive, portable, faster query processing, standardized and universal language to work with with RDBMS.
- SQL is divided into five types like DDL, DML, DQL, TCL and DCL.
- DDL (Data Definition Language) includes SQL statements such as, Create table, Alter table and Drop table.
- Create command is used to create database and its further objects like Table, View.
- DML (Data Manipulation Language) includes SQL statements such as, insert, select, update and delete.
- A table is a collection of rows and columns, where each row is a record and columns describe the feature of records.
- DESCRIBE TABLE statement is used to view the structure of an already existing table
- ALTER TABLE statement is used to make changes in the structure of a table like adding, removing column and changing datatype of column(s). It is also used to apply/remove any constraints like Primary Key, Foreign Key etc.
- DROP statement is used to remove a database or a table permanently from the database system.
- TRUNCATE statement is used to delete all records from the table but table structure will exist in database.
- INSERT INTO statement is used to insert new records in any existing table
- UPDATE statement is used to make required changes in records of any table.
- DELETE statement is used to delete/remove one or more records from a table.
- CREATE TABLE statement can also be used to create new table from existing tables/s.
- RENAME statement is used to change the name of existing tables of other database objects.
- Views in any database is a special kind of virtual table that is created from one or more table and having no data of its own.
- WHERE clause in SQL query is used to enforce condition(s).
- DISTINCT clause is used to eliminate repetition and display the values only once.
- The BETWEEN operator defines the range of values inclusive of boundary values.
- The IN operator selects values that match any value in the given list of values.
- NULL values can be tested using IS NULL and IS NOT NULL.
- ORDER BY clause is used to display the result of a SQL query in ascending or descending order with respect to specified attribute values. By default, the order is ascending.
- LIKE operator is used for pattern matching. % and \_ are two wild card characters. The per cent (%) symbol is used to represent zero or more characters. The underscore (\_) symbol is used to represent a single character.

## Check Your Progress

### A. Multiple choice questions

1. Which of the following is not a valid aggregate function? (a) COUNT (b) COMPUTE (c) SUM (d) MAX
2. DDL stands for (a) Data Describe Language (b) Definition Data Language (c) Data Definition Language (d) Data Distinct Language
3. Which of the following SQL command is used to remove data from table (a) Collapse (b) Remove (c) Alter (d) Delete

4. The records and structure of a table may be removed or deleted from the database using which command? (a) Remove (b) Delete (c) Drop (d) Truncate
5. SQL \_\_\_ statement can be used to delete or drop existing databases in a SQL schema. (a) Create Database (b) Rename Database (c) Drop Database (d) Select Database
6. Using DROP TABLE command in SQL (a) Drop the table structure (b) Drop the Integrity constraints (c) Drop the Relationship (d) All of the above
7. Using DROP TABLE command in SQL (a) Drop the table structure (b) Drop the Integrity constraints (c) Drop the Relationship (d) None of the above
8. TRUNCATE TABLE requires (a) Where Clause (b) Having Clause (c) Both A And B (d) None of the above
9. Which of the following clause is used to add a Primary Key constraint after creating table (a) Update (b) Add (c) Alter (d) Join
10. Which of the following clause is used to remove a primary key constraint (a) Delete (b) Drop (c) Alter (d) Remove
11. Which of the following SQL statement is used to give result in sorted order (a) Sort By (b) Order (c) Order By (d) Sort
12. Commands under DCL are (a) GRANT (b) REVOKE (c) Both A. and B. (d) None of the above
13. The SQL command to retrieve table records is (a) RETRIEVE (b) SELECT (c) CREATE (d) ALTER
14. Which of the following operator is used for pattern matching in SQL? (a) BETWEEN operator (b) LIKE operator (c) EXISTS operator (d) None of these
15. Which operator is used to check the absence of data in any column (a) EXISTS operator (b) NOT operator (c) IS NULL operator (d) None of these
16. Which of the following keyword is used to select only unique values from any column (a) DISTINCTIVE (b) UNIQUE (c) DISTINCT (d) DIFFERENT

#### B. Fill in the blanks

1. SQL is divided in \_\_\_\_\_ category.
2. The \_\_\_\_\_ command is used to see the structure of table.
3. The \_\_\_\_\_ command is used to remove all records.
4. The \_\_\_\_\_ command is used to add an attribute in an existing table.
5. The \_\_\_\_\_ command is used to remove all records only from a table.
6. The \_\_\_\_\_ command is used to remove a attribute from a table.
7. A view is a special kind of \_\_\_\_\_ table.
8. Views can be created form \_\_\_\_\_ or more tables.
9. Grant and Revoke are part of \_\_\_\_\_ in SQL.
10. Commit and Savepoint are part of \_\_\_\_\_ in SQL.
11. To sort the result of a query in descending order, we can use clause \_\_\_\_\_
12. To extract unique values from a column, user can use \_\_\_\_\_ clause.

#### C. State whether True or False

1. INSERT clause is used to add a Foreign key constraint.
2. ALTER clause is used to add a Primary key constraint after table is created.
3. DROP command is used to delete the structure of a table from the database.
4. Updation and deletion of records are part of DDL.
5. Insert into statement is useful to insert a new field in any table.
6. Aggregate functions are used to perform calculations on multiple values and returns a single value.
7. Aggregate functions are mostly used with the SELECT statement.
8. DML is used to create a new database objects like table and view.
9. A new table can be created from existing table(s).
10. The name of any tables once its created and records are inserted cannot be change.

**D. Short answers questions**

1. What do you understand by SQL?
2. SQL Statements are classified in how many ways?
3. Differentiate between DDL and DML?
4. Differentiate between DCL and TCL?
5. What is the difference between ALTER and UPDATE command.
6. Differentiate between DELETE and DROP command
7. What is create statement? How many database objects can be created using this?
8. Write the CREATE statement to create the following relations with given constraints.  
Book(ISBN (Text), Title (Text), Author (Text), PubID(Text), Price (Numeric), Pages (Numeric)).  
Here ISBN is Primary Key field and remaining all are Not Null.
9. Modify the Book table in previous question and add one more new field Discount (Numeric).
10. Shyam has created one database name Mycontacts but he is not able to create new table in this database. What command should Shyam be used before creating the table?
11. Mr. Sachin Agrawal created two tables with Course as Primary Key in Table1 and Foreign key in Table2 while inserting new row in second Table2 Mr Agrawal is not able to insert new value in the column City. What could be the possible reason for this?

**E. Practical Exercises**

1. Based on employee table, write SQL queries to –
  - display the list of employee belonging to the department 30.
  - display the list of employee number and name of managers.
  - display the list of clerks working in department 10
  - display the detailed list of those employees who have joined before the July 1995
  - display the the names of employees who are not managers.
  - display the List of employees whose employees numbers are 7438, 7216, 7019 and 7984
  - display the employee name and salary whose salary is between 9000 and 10500.
  - display the employee name who have joined after 30 June 1995.
  - display the List of different job available in the emp table
  - display the List of employee who are not getting/eligible commission.
  - display the list of employee whose name start with “M”
  - display the list of employee whose name has 6 characters.
  - display the List of employee having 'a' as second character.
  - display the List of all employee in descending order of salary.
  - display the List of employee in ascending order of hire date.
  - display the employee name, Salary, PF, HRA, DA and Gross; order the result in descending order of gross. Here PF is 10% of Salary, HRA is 50% of Salary and DA is 30% of Salary and Gross is sum of Salary, HRA and DA.
  - display the unique jobs available in emp table.
  - display the total salary which is sum of salary and commission.
  - create new table named NewEmp from existing table emp with all same field and records.

- add a new column address and mobno to the newly created table NewEmp.
- Suppose the DBMS admin forget to make empno as primary key and deptno as foreign key. Write the SQL query to make these changes.
- change emp name with your name for empno=7034 in table NewEmp
- change emp name with your friend name for empno=7550 in table NewEmp.
- insert mob no and address in your record and of and your friend's records.
- delete the column address from the new table NewEmp.
- to delete the newly created table NewEmp.

2. Consider the following table named "Product", showing details of products being sold in a grocery shop.

PCode	PName	UPrice	Manufacturer
P01	Washing Powder	130	Surf
P02	Toothpaste	58	Colgate
P03	Soap	29	Lux
P04	Toothpaste	75	Pepsodent
P05	Soap	44	Dove
P06	Shampoo	275	Dove
P08	Toothpaste	44	Patanjali
P09	Soap	48	Hamam
P10	Washing Powder	90	Henko

Write SQL queries for the following.

- Create the table Product with appropriate data types and constraints.
  - Identify the primary key in Product.
  - List the Product Code, Product name and price in descending order of their product name. If PName is same, then display the data in ascending order of price.
  - Add a new column Discount to the table Product.
  - Calculate the value of the discount in the table Product as 10 per cent of the UPrice for all those products where the UPrice is more than 100, otherwise the discount will be 0.
  - Increase the price by 12 per cent for all the products manufactured by Dove.
  - Display the total number of products manufactured by each manufacturer.
3. Consider the following MOVIE table and write the SQL queries based on it.

MID	MovieName	Category	ReleaseDate	ProdCost	BusiCost
1	Hindi_Movie	Musical	4/23/2018	124500	130000
2	Tamil_Movie	Action	5/17/2016	112000	118000
3	English_Movie	Horror	8/6/2017	245000	360000
4	Bengali_Movie	Adventure	1/4/2017	72000	100000
5	Telugu_Movie	Action			100000
6	Punjabi_Movie	Comedy			30500

- Display all the information from the Movie table.
- List business done by the movies showing only MID, MovieName and Total\_Earning. Total\_Earning to be calculated as the sum of ProdCost and BusiCost.
- List the different categories of movies.

- d) Find the net profit of each movie showing its MID, MovieName and NetProfit. Net Profit is to be calculated as the difference between BussCost and ProdCost.
- e) List MID, MovieName and Cost for all movies with ProdCost greater than 10,000 and less than 1,00,000.
- f) List details of all movies which fall in the category of comedy or action.
- g) List details of all movies which have not been released yet.

4. Suppose your school management has decided to conduct cricket matches between students of Class XI and Class XII. Students of each class are asked to join any one of the four teams – Team Titan, Team Rockers, Team Magnet and Team Hurricane. During summer vacations, various matches will be conducted between these teams. Help your sports teacher to do the following:

1. Create a database “Sports”.

12. Create a table “TEAM” with following considerations:

- It should have a column TeamID for storing an integer value between 1 to 9, which refers to unique identification of a team.
- Each TeamID should have its associated name (TeamName), which should be a string of length not less than 10 characters.

13. Using table level constraint, make TeamID as the primary key.

14. Show the structure of the table TEAM using a SQL statement.

15. As per the preferences of the students four teams were formed as given below. Insert these four rows in TEAM table:

Row 1: (1, Team Titan)

Row 2: (2, Team Rockers)

Row 3: (3, Team Magnet)

Row 3: (4, Team Hurricane)

16. Show the contents of the table TEAM using a DML statement.

17. Now create another table MATCH\_DETAILS and insert data as shown below. Choose appropriate data types and constraints for each attribute.

**Table: MATCH\_DETAILS**

MatchID	MatchDate	FirstTeamID	SecondTeamID	FirstTeamScore	SecondTeamScore
M1	7/17/2022	1	2	90	86
M2	7/18/2022	3	4	45	48
M3	7/19/2022	1	3	78	56
M4	7/19/2022	2	4	56	67
M5	7/18/2022	1	4	32	87
M6	7/17/2022	2	3	67	51

## Session 3: Functions In SQL

There are various readily available functions in SQL that can be used in queries. It includes single row functions, multiple row functions, group records based on some criteria, and working on multiple tables using SQL.

A function is used to perform some particular tasks and it returns zero or more values as a result. Functions are useful while writing SQL queries also. Functions can be applied to work on single or multiple records (rows) of a table.

### 3.1 SQL functions

SQL functions are categorized as Single Row functions and Aggregate functions, depending on their application in one or multiple rows.

Single Row Functions are also known as Scalar functions. Single row functions are applied on a single value and return a single value. These are used in SELECT, WHERE, and ORDER BY clause. MATH, STRING and DATE functions are examples of single row functions.

Aggregate functions are also called Multiple Row functions. These functions work on a set of records as a whole and return a single value for each column of the records on which the function is applied. These are used with SELECT clause only. MAX ( ), MIN ( ), AVG ( ), SUM ( ), COUNT ( ) and COUNT (\*) are examples of multiple row fun.

To demonstrate the use of SQL function, Let us create database called CARSHOWROOM having the schema with four relations as shown in Figure 3.1.

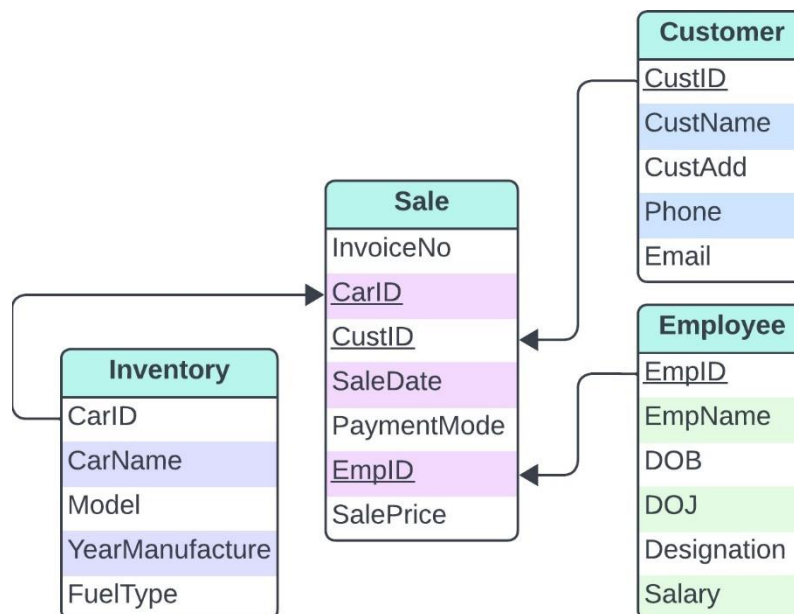


Fig 3.1: Car showroom database schema

**Inventory** – Stores Car id, Car Name, Price, Model, Year of manufacturing and fuel type for each car in inventory of the showroom.

**Table 3.1: Attribute specification of “Inventory” table**

Attribute	Data expected to be stored	Data type	Constraint
<u>CarID</u>	Alpha-Numeric value consisting of maximum 4 digits	Varchar (4)	Primary Key
CarName	Variant length string of maximum 20 characters	Varchar (20)	Not Null



Price	Numeric value consisting of car price.	Int	Not Null
Model	Variable length string of maximum 4 characters	Varchar (10)	Not Null
YearManufacturer	Variable length string of maximum 4 characters	Varchar (4)	Not Null
FuelType	Variable length string of max 10 characters	Varchar (10)	Not Null

**Customer** – Stores Customer id, name, address, phone number and email for each customer.

**Table 3.2: Attribute specification of “Customer” table**

Attribute	Data expected to be stored	Data type	Constraint
<u>CustID</u>	Alphanumeric value consisting of characters and digits, max 5 chars	Varchar (5)	Primary Key
CustName	Variable length string of max 30 characters	Varchar (30)	Not Null
CustAdd	Variable length string of max 50 characters	Varchar (50)	Not Null
Phone	Numeric value consisting of 10 digits	Char (10)	Not Null
Email	Variable length string of max 50 characters	Varchar (20)	Not Null

**Sale** – Stores the invoice number, car id, customer id, sale date, mode of payment, sales person’s employee id and selling price of the car sold,

**Table 3.3: Attribute specification of “Sale” table**

Attribute	Data expected to be stored	Data type	Constraint
<u>InvoiceNo</u>	Alpha-Numeric value consisting of Characters and digits, max 6 chars	Varchar (6)	Primary Key
CarID	Alpha-Numeric value consisting of maximum 4 digits	Varchar (4)	Foreign Key
CustID	Alpha-Numeric value consisting of Characters and digits, max 5 chars	Varchar (5)	Foreign Key
SaleDate	Date value	Date	Not Null
PaymentMode	Variant length string of max 20 characters	Varchar (20)	Not Null
EmpID	Alpha-Numeric value consisting of maximum 4 chars only	Varchar (4)	Foreign Key
SalePrice	Car price will be as Numeric Value	Int	Not Null

**Employee** – Stores employee id, name, date of birth, date of joining, designation and salary of each employee in the showroom.

**Table 3.4: Attribute specification of “Employee” table**

Attribute	Data expected to be stored	Data type	Constraint
<u>EmpID</u>	Alpha-Numeric value consisting of maximum 4 chars only	Varchar (4)	Primary Key
EmpName	String of max 20 characters	Varchar (20)	Not Null
DOB	Date value	Date	Not Null
DOJ	Date value	Date	Not Null
Designation	String of max 20 characters	Varchar (20)	Not Null
Salary	Numeric value	Int	Not Null

To proceed further, create database CARSHOWROOM and create all four tables as per the above specification.

Insert the records in tables *Inventory*, *Customer*, *Sale* and *Employee* using INSERT command. The records of these four relations can be viewed using the SELECT command.

Execute the following query to view the records of “*inventory*” table. After successful execution of the query, the records entered in the “*inventory*” table will be displayed.

```
mysql> SELECT * FROM inventory;
```

CarID	CarName	Price	Model	YearManufacturer	FuelType
B001	Baleno	567031	Sigma1.2	2019	Petrol
B002	Baleno	647858	Delta1.2	2018	Petrol
D001	Dzire	582613	LXI	2017	Petrol
D002	Dzire	673112	VXI	2018	Petrol
E001	EECO	355205	5 STR STD	2017	CNG
E002	EECO	654914	CARE	2018	CNG
S001	SWIFT	514000	LXI	2017	Petrol
S002	SWIFT	614000	VXI	2018	Petrol

```
8 rows in set (0.00 sec)
```

Execute the following query to view the records of “*customer*” table. After successful execution of the query, the records entered in the “*customer*” table will be displayed.

```
mysql> SELECT * FROM customer;
```

CustID	CustName	CustAdd	Phone	Email
C0001	Amit Saha	L-10, Pitampura	4564587852	amitsaha2@gmail.com
C0002	Rehnuma	J-12, SAKET	5527688761	rehnuma@hotmail.com
C0003	Charvi Nayyar	10/9, FF, Rohini	6811635425	charvi123@yahoo.com
C0004	Gurpreet	A-10/2, SF Mayur Vihar	3511056125	gur_singh@yahoo.com

```
4 rows in set (0.00 sec)
```

Execute the following query to view the records of “*sale*” table. After successful execution of the query, the records entered in the “*sale*” table will be displayed.

```
mysql> SELECT * FROM sale;
```

InvoiceNo	CarID	CustID	SaleDate	PaymentMode	EmpID	SalePrice	Commission
I00001	D001	C0001	2019-01-24	Credit Card	E004	613248	73589.76
I00002	S001	C0002	2018-12-12	Online	E001	590321	70838.52
I00003	S002	C0004	2019-01-25	Cheque	E010	604000	72480.00
I00004	D002	C0001	2018-10-15	Bank Finance	E007	659982	79197.84
I00005	E001	C0003	2018-12-20	Credit Card	E002	369310	44317.20
I00006	S002	C0002	2019-01-30	Bank Finance	E007	620214	74425.68

```
6 rows in set (0.00 sec)
```

Execute the following query to view the records of “*employee*” table. After successful execution of the query, the records entered in the “*employee*” table will be displayed.

```
mysql> SELECT * FROM employee;
```

EmpID	EmpName	DOB	DOJ	Designation	Salary
E001	Rushil	1994-07-10	2017-12-12	Salesman	25550
E002	Sanjay	1990-03-12	2016-06-05	Salesman	33100
E003	Zohar	1975-08-30	1999-01-08	Peon	20000
E004	Arpit	1989-06-06	2010-12-02	Salesman	39100
E006	Sanjucta	1985-11-03	2012-07-01	Receptionist	27350
E007	Mayank	1993-04-03	2017-01-01	Salesman	27352
E010	Rajkumar	1987-02-26	2013-10-23	Salesman	31111

```
7 rows in set (0.00 sec)
```

### 3.2 Single Row Functions

Figure 3.2 lists different single row functions under three categories — Numeric (Math), String, Date and Time.

- Math Functions accept numeric value as input and return a numeric value as a result.
- String Functions accept character value as input and return either character or numeric values as output.
- Date and Time functions accept date and time value as input and return numeric or string or Date and Time as output.

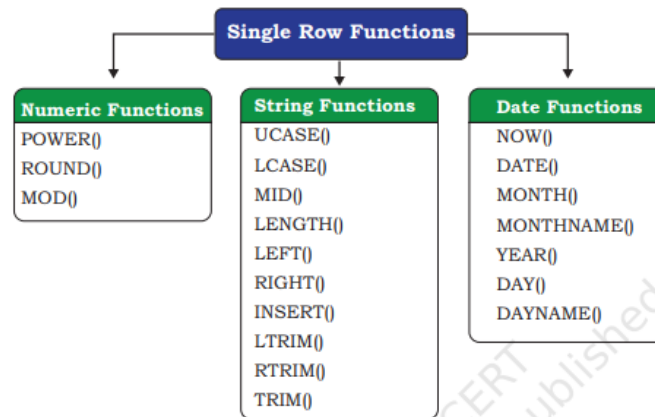


Fig 3.2: Categories of single row functions in SQL

#### C) 3.2.1 Math Functions

- d) Three commonly used numeric functions are POWER (), ROUND () and MOD (). Their usage along with syntax is given below.

1. POWER (X, Y) or POW (X, Y) – calculates X to the power Y

```
mysql> SELECT POWER (5,3);
+-----+
| POWER (5,3) |
+-----+
|          125 |
+-----+
1 row in set (0.00 sec)
```

2. ROUND (N, D) – Rounds off number N to D number of decimal places. If D=0, then it rounds off the number to the nearest integer.

```
mysql> SELECT ROUND (3412.567, 1);
+-----+
| ROUND (3412.567, 1) |
+-----+
|          3412.6 |
+-----+
1 row in set (0.00 sec)

mysql> SELECT ROUND (412.56);
+-----+
| ROUND (412.56) |
+-----+
|          413 |
+-----+
1 row in set (0.00 sec)
```

3. MOD (A, B) – Returns the remainder after dividing number A by number B.

```
mysql> SELECT MOD (44, 3);
+-----+
| MOD (44, 3) |
+-----+
|          2 |
+-----+
1 row in set (0.00 sec)
```

### Practical Activity 3.1 – Demonstrate to use math function ROUND

In order to increase sales, suppose the car dealer offers the customers to pay the total amount in 10 easy EMIs (equal monthly installments). Assume that EMIs are required in multiples of 10000. For that, the dealer wants to list the CarID and Price along with the following data from the Inventory table.

**Step 1.** Calculate GST as 12 per cent of Price and apply ROUND function to it. Execute the query to round off the GST to one decimal place and display the records with the fields CarID, CarName and GST.

```
mysql> SELECT CarID, CarName, ROUND (12/100*Price,1) "GST" FROM inventory;
+-----+-----+-----+
| CarID | CarName | GST |
+-----+-----+-----+
| B001  | Baleno  | 68043.7 |
| B002  | Baleno  | 77743.0 |
| D001  | Dzire   | 69913.6 |
| D002  | Dzire   | 80773.4 |
| E001  | EECO    | 42624.6 |
| E002  | EECO    | 78589.7 |
| S001  | SWIFT   | 61680.0 |
| S002  | SWIFT   | 73680.0 |
+-----+-----+-----+
8 rows in set (0.00 sec)
```

**Step 2.** Add a new column “FinalPrice” to the table “inventory”. Update the table “inventory” with “FinalPrice” as the sum of Price and 12 percent of the GST. Apply the ROUND function to round off the GST to one decimal place. Execute the following query to do this.

```
mysql> ALTER TABLE inventory ADD(FinalPrice Numeric(10,1));
Query OK, 0 rows affected (0.09 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> UPDATE inventory SET FinalPrice=Price+Round(Price*12/100,1);
Query OK, 8 rows affected (0.02 sec)
Rows matched: 8 Changed: 8 Warnings: 0
```

Display the values of “FinalPrice” for all the record by using the SELECT command.

```
mysql> SELECT * FROM inventory;
+-----+-----+-----+-----+-----+-----+-----+
| CarID | CarName | Price | Model | YearManufacturer | FuelType | FinalPrice |
+-----+-----+-----+-----+-----+-----+-----+
| B001  | Baleno  | 567031 | Sigma1.2 | 2019 | Petrol | 635074.7 |
| B002  | Baleno  | 647858 | Delta1.2 | 2018 | Petrol | 725601.0 |
| D001  | Dzire   | 582613 | LXI | 2017 | Petrol | 652526.6 |
| D002  | Dzire   | 673112 | VXI | 2018 | Petrol | 753885.4 |
| E001  | EECO    | 355205 | 5 STR STD | 2017 | CNG | 397829.6 |
| E002  | EECO    | 654914 | CARE | 2018 | CNG | 733503.7 |
| S001  | SWIFT   | 514000 | LXI | 2017 | Petrol | 575680.0 |
| S002  | SWIFT   | 614000 | VXI | 2018 | Petrol | 687680.0 |
+-----+-----+-----+-----+-----+-----+-----+
8 rows in set (0.00 sec)
```

**Step 3.** Calculate and display the amount to be paid each month in multiples of 1000, which is calculated after dividing the FinalPrice of the car into 10 installments. After dividing the amount into EMIs, find out the remaining amount to be paid immediately, by performing modular division. Use SELECT command to display the result. Execute the following query to do this.

```
mysql> SELECT CarId, CarName, FinalPrice,
-> Round((FinalPrice-Round(mod(FinalPrice,1000),0))/10,0) 'EMI Amt',
-> mod(FinalPrice,1000) 'Balance Amt' FROM inventory;
```

CarId	CarName	FinalPrice	EMI Amt	Balance Amt
B001	Baleno	635074.7	63500	74.7
B002	Baleno	725601.0	72500	601.0
D001	Dzire	652526.6	65200	526.6
D002	Dzire	753885.4	75300	885.4
E001	EECO	397829.6	39700	829.6
E002	EECO	733503.7	73300	503.7
S001	SWIFT	575680.0	57500	680.0
S002	SWIFT	687680.0	68700	680.0

```
8 rows in set (0.00 sec)
```

**Step 4.** Execute the following query to display the “InvoiceNo” and “Commission” value rounded off to zero decimal places.

```
mysql> Select InvoiceNo, round(Commission) from sale;
```

InvoiceNo	round(Commission)
I00001	73590
I00002	70839
I00003	72480
I00004	79198
I00005	44317
I00006	74426

```
6 rows in set (0.00 sec)
```

**Step 5.** Execute the following query to display the details of “sale” table where payment mode is credit card.

```
mysql> Select * from sale where PaymentMode='Credit Card';
```

InvoiceNo	CarID	CustID	SaleDate	PaymentMode	EmpID	SalePrice	Commission
I00001	D001	C0001	2019-01-24	Credit Card	E004	613248	73589.76
I00005	E001	C0003	2018-12-20	Credit Card	E002	369310	44317.20

```
2 rows in set (0.02 sec)
```

**Step 6.** Execute the query to add a new column “Commission” with total length of 7 with 2 decimal places to the “sale” table.

**Step 7.** Execute the query to calculate commission for sales agents as 12% of “SalePrice”.

```
mysql> ALTER TABLE sale ADD(Commission Numeric(7,2));
Query OK, 0 rows affected (0.05 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> UPDATE sale SET Commission=12 / 100 * SalePrice;
Query OK, 6 rows affected (0.09 sec)
Rows matched: 6 Changed: 6 Warnings: 0
```

**Step 8.** Execute the following query to insert the values to the newly added column “Commission” and then display all records of the “sale” table where Commission > 73000.

```
mysql> SELECT * FROM sale WHERE Commission > 73000;
+-----+-----+-----+-----+-----+-----+-----+
| InvoiceNo | CarID | CustID | SaleDate | PaymentMode | EmpID | SalePrice | Commission |
+-----+-----+-----+-----+-----+-----+-----+
| I00001   | D001  | C0001  | 2019-01-24 | Credit Card | E004  | 613248   | 73589.76   |
| I00004   | D002  | C0001  | 2018-10-15 | Bank Finance | E007  | 659982   | 79197.84   |
| I00006   | S002  | C0002  | 2019-01-30 | Bank Finance | E007  | 620214   | 74425.68   |
+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

**Step 9.** Execute the following query to display InvoiceNo, EmpID, SalePrice and Commission such that commission value is rounded off to 0.

```
mysql> SELECT InvoiceNo, EmpID, SalePrice, Round(Commission,0) FROM sale;
+-----+-----+-----+-----+
| InvoiceNo | EmpID | SalePrice | Round(Commission,0) |
+-----+-----+-----+-----+
| I00001   | E004  | 613248   | 73590               |
| I00002   | E001  | 590321   | 70839               |
| I00003   | E010  | 604000   | 72480               |
| I00004   | E007  | 659982   | 79198               |
| I00005   | E002  | 369310   | 44317               |
| I00006   | E007  | 620214   | 74426               |
+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

### 3.2.2 String Functions

String functions can perform various operations on alphanumeric data which are stored in a table. They can be used to change the case such as uppercase to lowercase or vice-versa, extract a substring, calculate the length of a string and so on. Some of the string functions with examples are given below.

1. UCASE (string) OR UPPER (string) – converts string into uppercase.

```
mysql> SELECT UCASE ('Vocational Education');
+-----+
| UCASE ('Vocational Education') |
+-----+
| VOCATIONAL EDUCATION           |
+-----+
1 row in set (0.00 sec)
```

```
mysql> SELECT UPPER ('Vocational Education');
+-----+
| UPPER ('Vocational Education') |
+-----+
| VOCATIONAL EDUCATION           |
+-----+
1 row in set (0.00 sec)
```

2. LCASE (string) OR LOWER (string) – converts string into lowercase.

```
mysql> SELECT LCASE ('Computer SCIENCE');
+-----+
| LCASE ('Computer SCIENCE') |
+-----+
| computer science           |
+-----+
1 row in set (0.00 sec)
```

```
mysql> SELECT LOWER ('Computer SCIENCE');
+-----+
| LOWER ('Computer SCIENCE') |
+-----+
| computer science           |
+-----+
1 row in set (0.00 sec)
```

3. MID (string, pos, n) OR SUBSTRING (string, pos, n) OR SUBSTR (string, pos, n) – Returns a substring of size n starting from the specified position (pos) of the string. If n is not specified, it

returns the substring from the position pos till end of the string.

```
mysql> SELECT MID('Vocational', 3, 4);
+-----+
| MID('Vocational', 3, 4) |
+-----+
| cati                      |
+-----+
1 row in set (0.00 sec)

mysql> SELECT MID('Vocational', 7);
+-----+
| MID('Vocational', 7) |
+-----+
| onal                  |
+-----+
1 row in set (0.00 sec)
```

4. LENGTH (string) – Return the number of characters in the specified string.

```
mysql> SELECT LENGTH ('Voc Edu');
+-----+
| LENGTH ('Voc Edu') |
+-----+
| 7 |
+-----+
1 row in set (0.00 sec)
```

5. LEFT (string, N) – Returns N number of characters from the left side of the string.

```
mysql> SELECT LEFT ('Computer', 4);
+-----+
| LEFT ('Computer', 4) |
+-----+
| Comp                  |
+-----+
1 row in set (0.00 sec)
```

6. RIGHT (string, N) – Returns N number of characters from the right side of the string.

```
mysql> SELECT RIGHT('Practical', 3);
+-----+
| RIGHT('Practical', 3) |
+-----+
| cal                    |
+-----+
1 row in set (0.00 sec)
```

7. INSTR (string, substring) – Returns the position of the first occurrence of the substring in the given string. Returns 0, if the substring is not present in the string.

```
mysql> SELECT INSTR ('Vocational', 'na');
+-----+
| INSTR ('Vocational', 'na') |
+-----+
| 8 |
+-----+
1 row in set (0.00 sec)
```

8. LTRIM (string) – Returns the given string after removing leading white space characters.

```
mysql> SELECT LENGTH(' AGRA'),LENGTH(LTRIM(' AGRA'));
```

LENGTH(' AGRA')	LENGTH(LTRIM(' AGRA'))
5	4

1 row in set (0.00 sec)

9. RTRIM (string) – Returns the given string after removing trailing white space characters.

```
mysql> SELECT LENGTH('PENCIL'), LENGTH (RTRIM ('PENCIL '));
```

LENGTH('PENCIL')	LENGTH (RTRIM ('PENCIL '))
6	6

1 row in set (0.00 sec)

10. TRIM (string) – Returns the given string after removing both leading and trailing white space characters.

```
mysql> SELECT LENGTH('MADAM '),LENGTH(TRIM('MADAM '));
```

LENGTH('MADAM ')	LENGTH(TRIM('MADAM '))
6	5

1 row in set (0.00 sec)

i.

### Practical Activity 3.2 – Demonstrate to use string function

Let us use *Customer* relation to understand the working of various string functions.

**Step 1.** Execute the following query to display customer name in lower case and customer email in upper case from “customer” table.

```
mysql> SELECT LOWER(CustName), UPPER(Email) FROM customer;
```

LOWER(CustName)	UPPER(Email)
amit saha	AMITSAHA2@GMAIL.COM
rehnuma	REHNUMA@HOTMAIL.COM
charvi nayyar	CHARVI123@YAHOO.COM
gurpreet	GUR_SINGH@YAHOO.COM

4 rows in set (0.00 sec)

**Step 2.** Execute the following query to display the length of email and part of the email from the email id before the character '@’.

```
mysql> SELECT LENGTH(Email), LEFT(Email, INSTR(Email, "@")-1) FROM customer;
```

LENGTH(Email)	LEFT(Email, INSTR(Email, "@")-1)
19	amitsaha2
19	rehnuma
19	charvi123
19	gur_singh

4 rows in set (0.00 sec)

The function INSTR will return the position of “@” in the email address. So, to print email id without “@” position -1 is used.

Let us assume that four-digit area code is reflected in the mobile number starting from position number 3. For example, 1851 is the area code of mobile number 9818511338.

**Step 3.** Execute the following query to display the area code of the customer living in Rohini.



```
mysql> SELECT MID(Phone,3,4) FROM customer WHERE CustAdd like '%Rohini%';
+-----+
| MID(Phone,3,4) |
+-----+
| 1163           |
+-----+
1 row in set (0.00 sec)
```

**Step 4.** Execute the following query to display emails after removing the domain name extension “.com” from emails of the customers.

```
mysql> SELECT TRIM('.com' from Email) FROM customer;
+-----+
| TRIM('.com' from Email) |
+-----+
| amitsaha2@gmail         |
| rehnuma@hotmail        |
| charvi123@yahoo         |
| gur_singh@yahoo         |
+-----+
4 rows in set (0.00 sec)
```

**Step 5.** Execute the following query to display details of all the customers having yahoo emails only.

```
mysql> SELECT * FROM customer WHERE Email LIKE '%yahoo%';
+-----+-----+-----+-----+-----+
| CustID | CustName      | CustAdd                | Phone      | Email                |
+-----+-----+-----+-----+-----+
| C0003  | Charvi Nayyar | 10/9, FF, Rohini      | 6811635425 | charvi123@yahoo.com |
| C0004  | Gurpreet     | A-10/2, SF Mayur Vihar | 3511056125 | gur_singh@yahoo.com |
+-----+-----+-----+-----+-----+
```

2 rows in set (0.00 sec)

Now let us use the table “*inventory*” from CARSHOWROOM database, write SQL queries for the following:

**Step 6.** Execute the following query to convert the “*CarMake*” to uppercase if its value starts with the letter ‘B’.

```
mysql> SELECT Upper(CarName) from inventory where Carname Like 'b%';
+-----+
| Upper(CarName) |
+-----+
| BALENO         |
| BALENO         |
+-----+
2 rows in set (0.00 sec)
```

**Step 7.** If the length of the car model is greater than 4 then Execute the following query to fetch the substring starting from position 3 till the end from attribute Model.

```
mysql> SELECT Model, SUBSTR(Model,3) FROM
-> inventory WHERE Length(Model)>4;
+-----+-----+
| Model      | SUBSTR(Model,3) |
+-----+-----+
| Signal.2   | gma1.2          |
| Delta1.2   | lta1.2          |
| 5 STR STD  | STR STD         |
+-----+-----+
3 rows in set (0.00 sec)
```

### 3.2.3 Date and Time Functions

There are various functions that are used to perform operations on date and time data. Some of the operations include displaying the current date, extracting each element of a date (day, month

and year), displaying day of the week and so on. Some of the date and time functions with examples are given below.

1. NOW() – It returns the current system date and time.

```
mysql> SELECT NOW();
+-----+
| NOW() |
+-----+
| 2022-03-09 10:04:07 |
+-----+
1 row in set (0.02 sec)
```

2. DATE() – It returns the date part from the given date/time expression.

```
mysql> SELECT DATE(NOW());
+-----+
| DATE(NOW()) |
+-----+
| 2022-03-09 |
+-----+
1 row in set (0.00 sec)
```

3. MONTH(date) – It returns the month in numeric form from the date.

```
mysql> SELECT MONTH('2022-03-09');
+-----+
| MONTH('2022-03-09') |
+-----+
| 3 |
+-----+
1 row in set (0.00 sec)
```

4. MONTHNAME(date) – It returns the month name from the specified date.

```
mysql> SELECT MONTHNAME(NOW());
+-----+
| MONTHNAME(NOW()) |
+-----+
| March |
+-----+
1 row in set (0.00 sec)
```

5. YEAR(date) – It returns the year from the date.

```
mysql> SELECT YEAR('2022-03-09');
+-----+
| YEAR('2022-03-09') |
+-----+
| 2022 |
+-----+
1 row in set (0.00 sec)
```

6. DAY(date) – It returns the day part from the date.

```
mysql> SELECT DAY('2022-03-09');
+-----+
| DAY('2022-03-09') |
+-----+
| 9 |
+-----+
1 row in set (0.00 sec)
```

7. DAYNAME(date) – It returns the name of the day from the date.

```
mysql> SELECT DAYNAME('2022-03-09');
+-----+
| DAYNAME('2022-03-09') |
+-----+
| Wednesday              |
+-----+
1 row in set (0.00 sec)
```

### Practical Activity 3.3 – Demonstrate to use DATE and Time function

Let us use the “emp” table of CARSHOWROOM database to illustrate the working of some of the date and time functions.

**Step 1.** Execute the following query to select the day, month number and year of joining of all employees.

```
mysql> SELECT DAY(DOJ), MONTH(DOJ), YEAR(DOJ) FROM employee;
+-----+-----+-----+
| DAY(DOJ) | MONTH(DOJ) | YEAR(DOJ) |
+-----+-----+-----+
| 12       | 12         | 2017      |
| 5        | 6          | 2016      |
| 8        | 1          | 1999      |
| 2        | 12         | 2010      |
| 1        | 7          | 2012      |
| 1        | 1          | 2017      |
| 23       | 10         | 2013      |
+-----+-----+-----+
7 rows in set (0.00 sec)
```

**Step 1.** Execute the following query to display the date in the format "Wednesday, 26, November, 1979", if the date of joining is not Sunday.

```
mysql> SELECT DAYNAME(DOJ), DAY(DOJ), MONTHNAME(DOJ), YEAR(DOJ) FROM
employee WHERE DAYNAME(DOJ) != 'Sunday';
+-----+-----+-----+-----+
| DAYNAME(DOJ) | DAY(DOJ) | MONTHNAME(DOJ) | YEAR(DOJ) |
+-----+-----+-----+-----+
| Tuesday      | 12       | December       | 2017      |
| Friday       | 8        | January        | 1999      |
| Thursday     | 2        | December       | 2010      |
| Wednesday    | 23       | October        | 2013      |
+-----+-----+-----+-----+
4 rows in set (0.01 sec)
```

**Step 2.** Execute the following query to list the Employee Name, date of birth and Salary for all employees whose salary is more than 25000, in “emp” table.

```
mysql> Select EmpName, DayName(DOB), Salary
-> FROM employee WHERE Salary>25000;
+-----+-----+-----+
| EmpName | DayName(DOB) | Salary |
+-----+-----+-----+
| Rushil  | Sunday       | 25550  |
| Sanjay  | Monday      | 33100  |
| Arpit   | Tuesday     | 39100  |
| Sanjucta | Sunday     | 27350  |
| Mayank  | Saturday    | 27352  |
| Rajkumar | Thursday    | 31111  |
+-----+-----+-----+
6 rows in set (0.00 sec)
```

**Step 3.** Execute the following query to list the invoice number, customer id and date of sale those payment are done using bank finance in “Sale” table.

```
mysql> Select InvoiceNo, CustId, SaleDate FROM sale
-> WHERE PaymentMode = 'Bank Finance';
+-----+-----+-----+
| InvoiceNo | CustId | SaleDate |
+-----+-----+-----+
| I00004   | C0001  | 2018-10-15 |
| I00006   | C0002  | 2019-01-30 |
+-----+-----+-----+
2 rows in set (0.00 sec)
```

**Step 4.** Execute the following query to list all the employee without peon whose salary is more than 30000 in “emp” table.

```
mysql> Select * FROM employee WHERE Salary>30000 AND Designation!='Peon';
+-----+-----+-----+-----+-----+-----+
| EmpID | EmpName | DOB       | DOJ       | Designation | Salary |
+-----+-----+-----+-----+-----+-----+
| E002  | Sanjay  | 1990-03-12 | 2016-06-05 | Salesman    | 33100 |
| E004  | Arpit   | 1989-06-06 | 2010-12-02 | Salesman    | 39100 |
| E010  | Rajkumar | 1987-02-26 | 2013-10-23 | Salesman    | 31111 |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

**Step 5.** Execute the following query to list all the records without LXI and VXI models in the table “inventory”.

```
mysql> SELECT * FROM inventory WHERE Model NOT IN ('LXI', 'VXI');
+-----+-----+-----+-----+-----+-----+-----+
| CarID | CarName | Price | Model      | YearManufacturer | FuelType | FinalPrice |
+-----+-----+-----+-----+-----+-----+-----+
| B001  | Baleno  | 567031 | Sigma1.2   | 2019              | Petrol   | 635074.7 |
| B002  | Baleno  | 647858 | Delta1.2   | 2018              | Petrol   | 725601.0 |
| E001  | EECO    | 355205 | 5 STR STD  | 2017              | CNG      | 397829.6 |
| E002  | EECO    | 654914 | CARE       | 2018              | CNG      | 733503.7 |
+-----+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

### 3.3 Aggregate Functions

In aggregate functions the column must be of numeric type. Some of the aggregate functions are given below.

1. MAX (column) – Returns the largest value from the specified column.

```
mysql> SELECT MAX(Price)FROM inventory;
+-----+
| MAX(Price) |
+-----+
|      673112 |
+-----+
1 row in set (0.02 sec)
```

2. MIN (column) – Returns the smallest value from the specified column.

```
mysql> SELECT MIN(Price)FROM inventory;
+-----+
| MIN(Price) |
+-----+
|      355205 |
+-----+
1 row in set (0.00 sec)
```

3. AVG (column) – Returns the average of the values in the specified column.

```
mysql> SELECT AVG(Price)FROM inventory;
+-----+
| AVG(Price) |
+-----+
| 576091.6250 |
+-----+
1 row in set (0.00 sec)
```

4. SUM (column) – Returns the sum of the values for the specified column.

```
mysql> SELECT SUM(Price) FROM inventory;
+-----+
| SUM(Price) |
+-----+
| 4608733 |
+-----+
1 row in set (0.00 sec)
```

5. COUNT (\*) – Returns number of records in a table. COUNT (\*) is used with WHERE clause to display the number of records that matches a particular criteria in the table.

```
mysql> SELECT COUNT(*)FROM inventory WHERE Model='VXI';
+-----+
| COUNT(*) |
+-----+
| 2 |
+-----+
1 row in set (0.00 sec)
```

### Practical Activity 3.4 – Demonstrate to use aggregate functions in SQL

Let us explore how can we use various aggregate functions in SQL statements to fulfill various requirements of real-world situations.

**Step 1.** Execute the following SQL query to display the total number of records from table “*inventory*” having a model as VXI.

```
mysql> SELECT COUNT(*)FROM inventory WHERE Model='VXI';
+-----+
| COUNT(*) |
+-----+
| 2 |
+-----+
1 row in set (0.00 sec)
```

**Step 2.** Execute the following SQL query to display the total number of different types of Models available from table “*inventory*”.

```
mysql> SELECT COUNT(DISTINCT(Model)) FROM inventory;
+-----+
| COUNT(DISTINCT(Model)) |
+-----+
| 6 |
+-----+
1 row in set (0.00 sec)
```

**Step 3.** Execute the following SQL query to display the average price of all the cars with Model LXI from table “*inventory*”.

```
mysql> SELECT AVG(Price) FROM inventory WHERE Model='LXI';
+-----+
| AVG(Price) |
+-----+
| 548306.5000 |
+-----+
1 row in set (0.00 sec)
```

### 3.3 GROUP BY CLAUSE IN SQL

Sometimes it may require to fetch a group of rows on the basis of common values in a column. GROUP BY clause is a special clause in SQL to do this. It groups the rows together that contains the same values in a specified column. The

aggregate functions (COUNT, MAX, MIN, AVG and SUM) can be used with GROUP BY clause. HAVING Clause in SQL is used to specify conditions on the rows with GROUP BY clause.

### Practical Activity 3.5 – Demonstrate to use GROUP BY and HAVING clause in SQL

Consider the “sale” table from the CARSHOWROOM database. Display the number of records in the “sale” table using the following SQL statement.

```
mysql> Select * from sale;
```

InvoiceNo	CarID	CustID	SaleDate	PaymentMode	EmpID	SalePrice	Commission
I00001	D001	C0001	2019-01-24	Credit Card	E004	613248	73589.76
I00002	S001	C0002	2018-12-12	Online	E001	590321	70838.52
I00003	S002	C0004	2019-01-25	Cheque	E010	604000	72480.00
I00004	D002	C0001	2018-10-15	Bank Finance	E007	659982	79197.84
I00005	E001	C0003	2018-12-20	Credit Card	E002	369310	44317.20
I00006	S002	C0002	2019-01-30	Bank Finance	E007	620214	74425.68

6 rows in set (0.00 sec)

In these records, it is observed that, the columns, CarID, CustID, SaleDate, PaymentMode, EmpID, SalePrice can have rows with the same values in it. So, GROUP BY clause can be used in these columns to find the number of records of a particular type (column), or to calculate the sum of the price of each car type.

**Step 1.** Execute the following SQL query to display the number of Cars purchased by each Customer from SALE table.

```
mysql> SELECT CustID, COUNT(*) 'Number of Cars' FROM sale GROUP BY CustID;
```

CustID	Number of Cars
C0001	2
C0002	2
C0003	1
C0004	1

4 rows in set (0.00 sec)

**Step 2.** Execute the following SQL query to display the Customer Id and number of cars purchased if the customer purchased more than 1 car from SALE table.

```
mysql> SELECT CustID, COUNT(*) FROM sale GROUP BY CustID HAVING Count(*)>1;
```

CustID	COUNT(*)
C0001	2
C0002	2

2 rows in set (0.02 sec)

**Step 3.** Execute the following SQL query to display the number of people in each category of payment mode from the table SALE.

```
mysql> SELECT PaymentMode, COUNT(PaymentMode) FROM sale GROUP BY Paymentmode ORDER BY Paymentmode;
```

PaymentMode	COUNT(PaymentMode)
Bank Finance	2
Cheque	1
Credit Card	2
Online	1

4 rows in set (0.00 sec)

**Step 4.** Execute the following SQL query to display the PaymentMode and number of payments made using that mode more than once.

```
mysql> SELECT PaymentMode, COUNT(PaymentMode) FROM sale GROUP BY
Paymentmode ORDER BY Paymentmode;
+-----+-----+
| PaymentMode | COUNT(PaymentMode) |
+-----+-----+
| Bank Finance | 2 |
| Cheque       | 1 |
| Credit Card  | 2 |
| OnLine       | 1 |
+-----+-----+
4 rows in set (0.00 sec)
```

### 3.4 OPERATIONS ON RELATIONS

It is possible to perform certain operations on relations like Union, Intersection and Set Difference to merge the tuples of two tables. These three operations are binary operations as they work upon two tables. Note here that these operations can only be applied if both the relations have the same number of attributes and corresponding attributes in both tables have the same domain.

#### 3.4.1 Union (U)

This operation is used to combine the selected rows of two tables at a time. If some rows are same in both the tables, then result of the Union operation will show those rows only once. Figure 3.3 shows union of two sets.

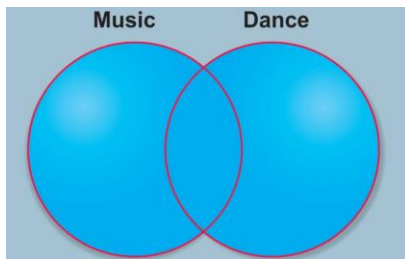


Fig 3.3: Union of two sets

Let us consider two relations DANCE and MUSIC shown in Tables 3.9 and 3.10 respectively.

**Table 3.9 DANCE**

Sno	Name	Class
1	Astha	7A
2	Pawani	6A
3	Mohit	7B
4	Vibhanshu	7A

Execute the following query to view the records of “dance” table. After successful execution of the query, the records entered in the “dance” table will be displayed.

```
mysql> SELECT * FROM dance;
+-----+-----+-----+
| Sno | Name   | Class |
+-----+-----+-----+
| 1   | Astha  | 7A    |
| 2   | Pawani | 6A    |
| 3   | Mohit  | 7B    |
| 4   | Vibhanshu | 7A    |
+-----+-----+-----+
4 rows in set (0.00 sec)
```

**Table 3.10 MUSIC**

Sno	Name	Class
1	Mahak	8A
2	Pawani	6A
3	Lavanya	7A
4	Vibhanshu	7A

5	Abhay	8A
---	-------	----

Execute the following query to view the records of “music” table. After successful execution of the query, the records entered in the “music” table will be displayed.

```
mysql> SELECT * FROM music;
+-----+-----+-----+
| Sno | Name | Class |
+-----+-----+-----+
| 1 | Mahak | 8A |
| 2 | Pawani | 6A |
| 3 | Lavanya | 7A |
| 4 | Vibhanshu | 7A |
| 5 | Abhay | 8A |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

**Step 1.** Execute the following SQL query to find the list of students participating in either of events by using UNION operation on relations **DANCE** and **MUSIC**. After execution it will display the union of DANCE and MUSIC relations.

```
mysql> SELECT * FROM dance UNION SELECT * from music;
+-----+-----+-----+
| Sno | Name | Class |
+-----+-----+-----+
| 1 | Astha | 7A |
| 2 | Pawani | 6A |
| 3 | Mohit | 7B |
| 4 | Vibhanshu | 7A |
| 1 | Mahak | 8A |
| 3 | Lavanya | 7A |
| 5 | Abhay | 8A |
+-----+-----+-----+
7 rows in set (0.01 sec)
```

### 3.4.2 Intersect ( $\cap$ )

Intersect operation is used to get the common tuples from two tables and is represented by symbol  $\cap$ . Figure 3.4 shows intersection of two sets.

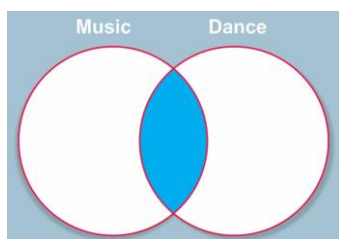


Fig 3.4: Intersection of two sets

Suppose, we have to display the list of students who are participating in both the events (DANCE and MUSIC), then intersection operation is to be applied on these two tables. The output of INTERSECT operation is shown in Table 3.11

**Table 3.11 DANCE  $\cap$  MUSIC**

Sno	Name	Class
2	Pawani	6A
4	Vibhanshu	7A

### 3.4.3 Minus (-)

This operation is used to get tuples/rows which are in the first table but not in the second table and the operation is represented by the symbol - (minus). Figure 3.5 shows difference operation between two sets.



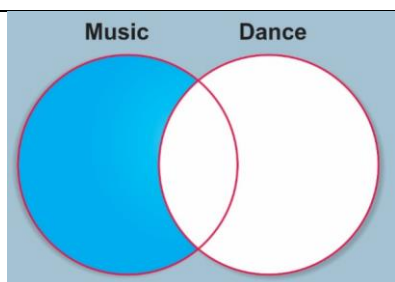


Fig 3.5: Difference of two sets

To find out the list of students who are only participating in MUSIC and not in DANCE event, use the MINUS operation. The output of MINUS operation is given in Table 3.12

**Table 3.12 DANCE – MUSIC**

Sno	Name	Class
1	Mahak	8A
3	Lavanya	7A
5	Abhay	8A

#### 3.4.4 Cartesian Product (×)

Cartesian product operation combines tuples from two relations. It results in all pairs of rows from the two input relations, regardless of whether or not they have the same values on common attributes. It is denoted as '×'.

The degree of the resulting relation is calculated as the sum of the degrees of both the relations under consideration. The cardinality of the resulting relation is calculated as the product of the cardinality of relations on which Cartesian product is applied. Let us use the relations DANCE and MUSIC to show the output of Cartesian product. Note that both relations are of degree 3. The cardinality of relations DANCE and MUSIC is 4 and 5 respectively. Applying Cartesian product on these two relations will result in a relation of degree 6 and cardinality 20, as shown in the output of the following query.

```
mysql> SELECT * FROM dance, music;
```

Sno	Name	Class	Sno	Name	Class
4	Vibhanshu	7A	1	Mahak	8A
3	Mohit	7B	1	Mahak	8A
2	Pawani	6A	1	Mahak	8A
1	Astha	7A	1	Mahak	8A
4	Vibhanshu	7A	2	Pawani	6A
3	Mohit	7B	2	Pawani	6A
2	Pawani	6A	2	Pawani	6A
1	Astha	7A	2	Pawani	6A
4	Vibhanshu	7A	3	Lavanya	7A
3	Mohit	7B	3	Lavanya	7A
2	Pawani	6A	3	Lavanya	7A
1	Astha	7A	3	Lavanya	7A
4	Vibhanshu	7A	4	Vibhanshu	7A
3	Mohit	7B	4	Vibhanshu	7A
2	Pawani	6A	4	Vibhanshu	7A
1	Astha	7A	4	Vibhanshu	7A
4	Vibhanshu	7A	5	Abhay	8A
3	Mohit	7B	5	Abhay	8A
2	Pawani	6A	5	Abhay	8A
1	Astha	7A	5	Abhay	8A

20 rows in set (0.00 sec)

### 3.5 USING TWO RELATIONS IN A QUERY

Till now we have written queries in SQL using a single relation only. Now let us see how to write queries using two relations.

#### 3.5.1 JOIN on two tables

JOIN operation combines tuples from two tables on specified conditions. This is unlike Cartesian product which make all possible combinations of tuples. While using the JOIN clause of SQL,

specify conditions on the related attributes of two tables within the FROM clause. Usually, such attribute is the *primary key* in one table and *foreign key* in another table.

Let us create two tables UNIFORM (UCode, UName, UColor) and COST (UCode, Size, Price) in the SchoolUniform database. “UCode” is *primary key* in table UNIFORM. “UCode” and “Size” is the *composite key* in table COST. Therefore, UCode is a common attribute between the two tables which can be used to fetch the common data from both tables. Define UCode as *foreign key* in the “Cost” table while creating this table. Enter the records in these tables as shown in Table 3.13 and 3.14.

**Table 3.13 Uniform table**

UCode	Uname	UColor
1	Shirt	White
2	Pant	Grey
3	Tie	Blue

**Table 3.14 Cost table**

UCode	Size	Price
1	L	580
1	M	500
2	L	890
1	M	810

### Practical Activity 3.6 – Demonstrate to join two tables in SQL

Let us consider two tables created, UNIFORM and COST to demonstrate the joining of two tables. The joining of two tables can be done in three different ways – using WHERE clause, JOIN clause and NATURAL JOIN clause

**Step 1.** Execute the following query to join the two tables using WHERE clause.

```
mysql> SELECT * FROM uniform U, cost C WHERE U.UCode = C.UCode;
+-----+-----+-----+-----+-----+
| UCode | Uname | UColor | UCode | Size | Price |
+-----+-----+-----+-----+-----+
|      1 | Shirt | White  |      1 | L   |  580 |
|      1 | Shirt | White  |      1 | M   |  500 |
|      2 | Pant  | Grey   |      2 | L   |  890 |
|      1 | Shirt | White  |      1 | M   |  810 |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

As the attribute “UCode” appears in both “uniform” and “cost” tables. Hence alias is used to remove ambiguity by specifying qualifier U with attribute UCode in SELECT and FROM clauses to indicate its scope.

**Step 2.** Execute the following query to join the two tables using JOIN clause.

```
mysql> SELECT * FROM uniform U JOIN cost C WHERE U.UCode = C.UCode;
+-----+-----+-----+-----+-----+
| UCode | Uname | UColor | UCode | Size | Price |
+-----+-----+-----+-----+-----+
|      1 | Shirt | White  |      1 | L   |  580 |
|      1 | Shirt | White  |      1 | M   |  500 |
|      2 | Pant  | Grey   |      2 | L   |  890 |
|      1 | Shirt | White  |      1 | M   |  810 |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

The output of the query is same as that of step 1. In this query the JOIN clause is used explicitly along with condition in FROM clause. Hence no condition is required in WHERE clause.

The output of queries in step 1 and 2 has a repetitive column UCode having exactly the same values. This redundant column provides no additional information. SQL provides the extension

of JOIN operation called as NATURAL JOIN, which works similar to JOIN clause in SQL to remove the redundant attribute. This operator can be used to join the contents of two tables if there is one common attribute in both the tables.

**Step 3.** Execute the following query to join the two tables using NATURAL JOIN clause.

```
mysql> SELECT * FROM uniform NATURAL JOIN cost;
+-----+-----+-----+-----+-----+
| UCode | Uname | UColor | Size | Price |
+-----+-----+-----+-----+-----+
| 1     | Shirt | White  | L    | 580   |
| 1     | Shirt | White  | M    | 500   |
| 2     | Pant  | Grey   | L    | 890   |
| 1     | Shirt | White  | M    | 810   |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

It is clear from the output that the result of this query is same as above in step 1 and 2, except that the attribute UCode appears only once.

It is important to note the following points while applying JOIN operations on two or more relations.

- If two tables are to be joined on equality condition on the common attribute, then one may use JOIN with ON clause or NATURAL JOIN in FROM clause. If three tables are to be joined on equality condition, then two JOIN or NATURAL JOIN are required.
- In general, N-1 joins are needed to combine N tables on equality condition.
- Any relational operators can be used with JOIN clause to combine tuples of two tables.

#### SUMMARY

- A Function is used to perform a particular task and return a value as a result.
- Single Row functions work on a single row of the table and return a single value.
- Multiple Row functions work on a set of records as a whole and return a single value. Examples include COUNT, MAX, MIN, AVG and SUM.
- GROUP BY function is used to group rows of a table that contain the same values in a specified column.
- Join is an operation which is used to combine rows from two or more tables based on one or more common fields between them.

## Check Your Progress

### A. Multiple choice questions

1. Which of the following is not an example of single row function (a) MATH (b) STRING (c) DATE (d) COUNT
2. Which of the following is not an example of multiple row function (a) MAX ( ) (b) MIN ( ) (c) STRING (d) COUNT (\*)
3. What is the functionality of SQL COUNT? (a) It returns the no of record of table (b) It returns the no of record of database (c) It returns the no of record of row (d) It returns the no of record of column
4. Date and Time functions accept date and time value as input and return output as (a) numeric (b) string (c) Date and Time (d) Any of the above
5. String Functions accept character value as input and return output as (a) either character or numeric values (b) string values (c) numeric values (d) character values
6. Which of the following is aggregate function in SQL (a) LEFT (b) AVG (c) JOIN (d) LEN
7. The SQL statement Select Round (47.956,-1) from Dual; (a) is illegal in SQL (b) prints a garbage value (c) 045.926 (d) prints 50

8. Which of the following SQL operation cannot be performed on relations (a) Union, (b) Intersection (c) Difference (d) Merge
9. Which of the following is used to join two tables on equality condition on the common attribute (a) JOIN with ON clause (b) NATURAL JOIN in FROM clause (c) Any of a or b (d) NATURAL JOIN
10. What will be the Cartesian product of the two relations having 4 rows and 3 columns for first relation and 3 rows and 4 columns in second relation. (a) degree 7 cardinality 12 (b) degree 6 cardinality 16 (c) degree 7 cardinality 16 (d) degree 9 cardinality 16

### B. Fill in the blanks

1. Single row functions are applied on a single \_\_\_\_\_ and return a single value.
2. Aggregate functions work on a \_\_\_\_\_ as a whole and return a single value.)
3. Math Functions accept numeric value as input and return a \_\_\_\_\_ value as a result.
4. MONTH (date) returns the month in \_\_\_\_\_ form from the date.
5. By default, the order by clause lists items in \_\_\_\_\_ order.
6. INSTR (string, substring) returns the position of the \_\_\_\_\_ of the substring in the given string.)
7. MID (string, pos, n) returns a substring of size \_\_\_ starting from the specified position \_\_\_\_\_ of the string. (n, pos).
8. LTRIM (string) returns the given string after removing \_\_\_\_\_ white space characters.
9. TRIM (string) returns the given string after removing both \_\_\_\_\_ and \_\_\_\_\_ white space characters.
10. The \_\_\_\_\_ operation is used to get common tuples from two tables.

### C. State True or False

1. Aggregate functions are also called Scalar functions.
2. A function always return a single value.
3. Functions can be applied to work on single or multiple records of a table.
4. INSTR (string, substring) returns 0, if the substring is not present in the string.
5. If n is not specified MID (string, pos, n), it returns the substring from the position 1 till end of the string.
6. RTRIM (string) returns the given string after removing leading white space characters.
7. NOW() returns the current system date and time.
8. Union operation eliminates the duplicate rows.
9. Cartesian product operation combines tuples from two relations.
10. Join statement is used to combine two tables on a specified condition.

### C. Short answer questions

1. Differentiate between single row functions and aggregate functions.
2. List the single row functions with example.
3. Differentiate between TRIM( ), LTRIM( ) and RTRIM( ) functions.
4. Demonstrate the use of LCASE( ) and UCASE( ) function with example.
5. List the date functions with example.
6. What is the difference between NOW( ) and DATE( ) function?
7. Demonstrate the difference between SUM( ) and AVG( ) function?
8. A table Student has 4 rows and 2 column and another table has 3 rows and 4 columns. How many rows and columns will be there if we obtain the Cartesian product of these two tables?
9. What will be the output of following SQL functions.
  - a) Select pow (3,2);
  - b) Select round (342.9234, 2);
  - c) Select length ('Vocational Education');

- d) Select year ('1978/08/17'), month ('1978/08/17'), day ('1978/08/17'), monthname ('1978/08/17');
- e) Select left ('Central', 3), right ('Institute', 4), mid ('Vocational', 3, 4), substr ('Education', 3);

10. Write the SQL functions to perform the following operations.

- a) To display the day like "Monday", "Tuesday", from the date when India got independence.
- b) To display the specified number of characters from a particular position of the given string.
- c) To display the name of the month in which you were born.
- d) To display your name in capital letters.

### Practical Exercise

Consider the following table named "Product", showing details of products being sold in a grocery shop.

PCode	PName	UPrice	Manufacturer
P01	Washing Powder	120	Surf
P02	Toothpaste	54	Colgate
P03	Soap	25	Lux
P04	Toothpaste	65	Pepsodent
P05	Soap	38	Dove
P06	Shampoo	245	Dov

#### A. Write SQL queries for the following:

- a) Create the table Product with appropriate data types and constraints.
- b) Identify the primary key in Product.
- c) List the Product Code, Product name and price in descending order of their product name. If PName is the same, then display the data in ascending order of price.
- d) Add a new column Discount to the table Product.
- e) Calculate the value of the discount in the table Product as 10 per cent of the UPrice for all those products where the UPrice is more than 100, otherwise the discount will be 0.
- f) Increase the price by 12 per cent for all the products manufactured by Dove.
- g) Display the total number of products manufactured by each manufacturer.

#### B. Write the output(s) produced by executing the following queries on the basis of the information given above in the table Product:

- h) SELECT PName, Average (UPrice) FROM Product GROUP BY Pname;
- i) SELECT DISTINCT Manufacturer FROM Product;
- j) SELECT COUNT (DISTINCT PName) FROM Product;
- k) SELECT PName, MAX(UPrice), MIN(UPrice) FROM Product GROUP BY PName;

**Module 2****Advanced Python  
Programming****Module Overview**

In Grade 11, you have learned the basic data structures – *list, set, tuples, and dictionary*. Each of the data structures is unique in its own way. A data structure defines a mechanism to store, organize and access data along with operations that can be efficiently performed on the data.

An Exception is an error that happens during the execution of a program. It is necessary to tackle the exception to prevent the program from getting crashed. File handling in Python is a fundamental skill for developers, enabling them to manage data effectively, perform data processing tasks, and work with various data sources. In Python, the lists data structure serve the purpose of arrays, but they are slow to process.

NumPy aims to provide an array object that is much faster than traditional Python lists. NumPy stands for Numerical Python is a Python library used for working with arrays. Pandas Series is similar to one-dimensional NumPy array consists of an array of data (values), and an array of labels (indices). It has additional functionality that allows values in the Series to be indexed using labels. A NumPy array does not have the flexibility to do this.

The Data visualisation in the form of charts, graphs, animation, and maps are very easy and simple to understand the trends, outliers, and patterns in data. Data visualization techniques for such big data are very important for the purpose of analysis of data.

In Python it is important to understand the database connectivity for software development. It is possible to connect with the database application to develop applications. It is required to use libraries that provide various connectivity functionalities.

In this unit the advanced topics of implementation of data structure using Stack & Queue, Exception Handling, File Handling, Numpy Array, Pandas Series, Graphical Representation using Matplotlib, and Database Connectivity with MySQL are covered.

**Learning Outcomes**

After completing this module, you will be able to:

- Code and execute the programs to execute Stack and Queue data structure in Python
- Code and execute the programs to handle exception in Python
- Code and execute the programs of file handling in Python
- Code and execute the programs to create and use NumPy array in Python
- Code and execute the programs to use Pandas and Series in Python
- Code and execute the programs to demonstrate Graphical Representation using Matplotlib
- Code and execute the programs to establish database connectivity with MySQL

**Module Structure**

Session 1. Implementing Data Structure using Stack & Queue

Session 2. Exception Handling in Python

Session 3. File Handling in Python

Session 4. Numpy Array

Session 5. Pandas and Series in Python

Session 6. Graphical Representation using MatpotLib

Session 7. Database Connectivity with MySQL

## Session 1. Implementing Data Structure using Stack & Queue

In Python different data types are used to handle the values. String, List, Set, Tuple, are the sequence data types that can be used to represent collection of elements either of the same type or different types. Multiple data elements are grouped in a particular way for fast accessibility and efficient storage of data. Thus Python uses different data types for storing data values. Such grouping is referred as data structure. A data structure defines a mechanism to store, organize and access data along with operations (processing) that can be efficiently performed on the data.

For example, string data structure has a sequence of elements where each element is a character. List is a sequence data structure in which each element may be of different types. It is possible to apply different operations like reversal, slicing, counting of elements on list and string. Data structure organizes multiple elements so that certain operations can be performed on each element as well as on the collective data unit.

Stack and Queue are two other popular data structures used in programming. It is not directly available in Python, but it is important to learn these concepts as they are extensively used in programming languages.

### 1.1 APPLICATIONS OF STACK

When the books are kept one after another it forms the stack of books as shown in the Figure 1.1. Any book to be placed on these books is always added on the top of the books. In the same way the top most book can be removed first. It is not possible to add or remove the book stored in the middle or bottom. Addition or removal of the book is possible from the top end only by following the LIFO (Last-In-First-Out) principle. This arrangement of elements in a linear order is called a stack. The element which was inserted last (the most recent element) will be the first one to be taken out from the stack.

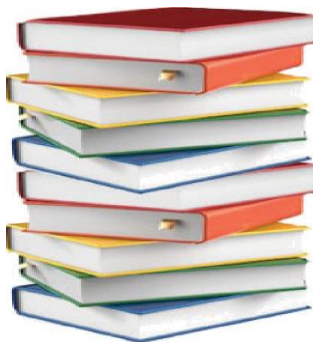


Fig. 1.1 Stack of books

In many applications stack can be used during programming. Some of the applications where stack is used are given here.

In the reversal of a string, the characters are traversed in the reverse order from last character to first character. This is very easily done by putting the characters of a string in a stack.

In editing the text or image the functions such as redo/undo has to be programmed. The most recent operations can be unto/ redo using this button. The stack can be usefull to do the programming of this type of operations.

In web browsing, the web pages are accessed sequentially. The stacked pages are stored in the browser history. The history of pages browsed is maintained as stack. To access the previous pages the most recently pages can be viewed first.

## 1.2 OPERATIONS ON STACK

As stack implements LIFO arrangement where elements are added and deleted from the stack at one end only. The end from which elements are added or deleted is called TOP of the stack. Two main operations performed on the stack are PUSH and POP. PUSH is performed to add an element in the stack and POP is performed to remove an element from the stack. These operations can be performed using programming by using Python.

### 1.2.1 PUSH and POP Operations

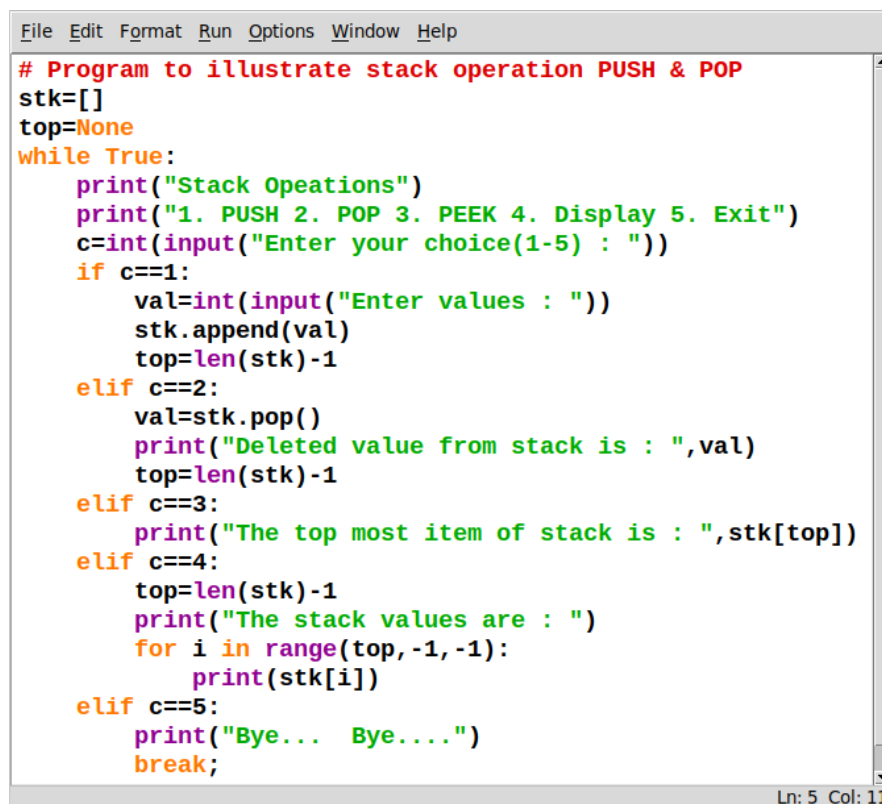
**PUSH** – This operation is used to add a new element at the TOP of the stack. It is an insertion operation. The elements can be added to a stack until it is full. Addition an element when the stack is full results into “*stack overflow*” error.

**POP** – This operation is used to remove the top most element of the stack. It is a deletion operation. The elements can be deleted from a stack until it is empty. Deleting an element from an empty stack results in “*stack underflow*” error.

Following programme illustrates the stack operation in Python.

### 1.3 IMPLEMENTATION OF STACK IN PYTHON

The various operations such as inserting an element in the stack (PUSH) and deleting an element from the stack (POP) can be implemented using Python programming. The following programme is coded to illustrate these operations.



```

File Edit Format Run Options Window Help
# Program to illustrate stack operation PUSH & POP
stk=[]
top=None
while True:
    print("Stack Operations")
    print("1. PUSH 2. POP 3. PEEK 4. Display 5. Exit")
    c=int(input("Enter your choice(1-5) : "))
    if c==1:
        val=int(input("Enter values : "))
        stk.append(val)
        top=len(stk)-1
    elif c==2:
        val=stk.pop()
        print("Deleted value from stack is : ",val)
        top=len(stk)-1
    elif c==3:
        print("The top most item of stack is : ",stk[top])
    elif c==4:
        top=len(stk)-1
        print("The stack values are : ")
        for i in range(top, -1, -1):
            print(stk[i])
    elif c==5:
        print("Bye... Bye....")
        break;
Ln: 5 Col: 11

```

After executing the above programme you can perform the stack operations.



```

File Edit Shell Debug Options Window Help
Stack Operations
1. PUSH 2. POP 3. PEEK 4. Display 5. Exit
Enter your choice(1-5) : 1
Enter values : 56
Stack Operations
1. PUSH 2. POP 3. PEEK 4. Display 5. Exit
Enter your choice(1-5) : 1
Enter values : 87
Stack Operations
1. PUSH 2. POP 3. PEEK 4. Display 5. Exit
Enter your choice(1-5) : 1
Enter values : 34
Stack Operations
1. PUSH 2. POP 3. PEEK 4. Display 5. Exit
Enter your choice(1-5) : 2
Deleted value from stack is : 34
Stack Operations
1. PUSH 2. POP 3. PEEK 4. Display 5. Exit
Enter your choice(1-5) : 4
The stack values are :
87
56
Stack Operations
1. PUSH 2. POP 3. PEEK 4. Display 5. Exit
Enter your choice(1-5) : 3
The top most item of stack is : 87
Stack Operations
1. PUSH 2. POP 3. PEEK 4. Display 5. Exit
Enter your choice(1-5) : 5
Bye... Bye....
>>>
Ln: 35 Col: 0

```

#### 1.4 QUEUE

As we have seen the Stack data structure works on Last-InFirst-Out (LIFO) principle. Queue is an another data structure which works on First-In-First-Out (FIFO) principle. Queue is an ordered linear list of elements, having different ends for adding and removing elements in it.

In our everyday life customers forming a queue at the cash counter in a bank and petrol pump, vehicles queued at fuel pumps. So whenever there is queue the first standing in the queue can exit first. Thus it follows the principle of First In First Out (FIFO). Queue is an arrangement in which new elements always get added at one end, usually called the REAR, and elements always get removed from the other end, usually called the FRONT of the queue. REAR is also known as TAIL and FRONT as HEAD of a queue.

##### 1.1.1 OPERATIONS ON QUEUE

The data structure queue supports the following operations.

**ENQUEUE** – Is used to insert a new element to the queue at the rear end. The element can be inserted in the queue from the rear end till there is space to add the elements. When there is no space left to insert an elements beyond capacity of the queue will result in an exception error “*Overflow*”.

**DEQUEUE** – Is used to remove one element at a time from the front of the queue. The element can be deleted from a queue until it is empty. Trying to delete an element from an empty queue will result in exception error “Underflow”.

To perform enqueue and dequeue efficiently on a queue, following operations are also required.

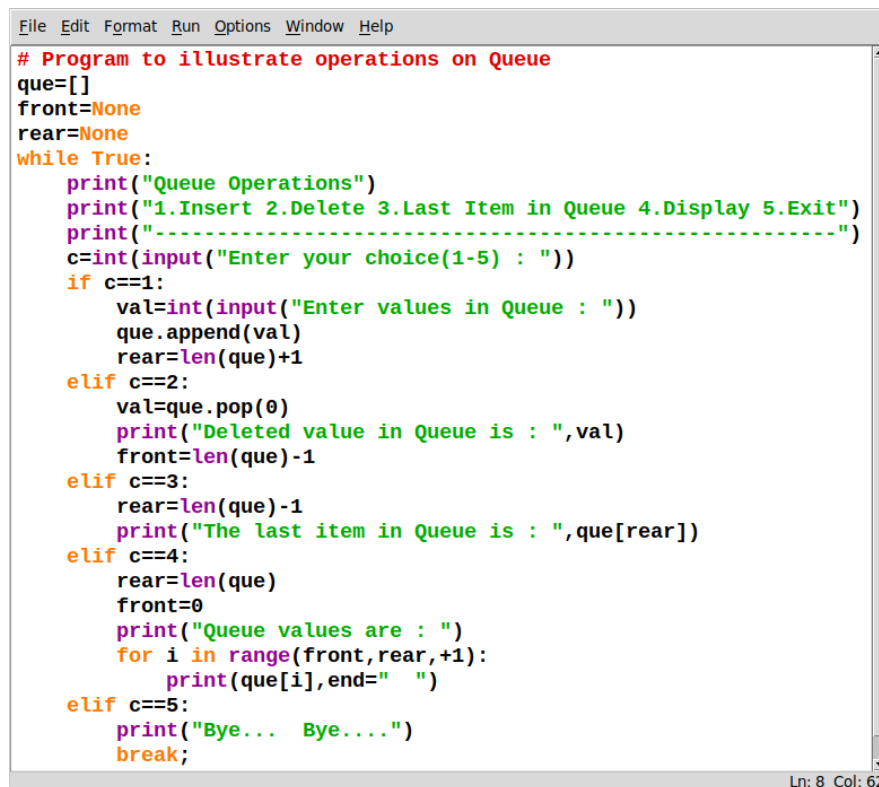
**IS EMPTY** – used to check whether the queue has any element or not, so as to avoid Underflow exception while performing dequeue operation.

**PEEK** – used to view elements at the front of the queue, without removing it from the queue.

**IS FULL** – used to check whether any more elements can be added to the queue or not, to avoid Overflow exceptions while performing enqueue operation.

### 1.3 IMPLEMENTATION OF QUEUE IN PYTHON

Following program illustrates the various operations performed on Queue.



```

File Edit Format Run Options Window Help
# Program to illustrate operations on Queue
que=[]
front=None
rear=None
while True:
    print("Queue Operations")
    print("1.Insert 2.Delete 3.Last Item in Queue 4.Display 5.Exit")
    print("-----")
    c=int(input("Enter your choice(1-5) : "))
    if c==1:
        val=int(input("Enter values in Queue : "))
        que.append(val)
        rear=len(que)+1
    elif c==2:
        val=que.pop(0)
        print("Deleted value in Queue is : ",val)
        front=len(que)-1
    elif c==3:
        rear=len(que)-1
        print("The last item in Queue is : ",que[rear])
    elif c==4:
        rear=len(que)
        front=0
        print("Queue values are : ")
        for i in range(front, rear, +1):
            print(que[i],end=" ")
    elif c==5:
        print("Bye... Bye....")
        break;
Ln: 8 Col: 62

```

```

File Edit Shell Debug Options Window Help
Queue Operations
1.Insert 2.Delete 3.Last Item in Queue 4.Display 5.Exit
-----
Enter your choice(1-5) : 1
Enter values in Queue : 11
Queue Operations
1.Insert 2.Delete 3.Last Item in Queue 4.Display 5.Exit
-----
Enter your choice(1-5) : 1
Enter values in Queue : 22
Queue Operations
1.Insert 2.Delete 3.Last Item in Queue 4.Display 5.Exit
-----
Enter your choice(1-5) : 1
Enter values in Queue : 33
Queue Operations
1.Insert 2.Delete 3.Last Item in Queue 4.Display 5.Exit
-----
Enter your choice(1-5) : 2
Deleted value in Queue is : 11
Queue Operations
1.Insert 2.Delete 3.Last Item in Queue 4.Display 5.Exit
-----
Enter your choice(1-5) : 4
Queue values are :
22 33 Queue Operations
1.Insert 2.Delete 3.Last Item in Queue 4.Display 5.Exit
-----
Enter your choice(1-5) : 3
The last item in Queue is : 33
Queue Operations
1.Insert 2.Delete 3.Last Item in Queue 4.Display 5.Exit
-----
Enter your choice(1-5) : 5
Bye... Bye....
>>>
Ln: 40 Col: 0

```

## SUMMARY

- Stack is a data structure in which insertion and deletion is done from one end only, usually referred to as TOP.
- Stack follows LIFO principle using which an element inserted in the last will be the first one to be out.
- PUSH and POP are two basic operations performed on a stack for insertion and deletion of elements, respectively.
- Trying to pop an element from an empty stack results into a special condition underflow.
- In Python, list is used for implementing a stack and its built-in-functions. Push and Pop are used for insertion and deletion. Hence, explicit declaration of TOP is not needed.
- Stack is commonly used data structure to convert an Infix expression into equivalent Prefix/Postfix notation.
- While conversion of an Infix notation to its equivalent Prefix/Postfix notation, only operators are PUSHed onto the Stack.
- Queue is an ordered linear data structure, following FIFO strategy.
- Front and Rear are used to indicate beginning and end of queue.
- In Python, the use of predefined methods takes care of Front and Rear.
- Insertion in a queue happens at the rear end. Deletion happens at the front.
- Insertion operation is known as enqueue and deletion operation is known as dequeue.
- To support enqueue and dequeue operations, is Empty, is full and peek operations are used.

## Check Your Progress

### A. Multiple Choice Questions

1. Process of inserting an element in stack is called \_\_\_\_\_ (a) Create (b) Push (c) Evaluation (d) Pop
2. Process of removing an element from stack is called \_\_\_\_\_ (a) Create (b) Push (c) Evaluation (d) Pop
3. Stack data structure is \_\_\_\_\_. (a) LILO (b) FIFO (c) LIFO (d) None of these
4. A queue is a \_\_\_\_\_ ? (a) FIFO (First In First Out) (b) LIFO (Last In First Out) list (c) Ordered array (d) Linear tree
5. A linear list of elements in which deletion can be done from one end (front) and insertion can take place only at the other end (rear) is known as \_\_\_\_\_ (a) Queue (b) Stack (c) Tree (d) Linked list

### B. State whether True or False

1. Stack is a linear data structure.
2. Stack does not follow LIFO rule.
3. PUSH operation may result into underflow condition.
4. Front and Rear are used to indicate beginning and end of queue.
5. Insertion operation is known as dequeue and deletion operation is known as enqueue.

### C. Fill in the blanks

1. Stack follows \_\_\_\_\_ principle using which an element inserted in the last will be the first one to be out.
2. \_\_\_\_\_ is a linear list of elements in which insertion and deletion takes place from different ends.
3. Operations on a queue are performed in \_\_\_\_\_ order.
4. Insertion in a queue is called \_\_\_\_\_ and deletion in a queue is called \_\_\_\_\_
5. Deletion of elements is performed from \_\_\_\_\_ end of the queue.

### D. Answer the following questions in short

1. What is stack and write name of three application of stack.
2. What is the difference between stack and queue.
3. What is queue and explain its application.
4. What is performed different type of operation in queue.
5. Write short note on these functions: (a) append( ) (b) pop( ) (c) len( ) (d) range( )

### Practical Exercise

1. Write a python program for the following.
2. Push( ) operation to insert data value in stack.
3. Pop( ) operation to remove data value from stack.
4. Display( ) operation to print values in the stack.
5. Enqueue( ) operation to insert data into queue.
6. Dequeue( ) operation to remove data from queue.
7. Display( ) operation display remaining value from the queue.

## Session 2: Exception Handling in Python

Sometimes a Python program does not execute or generates unexpected output or behaves abnormally. This happens due to presence of syntax errors, runtime errors or logical errors in the code. In Python, exceptions are errors that get triggered automatically. However, exceptions can be forcefully triggered and handled through program code. In this chapter, you will learn about exception handling in Python.

### 2.1 Syntax Errors

Syntax errors are also known as parsing errors, detected when the program is not coded according to syntax defined in that programming language. When syntax error encountered, the interpreter does not execute the program unless the errors are rectified. Python displays the syntax error with small description about the error in shell mode as shown in the Figure 2.1.

```

IDLE Shell 3.10.12
File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> m=10
>>> if m>20:
...     print "Good Job"
...
SyntaxError: Missing parentheses in call to 'print'. Did you mean print(...)?
>>>
Ln: 8 Col: 0

```

**Figure 2.1: A syntax error displayed in Python shell mode**

Syntax error is reported by the Python interpreter giving a brief explanation about the error and a suggestion to rectify it. Similarly in script mode the syntax error is displayed with its description through dialog box in script mode as shown in Figure 2.2.

```

File Edit Format Run Options Window Help
Python 3.10.12 (main, Jun 11 2
Type "help", "copyright", "cre
def tests():
    m=20
    if m>20:
        print "Good Job"

```

SyntaxError

leading zeros in decimal integer literals are not permitted; use an 0o prefix for octal integers

OK

**Fig. 2.2: A syntax error displayed in Python script mode**

### 2.2 Exceptions

Syntactically correct statement or expression may also generate error during its execution. These errors disrupt the normal execution of the program and are called exceptions. Such types of errors occurs when we are “Trying to open a file that does not exist” or “Division by zero”.

An exception is a Python object that represents an error. An exception is raised, when error occurs during the execution of program. In such cases an exception needs to be handled by the programmer so that the program does not terminate abnormally. Therefore, while coding a program, programmer may anticipate such erroneous situations that may arise during its execution and can address them by including appropriate code to handle that exception. The

syntax errors of “**Missing parentheses**” as shown in above examples are also comes under exception. But these errors can be handled by correcting the syntax, while other exceptions which are generated even the with the correct syntax need to be tackled logically.

### 2.3 Built-in Exceptions

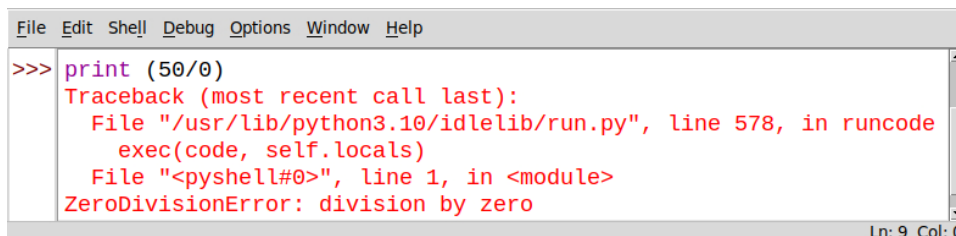
These are commonly occurring exceptions that are usually defined in the compiler or interpreter. These are called as built-in exceptions. Python’s standard library is an extensive collection of built-in exceptions that deals with the commonly occurring errors (exceptions) by providing the standardized solutions for such errors. When such errors occurs an appropriate exception handler code is executed which displays the raised exception name and the reason for the same. A programmer has to take appropriate action to handle it. Some of the commonly occurring built-in exceptions that can be raised in Python are stated in Table 2.1.

Table 2.1 Built-in exceptions in Python

SN	Built-in Exception	Raised when...
1	SyntaxError	there is an syntax error in the code.
2	ValueError	built-in method or operation receives an argument that has the right data type but mismatched or inappropriate values.
3	IOError	the file specified in a program statement cannot be opened.
4	KeyboardInterrupt	the user accidentally hits the Delete or Esc key while executing a program due to which the normal flow of the program is interrupted.
5	ImportError	when the requested module definition is not found.
6	EOFError	End of file condition is reached without reading any data by input ().
7	ZeroDivisionError	the denominator in a division operation is zero.
8	IndexError	the index or subscript in a sequence is out of range.
9	NameError	a local or global variable name is not defined.
10	IndentationError	there is incorrect indentation in the program code.
11	TypeError	operator is supplied with a value of incorrect data type.
12	OverFlowError	Result of calculation exceeds the maximum limit for numeric data type.

Some of the built-in exceptions viz, ZeroDivisionError, NameError, and TypeError raised by the Python interpreter are shown below.

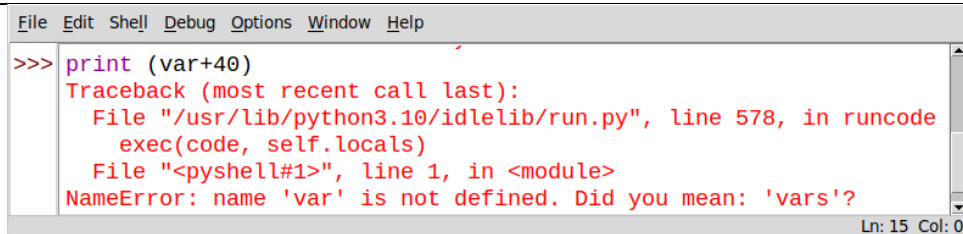
**ZeroDivisionError** – It is raised when the denominator in a division operation is zero as shown in Figure 2.3.



```
File Edit Shell Debug Options Window Help
>>> print (50/0)
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
    exec(code, self.locals)
  File "<pyshell#0>", line 1, in <module>
ZeroDivisionError: division by zero
Ln: 9 Col: 0
```

Fig. 2.3: ZeroDivisionError

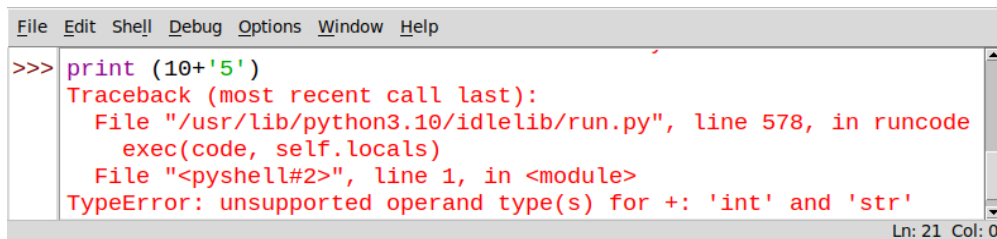
**NameError** – It is raised when a local or global variable name is not defined as shown in Figure 2.4.



```
File Edit Shell Debug Options Window Help
>>> print (var+40)
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
    exec(code, self.locals)
  File "<pyshell#1>", line 1, in <module>
NameError: name 'var' is not defined. Did you mean: 'vars'?
Ln: 15 Col: 0
```

**Fig. 2.4: NameError**

**TypeError** – It is raised when an operator is supplied with a value of incorrect data type as shown in Figure 2.5.



```
File Edit Shell Debug Options Window Help
>>> print (10+'5')
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
    exec(code, self.locals)
  File "<pyshell#2>", line 1, in <module>
TypeError: unsupported operand type(s) for +: 'int' and 'str'
Ln: 21 Col: 0
```

**Fig. 2.5: TypeError**

A programmer can also create custom exceptions to suit the requirements. These are called user-defined exceptions.

## 2.4 Raising Exceptions

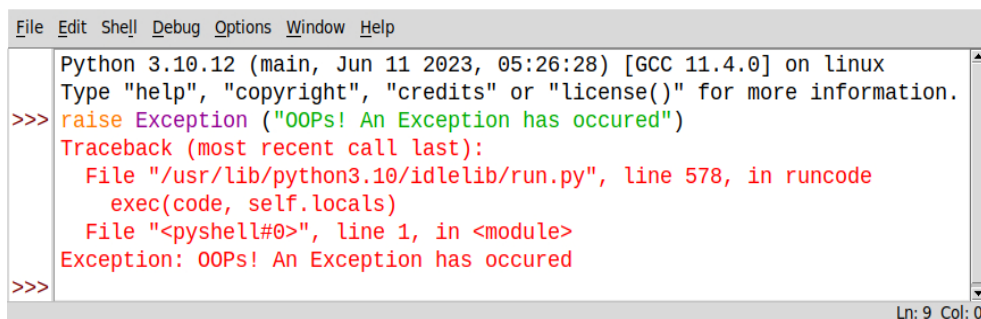
Each time when an error is detected in a program, the Python interpreter raises (throws) an exception. Exception handlers are designed to execute when a specific exception is raised. Programmer can also forcefully raise exceptions in a program using the raise and assert statements. Once an exception is raised, no further statements are executed in the current block of code. So, raising an exception involves interrupting the normal flow execution of program and jumping to that part of the program (exception handler code) which is written to handle such exceptional situations.

### 2.4.1 The raise Statement

The raise statement can be used to throw an exception. The syntax of raise statement is:

**raise exception-name[(optional argument)]**

The argument is generally a string that is displayed when the exception is raised. For example, when an exception is raised the message **"OOPS : An Exception has occurred"** is displayed along with a brief description of the error as shown in Figure 2.6. In addition to the error message, Python also displays a stack Traceback. This is a structured block of text that contains information about the sequence of function calls that have been made in the branch of execution of code in which the exception was raised as shown in Figure 2.6.

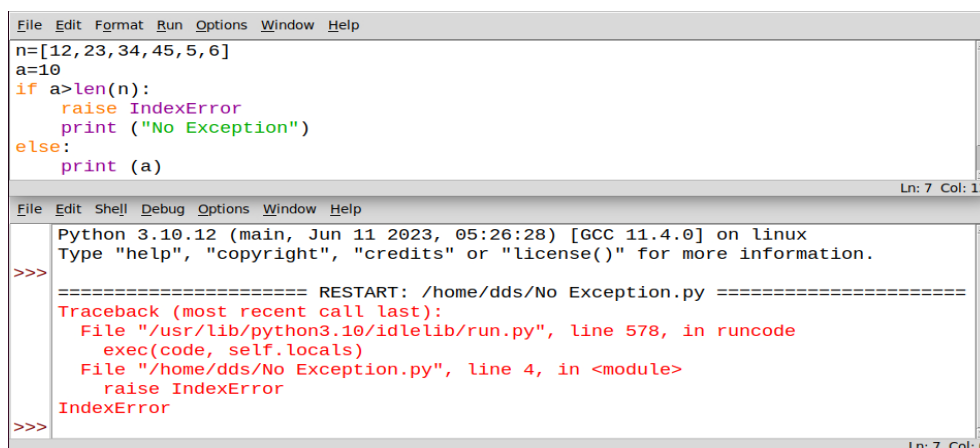


```
File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> raise Exception ("OOPS! An Exception has occurred")
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
    exec(code, self.locals)
  File "<pyshell#0>", line 1, in <module>
Exception: OOPS! An Exception has occurred
>>>
Ln: 9 Col: 0
```

**Fig. 2.6: raise Exception**

**Note:** In this case, the user has only raised the exception but not displayed any error message explicitly.

In the code shown in Figure 2.7, since the value of variable length is greater than the length of the list numbers, an `IndexError` exception will be raised. The statement following the `raise` statement will not be executed. So the message **"NO EXCEPTION"** will not be displayed in this case.



```
File Edit Format Run Options Window Help
n=[12,23,34,45,5,6]
a=10
if a>len(n):
    raise IndexError
    print ("No Exception")
else:
    print (a)
Ln: 7 Col: 13

File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/dds/No Exception.py =====
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
    exec(code, self.locals)
  File "/home/dds/No Exception.py", line 4, in <module>
    raise IndexError
IndexError
>>>
Ln: 7 Col: 0
```

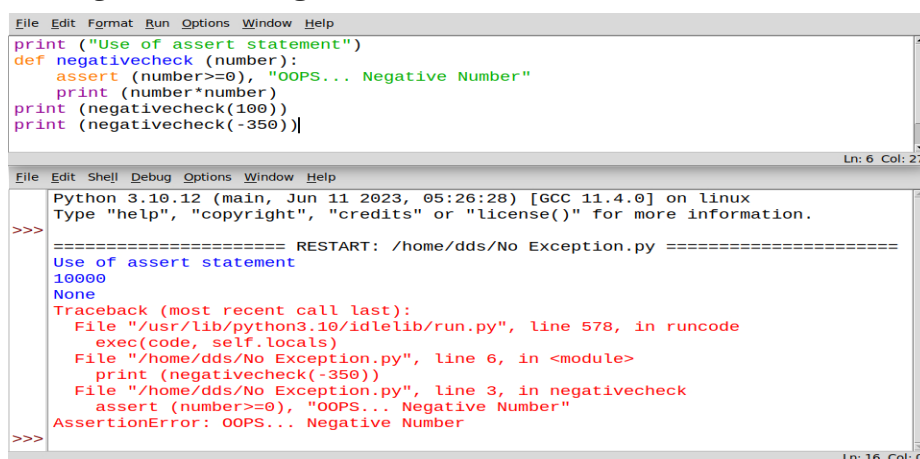
**Fig. 2.7: Use of raise statement with built-in exception**

### 2.4.2 The assert Statement

An `assert` statement in Python is used to test an expression in the program code. If the result after testing comes false, then the exception is raised. This statement is generally used in the beginning of the function or after a function call to check for valid input. The syntax for `assert` statement is:

```
assert Expression[,arguments]
```

On encountering an `assert` statement, Python evaluates the expression given immediately after the `assert` keyword. If this expression is false, an `AssertionError` exception is raised which can be handled like any other exception. Consider the following code with output as shown in Figure 2.8. The `assert` statement checks for the value of the variable number. In case the number gets a negative value, `AssertionError` will be thrown, and subsequent statements will not be executed. Hence, on passing a negative value (-350) as an argument, it results in `AssertionError` and displays the message **"OOPS... Negative Number"**.



```
File Edit Format Run Options Window Help
print ("Use of assert statement")
def negativecheck (number):
    assert (number>=0), "OOPS... Negative Number"
    print (number*number)
print (negativecheck(100))
print (negativecheck(-350))
Ln: 6 Col: 27

File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/dds/No Exception.py =====
Use of assert statement
10000
None
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
    exec(code, self.locals)
  File "/home/dds/No Exception.py", line 6, in <module>
    print (negativecheck(-350))
  File "/home/dds/No Exception.py", line 3, in negativecheck
    assert (number>=0), "OOPS... Negative Number"
AssertionError: OOPS... Negative Number
>>>
Ln: 16 Col: 0
```

**Fig. 2.8: Use of assert statement**

## 2.6 Handling Exceptions

Each and every exception has to be handled by the programmer to avoid the program from crashing abruptly. This is done by writing additional code in a program to give proper messages or instructions to user on encountering an exception. This process is known as exception handling.

### 2.2.1 Need for Exception Handling



There is a need for exception handling in Python just like other programming languages such as C++, Java, Ruby. It is a useful technique that helps in capturing runtime errors and handling them so as to avoid the program getting crashed. Following are some of the important points regarding exceptions handling.

- Python categorises exceptions into distinct types so that specific exception handlers (code to handle that particular exception) can be created for each type.
- Exception handlers separate the main logic of the program from the error detection and correction code. The segment of code where there is any possibility of error or exception, is placed inside one block. The code to be executed in case the exception has occurred, is placed inside another block. These statements for detection and reporting the exception do not affect the main logic of the program.
- The compiler or interpreter keeps track of the exact position where the error has occurred.
- Exception handling can be done for both user-defined and built-in exceptions.

### 2.2.2 Process of Handling Exception

When an error occurs, Python interpreter creates an object called the exception object. This object contains information about the error like its type, file name and position in the program where the error has occurred. The object is handed over to the runtime system so that it can find an appropriate code to handle this particular exception. This process of creating an exception object and handing it over to the runtime system is called *throwing an exception*. It is important to note that when an exception occurs while executing a particular program statement, the control jumps to an exception handler, abandoning execution of the remaining program statements.

A runtime system refers to the execution of the statements given in the program. It is a complex mechanism consisting of hardware and software that comes into action as soon as the program, written in any programming language, is put for execution.

The runtime system searches the entire program for a block of code, called the exception handler that can handle the raised exception. It first searches for the method in which the error has occurred and the exception has been raised. If not found, then it searches the method from which this method (in which exception was raised) was called. This hierarchical search in reverse order continues till the exception handler is found. This entire list of methods is known as call stack. When a suitable handler is found in the call stack, it is executed by the runtime process. This process of executing a suitable handler is known as catching the exception. If the runtime system is not able to find an appropriate exception after searching all the methods in the call stack, then the program execution stops. Figure 2.9 describes the exception handling process.

### 2.2.3 Catching Exceptions

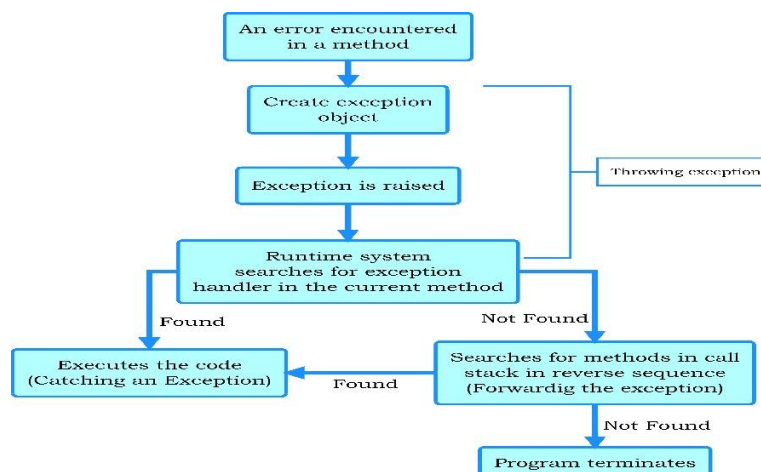


Fig. 2.9: Steps of handling exception

An exception is said to be caught when a code that is designed to handle a particular exception is executed. Exceptions, if any, are caught in the **try** block and handled in the **except** block. While writing or debugging a program, a user might doubt an exception to occur in a particular part of the code. Such suspicious lines of codes are put inside a **try** block. Every **try** block is followed by an **except** block. The appropriate code to handle each of the possible exceptions (in the code inside the try block) are written inside the except clause.

While executing the program, if an exception is encountered, further execution of the code inside the try block is stopped and the control is transferred to the **except** block. The syntax of **try... except** clause is as follows.

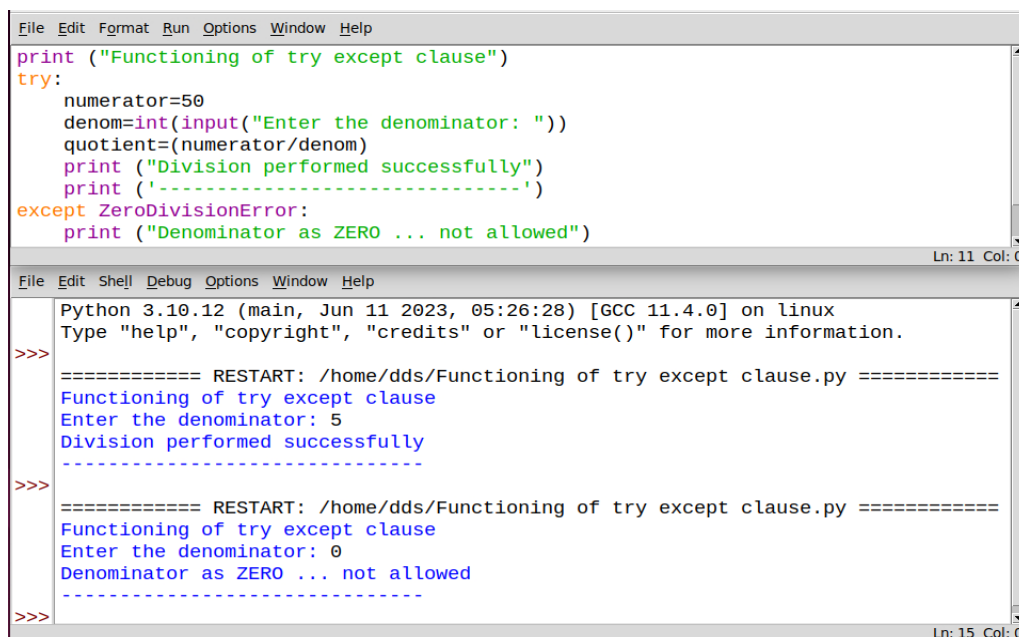
try:

[program statements where exceptions might occur]

except [exception-name]:

[code for exception handling if the exception-name error is encountered]

The python programme code shown in Figure 2.10 illustrates the functioning of **try... except** clause.



```

File Edit Format Run Options Window Help
print ("Functioning of try except clause")
try:
    numerator=50
    denom=int(input("Enter the denominator: "))
    quotient=(numerator/denom)
    print ("Division performed successfully")
    print ('-----')
except ZeroDivisionError:
    print ("Denominator as ZERO ... not allowed")
Ln: 11 Col: 0

File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/dds/Functioning of try except clause.py =====
Functioning of try except clause
Enter the denominator: 5
Division performed successfully
-----
>>>
===== RESTART: /home/dds/Functioning of try except clause.py =====
Functioning of try except clause
Enter the denominator: 0
Denominator as ZERO ... not allowed
-----
>>>
Ln: 15 Col: 0

```

**Fig. 2.10: Functioning of try... except clause.**

Above python code illustrate to handle ZeroDivisionError exception. If the user enters any non-zero value as denominator, the quotient will be displayed along with the message "Division performed successfully". The except clause will be skipped in this case. However, if the user enters the value of denom as zero (0), then the execution of the try block will stop. The control will shift to the except block and the message "Denominator as Zero.... not allowed" will be displayed. Thereafter, the statement following the **try... except** block is executed and the message "OUTSIDE try...except block" is displayed in this case also.

Sometimes, a single piece of code might be suspected to have more than one type of error. For handling such situations, we can have multiple except blocks for a single try block as shown in the Python code with output in Figure 2.11.

```

File Edit Format Run Options Window Help
print ("Handling multiple exceptions")
try:
    numerator=50
    denom=int(input("Enter the denominator: "))
    print (numerator/denom)
    print ("Division performed successfully")
except ZeroDivisionError:
    print ("Denominator as ZERO is not allowed")
except ValueError:
    print ("Denominator should be INTEGERS only")
Ln: 10 Col: 0

File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/dds/Handling multiple exceptions.py =====
Handling multiple exceptions
Enter the denominator: 5
10.0
Division performed successfully
>>>
===== RESTART: /home/dds/Handling multiple exceptions.py =====
Handling multiple exceptions
Enter the denominator: 0
Denominator as ZERO is not allowed
>>>
===== RESTART: /home/dds/Handling multiple exceptions.py =====
Handling multiple exceptions
Enter the denominator: p
Denominator should be INTEGERS only
>>>
Ln: 19 Col: 0

```

**Fig. 2.12: Handling multiple exception in Python**

In the above code, two types of exceptions (ZeroDivisionError and ValueError) are handled using two except blocks for a single try block. When an exception is raised, a search for the matching except block is made till it is handled. If no match is found, then the program terminates.

However, if an exception is raised for which no handler is created by the programmer, then such an exception can be handled by adding an except clause without specifying any exception. This except clause should be added as the last clause of **try... except** block.

#### **2.6.4 try...except...else clause**

We can put an optional else clause along with the **try... except** clause. An except block will be executed only if some exception is raised in the try block. But if there is no error then none of the except blocks will be executed. In this case, the statements inside the else clause will be executed. Program code along with its output as shown in Figure 2.13 explains the use of else block with the **try... except** block.

```

File Edit Format Run Options Window Help
print ("Handling exceptions using try...except...else")
try:
    numerator=50
    denom=int(input("Enter the denominator: "))
    quotient=(numerator/denom)
    print ("Division performed successfully")
except ZeroDivisionError:
    print ("Denominator as ZERO is not allowed")
except ValueError:
    print ("Denominator should be INTEGERS only")
else:
    print ("The result of division operation is ", quotient)
Ln: 6 Col: 4

File Edit Shell Debug Options Window Help
Python 3.10.12 (main, Jun 11 2023, 05:26:28) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/dds/Handling exceptions using try except else.py =====
Handling exceptions using try...except...else
Enter the denominator: 5
Division performed successfully
The result of division operation is 10.0
>>>
===== RESTART: /home/dds/Handling exceptions using try except else.py =====
Handling exceptions using try...except...else
Enter the denominator: 0
Denominator as ZERO is not allowed
>>>
===== RESTART: /home/dds/Handling exceptions using try except else.py =====
Handling exceptions using try...except...else
Enter the denominator: p
Denominator should be INTEGERS only
>>>
===== RESTART: /home/dds/Handling exceptions using try except else.py =====
Handling exceptions using try...except...else
Enter the denominator: 8
Division performed successfully
The result of division operation is 6.25
>>>
Ln: 7 Col: 0

```

**Fig. 2.13: Handling exception using try...except...else in Python**

### 2.7 Finally Clause

The try statement in Python can also have an optional finally clause. The statements inside the finally block are always executed regardless of whether an exception has occurred in the try block or not. It is a common practice to use finally clause while working with files to ensure that the file object is closed. If used, finally should always be placed at the end of try clause, after all except blocks and the else block. Program code along with its output as shown in Figure 2.14 explains the use of finally clause. The message **“OVER AND OUT”** will be displayed irrespective of whether an exception is raised or not.

```

File Edit Format Run Options Window Help
print ("Handling exceptions using try...except...else...finally")
try:
    numerator=50
    denom=int(input("Enter the denominator: "))
    quotient=(numerator/denom)
    print ("Division performed successfully")
except ZeroDivisionError:
    print ("Denominator as ZERO is not allowed")
except ValueError:
    print ("Denominator should be INTEGERS only")
else:
    print ("The result of division operation is ", quotient)
finally:
    print ("FINALLY OVER AND OUT")

IDLE Shell 3.10.12
File Edit Shell Debug Options Window Help
Enter the denominator: 5
Division performed successfully
The result of division operation is 10.0
FINALLY OVER AND OUT
>>>
==== RESTART: /home/dds/Handling exceptions using try except else finally.py ====
Handling exceptions using try...except...else...finally
Enter the denominator: 0
Denominator as ZERO is not allowed
FINALLY OVER AND OUT
>>>
==== RESTART: /home/dds/Handling exceptions using try except else finally.py ====
Handling exceptions using try...except...else...finally
Enter the denominator: p
Denominator should be INTEGERS only
FINALLY OVER AND OUT
>>>
==== RESTART: /home/dds/Handling exceptions using try except else finally.py ====
Handling exceptions using try...except...else...finally
Enter the denominator: 8
Division performed successfully
The result of division operation is 6.25
FINALLY OVER AND OUT
Ln: 10 Col: 0

```

**Fig. 2.14: Handling exception using try...except...else...finally in Python**

## Summary

1. Syntax errors or parsing errors are detected when we have not followed the rules of the particular programming language while writing a program.
  - When syntax error is encountered, Python displays the name of the error and a small description about the error.
  - The execution of the program will start only after the syntax error is rectified.
  - An exception is a Python object that represents an error.
  - Syntax errors are also handled as exceptions.
  - The exception needs to be handled by the programmer so that the program does not terminate abruptly.
  - When an exception occurs during execution of a program and there is a built-in exception defined for that, the error message written in that exception is displayed. The programmer then has to take appropriate action and handle it.
  - Some of the commonly occurring built-in exceptions are SyntaxError, ValueError, IOError, KeyboardInterrupt, ImportError, EOFError, ZeroDivisionError, IndexError, NameError, IndentationError, TypeError, and OverflowError.
  - When an error is encountered in a program, Python interpreter raises or throws an exception. Exception Handlers are the codes that are designed to execute when a specific exception is raised.
  - Raising an exception involves interrupting the normal flow of the program execution and jumping to the exception handler.
  - Raise and assert statements are used to raise exceptions.

- The process of exception handling involves writing additional code to give proper messages or instructions to the user. This prevents the program from crashing abruptly. The additional code is known as an exception handler.
- An exception is said to be caught when a code that is designed to handle a particular exception is executed.
- An exception is caught in the try block and handles in except block.
- The statements inside the finally block are always executed regardless of whether an exception occurred in the try block or not.

## CHECK YOUR PROGRESS

### A. Multiple choice questions

1. Which of the following is raised when there is an error in the syntax of the Python code. (a) SyntaxError (b) IOError (c) ZeroDivisionError (d) TypeError
2. Which of the following is raised when the file specified in a program statement cannot be opened. (a) SyntaxError (b) IOError (c) ZeroDivisionError (d) TypeError
3. Which of the following is raised when the denominator in a division operation is zero. SyntaxError (b) IOError (c) ZeroDivisionError (d) TypeError
4. Which of the following is raised when an operator is supplied with a value of incorrect data type. (a) SyntaxError (ans) (b) IOError (c) ZeroDivisionError (d) TypeError
5. Which of the following is statement in Python is used to test an expression in the program code. (a) Assert (b) raise (c) interpreter (d) compiler

### B. Fill In the Blank

1. On encountering a syntax error, the \_\_\_\_\_ does not execute the program.
2. An \_\_\_\_\_ is a Python object that represents an error.
3. Commonly occurring exceptions are usually defined in the compiler/interpreter, these are called \_\_\_\_\_ exceptions.
4. The \_\_\_\_\_ statement can be used to throw an exception.
5. An \_\_\_\_\_ statement in Python is used to test an expression in the program code.

### C. State whether True or False

1. The Raise statement can be used to throw an exception.
2. An Assert statement in Python is used to test an expression in the program code.
3. An exception is caught in the try block and handles in except block.
4. Syntax error raise at run time.
5.  $10/2$  will generate syntax error.

### D. Answer the following in short

1. "Every syntax error is an exception but every exception cannot be a syntax error." Justify the statement.
2. When are the following built-in exceptions raised? Give examples to support your answers. (a) ImportError (b) IOError (c) NameError (d) ZeroDivisionError
3. What is the use of a raise statement?
4. Define the following: (a) Exception Handling (b) Throwing an exception (c) Catching an exception
5. Explain catching exceptions using try and except block.

### Practical Exercise

1. Write a code to accept two numbers and display the quotient. Appropriate exception should be raised if the user enters the second number (denominator) as zero (0).
2. Use assert statement in Question No. 1 to test the division expression in the program.
3. Write a python code to use finally clause in the problem given in Question No. 1.

## Session 3: File Handling in Python

### 3.1 Introduction to Files

So far, we have created programs in Python that accept the input, manipulate it and display the output. Output of the program is available only during execution of the program and input is to be entered through the keyboard. This is because the variables used in a program have a lifetime that lasts till the time the program is under execution.

Usually, organisations would want to permanently store information about employees, inventory, sales to avoid repetitive tasks of entering the same data. Hence, data are stored permanently on secondary storage devices for reusability. We store Python programs written in script mode with a .py extension. Each program is stored on the secondary device as a file. Likewise, the data entered, and the output can be stored permanently into a file.

So, what is a file? A file is a named location on a secondary storage media where data are permanently stored for later access.

Text files contain only the ASCII equivalent of the contents of the file whereas a.docx file contains many additional information like the author's name, page settings, font type and size, date of creation and modification, etc.

### 3.2. Types of Files

Computers store every file as a collection of 0s and 1s i.e., in binary form. Therefore, every file is just a series of bytes stored one after the other. There are mainly two types of data files — text file and binary file. A text file consists of human readable characters, which can be opened by any text editor. On the other hand, binary files are made up of non-human readable characters and symbols, which require specific programs to access its contents.

#### 3.2.1 Text file

A text file can be understood as a sequence of characters consisting of alphabets, numbers and other special symbols. Files with extensions like .txt, .py, .csv are some examples of text files. When we open a text file using a text editor such as Notepad, it shows several lines of text. However, file contents are not stored in such a way internally. Rather, they are stored in sequence of bytes consisting of 0s and 1s. In ASCII, UNICODE or any other encoding scheme, the value of each character of the text file is stored as bytes. So, while opening a text file, the text editor translates each ASCII value and shows the equivalent character that is readable by the human being. For example, the ASCII value 65 (binary equivalent 1000001) will be displayed by a text editor as the letter 'A' since the number 65 in ASCII character set represents 'A'.

Each line of a text file is terminated by a special character, called the End of Line (EOL). For example, the default EOL character in Python is the newline (\n). However, other characters can be used to indicate EOL. When a text editor or a program interpreter encounters the ASCII equivalent of the EOL character, it displays the remaining file contents starting from a new line. Contents in a text file are usually separated by whitespace, but comma (,) and tab (\t) are also commonly used to separate values in a text file.

Activity 3.1. Create a text file using notepad and write your name and save it. Now, create a .docx file using Microsoft Word and write your name and save it as well. Check and compare the file size of both the files. You will find that the size of .txt file is in bytes whereas that of .docx is in KBs.

#### 3.2.2 Binary Files

Binary files are also stored in terms of bytes (0s and 1s), but unlike text files, these bytes do not represent the ASCII values of characters. Rather, they represent the actual content such as

image, audio, video, compressed versions of other files, executable files, etc. These files are not human readable. Thus, trying to open a binary file using a text editor will show some garbage values. We need specific software to read or write the contents of a binary file.

Binary files are stored in a computer in a sequence of bytes. Even a single bit change can corrupt the file and make it unreadable to the supporting application. Also, it is difficult to remove any error which may occur in the binary file as the stored contents are not human readable. We can read and write both text and binary files through Python programs.

### 3.3 Opening and Closing a Text File

In real world applications, computer programs deal with data coming from different sources like databases, CSV files, HTML, XML, JSON, etc. We broadly access files either to write or read data from it. But operations on files include creating and opening a file, writing data in a file, traversing a file, reading data from a file and so on. Python has the IO module that contains different functions for handling files.

#### 3.3.1 Opening a file

To open a file in Python, we use the `open()` function. The syntax of `open()` is as follows:

```
file_object= open(file_name, access_mode)
```

This function returns a file object called file handle which is stored in the variable `file_object`. We can use this variable to transfer data to and from the file (read and write) by calling the functions defined in the Python's `io` module. If the file does not exist, the above statement creates a new empty file and assigns it the name we specify in the statement.

The `file_object` has certain attributes that tells us basic information about the file, such as:

<`file.closed`> returns true if the file is closed and false otherwise.

<`file.mode`> returns the access mode in which the file was opened.

<`file.name`> returns the name of the file.

The `file_name` should be the name of the file that has to be opened. If the file is not in the current working directory, then we need to specify the complete path of the file along with its name. The `access_mode` is an optional argument that represents the mode in which the file has to be accessed by the program. It is also referred to as processing mode. Here mode means the operation for which the file has to be opened like `<r>` for reading, `<w>` for writing, `<+>` for both reading and writing, `<a>` for appending at the end of an existing file. The default is the read mode. In addition, we can specify whether the file will be handled as binary (`<b>`) or text mode. By default, files are opened in text mode that means strings can be read or written. Files containing non-textual data are opened in binary mode that means read/write are performed in terms of bytes. Table 3.1 lists various file access modes that can be used with the `open()` method. The file offset position in the table refers to the position of the file object when the file is opened in a particular mode.

Table 3.1 File Open Modes

File Mode	Description	File Offset position
<r>	Opens the file in read-only mode.	Beginning of the file
<rb>	Opens the file in binary and read-only mode.	Beginning of the file
<r+> or <+r>	Opens the file in both read and write mode.	Beginning of the file
<w>	Opens the file in write mode. If the file already exists, all the Beginning of the file contents will be overwritten. If the file doesn't exist, then a new file will be created.	Beginning of the file



<wb+> or <+wb>	Opens the file in read,write and binary mode. If the file already exists, the contents will be overwritten. If the file doesn't exist, then a new file will be created.	Beginning of the file
<a>	Opens the file in append mode. If the file doesn't exist, then a new file will be created.	End of the file
<a+> or <+a>	Opens the file in append and read mode. If the file doesn't exist, then it will create a new file.	End of the file

Consider the following example.

```
myObject=open("myfile.txt", "a+")
```

In the above statement, the file *myfile.txt* is opened in append and read modes. The file object will be at the end of the file. That means we can write data at the end of the file and at the same time we can also read data from the file using the file object named *myObject*.

### 3.3.2 Closing a file

It is always a good practice to close the file after performing read/write operations on a file. Closing a file frees the memory allocated to it. Python provides a `close()` method to close a file. The syntax of `close()` is:

```
file_object.close()
```

Here, `file_object` is the object that was returned while opening the file.

Python makes sure that any unwritten or unsaved data is flushed off (written) to the file before it is closed. Hence, it is always advised to close the file after completing the work. Also, if the file object is re-assigned to some other file, the previous file is automatically closed.

### 3.3.3 Opening a file using with clause

In Python, we can also open a file using with clause. The syntax of with clause is:

```
with open (file_name, access_mode) as file_object:
```

The advantage of using with clause is that any file that is opened using this clause is closed automatically, once the control comes outside the with clause. In case the user forgets to close the file explicitly or if an exception occurs, the file is closed automatically. Also, it provides a simple syntax.

```
with open("myfile.txt","r+") as myObject:
    content = myObject.read()
```

Here, we don't have to close the file explicitly using `close()` statement. Python will automatically close the file.

## 3.4 Writing to a Text File

For writing to a file, first it need to open in write or append mode. Opening an existing file in write mode, will erase the previous data, and the file object will be positioned at the beginning of the file. On the other hand, in append mode, new data will be added at the end of the previous data as the file object is at the end of the file. After opening the file, the following methods can be used to write data in the file.

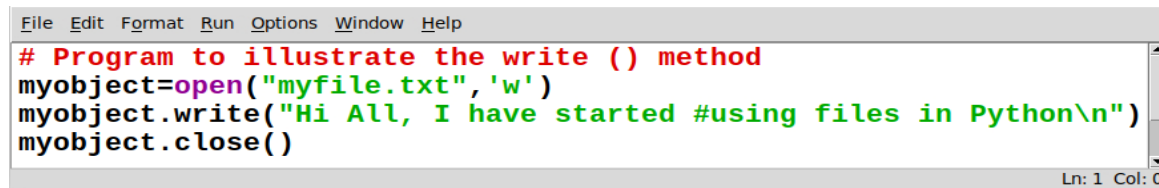
Think and Reflect for a newly created file, is there any difference between `write()` and `writeline()` methods?

`write()` - for writing a single string

`writeline()` - for writing a sequence of strings

### 3.4.1 The write() method

The write() method takes a string as an argument and writes it to the text file. It returns the number of characters being written on single execution of the write() method. Also, it is required to add a newline character (\n) at the end of every sentence to mark the end of line. Consider the following Python code.



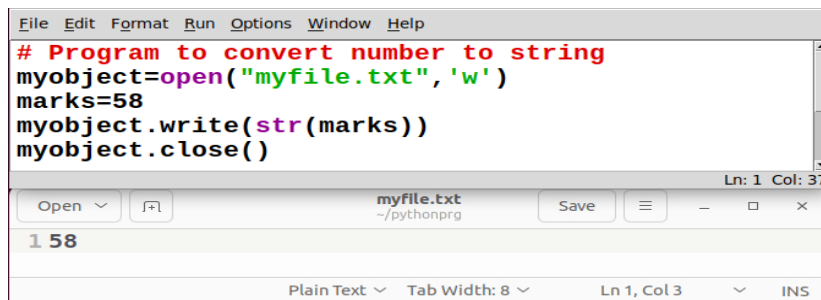
```
File Edit Format Run Options Window Help
# Program to illustrate the write () method
myobject=open("myfile.txt",'w')
myobject.write("Hi All, I have started #using files in Python\n")
myobject.close()
Ln: 1 Col: 0
```

Executing this code will create the file with the name myfile.txt with the text as mentioned under the write () method. Note that '\n' is treated as a single character.

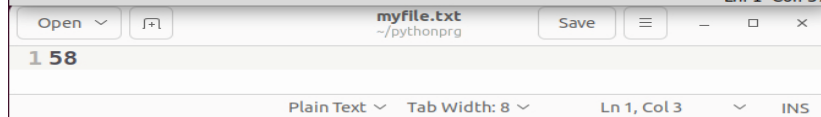


```
Open [F+] myfile.txt Save [F1]
~/pythonprg
1 Hi All, I have started #using files in Python
Plain Text Tab Width: 8 Ln 1, Col 29 INS
```

If numeric data are to be written to a text file, the data need to be converted into string before writing to the file. Following program code and its output text file below that shows the numeric marks 58 is converted to string using the str () function and then it is written in the text file.



```
File Edit Format Run Options Window Help
# Program to convert number to string
myobject=open("myfile.txt",'w')
marks=58
myobject.write(str(marks))
myobject.close()
Ln: 1 Col: 37
```

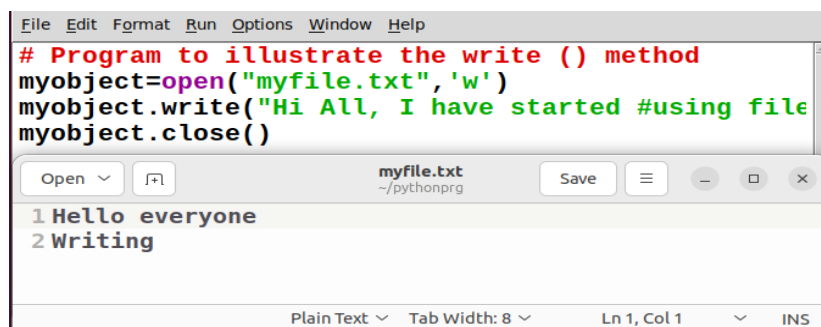


```
Open [F+] myfile.txt Save [F1]
~/pythonprg
1 58
Plain Text Tab Width: 8 Ln 1, Col 3 INS
```

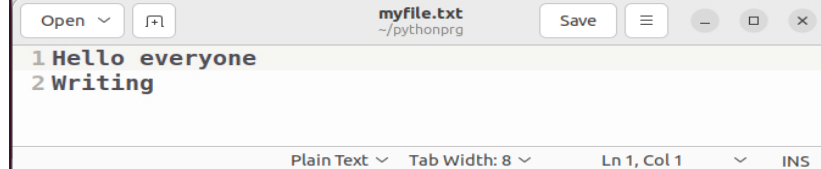
The write() actually writes data onto a buffer. When the close() method is executed, the contents from this buffer are moved to the file located on the permanent storage. The flush() method can also be used to clear the buffer and write contents in buffer to the file. This is how programmers can forcefully write to the file as and when required.

### 3.4.2 The writelines() method

This method is used to write multiple strings to a file. We need to pass an iterable object like lists, tuple, etc. containing strings to the writelines() method. Unlike write(), the writelines() method does not return the number of characters written in the file. The following code and its output shows that the writeln function writes by breaking the text to the next line when \n character encountered. This illustrates the use of writelines() method.



```
File Edit Format Run Options Window Help
# Program to illustrate the write () method
myobject=open("myfile.txt",'w')
myobject.write("Hi All, I have started #using file\n")
myobject.close()
Ln: 1 Col: 37
```



```
Open [F+] myfile.txt Save [F1]
~/pythonprg
1 Hello everyone
2 Writing
Plain Text Tab Width: 8 Ln 1, Col 1 INS
```

## 3.5 Reading from a Text File

We can write a program to read the contents of a file. Before reading a file, we must make sure that the file is opened in “r”, “r+”, “w+” or “a+” mode. There are three ways to read the contents of a file:

### 3.5.1 The read () method

This method is used to read a specified number of bytes of data from a data file. The syntax of read () method is:

```
file_object.read(n)
```

Consider the following set of statements. The value 10 is passed to read () method, produce the output of reading 10 characters from the line. This illustrates the usage of read() method:

```
File Edit Format Run Options Window Help
# Program to illustrate the readline() method
myobject=open("myfile.txt", 'r')
print(myobject.read(10))
myobject.close()

IDLE Shell 3.10.12
File Edit Shell Debug Options Window Help
>>>
= RESTART: /home/dds/pythonprg/Example code 4.py
Hello ever
>>>
Ln: 14 Col: 0
```

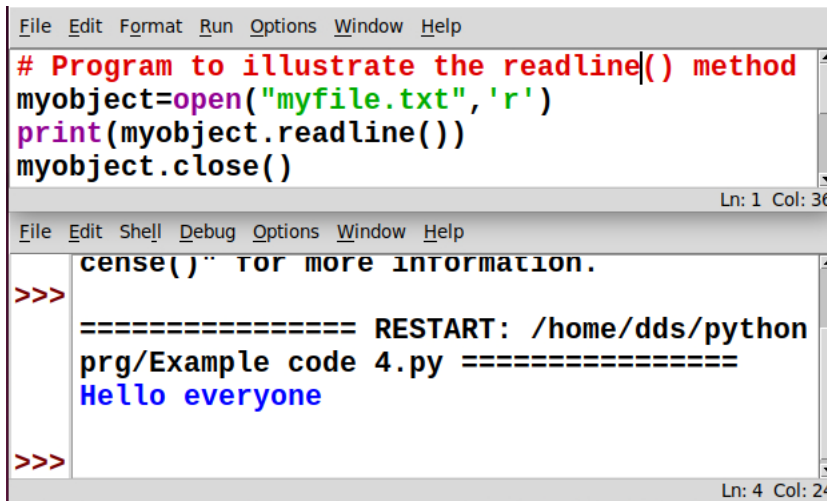
If no argument or a negative number is specified in read(), the entire file content is read as shown in the following code.

```
File Edit Format Run Options Window Help
# Program to illustrate the readline() method
myobject=open("myfile.txt", 'r')
print(myobject.read())
myobject.close()

IDLE Shell 3.10.12
File Edit Shell Debug Options Window Help
= RESTART: /home/dds/pythonprg/Example code 4.py
Hello everyone
Writing
>>>
Ln: 11 Col: 0
```

### 3.5.2 The readline([n]) method

This method reads one complete line from a file where each line terminates with a newline (\n) character. It can also be used to read a specified number (n) of bytes of data from a file but maximum up to the newline character (\n). Following code illustrates that it reads the first line of text file and displays them on the screen.



```
File Edit Format Run Options Window Help
# Program to illustrate the readline() method
myobject=open("myfile.txt", 'r')
print(myobject.readline())
myobject.close()
Ln: 1 Col: 36

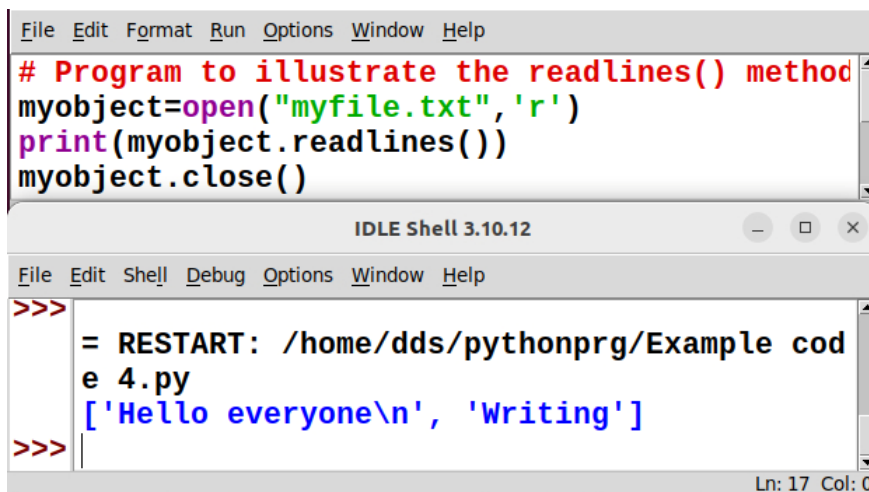
File Edit Shell Debug Options Window Help
cense() for more information.
>>>
===== RESTART: /home/dds/python
prg/Example code 4.py =====
Hello everyone
>>>
Ln: 4 Col: 24
```

If no argument or a negative number is specified, it reads a complete line and returns string.

To read the entire file line by line using the `readline()`, we can use a loop. This process is known as looping/ iterating over a file object. It returns an empty string when EOF is reached.

### 3.5.3 The `readlines()` method

The method reads all the lines and returns the lines along with newline as a list of strings. The following code illustrates the use of `readlines()` to read data from the text file `myfile.txt`.



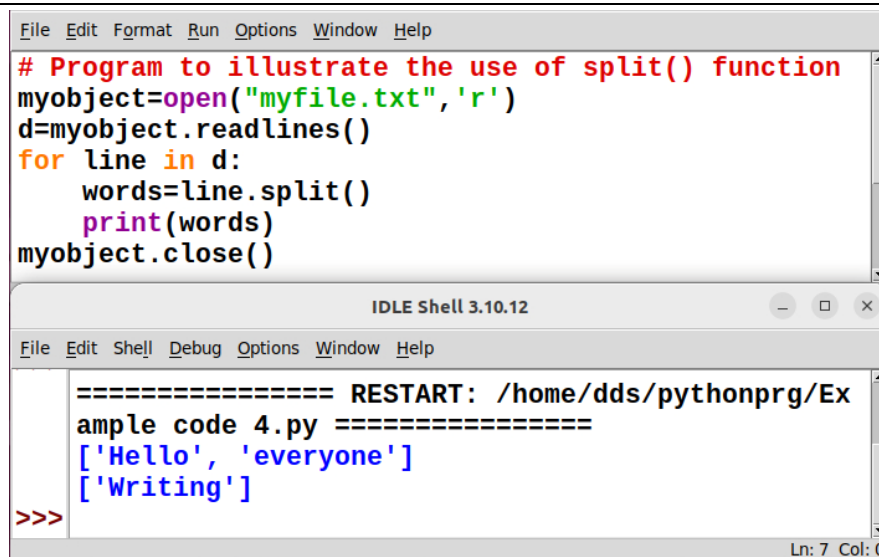
```
File Edit Format Run Options Window Help
# Program to illustrate the readlines() method
myobject=open("myfile.txt", 'r')
print(myobject.readlines())
myobject.close()

IDLE Shell 3.10.12

File Edit Shell Debug Options Window Help
>>>
= RESTART: /home/dds/pythonprg/Example cod
e 4.py
['Hello everyone\n', 'Writing']
>>>
Ln: 17 Col: 0
```

The above output shows that when a file is read using `readlines()` function, lines in the file become members of a list, where each list element ends with a newline character (`'\n'`).

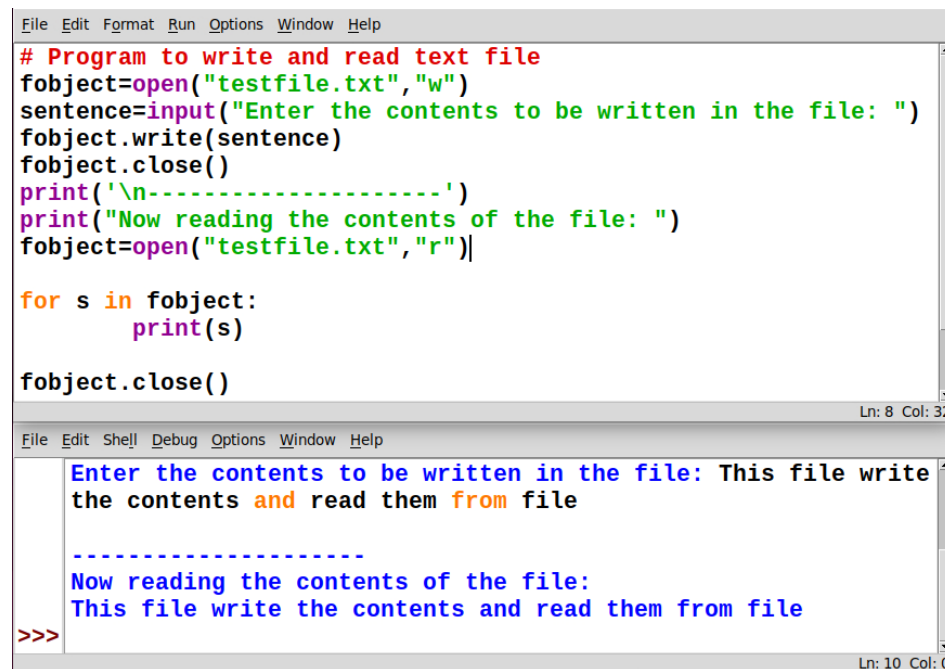
To display each word of a line separately as an element of a list, the `split()` function can be used. The following code demonstrates the use of `split()` function.



```
File Edit Format Run Options Window Help
# Program to illustrate the use of split() function
myobject=open("myfile.txt",'r')
d=myobject.readlines()
for line in d:
    words=line.split()
    print(words)
myobject.close()

IDLE Shell 3.10.12
File Edit Shell Debug Options Window Help
===== RESTART: /home/dds/pythonprg/Ex
ample code 4.py =====
['Hello', 'everyone']
['Writing']
>>>
Ln: 7 Col: 0
```

Let us now write a program that accepts a string from the user and writes it to a text file. Thereafter, the same program reads the text file and displays it on the screen. Following program illustrates how the text written in the text file is read.



```
File Edit Format Run Options Window Help
# Program to write and read text file
fobject=open("testfile.txt","w")
sentence=input("Enter the contents to be written in the file: ")
fobject.write(sentence)
fobject.close()
print('\n-----')
print("Now reading the contents of the file: ")
fobject=open("testfile.txt","r")

for s in fobject:
    print(s)

fobject.close()

Ln: 8 Col: 32

File Edit Shell Debug Options Window Help
Enter the contents to be written in the file: This file write
the contents and read them from file

-----
Now reading the contents of the file:
This file write the contents and read them from file
>>>
Ln: 10 Col: 0
```

### 3.6 Setting Offsets in a File

The functions that we have learnt till now are used to access the data sequentially from a file. But if we want to access data in a random fashion, then Python gives us `seek()` and `tell()` functions to do so.

#### 3.3.1 The `tell()` method

This function returns an integer that specifies the current position of the file object in the file. The position so specified is the byte position from the beginning of the file till the current position of the file object. The syntax of using `tell()` is:

```
file_object.tell()
```

#### 3.3.2 The `seek()` method

This method is used to position the file object at a particular position in a file. The syntax of `seek()` is:

```
file_object.seek(offset [, reference_point])
```

In the above syntax, offset is the number of bytes by which the file object is to be moved. `reference_point` indicates the starting position of the file object. That is, with reference to which position, the offset has to be counted. It can have any of the following values:

- 0 - beginning of the file
- 1 - current position of the file
- 2 - end of file

By default, the value of `reference_point` is 0, i.e. the offset is counted from the beginning of the file. The following program illustrates the usage of `seek()` and `tell()`.

```
File Edit Format Run Options Window Help
# Program to illustrate the usage of seek() and tell()

fobject=open("testfile.txt","r")
print('\nFile opening location at begening is: ',fobject.tell())

print('\nThe content of file are as under\n')
print('-----')
for s in fobject:
    print(s)
print('-----')
fobject.seek(6)
print('\nNow file pointer will shift to 6th location',fobject.tell())

print('After shifting the content of file are as under\n')
print('-----')
for s in fobject:
    print(s)
print('-----')
print('\nCurrent Position of File pointer is : ', fobject.tell())

fobject.close()
Ln: 26 Col: 0
```

Executing the above program, the statement `fileObject.seek(6,0)` will position the file object at 6<sup>th</sup> byte position from the beginning of the file and print the text from the 6<sup>th</sup> character as shown in the following output screen.

```
File Edit Shell Debug Options Window Help
File opening location at begening is: 0
The content of file are as under
-----
This file write the contents and read them from file
-----
Now file pointer will shift to 6th location 6
After shifting the content of file are as under
-----
ile write the contents and read them from file
-----
Current Position of File pointer is : 52
>>>
Ln: 12 Col: 41
```

### 3.7 Creating and Traversing a Text File

Having learnt various methods that help us to open and close a file, read and write data in a text file, find the position of the file object and move the file object at a desired location, let us now perform some basic operations on a text file. To perform these operations, let us assume that we will be working with `practice.txt`.

#### 3.7.1 Creating a file and writing data

To create a text file, we use the `open()` method and provide the filename and the mode. If the file already exists with the same name, the `open()` function will behave differently depending on the mode (write or append) used. If it is in write mode (`w`), then all the existing contents of file will be lost, and an empty file will be created with the same name. But, if the file is created in append

mode (**a**), then the new data will be written after the existing data. In both cases, if the file does not exist, then a new empty file will be created. In the following program a file, practice.txt is opened in write (w) mode and three sentences are stored in it as shown in the output screen that follows it.

```
File Edit Format Run Options Window Help
# Program to illustrate Creating and Traversing a Text File
fileobject=open("practice.txt","a")
data= input("Enter data to save in the text file: ")
fileobject.write(data)

fileobject=open("practice.txt","r")
s = fileobject.readline()
print (s)
fileobject.close()
Ln: 10 Col: 0
```

Executing the above program allows to enter the data to be stored in the practice.txt file, and thereafter read the contents entered as shown below.

```
File Edit Shell Debug Options Window Help
y
Enter data to save in the text file: The various job roles are
offered as vocational subjects in the schools for skill develop
ment of the students.
The various job roles are offered as vocational subjects in the
schools for skill development of the students.
>>>
Ln: 7 Col: 0
```

The practice.txt file is created as below.

```
Open [ ] Save [ ] [ ] [ ] [ ]
practice.txt
~/pythonprg
1 The various job roles are offered as vocational
  subjects in the schools for skill development of the
  students.
Plain Text Tab Width: 8 Ln 1, Col 111 INS
```

### 3.8 The Pickle Module

We know that Python considers everything as an object. So, all data types including list, tuple, dictionary are also considered as objects. During execution of a program, we may require to store current state of variables so that we can retrieve them later to its present state. Suppose you are playing a video game, and after some time, you want to close it. So, the program should be able to store the current state of the game, including current level/stage, your score as a Python object. Likewise, you may like to store a Python dictionary as an object, to be able to retrieve later. To save any object structure along with data, Python provides a module called **Pickle**. The module Pickle is used for serializing and de-serializing any Python object structure. Pickling is a method of preserving food items by placing them in some solution, which increases the shelf life. In other words, it is a method to store food items for later consumption.

Serialization is the process of transforming data or an object in memory (RAM) to a stream of bytes called byte streams. These byte streams in a binary file can then be stored in a disk or in a database or sent through a network. Serialization process is also called pickling.

De-serialization or unpickling is the inverse of pickling process where a byte stream is converted back to Python object.

The pickle module deals with binary files. Here, data are not written but dumped and similarly, data are not read but loaded. The Pickle Module must be imported to load and dump data. The pickle module provides two methods - dump() and load() to work with binary files for pickling and unpickling, respectively.

### 3.8.1 The dump() method

This method is used to convert (pickling) Python objects for writing data in a binary file. The file in which data are to be dumped, needs to be opened in binary write mode (wb).

Syntax of dump() is as follows:

**dump(data\_object, file\_object)**

where data\_object is the object that has to be dumped to the file with the file handle named **file\_object**. For example, Program 2-6 writes the record of a student (roll\_no, name, gender and marks) in the binary file named mybinary.dat using the dump(). We need to close the file after pickling.

### 3.8.2 The load() method

This method is used to load (unpickling) data from a binary file. The file to be loaded is opened in binary read (rb) mode. Syntax of load() is as follows:

Store\_object = load(file\_object)

Here, the pickled Python object is loaded from the file having a file handle named **file\_object** and is stored in a new file handle called **store\_object**.

### 3.8.3 File handling using pickle module

As we read and write data in a text file, similarly we will be adding and displaying data for a binary file. Following program illustrates to accepts the data for employee record from the user and appends it in the binary file. Then, the records are read from the binary file and displayed on the screen using the same object. The user can enter as many records as they wish to. The program also displays the size of binary files before starting with the reading process.

As each employee record is stored as a list in the file empfile.dat, hence while reading the file, a list is displayed showing record of each employee. Notice that in this program **try... except** block is used to handle the end-of-file exception.

When second time program is executed, new record will be added to existing data file. You can observe the output below for two employee records carefully.



```

File Edit Format Run Options Window Help
# Program to write and read employee records in a binary file using pickle modul
import pickle
print("Writing and Reading Employee Record in Binary file using Pickle Module")
bfile=open("empdata.dat","ab")
recno=1
print ("Enter Records of Employees")
print()
while True:
    eno=int(input("\tEmployee number : "))
    ename=input("\tEmployee Name : ")
    ebasic=int(input("\tBasic Salary : "))
    allow=int(input ("\tAllowances : "))
    totals=ebasic+allow
    print("\tTOTAL SALARY :",totals)
    edata=[eno,ename,ebasic,allow,totsal]
    pickle.dump(edata,bfile)
    ans=input("Do you wish to enter more records (y/n)? ")
    recno=recno+1
    if ans.lower()=='n':
        print("Record entry OVER ")
        print()
        break
# Retrieving the size of file
print("Size of binary file (in bytes):",bfile.tell())
bfile.close()
# Reading the employee records from the file using load() module
print("Now reading the employee records from the file")
print()
readrec=1
try:
    bfile=open("empdata.dat","rb")
    while True:
        edata=pickle.load(bfile)
        print("Record Number : ",readrec," ",edata)
        #print(edata)
        readrec=readrec+1
except EOFError:
    pass
bfile.close()
Ln: 39 Col: 13

```

```

File Edit Shell Debug Options Window Help
Writing and Reading Employee Record in Binary file using Pickle Module
Enter Records of Employees

    Employee number : 101
    Employee Name   : Vishal
    Basic Salary    : 62500
    Allowances      : 3000
    TOTAL SALARY    : 65500
Do you wish to enter more records (y/n)? y
    Employee number : 102
    Employee Name   : Shubham
    Basic Salary    : 66000
    Allowances      : 4000
    TOTAL SALARY    : 70000
Do you wish to enter more records (y/n)? n
Record entry OVER

Size of binary file (in bytes): 77
Now reading the employee records from the file

Record Number : 1    [101, 'Vishal', 62500, 3000, 65500]
Record Number : 2    [102, 'Shubham', 66000, 4000, 70000]
>>>
Ln: 27 Col: 0

```

## Summary

- A file is a named location on a secondary storage media where data are permanently stored for later access.

- A text file contains only textual information consisting of alphabets, numbers and other special symbols. Such files are stored with extensions like .txt, .py, .c, .csv, .html, etc. Each byte of a text file represents a character.
- Each line of a text file is stored as a sequence of ASCII equivalent of the characters and is terminated by a special character, called the End of Line (EOL).
- Binary file consists of data stored as a stream of bytes.
- open() method is used to open a file in Python and it returns a file object called file handle. The file handle is used to transfer data to and from the file by calling the functions defined in the Python's IO module.
- close() method is used to close the file. While closing a file, the system frees up all the resources like processor and memory allocated to it.
- write() method takes a string as an argument and writes it to the text file.
- writelines() method is used to write multiple strings to a file. We need to pass an iterable object like lists, tuple etc. containing strings to writelines() method.
- read([n]) method is used to read a specified number of bytes (n) of data from a data file.
- readline([n]) method reads one complete line from a file where lines are ending with a newline (\n). It can also be used to read a specified number (n) of bytes of data from a file but maximum up to the newline character (\n).
- readlines() method reads all the lines and returns the lines along with newline character, as a list of strings.
- tell() method returns an integer that specifies the current position of the file object. The position so specified is the byte position from the beginning of the file till the current position of the file object.
- seek() method is used to position the file object at a particular position in a file.
- Pickling is the process by which a Python object is converted to a byte stream.
- dump() method is used to write the objects in a binary file.
- load() method is used to read data from a binary file.

## CHECK YOUR PROGRESS

### A. Multiple choice questions

1. Which of the following method returns an integer that specifies the current position of the file object. (a) tell () b. dump () c. read () d. Seek ()
2. Which of the following method is used to position the file object at a particular position in a file (a) tell() (b) dump() (c) read() (d) seek()
3. Which of the following method is used to write the objects in a binary file. (a) tell() (b) dump() (c) read() (d) seek()
4. Which of the following method is used to read data from a binary file (a) tell() (b) dump() (c) load() (d) seek()
5. What is name of package/library is used to read/write data from a binary file (a) matplotlib (b) pandas (c) pickle (d) math

### B. Fill in The Blank

1. A \_\_\_\_\_ contains only textual information consisting of alphabets, numbers and other special symbols.
2. Binary file consists of data stored as \_\_\_\_\_ bytes.
3. Each line of a text file is terminated by a special character, called the \_\_\_\_\_.

4. To open a file in Python, we use the \_\_\_\_\_ function.
5. To mark the end of line at the end of every sentence need to add a \_\_\_\_\_

**C. State whether True or False**

1. open() method is used to close a file.
2. close() method is used to open a file.
3. write() method is used to read a file.
4. read() method is used to write a file.
5. readlines() function is read multiple line from file.

**D. Answer the following in short**

1. Differentiate between: (a) text file and binary file (b) readline() and readlines() (c) write() and writelines()
2. Write the use and syntax for the following methods: (a) open() (b) read() (c) seek() (d) dump()
3. Write the file mode that will be used for opening the following files. Also, write the Python statements to open the following files:
  - a) a text file "example.txt" in both read and write mode
  - b) a binary file "bfile.dat" in write mode
  - c) a text file "try.txt" in append and read mode
  - d) a binary file "btry.dat" in read only mode.

Why is it advised to close a file after we are done with the read and write operations? What will happen if we do not close it? Will some error message be flashed?

4. What is the difference between the following set of statements (a) and (b):
  - a) `P = open("practice.txt", "r")`  
`P.read(10)`
  - b) with `open("practice.txt", "r")` as P:  
`x = P.read()`
5. What will be the difference if the file was opened in write mode instead of append mode?

**Practical Exercise**

1. Write a command(s) to write the following lines to the text file named hello.txt. Assume that the file is opened in append mode.

```

>Welcome my class"
"It is a fun place"
"You will learn and play"

```

2. Write a Python program to open the file hello.txt used in question no 5 in read mode to display its contents.
3. Write a program to accept string/sentences from the user till the user enters "END" to. Save the data in a text file and then display only those sentences which begin with an uppercase alphabet.
4. Write a program to enter the following records in a binary file:

Item No	integer
Item_Name	string
Qty	integer
Price	float

Number of records to be entered should be accepted from the user. Read the file to display the records in the following format:

Item No:                      Item Name :                      Quantity:                      Price per item:                      Amount:  
 Amount to be calculated as Price \* Qty.

## Session 4. NumPy Array

In Python, the lists data structure serves the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is much faster than traditional Python lists. NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently. The elements of a NumPy array must all be of the same type, whereas the elements of a Python list can be of completely different types.

NumPy stands for Numerical Python is a Python library used for working with arrays. It provides functions for fast mathematical computation on arrays and matrices. NumPy objects are primarily used to create arrays or matrices that can be applied to Deep Learning or Machine Learning models. Pandas is used for creating heterogeneous, two-dimensional data objects, NumPy makes N-dimensional homogeneous objects. Pandas functions return result in the form of NumPy array.

### Installation and Importing of NumPy

To use NumPy, it is required to install and import NumPy. If Python and PIP already installed on a system, then installation of NumPy is very easy by using the command,

#### install numpy

Once NumPy is installed, it is possible to import it in by adding the **import** keyword.

#### import numpy

NumPy is usually imported under the np alias as follows.

#### import numpy as np

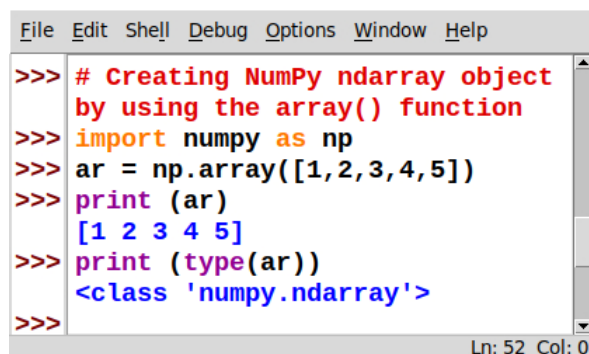
NumPy is ready to use after imported.

### NumPy Creating Arrays

There are 6 general mechanisms for creating arrays.

1. Conversion from other Python structures such as lists and tuples.
2. Intrinsic NumPy array creation functions (e.g. arange, ones, zeros, etc.).
3. Replicating, joining, or mutating existing arrays.
4. Reading arrays from disk, either from standard or custom formats.
5. Creating arrays from raw bytes through the use of strings or buffers.
6. Use of special library functions such as random.

NumPy is used to work with arrays. The array object in NumPy is called ndarray. NumPy ndarray object can be created by using the array() function.



```
File Edit Shell Debug Options Window Help
>>> # Creating NumPy ndarray object
>>> # by using the array() function
>>> import numpy as np
>>> ar = np.array([1,2,3,4,5])
>>> print (ar)
[1 2 3 4 5]
>>> print (type(ar))
<class 'numpy.ndarray'>
>>>
```

**type()** is built-in Python function display the type of the object passed to it. Like in above code it shows that arr is numpy.ndarray type.

To create an ndarray, you can pass a list, tuple or any array-like object into the array() method, and it will be converted into an ndarray, as illustrated in the following example.

```
File Edit Shell Debug Options Window Help
>>> # Creating NumPy array using a tuple
>>> import numpy as np
>>> ar = np.array((1,2,3,4,5))
>>> print(ar)
[1 2 3 4 5]
>>>
Ln: 17 Col: 0
```

### Dimensions in Arrays

There are two forms of NumPy arrays – one dimensional array, known as *vectors*, and multidimensional arrays, known as *matrices*. NumPy has a whole sub module dedicated towards matrix operations called `numpy.mat`

Following example illustrate to create 1D, 2D and 3D arrays.

```
File Edit Shell Debug Options Window Help
>>> # Creating 1-Dimensional array
>>> import numpy as np
>>> ar1 = np.array([1, 2, 3, 4, 5])
>>> print("1 Dimensional array created as : \n", ar1)
1 Dimensional array created as :
[1 2 3 4 5]
>>> # Creating 2-Dimensional array
>>> ar2 = np.array([[1, 2, 3], [4, 5, 6]])
>>> print("2 Dimensional array created as : \n", ar2)
2 Dimensional array created as :
[[1 2 3]
 [4 5 6]]
>>> # Creating 3-Dimensional array
>>> ar3 = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])
>>> print("3 Dimensional array created as : \n", ar3)
3 Dimensional array created as :
[[[1 2 3]
 [4 5 6]]
 [[1 2 3]
 [4 5 6]]]
>>>
Ln: 24 Col: 0
```

### Checking Dimensions of Array

It is possible to check dimension of array. NumPy Arrays provides the `ndim` attribute that returns an integer value indicating the number of dimensions the array. Following example illustrates to check number of dimensions of the array.

```

File Edit Shell Debug Options Window Help
>>> # Checking the dimension of array
>>> import numpy as np
>>> a = np.array(25)
>>> print ("The array is :\n", a)
The array is :
25
>>> print ("The dimension of array is :\n", a.ndim)
The dimension of array is :
0
>>> b = np.array([1, 2, 3, 4, 5])
>>> print ("The array is :\n", b)
The array is :
[1 2 3 4 5]
>>> print ("The dimension of array is :\n", b.ndim)
The dimension of array is :
1
>>> c = np.array([[1, 2, 3], [4, 5, 6]])
>>> print ("The array is :\n", c)
The array is :
[[1 2 3]
 [4 5 6]]
>>> print ("The dimension of array is :\n", c.ndim)
The dimension of array is :
2
>>> d = np.array([[[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]]])
>>> print ("The array is :\n", d)
The array is :
[[[1 2 3]
 [4 5 6]]

 [[1 2 3]
 [4 5 6]]]
>>> print ("The dimension of array is :\n", d.ndim)
The dimension of array is :
3
>>>
Ln: 41 Col: 0

```

### NumPy Arrays Vs Python Lists

Although NumPy array also holds elements like Python List, yet NumPy arrays are different data structures from Python list. The key differences are as follows.

NumPy Array	Python List
It is required to install and import Numpy Library to access Numpy Arrays.	It is built-in function of python available in the core library of python.
Once a NumPy array is created, it is not possible to change its size. It is required to create a new array or overwrite the existing one.	It is possible to append/insert values in List.
NumPy array contain elements of same type (homogeneous)	List contain elements of different type (heterogeneous)
Element wise operation is possible in NumPy array.	Element wise operation is not possible on the list.
An equivalent NumPy array occupies much less space than a Python list.	List occupies much more space than a array.
It is faster as compared to list.	It is slow as compared to NumPy Array.
NumPy array supports vectorized operations.	List does not support vectorized operations.

**Example:** It is possible to convert python list into NumPy array with the help of `numpy.array()` function. The following example illustrates this.

```

File Edit Shell Debug Options Window Help
>>> # Converting List into Array using NumPy
array() function
>>> import numpy as np
>>> L=[11,22,33,44,55]
>>> ar=np.array(L)
>>> print("Array values are : ",ar)
Array values are : [11 22 33 44 55]
>>>
Ln: 9 Col: 0

```

In this example, L = [11,22,33,44,55] is list. The statement ar=np.array(L) will convert this list into array and store into "ar".

**Example:** Following python code illustrates that NumPy array can perform vector addition but List cannot perform it.

```

File Edit Shell Debug Options Window Help
>>> # Program to illustrate the vector addition on NumPy Array but not possible on List
>>> import numpy as np
>>> L=[1,2,3,4]
>>> ar=np.array(L)
>>> print('Adding value to array adds the value to each element of array','\n',ar+5)
Adding value to array adds the value to each element of array
[6 7 8 9]
>>> print('Adding value to list generates Error Message as follows','\n',L+5)
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
    exec(code, self.locals)
  File "<pyshe11#5>", line 1, in <module>
TypeError: can only concatenate list (not "int") to list
>>>
Ln: 16 Col: 0

```

### Accessing Array Elements

It is possible to access an array element by referring to its index number. The indexes in NumPy arrays start with 0, means the first element has index 0, and the second has index 1 and so on. Following example illustrate to access the array elements.

```

File Edit Shell Debug Options Window Help
>>> # Example to illustrate to access elements of NumPy array
>>> import numpy as np
>>> arr = np.array([1, 2, 3, 4])
>>> print('First element of array is : ', arr[0])
First element of array is : 1
>>> print('Second element of array is : ', arr[1])
Second element of array is : 2
>>> print('Third element of array is : ', arr[2])
Third element of array is : 3
>>>
Ln: 12 Col: 0

```

A table is 2-D array with rows and columns, where the dimension represents the row and the index represents the column. To access elements from 2-D arrays, a comma is used to separate integers representing the dimension and the index of the element.

Similarly, to access elements from 3-D arrays we can use comma separated integers representing the dimensions and the index of the element.

Following example illustrate to access the elements of 2-Dim and 3-Dim array.

```

File Edit Shell Debug Options Window Help
>>> # Example to illustrate to acces the elements of 2-Dim and 3-Dim array
>>> import numpy as np
>>> ar2 = np.array([[1,2,3,4,5], [6,7,8,9,10]])
>>> print('Third element of first row is :', ar2[0,2])
Third element of first row is : 3
>>> print('Fifth element of second row is :', ar2[1,4])
Fifth element of second row is : 10
>>> ar3 = np.array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])
>>> print('Third element of second array is :', ar3[0,1,2])
Third element of second array is : 6
>>> print('Second element of second array is :', ar3[0,1,1])
Second element of second array is : 5
>>>
Ln: 15 Col: 0

```

### Data Types in NumPy

By default Python has string, integer, float, boolean data types. NumPy has some extra data types, and refer to data types with one character, like i for integers, u for unsigned integers. The NumPy array object has a property called dtype that returns the data type of the array.

Following example illustrates to create NumPy array of different data types.

```

File Edit Shell Debug Options Window Help
>>> # Example to check the data type of array
>>> import numpy as np
>>> ar1 = np.array([1,2,3,4])
>>> print ('Data type of array is :', ar1.dtype)
Data type of array is : int64
>>> ar2 = np.array(['a', 'b', 'c'])
>>> print ('Data type of array is :', ar2.dtype)
Data type of array is : <U1
>>> ar3 = np.array([1.1,2.2,3.3,4.4])
>>> print ('Data type of array is :', ar3.dtype)
Data type of array is : float64
>>>
Ln: 14 Col: 0

```

### Creating Arrays With a Defined Data Type

The array() function used to create arrays can take an optional argument as dtype that allows to define the expected data type of the array elements. If the data element is not of specified data type, it generates error. Following example illustrates to create an array with specified data types.

```

File Edit Shell Debug Options Window Help
>>> # Example to create an array with the specified data type
>>> import numpy as np
>>> ar1 = np.array([1, 2, 3, 4], dtype='S')
>>> print('The array is :', ar1)
The array is : [b'1' b'2' b'3' b'4']
>>> print('The data type of array is :', ar1.dtype)
The data type of array is : |S1
>>> # Create an array with data type 4 bytes integer
>>> ar2 = np.array([1, 2, 3, 4], dtype='i4')
>>> print('The array is :', ar2)
The array is : [1 2 3 4]
>>> print('The data type of array is :', ar2.dtype)
The data type of array is : int32
>>> ar3 = np.array(['1', '2', 'a'], dtype='i')
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
    exec(code, self.locals)
  File "<pyshell#9>", line 1, in <module>
ValueError: invalid literal for int() with base 10: 'a'
>>>
Ln: 16 Col: 38

```

It is possible to create NumPy array containing the data element of the same data type as well as the data element of different data types. Following example illustrate to create the NumPy array of data element of integer, float, string as well as mixed data types.

### Converting Data Type on Existing Arrays

The data type of an existing array can be changed. To do this make a copy of the array with the astype() method. The astype() function creates a copy of the array, and allows to specify the data



type as a parameter. The data type can be specified using a string, like 'f' for float, 'i' for integer or you can use the data type directly like float for float and int for integer.

```
File Edit Shell Debug Options Window Help
>>> # Change data type from float to integer by using 'i' as parameter
>>> import numpy as np
>>> arr = np.array([1.1,2.2,3.3])
>>> print ('The array created as \n',arr)
The array created as
[1.1 2.2 3.3]
>>> arr1 = arr.astype('i')
>>> print('The array converted to integer is as \n',arr1)
The array converted to integer is as
[1 2 3]
>>> print('The data type of converted array using i is:',arr1.dtype)
The data type of converted array using i is: int32
>>> arr1 = arr.astype('int')
>>> print('The array converted to integer is as \n',arr1)
The array converted to integer is as
[1 2 3]
>>> print('The data type of converted array using int is:',arr1.dtype)
The data type of converted array using int is: int64
>>>
```

Ln: 15 Col: 0

### NumPy Array Reshaping

Reshaping means changing the shape of an array. The shape of an array is the number of elements in each dimension. Reshaping can change the shape of an array. It is possible to add or remove dimensions or change number of elements in each dimension, provided the number of elements should be of exact count. So, it is possible to convert 9 elements of 1 dimensional array to 2-dimensional array, but it is not possible to convert 8 elements of 1 dimensional array to 2-dimensional array, it generates error.

Following example illustrate to reshape the array from 1 dimension to 2 and 3 dimensions.

```
File Edit Shell Debug Options Window Help
>>> # Converting 1-Dim array with 12 elements into a 2-Dim array.
>>> import numpy as np
>>> ar1 = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])
>>> ar2 = ar1.reshape(4, 3)
>>> print('1-Dim array converted to 2-Dim array is as \n',ar2)
1-Dim array converted to 2-Dim array is as
[[ 1  2  3]
 [ 4  5  6]
 [ 7  8  9]
 [10 11 12]]
>>> # Converting 1-Dim array with 12 elements into a 3-Dim array.
>>> ar3 = ar1.reshape(2, 3, 2)
>>> print('1-Dim array converted to 3-Dim array is as \n',ar3)
1-Dim array converted to 3-Dim array is as
[[[ 1  2]
 [ 3  4]
 [ 5  6]]

 [[ 7  8]
 [ 9 10]
 [11 12]]]
>>> # Converting 1D array with 8 elements to a 2D array with 3 elements in each d
imension (will raise an error):
>>> ar = np.array([1, 2, 3, 4, 5, 6, 7, 8])
>>> ar4 = ar.reshape(3, 3)
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py", line 578, in runcode
    exec(code, self.locals)
  File "<pyshell#20>", line 1, in <module>
NameError: name 'arr' is not defined. Did you mean: 'ar1'?
>>>
```

Ln: 55 Col: 0

### NumPy Array Slicing

Slicing means taking elements from one given index to another given index. It is possible to slice instead of index like this: `[start:end]`. It is possible to define the step, like this: `[start:end:step]`. If start or end is not passed, by default the start is considered 0, and end is considered length of array in that dimension. If step is not passed it is considered as 1.

Following example illustrates the array slicing.

```
File Edit Shell Debug Options Window Help
>>> # Example to slice elements from index 2 to index 5 from the array
>>> import numpy as np
>>> arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9])
>>> print('The array created is as \n',arr)
The array created is as
[1 2 3 4 5 6 7 8 9]
>>> print('The array sliced from index 2 to 5 is as \n',arr[2:5])
The array sliced from index 2 to 5 is as
[3 4 5]
>>> print('The array sliced from index 5 is as \n',arr[5:])
The array sliced from index 5 is as
[6 7 8 9]
>>> print('The array sliced from beginning to index 4 is as \n',arr[:4])
The array sliced from beginning to index 4 is as
[1 2 3 4]
>>>
```

Ln: 84 Col: 0

### Negative Slicing

The index of the array when referred from the end by using minus operator, then it is negative indexing. Slicing can be done in steps by specifying steps.

Following example illustrates the negative indexing performed on the array and slicing in steps.

```
File Edit Shell Debug Options Window Help
>>> # Slice from the index 5 to 2 from the end
>>> import numpy as np
>>> arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9])
>>> print ('Slice from the index 5 to 2 from the end\n', arr[-5:-2])
Slice from the index 5 to 2 from the end
[5 6 7]
>>> # Returning element of array in the step of 2
>>> print ('Element of array from index 2 to 8 in step of 2\n', arr[2:8:2])
Element of array from index 2 to 8 in step of 2
[3 5 7]
>>> print ('Element of array in step of 2 from entire array\n', arr[:,2])
Element of array in step of 2 from entire array
[1 3 5 7 9]
>>>
```

Ln: 16 Col: 0

Following examples illustrates some more programs.

```

File Edit Shell Debug Options Window Help
>>> # Python code to create an array with integer values [24,46,57,14,68,34,89,92] and display array values from index 2 to 6 and negative index from 7 to 3
>>> import numpy as np
>>> A=np.array([24,46,57,14,68,34,89,92])
>>> print('Array from index 2 to 6 is \n', A[2:6])
Array from index 2 to 6 is
[57 14 68 34]
>>> print('Array from negative index 7 to 3 is \n', A[7:3:-1])
Array from negative index 7 to 3 is
[92 89 34 68]
>>>
>>> # Python code to create an array with integer values from 0 to 9 and display its values in reverse order
>>> import numpy as np
>>> arr= np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
>>> print('Array created with values \n', arr)
Array created with values
[0 1 2 3 4 5 6 7 8 9]
>>> print('Array in reverse order is \n', arr[::-1])
Array in reverse order is
[9 8 7 6 5 4 3 2 1 0]
>>>
>>> # Python code to create 2 arrays as 'a' and 'b' and display sum of array values.
>>> import numpy as np
>>> a=np. array([2,3,4,5])
>>> b=np.array([6,7,8,9])
>>> print('Sum of values of two array is \n', a + b )
Sum of values of two array is
[ 8 10 12 14]
>>>
>>> # Python code to create an array as 'a' and increase array values by 10.
>>> import numpy as np
>>> a=np. array([ 2,3,4,5 ])
>>> print('Array values after adding 10 \n', a + 10)
Array values after adding 10
[12 13 14 15]
>>>
Ln: 37 Col: 0

```

## CHECK YOUR PROGRESS

### A. Multiple Choice Questions

- Which of the following is not valid function? (a) Mod() (b) Add() (c) Maxx() (d) Divide()
- Point out the Correct Statement: (a) We cannot change the size of NumPy array (b) NumPy array can contain elements of non-homogenous type (c) Python array occupy less space than a List. (d) All of the Above.
- Which of the following is not a valid data type of elements of NumPy array created by the linspace() method? (a) float64 (b) int32 (c) bool (d) int16
- What is the use of reshape () method? (a) To create 2D array from 1D array (b) To create 1D array from 2D array (c) Both 1 and 2 (d) None of the above
- We can convert list into array with the help of \_\_\_\_\_ function. (a) id() (b) itemtype() (c) array() (d) None of the above

### B. Fill in the Blank

- Slicing is specified by using \_\_\_\_\_ operator.

2. NumPy array supports \_\_\_\_\_ operations which is not supported in Python List.
3. NumPy is an \_\_\_\_\_ module of Python.
4. NumPy support \_\_\_\_\_ array.
5. We can convert list into array with the help of \_\_\_\_\_ function.

**C. State whether True or False**

**D. Answer the following in short**

1. How One Dimension array is different from Two Dimension array?
2. How head() and head(3) is different with each other. Justify your answer with suitable example.
3. What is the use of MOD() function? Explain with example.
4. What is use of hstack() and vstack()? Explain with example.
5. What is use of concatenate ( ) function? Explain with example.

**Practical Exercise**

1. Write a python program to reverse the rows in a 2D numpy array?
1. Write a python program for: Given a 1D array to negate all elements which are between 3 and 8.
2. Write a python program for to Create a integer array from a range between 100 to 200 such that the difference between each element is 10
3. Write a Program to store 30 zeros in an array.

## Session 5. Pandas and Series in Python

A pandas Series is very similar to a one-dimensional NumPy array consists of an array of data (values), and an array of labels (indices). It has additional functionality that allows values in the Series to be indexed using labels. A NumPy array does not have the flexibility to do this. As the most basic element in Pandas, a Pandas Series is also the building block of Pandas DataFrames. Pandas or Python Pandas is a library of Python which is used for data analysis. The term Pandas is derived from “*Panel Data System*”, which is an econometric term for multidimensional, structured dataset. Pandas has become a popular option for Data Analysis. Pandas provide various tools for data analysis in very simple and easy form. Pandas are an Open Source, BSD library specially built for Python Programming language. Pandas offer high performance, easy to use data structure and data analysis tools for real world need of individual or any organisation. The main author of Pandas is Wes McKinney.

**Key Features of Pandas**

- Pandas, is the most popular library in Scientific Python ecosystem for data analysis.
- Quick and efficient data manipulation and analysis.
- It has functionality to find and fill missing data.
- It allows you to apply operations to independent groups within the data.
- It supports reshaping of data into different forms.
- It supports advanced time-series functionality (which is the use of a model to predict future values based on previously observed values).
- It supports visualization by integrating matplotlib.
- Pandas is best for handling huge tabular (like excel, mysql) data sets comprising different data formats.

- Tools for loading data from different file formats into in-memory data objects.
- Label-based Slicing, Indexing, and Subsetting can be performed on large datasets.
- Merges and joins two datasets easily.
- Pivoting and reshaping data sets
- Easy handling of missing data (represented as NaN) in both floating point and non-floating point data.
- Represents the data in tabular form.
- Size mutability: DataFrame and higher-dimensional object columns can be added and deleted.
- It provides time-series functionality.
- Effective grouping by functionality for splitting, applying, and combining data sets.

### Installation of Pandas

If Python and PIP already installed on a system, then Pandas can be installed using this command in Windows:

```
C:\Users\Your Name>pip install pandas
```

If this command fails, then use a python distribution that already has Pandas installed.

In Linux use the following command to install Pandas.

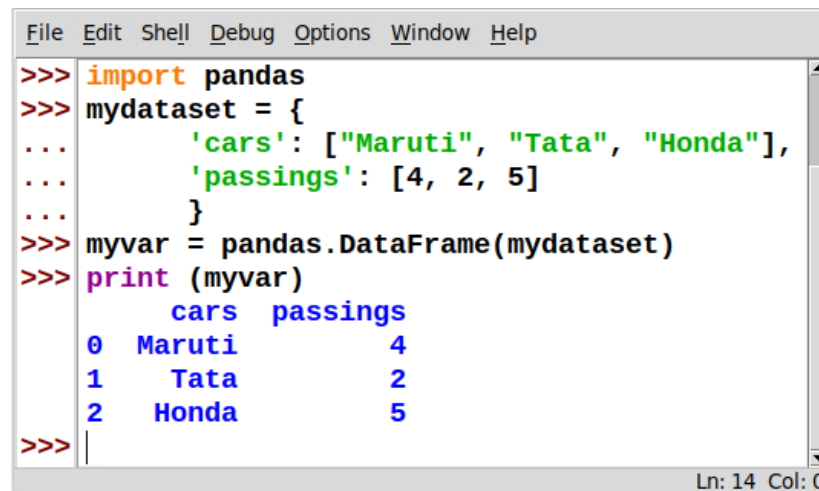
```
$ sudo apt install python3-pandas
```

### Import Pandas

Once Pandas is installed, import it in your applications by adding the import keyword as follows:

```
import pandas
```

Now Pandas is imported and ready to use. Let us test it using the following code.



```
File Edit Shell Debug Options Window Help
>>> import pandas
>>> mydataset = {
...     'cars': ["Maruti", "Tata", "Honda"],
...     'passings': [4, 2, 5]
... }
>>> myvar = pandas.DataFrame(mydataset)
>>> print (myvar)
      cars  passings
0  Maruti         4
1   Tata         2
2  Honda         5
>>>
```

### Pandas as pd

In Python alias are an alternate name for referring to the same thing. Pandas is usually imported under the pd alias. To create an alias with the as keyword while importing as.

```
import pandas as pd
```

Now the Pandas package can be referred to as pd instead of pandas. Here **pd** is an object of pandas library to which you can use in your program.

Pandas is a high level data manipulation tools used for analyzing data. It is very easy to import and export pandas libraries which has very rich set of functions. Pandas have three important data structure as under – *Series*, *DataFrame*, *Panel*.

Pandas data frame can contain different data types (int, float, double and string). Pandas data frame have column name, make it is keep track of data. Pandas are used when data is available in any tabular format like in a spreadsheet or in a database table. In this chapter we will discuss Series data structure only as others are beyond the scope of this book.

### Series

A Pandas Series is like a column in a table. It is a one-dimensional array holding data of any type. It contains a sequence of value of any data type like int, float or string. By default the index will be of type integer and start with zero. Series index can also be given by programmer like in the form of dictionary keys. It could be of numeric or character or even as string given by the programmer.

Series is a 1 dimension array of homogeneous (same type) elements with mutable (can be change) values of immutable size, i.e. size of the series once created cannot be changed.

### Creation of Series Objects

There are many ways to create series. Series can be created with the help of list, dictionary, Series () function, empty () function, zero () function.

#### Creation of series using list

Following program illustrates to crate empty series and non empty series using Series() function.

```
File Edit Shell Debug Options Window Help
>>> # Creating empty series using Series() function
>>> import pandas as pd
>>> ob = pd.Series ()
>>> print('The empty series is created as \n', ob)
The empty series is created as
Series([], dtype: float64)
>>>
>>> # Creating non-empty series using Series() function
>>> ob = pd.Series(range(5))
>>> print('The non empty series is created as \n', ob)
The non empty series is created as
0    0
1    1
2    2
3    3
4    4
dtype: int64
>>> import pandas as pd
>>> obj = pd.Series([3,5,4,4.5])
>>> print('The non empty series is created as \n', obj)
The non empty series is created as
0    3.0
1    5.0
2    4.0
3    4.5
dtype: float64
>>>
```

Here in this list, 3 values are integer and one is float type, so finally series type will be float.

#### Creation of series using dictionary

Following program illustrates to crate series with the help of dictionary. Dictionary's keys act as series index and dictionary's value's act as series value.

```

File Edit Shell Debug Options Window Help
>>> # Creation of series using dictionary
>>> import pandas as pd
>>> obj = pd.Series({'Jan':31, 'Feb':28, 'Mar':31})
>>> obj
Jan      31
Feb      28
Mar      31
dtype: int64
>>> |
Ln: 11 Col: 0

```

In this example, “Jan”, “Feb”, “Mar” is called series index and 31, 28, 31 are as series values and the type of series is int type.

### Creation of series with Scalar value

Let us understand different ways to create series using pandas library using following code.

```

File Edit Shell Debug Options Window Help
>>> # Creation of series with Scalar value
>>> import pandas as pd
>>> a=pd.Series(10, index=range(0, 3))
>>> a
0      10
1      10
2      10
dtype: int64
>>> |
Ln: 11 Col: 0

```

In this code snippet, range (0, 3) function will generate indexes 0, 1 and 2 given by programmer. There is only one value “10” which will store in all indexes.

```

File Edit Shell Debug Options Window Help
>>> b=pd.Series(15, index=range(1, 6, 2))
>>> b
1      15
3      15
5      15
dtype: int64
>>> |
Ln: 19 Col: 0

```

In this code snippet, range () functions is used to create *index 1, 3 and 5*. Here index range is 1 to 6, which will start from 1 and increment by 2 and goes up to 5 only. There is only one value “15” given by programmer which will store in all indexes.

```

File Edit Shell Debug Options Window Help
>>> a=pd.Series('Welcome to My School',
index=['Hema', 'Rahul', 'Anup'])
>>> a
Hema      Welcome to My School
Rahul     Welcome to My School
Anup      Welcome to My School
dtype: object
>>> |
Ln: 39 Col: 0

```

Indexes can also be given in the form list that is ‘Hema’, ‘Rahul’, ‘Anup’. This is given by programmer and there is only one value “Welcome to CIVE” which will store in all indexes.

**Example:** Following examples demonstrate the use of pandas library to demonstrate mathematical function/Expression in series.

```
File Edit Shell Debug Options Window Help
>>> import pandas as pd
>>> import numpy as np
>>> a=np.arange(9,13)
>>> a
array([ 9, 10, 11, 12])
>>> |
Ln: 52 Col: 0
```

arange() is numpy library function which stored 9 to 12 numbers into “a” object and we can used this “a” object values as series index and we can used “**data= a \* 2**” for data values.

```
File Edit Shell Debug Options Window Help
>>> import pandas as pd
>>> import numpy as np
>>> a=np.arange(9,13)
>>> a
array([ 9, 10, 11, 12])
>>> od=pd.Series(index=a, data=a*3)
>>> od
9      27
10     30
11     33
12     36
dtype: int64
>>> |
Ln: 16 Col: 0
```

arange() is numpy library function which stored 9 to 12 numbers into “a” object and we can used this “a” object values as series index and we can used “**data= a \*\* 3**” for data values. \*\* means a raised to power 3.

### Series Object Attributes

Series Attribute	Description
Series.index	Returns the index of a series
Series.values	Returns values of the series in the form of ndarray.
Series.dtype	Returns the data type of the data object
Series.shape	Returns tuple of the shape of underlying data
Series.nbytes	Return number of bytes of underlying data. The formula is: number of element in series * data types size.
Series.ndim	Returns the number of dimension in series.
Series.size	Returns number of elements present in series



Let us create a series to display its different attributes.

```
File Edit Shell Debug Options Window Help
>>> # Creating a series to display its different attributes.
>>> import pandas as pd
>>> print('Index of series \n',s.index)
Index of series
Index(['a', 'b', 'c', 'd', 'e'], dtype='object')
>>> print('Values of series \n',s.values)
Values of series
[ 1  4  7 10 13]
>>> print('Tuple of the shape of underlying data \n',s.shape)
Tuple of the shape of underlying data
(5,)
>>> print('Number of elements present in series \n',s.size)
Number of elements present in series
5
>>> print('Number of bytes underlying data in series \n',s.nbytes)
Number of bytes underlying data in series
40
>>> print('Number of dimension \n',s.ndim)
Number of dimension
1
>>>
```

In the above example:

s.index: It will display index of the series i.e. Index(['a', 'b', 'c', 'd', 'e'] dtype = 'object')

s.values: It will display values of the series i.e. array([1,4,7,1,13], dtype = int64).

s.shape: It will display tuple of the shape of underlying data i.e. (5,)

s.size: It will display number of elements present in series i.e. 5

s.nbytes: It will display number of bytes of underlying data(formula is: number of element present in series \* (multiply by) size of individual element) i.e. 40

s.ndim: It will display the number of dimension(1-D, 2-D) i.e. 1

It is possible to display series data values as per user need. It is known as Series Slicing. For this we need to pass three parameters i.e. [<start>:<stop>:<step>] It is illustrated in the following code.

```

File Edit Shell Debug Options Window Help
>>> import pandas as pd
>>> import numpy as np
>>> a=np.arange(9,13)
>>> ob=pd.Series(index=a, data=a**2)
>>> print('Values in the series are \n',ob)
Values in the series are
  9    81
 10   100
 11   121
 12   144
dtype: int64
>>> print('Value at index 10 are \n',ob[10])
Value at index 10 are
 100
>>> print('Value at index 10 are \n',ob[2:4])
Value at index 10 are
 11   121
 12   144
dtype: int64
>>> print('Series values starting at index 1 are \n',ob[1:])
Series values starting at index 1 are
 10   100
 11   121
 12   144
dtype: int64
>>> print('Series values starting at index 0 upto with step 2\n',ob[0::2])
Series values starting at index 0 upto with step 2
  9    81
 11   121
dtype: int64
>>> print('Series values in reverse order\n',ob[::-1])
Series values in reverse order
 12   144
 11   121
 10   100
  9    81
dtype: int64
>>>
Ln: 41 Col: 0

```

In the above example,

**ob** – It will display all values in the series.

**Ob[10]** – It will display the value at index 10 means 100.

**ob[2 : 4]** – It will display values 121, 144 at 11, 12 as per given index by user. Python automatically generates index 0, 1, 2, 3 internally against user's indexes 9, 10, 11 and 12. So in this code snippet indexes will be treated as 2, 3 means 11 and 12 only. Index no 4 is not included here.

**ob[1: ]** – It will display series values which start with “1” in index up-to end in normal order i.e. 100, 121 and 144

**ob[0 : 2]** – Here starting index is 0, there is no stop point but step is 2. So, it will display alternative values 81 and 121.

**ob[ : -1]** – It will display series values in reverse order, because there is no start and stop point. The step is given as -1 means index will increase negatively. This is similar to list slicing concept.

**Example:** Write a python code to modify/update a data series.

```

File Edit Shell Debug Options Window Help
>>> # Example to modify/update a data series
>>> import pandas as pd
>>> s=pd.Series(range(1,15,3),index=[x for x in 'abcde'])
>>> print('The data series is created with the elemens as \n',s)
The data series is created with the elemens as
a    1
b    4
c    7
d   10
e   13
dtype: int64
>>> s['c']=25
>>> print('The data series after replacing the value of c to 25\n',s)
The data series after replacing the value of c to 25
a    1
b    4
c   25
d   10
e   13
dtype: int64
>>>
Ln: 46 Col: 0

```

Here in this example, we have update the value of series at index 'c'.

**Example :** Write a python code to modify / update index of a data series.

```

File Edit Shell Debug Options Window Help
>>> # Example to modify/update a data series
>>> import pandas as pd
>>> s=pd.Series(range(1,15,3),index=[x for x in 'abcde'])
>>> print('The data series is created with the elemens as \n',s)
The data series is created with the elemens as
a    1
b    4
c    7
d   10
e   13
dtype: int64
>>> s.index=['u','v','w','x','y']
>>> print('The data series after modifying index is\n',s)
The data series after modifying index is
u    1
v    4
w    7
x   10
y   13
dtype: int64
>>>
Ln: 88 Col: 0

```

```

File Edit Shell Debug Options Window Help
>>> # Example to modify/update a data series
>>> import pandas as pd
>>> s=pd.Series(range(1,15,3),index=[x for x in 'abcde'])
>>> print('The data series is created with the elemens as \n',s)
The data series is created with the elemens as
a    1
b    4
c    7
d   10
e   13
dtype: int64
>>> s.index=['u','v','w','x','y']
>>> print('The data series after modifying index is\n',s)
The data series after modifying index is
u    1
v    4
w    7
x   10
y   13
dtype: int64
>>>
Ln: 88 Col: 0

```

Here in this example, the series is created with the index using for loop as ['a', 'b', 'c', 'd', 'e']. Then the index of the series is changed to ['u', 'v', 'w', 'x', 'y']. So the new indexes in series are u, v, w, x and y.

**head() and tail () Function**

Python provides two important functions to access the data values from the series directly. You can access starting values using head() function and or last values using tail() function.

**head (<n>)** : The head () function is used to display first/top n rows from a Series. By default it will display top/first 5 rows.

**tail(<n>)** : The tail () function is used to display last/bottom n rows from a Series. By default it will display last / bottom 5 rows.

Following program illustrates the use of head() and tail() function.

```
File Edit Shell Debug Options Window Help
>>> # Program to illustrate the head() and tail() function
>>> import pandas as pd
>>> s=pd.Series(range(1,20,2),index=[x for x in range(0,10)])
>>> print('Display the series values from 1 to 20 with step 2\n',s)
Display the series values from 1 to 20 with step 2
  0    1
  1    3
  2    5
  3    7
  4    9
  5   11
  6   13
  7   15
  8   17
  9   19
dtype: int64
>>> print('Display default top values of series\n',s.head())
Display default top values of series
  0    1
  1    3
  2    5
  3    7
  4    9
dtype: int64
>>> print('Display top 3 values of series \n',s.head(3))
Display top 3 values of series
  0    1
  1    3
  2    5
dtype: int64
>>> print('Display default bootom values of series\n',s.tail())
Display default bootom values of series
  5   11
  6   13
  7   15
  8   17
  9   19
dtype: int64
>>> print('Display bottom 2 values of series \n',s.tail(2))
Display bottom 2 values of series
  8   17
  9   19
dtype: int64
>>>
```

**Vector Operations and Arithmetic Operations on series**

You can do various arithmetic operations on series data in python just like you perform it on normal variable.

Following program illustrate vector operations and arithmetic operations on series. For this first create the three series as shown in the following code.

```

File Edit Shell Debug Options Window Help
>>> import pandas as pd
>>> s1=pd.Series(range(11,18))
>>> s2=pd.Series(range(41,48))
>>> s3=pd.Series(range(101,106), index=[10,20,30,40,50
])
>>> print('Series s1 created with values ranging from
11 to 18 from index 0 to 6 \n',s1)
Series s1 created with values ranging from 11 to 1
8 from index 0 to 6
  0    11
  1    12
  2    13
  3    14
  4    15
  5    16
  6    17
dtype: int64
>>> print('Series s2 created with values ranging from
41 to 48 from index 0 to 6 \n',s2)
Series s2 created with values ranging from 41 to 4
8 from index 0 to 6
  0    41
  1    42
  2    43
  3    44
  4    45
  5    46
  6    47
dtype: int64
>>> print('Series s3 created with values ranging from
101 to 106 with index as 10,20,30,40,50\n',s3)
Series s3 created with values ranging from 101 to
106 with index as 10,20,30,40,50
  10    101
  20    102
  30    103
  40    104
  50    105
dtype: int64
Ln: 37 Col: 26

```

Now perform arithmetic operation (+, -, \* and /) on s1 and s2 as the indexes are same in both but not using s3 as it has different indexes.

Following code illustrates the arithmetic operation and vector operation on series.

```

File Edit Shell Debug Options Window Help
>>> # Arithmetic operation of adding elements of series s1 and s2
>>> print('Arithmetic operation to add the elements of series s1 a
nd s2 to create a new series \n', s1+s2)
Arithmetic operation to add the elements of series s1 and s2 t
o create a new series
 0    52
 1    54
 2    56
 3    58
 4    60
 5    62
 6    64
dtype: int64
>>> # Vector operation of adding a vector value to series s1
>>> print('Vector value 2 is added to series s1 to form a new seri
es \n', s1+2)
Vector value 2 is added to series s1 to form a new series
 0    13
 1    14
 2    15
 3    16
 4    17
 5    18
 6    19
dtype: int64
>>> # Vector operation to multiply each elements of series s1 by 2
>>> print('Vector value 2 is multiplied to series s1 to form a new
series \n', s1*2)
Vector value 2 is multiplied to series s1 to form a new series
 0    22
 1    24
 2    26
 3    28
 4    30
 5    32
 6    34
dtype: int64
>>> # Addition of series with different index is not possible
>>> print('Arithmetic operation to add the elements of series s1 a
nd s3 is not possible due to different index\n', s1+s3)
Arithmetic operation to add the elements of series s1 and s3 i
s not possible due to different index
 0    NaN
 1    NaN
Ln: 131 Col: 0

```

In the above code the arithmetic and vector operations performed on series are as follows.

**s1+s2:** This operation will add each element of series s1 and s2. It will successfully done as both the series have the similar in nature in term of their index number.

**s1 + 2:** This operation will add 2 to each item of the data series. So we can get 13, 14, 15, 16, 17, 18, and 19 instead of 11, 12, 13, 14, 15, 16 and 17.

**s1 \* 2:** This operation will multiply each item of the data series by 2. So we will get 22, 24, 26, 28, 30, 32 and 34 instead of 11, 12, 13, 14, 15, 16 and 17.

**s1 + s3:** This operation will not do appropriately as both series had different types of indexes. The index of series s1 is [0,1,2,3,4,5,6] while series s3 has index [10,20,30,40,50] . if indexes are not matched then Python will result in NaN (Not a number) in Output.

### Relational Operations on series

It is also possible to perform various relational operations (>, <, >=, <=, ==, !=) on series data in python to generate Boolean results in the form of True/False. These operations are also known as filtration in python.

First create a series and then perform the relational operations and delete data from data Series as shown in the following code.

```

File Edit Shell Debug Options Window Help
>>> # Creating a Series to do relational operations
>>> import pandas as pd
>>> s1=pd.Series(data=[12,76,9,34,65])
>>> print('Series s1 created with specified data values with index
from 0 to 4 is \n',s1)
Series s1 created with specified data values with index from 0
to 4 is
  0    12
  1    76
  2     9
  3    34
  4    65
dtype: int64
>>> print('Data values in Series s1 which satisfy the conditon gene
rates the result as True else False\n',s1<50)
Data values in Series s1 which satisfy the conditon generates t
he result as True else False
  0     True
  1    False
  2     True
  3     True
  4    False
dtype: bool
>>> print('Data values in Series s1 which satisfy the conditon gene
rates the result as original values. The data values which dono
t satisfy the condition will not be displayed\n',s1[s1<50])
Data values in Series s1 which satisfy the conditon generates t
he result as original values. The data values which donot satis
fy the condition will not be displayed
  0    12
  2     9
  3    34
dtype: int64
>>> print('Drop function delete data value at the specified index n
umber\n',s1.drop(3))
Drop function delete data value at the specified index number
  0    12
  1    76
  2     9
  4    65
dtype: int64
>>>
Ln: 165 Col: 0

```

### Difference between NumPy array and Series data objects

Series	Numpy
In series data structure, we can create our own type of index to access data elements from a series.	NumPy array data structures are accessed via their index which is always an integer value.
The index could be a numbers or character or even string.	It always starts with 0 and goes up-to N where N is size of array.
If two series are not aligned/ similar in nature, NaN or missing values are generated.	There is no concept of NaN values. If the size of array mismatched, vector operation is not possible.
Series require more memory.	NumPy occupyes lesser memory.

Following program illustrates that arithmetic addition operation is not possible on two arrays of different size.

```
File Edit Shell Debug Options Window Help
>>> import numpy as np
>>> a=np.array([1,2,3])
>>> b=np.array([1,2,3,4,5,6])
>>> print('Addition of two arrays of different
size is not possible',a+b)
Traceback (most recent call last):
  File "/usr/lib/python3.10/idlelib/run.py"
, line 578, in runcode
    exec(code, self.locals)
  File "<pyshell#60>", line 1, in <module>
ValueError: operands could not be broadcast
together with shapes (3,) (6,)
>>>
Ln: 193 Col: 0
```

### Summary

- Data Structure refer to specialized way of storing data so as to apply a specific type of functionality on them.
- Series is a pandas data structure the represents a 1D array -like object containing an array of data and an associated array of data labels call its INDEX.
- The shape of a series object tells how big it is, i.e. how many elements it contains including missing or empty values(NaN).
- If you use len() function on series object , then it return total elements in it including NaNs but Series.count() return only the count of non- NaN values in a series object.
- When you perform arithmetic operations on 2 series type objects, the data is aligned on the basis of matching indexes.
- NaN means “Not a Number”.
- Missing values are a hindrance in data analysis and must be handled properly.

### Solved Practical

1. Write a python code to create a series using list [3, 4, 5, 6, 7] and ['a', 'b', 'c', 'd'] .
2. Write a python code to create a series using an ndarray that has 19 elements in the range 100 to 201

```
File Edit Shell Debug Options Window Help
>>> # Create a series using list [3,4,5,6,7]
>>> import pandas as pd
>>> s1=pd.Series([3,4,5,6,7])
>>> print('The series using list is\n',s1)
The series using list is
0    3
1    4
2    5
3    6
4    7
dtype: int64
>>> # Create a series using list [3,4,5,6,7]
>>> import pandas as pd
>>> s2=pd.Series(['a','b','c','d'])
>>> print('The series using list is\n',s2)
The series using list is
0    a
1    b
2    c
3    d
dtype: object
Ln: 128 Col: 0
```



```

File Edit Shell Debug Options Window Help
>>> # Create a series using an ndarray that has
>>> 19 elements in the range 100 to 201
>>> import pandas as pd
>>> import numpy as np
>>> z1=pd.Series(np.linspace(100,201,20))
>>> print('The series created using ndarray is
>>> \n',z1)
The series created using ndarray is
  0    100.000000
  1    105.315789
  2    110.631579
  3    115.947368
  4    121.263158
  5    126.578947
  6    131.894737
  7    137.210526
  8    142.526316
  9    147.842105
 10    153.157895
 11    158.473684
 12    163.789474
 13    169.105263
 14    174.421053
 15    179.736842
 16    185.052632
 17    190.368421
 18    195.684211
 19    201.000000
dtype: float64
>>>
Ln: 122 Col: 19

```

3. Write a python code to create a series using a dictionary that store number of days in a months.

```

File Edit Shell Debug Options Window Help
>>> # Create a series using a dictionary that
>>> store number of days in a months
>>> import pandas as pd
>>> d={'jan':31, 'feb':28, 'march':31}
>>> p1=pd.Series(d)
>>> print('The series created using dictionary
>>> as\n',p1)
The series created using dictionary as
jan      31
feb      28
march    31
dtype: int64
>>>
Ln: 15 Col: 0

```

4. Write a python code to create a series using list for player name p = ["Virat", "Khan", "Karan", "Sawan"] and run score r = [12, 34, 56, 78] by the player , use player name as series index and run as series values.

```

File Edit Shell Debug Options Window Help
>>> # Create a series using list for player name
>>> # as index and run as series values
>>> import pandas as pd
>>> p=["Virat","Khan","Karan", "Sawan"]
>>> r=[12,34,56,78]
>>> p1=pd.Series(data=r, index=p)
>>> print('The series created with name of player
>>> as index and run scored as values \n',p1)
The series created with name of player as index
and run scored as values
Virat    12
Khan     34
Karan    56
Sawan    78
dtype: int64
>>>
Ln: 29 Col: 0

```

## CHECK YOUR PROGRESS

### A. Multiple Choice Questions

- Which of the following option is used to create empty series, (a) pd.Series(empty) (b) pd.Series(np.NaN) (c) pd.Series() (d) all of the above
- Which of the following option is used to display 3<sup>rd</sup> element of a series of S series (a) S[ : 3] (b) S[2] (c) S[3] (d) S[: 2]
- How many number of rows are displayed by default using head() and tail() function (a) 1 (b) 9 (c) 3 (d) 5
- Which of the following attribute is used to get the number of bytes of the series data (a) index (b) size (c) itemsize (d) ndim
- The function S.head(3) will display (a) first 5 rows from series (b) last 5 rows from series (c) no output (d) first 3 rows from series

### B. Fill in the blanks

- Pandas is popular for \_\_\_\_\_ .
- A \_\_\_\_\_ is a pandas data structure that represents a 1 D array like object.
- A \_\_\_\_\_ is a pandas data structure that represents a 2 D array like object.
- Series is \_\_\_\_\_ ( mutable / immutable).
- Data Frame is \_\_\_\_\_ data structure.

### C. State whether True or False

- Series values is mutable.(T)
- Series index is mutable. (F)
- Data Frame values is immutable. (T)
- Can we increase the size of Data Frame? (T)
- There is no difference between numpy array and series(F)

### D. Answer the following questions in short

- What is the difference numpy array and Series?
- What is the difference 1D array and 2D array?
- What is the difference numpy array and Series?
- What is the difference between 1D numpy array and list?
- Write any four data types support by numpy array and Series.

### Practical Exercise

- Write a python code to create series with 6 random integer and having index as [1,2,3,4,5,6].
- Write a python code to create series with 6 random integer change their index.

## Session 6. Graphical Representation using Matplotlib

Data is very important for any organization for any type of acknowledgement, research, analysis, decision making and future forecasting. From a huge amount of data, picking up or point out the required data at right time is very important and tedious task.

Data visualisation is the graphical representation of data or information. It is used to display data in more expressive way to fulfill the audience requirement. The Data visualisation in the form of charts, graphs, animation, and maps are very easy and simple to understand the trends, outliers, and patterns in data. Data visualization techniques for such big data are very important for the purpose of analysis of data.

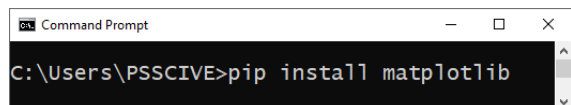
### Use of PYPLOTT MATPLOTLIB Library

The Matplotlib is a python library that provides many interfaces and functionality for 2D-graphics similar to MATLAB. Python scripts can be used to create 2D graphs and plots using the Matplotlib module. With features to control line styles, font attributes, formatting axes, and other features, it offers a module called pyplot that makes things simple for plotting. It offers a huge range of graphs and plots, including error charts, bar charts, power spectra, and histograms. It is combined with NumPy to provide a powerful open source MatLab substitute environment.

### Installing Matplotlib

To install Matplotlib library, you need to open command prompt with administrator rights and make sure internet connectivity is on. Matplotlib library and it's all dependencies can be easily downloaded as binary file (pre-compiled) package from internet very easily.

To install Matplotlib in Windows operating system, issue the following command on the command prompt.



```

C:\Users\PSSCIVE>pip install matplotlib
  
```

It will give the message for successful installation of Matplotlib library.

To install Matplotlib in Ubuntu Linux run the following command in the command prompt.

### pip install matplotlib

It will start downloading and installing packages related to the matplotlib library.

To verify that matplotlib is successfully installed, execute the following command in the Python idle. If matplotlib is successfully installed, the version of matplotlib installed will be displayed.

To find out version of Matplotlib, open the Python Idle and find the version using following command. The version of Matplotlib is displayed as '3.5.1'

```

>> import matplotlib
>> matplotlib.__version__
'3.5.1'
  
```

The version of Matplotlib is displayed as '3.5.1'

You can find what directory Matplotlib is installed in by importing it and printing the `__file__` attribute:

### Importing Pyplot

Pyplot is a set of functions in Matplotlib library. A figure's elements can be changed by using its functions, which include constructing a figure, a plotting area, plot lines, adding plot labels. Keep in mind that you can alter the line's colour and style by including the arguments *linecolor* and *linestyle*. You need to use import keyword as under to use it.

**import matplotlib.pyplot as pp**

Here pp is user defined object for pyplot. You can use all the functions in pyplot library using this object as per your need.

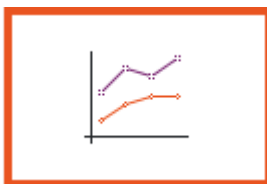
**Basics of Simple Plotting**

Graphical representation of compiled data is known as data visualization. With the help of Pyplot we can create following type Graphs or Charts.

**Line chart** – Line charts are used to represent the relation between two data X and Y on a different axis.

**Bar Chart** – A bar plot or bar chart is a graph that represents the category of data with rectangular bars with lengths and heights that is proportional to the values which they represent. The bar plots can be plotted horizontally or vertically.

**Pie Chart** – A Pie Chart is a circular statistical plot that can display only one series of data. The area of the chart is the total percentage of the given data. The area of slices of the pie represents the percentage of the parts of the data. The slices of pie are called wedges.



Line chart



Bar Chart



Pie Chart

**CREATING LINE CHART**

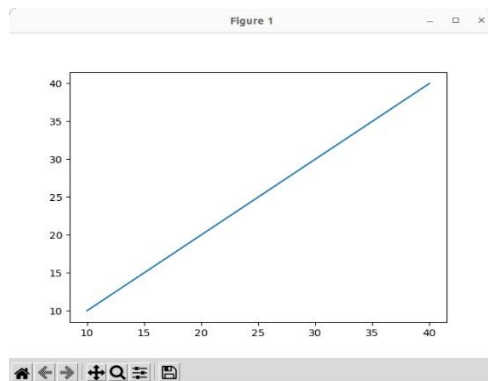
A line chart or line graph is a type of chart which displays information as a series of data points called 'markers' connected by a straight line segments. The pyplot interface offers *plot()* function for creating a line graph.

**Example 1:** Create Line Graph with the help of two given List.

The python code is shown below with the list a and b. The output as plotted graph is shown below the program.

```
File Edit Shell Debug Options Window Help
>>> # Create Line Graph with the help of two given List a & b
>>> import matplotlib.pyplot as pp
>>> a=[10,20,30,40]
>>> b=[10,20,30,40]
>>> pp.plot(a,b)
[<matplotlib.lines.Line2D object at 0x7f105d478400>]
>>> print('The line graph is plotted as')
The line graph is plotted as
>>> pp.show()
Ln: 26 Col: 0
```

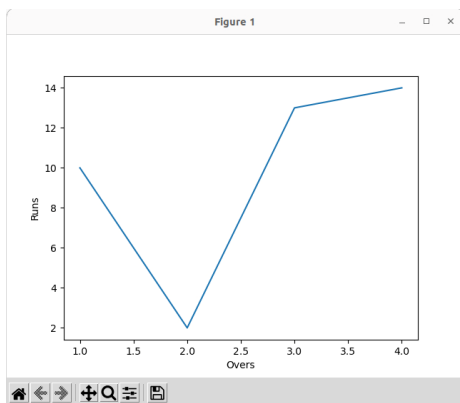
The output is shown with the line graph plotted as below.



**Example 2:** Write a program to plot a Line Graph of number of runs and over provided in two different lists.

The python program and output is given below.

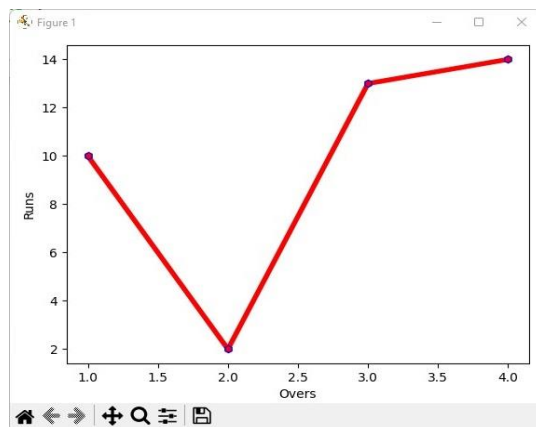
```
File Edit Format Run Options Window Help
# Plot a line graph of runs and over
import matplotlib.pyplot as pp
a=[1,2,3,4]
b=[10,2,13,14]
pp.xlabel("Overs")
pp.ylabel("Runs")
pp.plot(a,b)
pp.show()
Ln: 8 Col: 0
```



**Example 3.** Let us modify previous example for changing marker size, edge color and increase the line width using various parameters of plot () function.

```
File Edit Format Run Options Window Help
# Changing marker size, edge color, line width
import matplotlib.pyplot as pp
a=[1,2,3,4]
b=[10,2,13,14]
pp.xlabel("Overs")
pp.ylabel("Runs")
pp.plot(a,b, "r", marker="h", markersize=6,
        markeredgecolor="b", linewidth=4)
pp.show()
Ln: 9 Col: 9
```

Here in this example, the parameters like *marker*, *markersize*, *markeredgecolor* and *linewidth* are used to give specification in line plot.



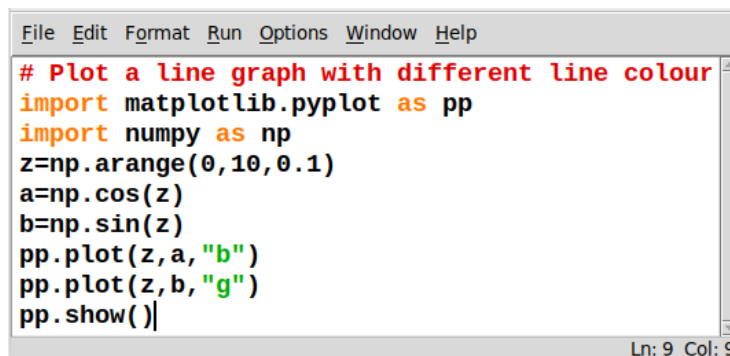
The various codes for marker parameter are given below.

Character	Description
'p'	pentagon marker
'*'	star marker
'h'	hexagon1 marker
'H'	hexagon2 marker
'+'	plus marker
'x'	x marker
'D'	diamond marker
'd'	thin_diamond marker
' '	vline marker
'_'	hline marker

**Example 4.** Write a python code for creating a line chart with different line color.

In this example we have used arange() function. Variable z is initialized with multiple instances of values from 0 to 10 with the interval of 0.1 using this function. These multiple values of z will be passed to sin and cos functions respectively and result will be stored in variable a, b.

Now using pp object values of z will be plot in line chart along with a and b within blue and green color respectively.



```
File Edit Format Run Options Window Help
# Plot a line graph with different line colour
import matplotlib.pyplot as pp
import numpy as np
z=np.arange(0,10,0.1)
a=np.cos(z)
b=np.sin(z)
pp.plot(z,a,"b")
pp.plot(z,b,"g")
pp.show()
Ln: 9 Col: 9
```

In the above program, the various color codes as given below can be used while preparing chart as below.

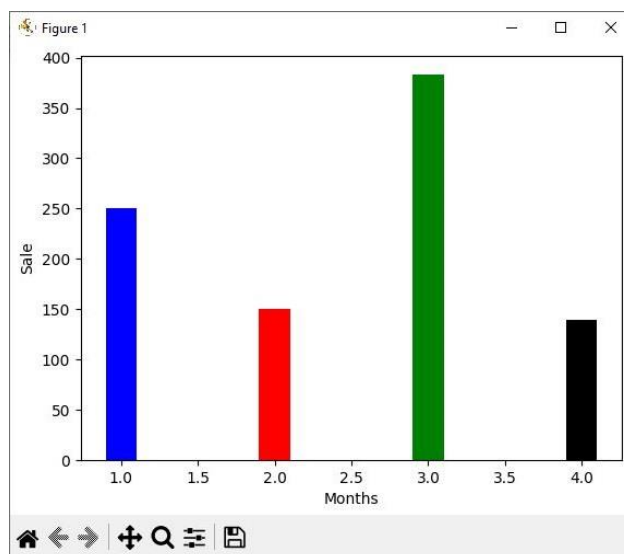
Code	Colour Name
"b"	Blue
"g"	Green
"r"	Red
"c"	Cyan
"m"	Magenta
"y"	Yellow
"k"	Black
"w"	White

#### CREATING BAR CHART

A Bar Graph/Chart a graphical display of data using bars of different heights. We can use `bar()` and `barh()` for this purpose. We can use width and color parameter of bar Graph

**Example 5:** Let us create bar Graph for monthly sale of an electronic items shop using list.

```
File Edit Format Run Options Window Help
# Creating bar graph for monthly sale of electronic item
import matplotlib.pyplot as pp
a=[1,2,3,4]
b=[250,150,383,140]
pp.xlabel("Months")
pp.ylabel("Sale")
pp.bar(a,b,width=[1/5],color=['b','r','g','k'])
pp.show()
Ln: 1 Col: 10
```

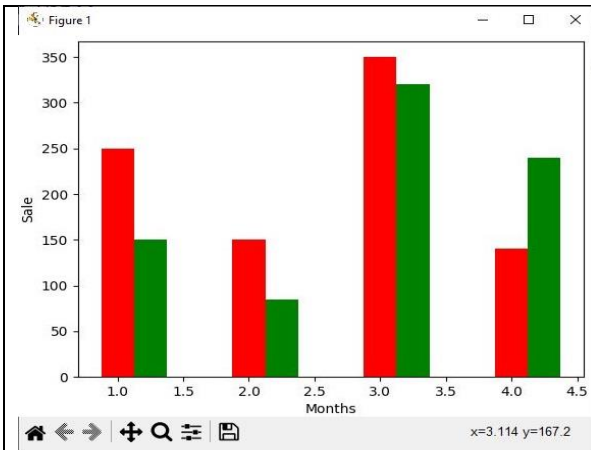


### Creating Multiple Bar Chart

Grouped Bar chart is another name for a multiple bar chart. There are several ways to customize a bar plot or bar chart, including multiple bar plots, stacked bar plots, and horizontal bar charts. Typically, multiple bar charts are used to compare multiple items using chart graphically.

**Example 6:** Let us create multiple Bar Graph for monthly sale of Keyboard and mouse with different color.

```
File Edit Format Run Options Window Help
# Creating multiple bar graph for monthly sale
import matplotlib.pyplot as pp
import numpy as np
month=np.arange(1.0,5.0,1.0)
mousesale=[250,150,350,140]
kbsale=[150,85,320,240]
pp.xlabel("Months")
pp.ylabel("Sale")
pp.bar(month,mousesale,color="r",width=0.25)
pp.bar(month+0.25,kbsale,color="g",width=0.25)
pp.show()
Ln: 3 Col: 0
```

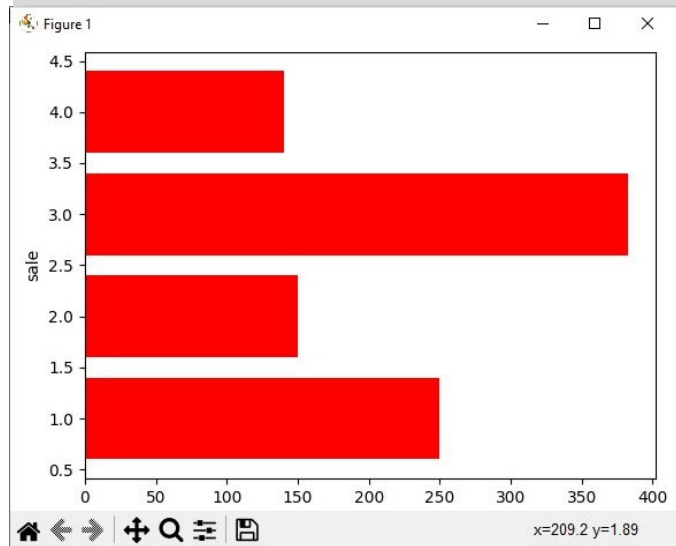


### Creating Horizontal Bar Chart

As you have prepared vertical bar graph in example 5, you can prepare a horizontal bar chart using another bar chart function i.e. `barh()`.

**Example 7:** Modify Example 5 to create Horizontal Bar Chart graph using `barh()` function.

```
File Edit Format Run Options Window Help
# Creating Horizontal bar graph for monthly sale
import matplotlib.pyplot as pp
import numpy as np
month=np.arange(1.0,5.0,1.0)
mousesale=[250,150,350,140]
pp.ylabel("Sale")
pp.barh(month,mousesale,color="r")
pp.show()
Ln: 7 Col: 7
```



### CREATING PIE CHART

Pie chart is made up of different parts of a circle where each part shows a particular ratio of data. We can use `pie()` function to create this type of chart.

The following property we can use with pie Graph like:

**Label:** Name of Pie

**Autopct:** Percentage value of Pie in circle

**Color:** Colours of Pie



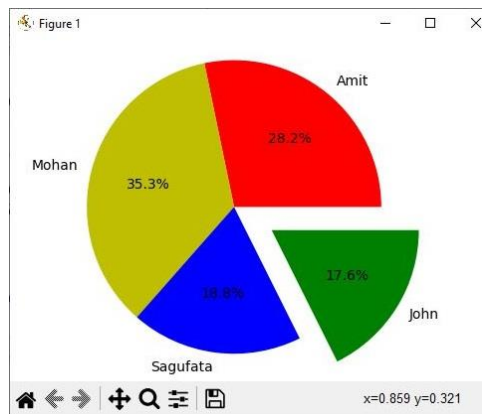
**Explode:** Detached from circle.

**Example 8:** Create a Pie Graph Using salary and name as a list and to use color, autopct, label, explode property.

Program to create a pie graph is given below.

```
File Edit Format Run Options Window Help
# Creating Pie Graph using salary and name
import matplotlib.pyplot as pp
import numpy as np
salary=[12000,15000,8000,7500]
empname=["Amit","Mohan","sagufata","John"]
c=['r','y','b','g']
e=[0,0,0,.3]
pp.pie(salary,colors=c,labels=empname,autopct="%1.1f%%",explode=e)
pp.show()
Ln: 9 Col: 9
```

On executing the above program the pie graph is created as shown below.



### Chart Anatomy and Saving Graph

Every type of chart has a special structure with its contents/values. The values for each bar represent a specific category of data. The y-axis is the vertical axis that runs along either the left or right side of the bar graph. The x-axis is the horizontal axis located at the bottom of a bar graph. The value of the data is represented by the height or length of the bars. The common key points in any bar chart are as under.

**Figure** – Any chart will be made under this area only. This is the area of plot.

**Axes** – This is that area which has actual plotting.

Axis Label – This is made up of x-axis and y-axis.

Limits – This is the limit of values marked on x-axis and y-axis.

Tick Marks – This is the individual value on x-axis and y-axis.

**Title** – It is the text to be shown at the top of plot.

**Legends** – This is the set of data of different color which is to be used during plotting.

**Example 9.** Create a graph to show anatomy of Chart using *legend*, *xlabel*, *ylabel*, *title of Chart* and understand how to save the graph as image.

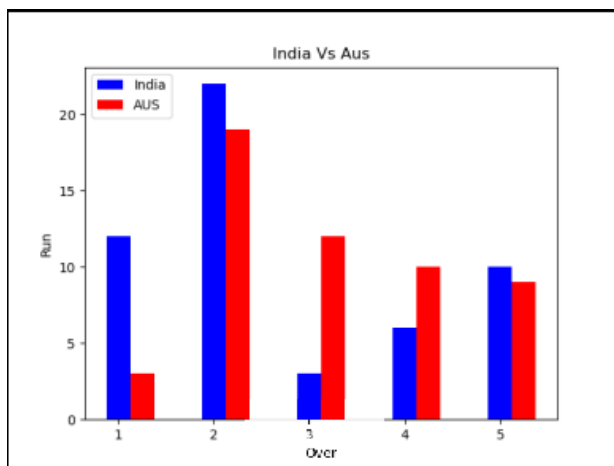
```

File Edit Format Run Options Window Help
# Creating a Graph to show anatomy of Chart
import matplotlib.pyplot as pp
import numpy as np
over=np.arange(1,6,1)
India=[12,22,3,6,10]
Aus=[3,19,12,10,9]
pp.title("India Vs Aus")

pp.bar(over, India,color="b",width=0.25,label="India")
pp.bar(over+0.25, Aus,color="r",width=0.25,label="AUS")
pp.legend(loc="upper left")
pp.xlabel("Over")
pp.ylabel("Run")
# Specify the location of computer to save the file
pp.savefig("/home/dds/WinIndia.png")
pp.show()
Ln: 14 Col: 51

```

In this code snippet, `pp.savefig()` function is used to save the chart as image. It will save chart at the given path/location in the function as parameter.



### Summary

- Data visualisation is the graphical representation of data or information using visual elements like, chart, graph, maps and so forth.
- The Matplotlib is a python library that provides many interfaces and functionality for 2D-graphics similar to MATLAB.
- To install Matplotlib, you need to use “pip install Matplotlib” at command prompt.
- Pyplot is a set of functions in Matplotlib library to utilize various graphical objects like Line, Bar, Pie chart etc.
- Data point are called marker.
- A line chart or line graph can be created using `plot ()` function.
- The `arange ()` function is used to generate numerical values at a fixed interval for array.
- The `show ()` function is used to display the figure /graph on the screen.
- The `savefig ()` function is used to save the figure of the graph at user given path.
- The pie chart is a type of graph in which a circle is divided into sector that each represents a proportion of the whole.
- Explode property of pie graph separate the slice of pie chart to display it separately.

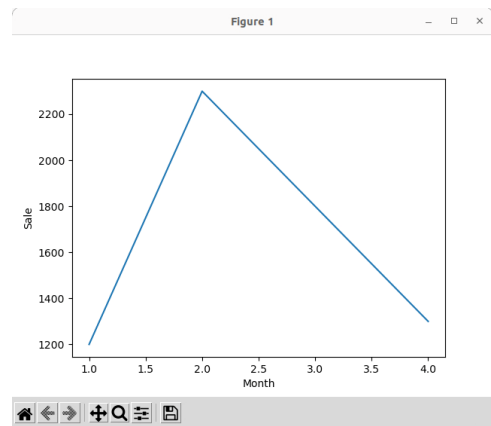
### Solved Programs

1. Write a program to plot a line chart to display monthly sale of computer.

#### Program

```
File Edit Format Run Options Window Help
# program to plot a line chart to
# display monthly sale of computer
import matplotlib.pyplot as pp
Month=[1,2,3,4]
Sale=[1200, 2300, 1800, 1300]
pp.plot(Month,Sale)
pp.xlabel("Month")
pp.ylabel("Sale")
pp.show()
Ln: 2 Col: 34
```

#### Output

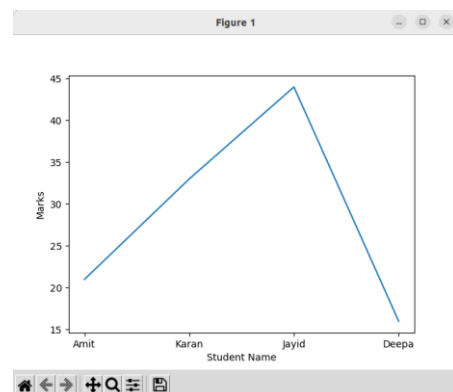


2. Write a program to plot a line chart to display marks of students secure in unit test.

#### Program

```
File Edit Format Run Options Window Help
# Program to plot a line chart to display
# marks of students secure in unit test
import matplotlib.pyplot as pp
SName=["Amit","Karan","Jayid","Deepa"]
Marks = [21, 33, 44, 16]
pp.plot(SName,Marks)
pp.xlabel("Student Name")
pp.ylabel("Marks")
pp.show()
Ln: 3 Col: 6
```

#### Output

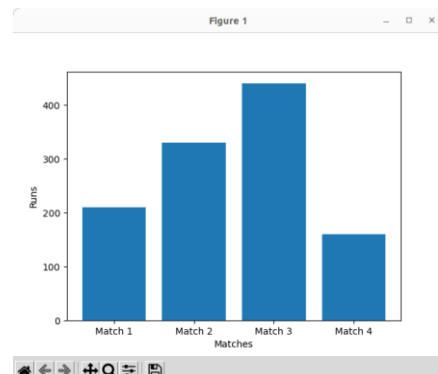


3. Write a program to draw a Bar chart to display run score by India in last 4 test matches.

#### Program

```
File Edit Format Run Options Window Help
# Program to plot a Bar chart to display runs
# scored by India in last 4 test marches
import matplotlib.pyplot as pp
M = ["Match 1", "Match 2", "Match 3", "Match 4"]
Run = [210, 330, 440, 160]
pp.bar(M, Run)
pp.xlabel("Matches")
pp.ylabel("Runs")
pp.show()
Ln: 5 Col: 31
```

#### Output



#### Practical Exercise

1. Add title for x-axis and y-axis and for whole chart title.
2. In an match player and their score runs are as follow:

Virat=12, Surya=34, Mohit=45, Zaheer=23

Create a pie Graph for the above data.

3. In an company, salesman and their sale are as fellow:

Madan=1200, Rohit=3994, Veena=2345, Zara=2003

Create Line Graph for the above data.

## CHECK YOUR PROGRESS

### A. Multiple Choice Questions

1. Pyplot is an interface of python library. (a) Seaborn (b) plotly (c) ggplot (d) Matplotlib
2. Which of the following not valid chart type? (a) static (b) bar (c) pie (d) plot
3. To create line chart we use following function (a) Seaborn (b) histogram (c) plot (d) bar
4. What is use for thickness of bar? (a) Title (b) plotly (c) barwidth (d) width
5. What is used for legend in graph? (a) Legend (b) savefig (c) show (d) display
6. What is used for to save figure in Matplotlib? (a) Legend (b) savefig (c) show (d) display
7. What is used for colour in bar graph? (a) color (b) savefig (c) colour (d) display

### B. Fill in the blanks

1. To save the figure in matplotlib, \_\_\_\_\_ is used.
2. To set color in bar graph \_\_\_\_\_ property is used.
3. To give title of any graph, we can use \_\_\_\_\_ property.
4. To set X axis and Y axis label, we use \_\_\_\_\_ and \_\_\_\_\_.
5. To display graph, \_\_\_\_\_ function is used.

### C. State whether True or False

1. Pyplot is sub-library of matplotlib library.
2. Statement "import pyplot.matplotlib is valid statement.
3. Line Graph can be create with the help of Line() function.
4. Can we create horizontal bar graph?
5. We cannot save chart using savefig function.

### D. Answer the following questions in short

1. What is Data Visualization? What is its significance?
2. What is PyPlot and matplotlib?
3. Name some common type of graphs and it their uses?
4. List down any 5 components of any graph?
5. What is the role of autopct in pie chart?

## Session 7. Database Connectivity with MySQL

In real life applications we always deal with data. In many applications we see that data is stored, manipulated, sorted, searched and retrieved as per the requirement through programs designed by the software developer. Like other programming language in Python also it is possible to connect with the database application to develop such types of applications. It is required to use libraries that provide various connectivity functionalities. To work with python MySQL connector, you need to install MySQL connector. In this chapter, you will understand to install mysql connector for Python connectivity, connect with the MySQL connector library for database connectivity, front-end interface and back-end interface. After database connecting, you will be able to extract from database, insert data into database, search data from database, modify data in database and delete data from database.

### Front-end interface

Front-end interface of any software is the screen where user can interact with the software. Front-end interface allows to enter data to the computer using a well-designed form as shown in Figure 7.1. The information filled in this form will store into connected database for future reference.

The image shows a web form titled "Student Registration Form". It contains the following fields and controls:

- Name :** A text input field with the placeholder text "Enter full name".
- Father's Name :** A text input field.
- Mother's Name :** A text input field.
- Phone Number :** A text input field with the placeholder "017xxxxxxxx".
- Email :** A text input field with the placeholder "sample@example.com".
- Gender :** Three radio buttons labeled "Male", "Female", and "Other".
- Date Of Birth :** Three separate input fields for day, month, and year, with a "(dd-mm-yyyy)" label.
- Address :** A text input field with the placeholder "Street:- House:- Road:-".
- Blood Group :** A dropdown menu with "Select" as the current selection.
- Department :** Three radio buttons labeled "CSE", "EEE", and "BBA".
- Course :** Six checkboxes labeled "C", "C++", "Java", "AI", "Machine learning", and "Robotics".
- Photo :** A "Choose File" button and the text "No file chosen".

At the bottom of the form, there are two buttons: "Submit" and "Reset".

**Fig. 7.1 Front end interface**

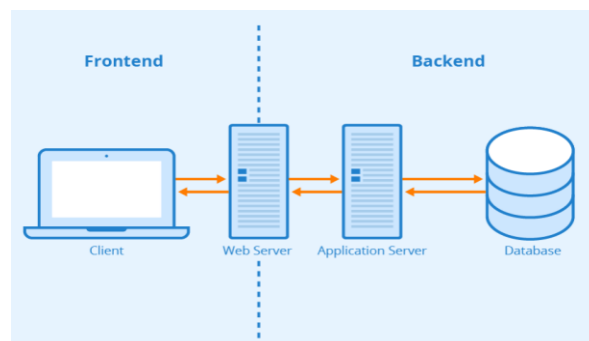
### Back-end interface

The back end refers to database, which store the data for the application. The back-end is the code that runs on the server, that receives requests from the clients, and contains the logic to send the appropriate data back to the client.

To give a real life example, when customer browses on website, they are interact with the front end. After selecting the item when they put it in the shopping cart, and authorize the purchase, the information is kept inside the database which resides on the server is the back-end.

HTML, CSS, and JavaScript are the programming languages used for the front end while Java, Ruby, Python, and . Net. Are used for back-end.

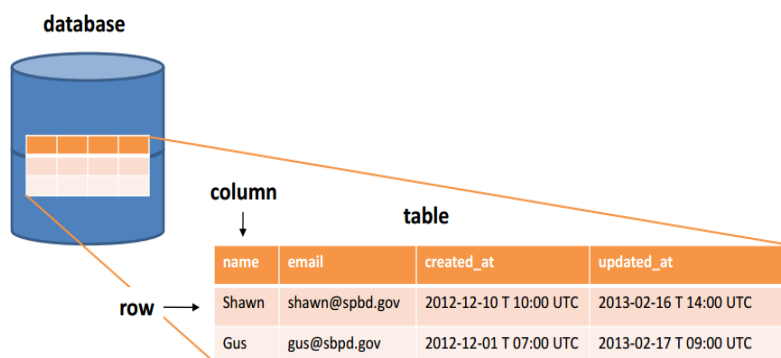
Figure 7.2 illustrates front-end and back-end technology. The data is entered via front end application and stored into database using back-end technology.



**Fig. 7.2 Front-end and back-end technology**

The frontend includes visual elements like buttons, check-boxes, graphics, and text messages that allows users to interact with application. The back-end stores and processes application data for users. The back-end is part of the application that is hidden from the end user. Database

administration can work on database like table creation, insert data into table, update data, and delete data. Figure 7.3 illustrates the data stored in the database is the back-end which is displayed on the screen through front-end programs.



**Fig. 7.3 Front-end and back-end**

### Creating database connectivity application

Following are the steps to be followed for creating database connectivity in Python.

Step 1. Start Python.

Step 2. Install MySQL-connector online with “pip install MySQL-connector-python”.

Step 3. Import the package required for database programming (import mysql.connector).

Step 4. Open a connection to database (connect () is used for this purpose, it required following parameters host, user, passwd, databasename).

Step 5. Create cursor (for this purpose used cursor ()) When you connect to a database from with python program, then the query gets sent to the server, where it gets executed, and the resultset is sent over the connection to you, in one burst of activity in one go. But you may want to access the retrieved data, one row at a time. Cursor is used for handle the resultset.

Example: mycur=mydb.cursor()

Step 6. Execute a query using execute () function by passing a query as a parameter. For example: cursor.execute (“select \* from student”).

Step 7. Extract data from resultset: once the result of query is available in the form of a resultset stored in a cursor object, you can extract data from the resultset using any of the following fetch () function.

**data=cursor.fetchall().** It will return all the records retrieved as per query in a tuple form.

**data=cursor.fetchone().** It will return one record from the resultset as a tuple or a list. This method returns one record as a tuple, if there are no more records then it returns None.

**data=cursor.fetchmany(n).** This method accepts number of records to fetch and returns a tuple where each record itself is a tuple. If there are not more records then it returns an empty tuple.

**variable=cursor.rowcount.** The rowcount is a property of cursor object that returns the number of rows retrieved from the cursor so far.

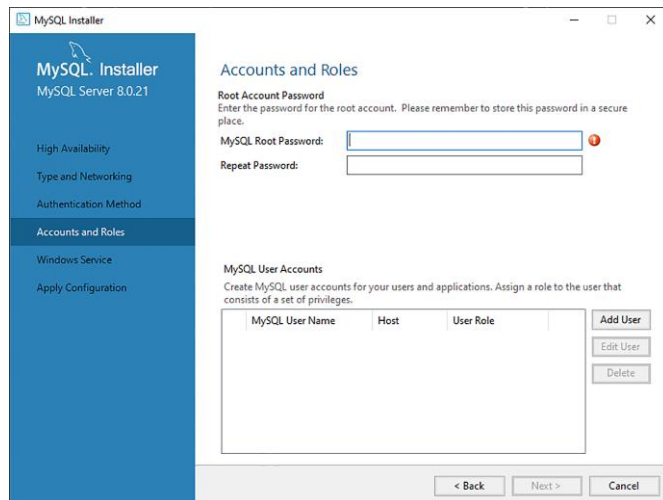
**Step 8.** Clean up the environment this is the final step we need to close the connection by using mycon.close().

### MySQL and Other SQL Databases

**SQL** stands for Structured Query Language and is a widely used programming language for managing relational databases. There are different flavors of SQL-based DBMSs compliant with the SQL standards. MySQL is the most popular apart from other DBMSs such as PostgreSQL, SQLite, and SQL Server. MySQL is now a part of the Oracle, while its core functionality is completely free through MariaDB. It is free and open source.

### Installing MySQL Server

In Windows, download MySQL Installer and install it in computer. The installation manager helps to configure the security settings of the MySQL server. On the Accounts and Roles page, enter a password for the root (admin) account and also optionally add other users with varying privileges.



### Installing MySQL Connector/Python

In Python it is required to install a Python MySQL connector to interact with a MySQL database. Many packages follow the DB-API standards, but the most popular among them is MySQL Connector/Python. You can get it with pip

```
$ pip install mysql-connector-python
```

pip installs the connector as a third-party module in the currently active virtual environment.

To test for successful installation, type the following command on Python terminal:

```
>>> import mysql.connector
```

If the above code executes with no errors, then mysql.connector is installed and ready to use.

### Establishing a Connection With MySQL Server

MySQL is a server-based database management system. One server might contain multiple databases. To interact with a database, you must first establish a connection with the server. The general workflow of a Python program that interacts with a MySQL-based database is as follows:

1. Connect to the MySQL server.
2. Create a new database.
3. Connect to the newly created or an existing database.
4. Execute a SQL query and fetch results.
5. Inform the database if any changes are made to a table.
6. Close the connection to the MySQL server.

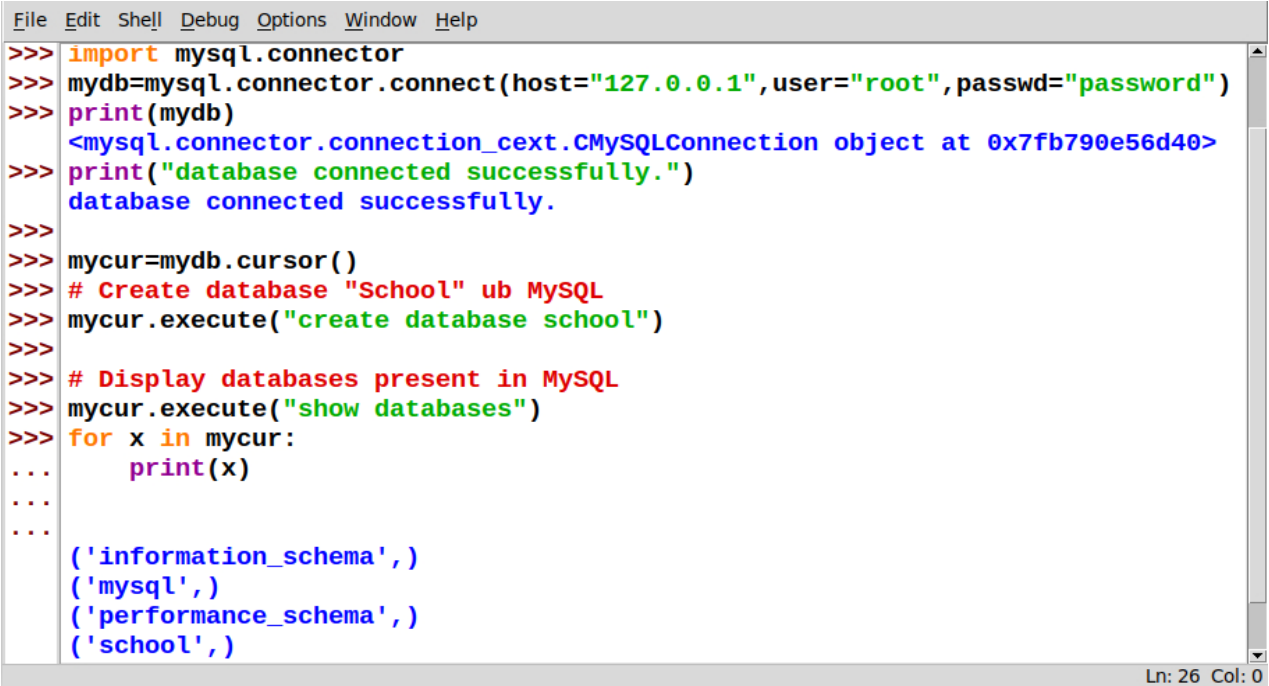
**Example 1 :** Connecting MySQL database in Python.

```
File Edit Shell Debug Options Window Help
>>> import mysql.connector
>>> mydb=mysql.connector.connect(host="127.0.0.1", user="root", passwd="password")
>>> print(mydb)
<mysql.connector.connection_cext.CMySQLConnection object at 0x7f2e24a6f100>
>>> print("database connected sucessfully.")
database connected sucessfully.
>>> |
```

Ln: 15 Col: 0

Here, in the above example, the first line is used to import the mysql connector using the command as “import mysql.connector”. In next line, we have created an object named mydb to connect with MySQL database by passing host id, user name and password of MySQL. The host name or IP address, user name and password are given as sample. Check these credential from your school lab before making the connection. Finally you can print the mydb object to see the connection status from MySQL database server. You can also print your own message such as “database connected successfully” as seen in the output.

**Example 2 :** Create database on MySQL and display all database name available in MySQL.



```
File Edit Shell Debug Options Window Help
>>> import mysql.connector
>>> mydb=mysql.connector.connect(host="127.0.0.1",user="root",passwd="password")
>>> print(mydb)
<mysql.connector.connection_cext.CMySQLConnection object at 0x7fb790e56d40>
>>> print("database connected successfully.")
database connected successfully.
>>>
>>> mycur=mydb.cursor()
>>> # Create database "School" ub MySQL
>>> mycur.execute("create database school")
>>>
>>> # Display databases present in MySQL
>>> mycur.execute("show databases")
>>> for x in mycur:
...     print(x)
...
...
('information_schema',)
('mysql',)
('performance_schema',)
('school',)
```

Let us understand the new things in this code. Here, in this example 2, we have created new cursor object as “mycur”. This object is used to refer to the result set returned from MySQL database after executing a query. We can perform multiple operations row by row against a result set, with or without returning to the original table.

After creating mycur object, *mycur.execute (“create database school”)* will create new database **school** in MySQL database. *mycur.execute(show database”)* will prepare result set of all database in MySQL database. Now using for loop, we can print all existing databases with our newly created database **school**.

**Example 3:** Write a python code to create table “student” in “school” database and show the Tables name available in database “school”, as we have created database “school” in Example 2.

In Example 3, we need to create one new table “student” as per question in the “school” database. For this purpose we have used *mycur.execute(“create table student (.....)”) with requested fields. After that we can prepare a result set object using mycur.execute(show tables). Now using for loop, we can print the all tables available in the result set. Finally mycur.execute(“desc student”) will display all table fields and its data types. Here for loop is used to display record one by one from cursor “mycur” and display on the screen.*



```

File Edit Format Run Options Window Help
import mysql.connector
mydb=mysql.connector.connect(host="127.0.0.1", user="root", passwd="password", database="school")
print(mydb)
print ("Database Connected successfully... GOOD")
mycur=mydb.cursor()

# This statement will create table "student" in "school" database.
mycur.execute("create table student( rollno int(4) primary key,\
name varchar(34), age int(2), city varchar(23))")

# This statement will display all tables name present in "school" database.
mycur.execute("show tables")

for x in mycur:
    print("Recently created table name is : ",x)

# This statement will display all table fields with their datatypes.
mycur.execute("desc student")

for x in mycur:
    print(x)

```

Ln: 2 Col: 76

**Example 4:** Write a python code to insert data in table “student” which is in “school” database under MySQL. under MySQL.

```

File Edit Shell Debug Options Window Help
<mysql.connector.connection_cext.CMySQLConnection object at 0x7fdeef49c60>
Database Connected successfully... GOOD
Recently created table name is : ('student',)
('rollno', 'int(4)', 'NO', 'PRI', None, '')
('name', 'varchar(34)', 'YES', '', None, '')
('age', 'int(2)', 'YES', '', None, '')
('city', 'varchar(23)', 'YES', '', None, '')
>>>

```

Ln: 12 Col: 0

In Example 1, 2 and 3, we have created “student” table in “school” database. It is now required to insert data into “student” table. It is illustrated in the following example.

```

File Edit Format Run Options Window Help
import mysql.connector
mydb=mysql.connector.connect(host="127.0.0.1", user="root", passwd="password", database="school")
print(mydb)
mycur=mydb.cursor()
# The below 4 statements will Insert 4 new records into table "student"

r1="insert into student(rollno,name,age,city) values(%s,%s,%s,%s)"
v1=(101,"Ravi",23,"Bhopal")

r2="insert into student(rollno,name,age,city) values(%s,%s,%s,%s)"
v2=(102,"Gagan",31,"New Delhi")

r3="insert into student(rollno,name,age,city) values(%s,%s,%s,%s)"
v3=(103,"Veena",40,"Nanital")

r4="insert into student(rollno,name,age,city) values(%s,%s,%s,%s)"
v4=(104,"Zaid",34,"Mumbai")

mycur.execute(r1,v1)
mycur.execute(r2,v2)
mycur.execute(r3,v3)
mycur.execute(r4,v4)

mydb.commit() # This statement will save data permanently into table

```

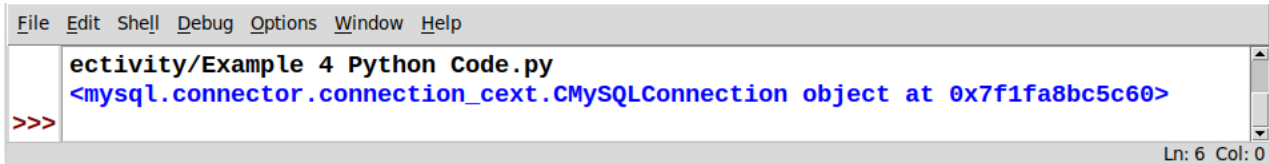
Ln: 1 Col: 0

In this example, a new variable r1 is created that contain SQL statement for inserting data into table. In this statement %s is used for each value which we need to insert in table. Another

variable of type tuple is created to hold values of one student like roll number, name, age and city of the student.

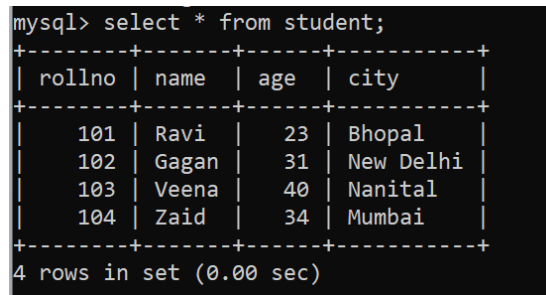
Similarly, three more SQL statement variables and tuples are created for other students. Now using `mycur.execute(r1,v1)`, execute these SQL queries to insert records in the database. Finally update and save database permanently using `mydb.commit()` function.

All the 4 records inserted successfully into student table can be displayed using the query as shown in the following output screen.



```
File Edit Shell Debug Options Window Help
activity/Example 4 Python Code.py
<mysql.connector.connection_cext.CMySQLConnection object at 0x7f1fa8bc5c60>
>>>
Ln: 6 Col: 0
```

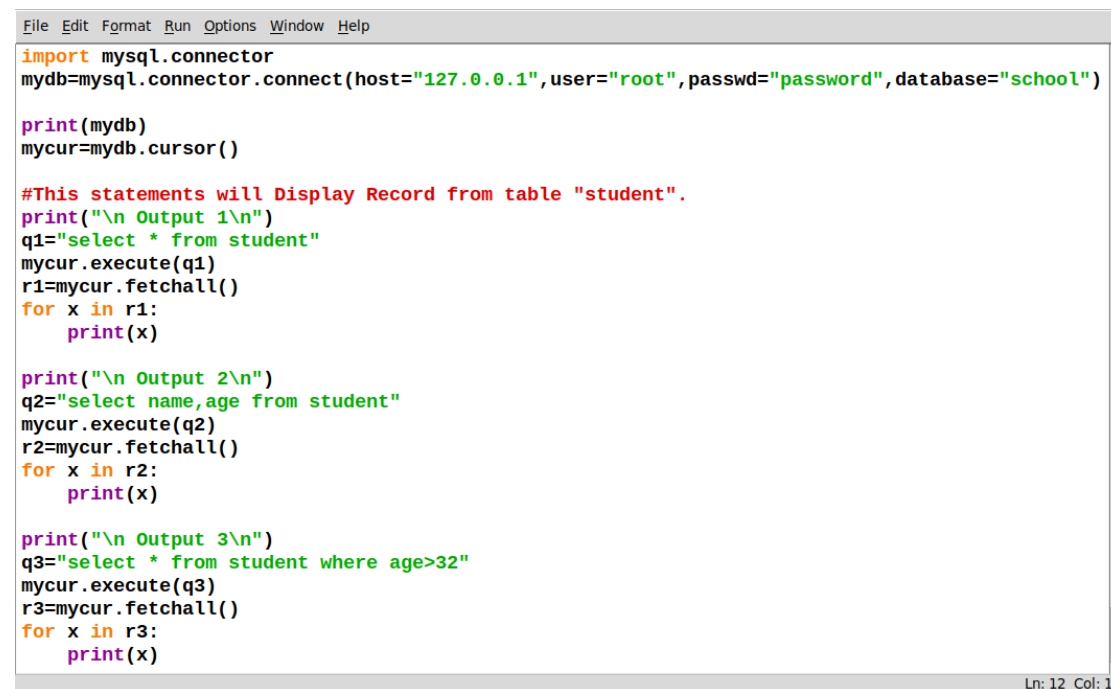
All the 4 records inserted successfully into student table.



```
mysql> select * from student;
+-----+-----+-----+-----+
| rollno | name  | age  | city  |
+-----+-----+-----+-----+
| 101    | Ravi  | 23   | Bhopal |
| 102    | Gagan | 31   | New Delhi |
| 103    | Veena | 40   | Nanital |
| 104    | Zaid  | 34   | Mumbai |
+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

**Example 5:** Write a python code to display data from table “student” with select command.

In Example 4, we have inserted 4 records into “student” table in “school” database. Now we need to display/show data from “student” table with different SQL statement. Explore example below for said purpose.



```
File Edit Format Run Options Window Help
import mysql.connector
mydb=mysql.connector.connect(host="127.0.0.1",user="root",passwd="password",database="school")

print(mydb)
mycur=mydb.cursor()

#This statements will Display Record from table "student".
print("\n Output 1\n")
q1="select * from student"
mycur.execute(q1)
r1=mycur.fetchall()
for x in r1:
    print(x)

print("\n Output 2\n")
q2="select name,age from student"
mycur.execute(q2)
r2=mycur.fetchall()
for x in r2:
    print(x)

print("\n Output 3\n")
q3="select * from student where age>32"
mycur.execute(q3)
r3=mycur.fetchall()
for x in r3:
    print(x)
Ln: 12 Col: 12
```

Here in this example, a variable `q1` is created with SQL query to select all records from student table, then execute the SQL statement `q1` using `mycur.execute(q1)` statement.

In next statement, with the help of *fetchall()* function, *mycur* resultset object will transfer all records to *r1* variable which is part of python environment. Now using for loop, we can display all records stored in *r1*. Similarly, we have created *r2* and *r3* result set for other two SQL queries.

**Example 6:** Write a python code to delete data from “student” table using delete command.

```

File Edit Shell Debug Options Window Help
<mysql.connector.connection_cext.CMySQLConnection object at 0x7fbe7d3fdc60>

Output 1

(101, 'Ravi', 23, 'Bhopal')
(102, 'Gagan', 31, 'New Delhi')
(103, 'Veena', 40, 'Nanital')
(104, 'Zaid', 34, 'Mumbai')

Output 2

('Ravi', 23)
('Gagan', 31)
('Veena', 40)
('Zaid', 34)

Output 3

(103, 'Veena', 40, 'Nanital')
(104, 'Zaid', 34, 'Mumbai')
>>>
Ln: 21 Col: 0

```

In Example 4 and 5, we have inserted and displayed data in the table. Now to delete data from “student” table is illustrated in the following code.

```

File Edit Format Run Options Window Help
import mysql.connector
mydb=mysql.connector.connect(host="127.0.0.1", user="root", passwd="password", database="school")
print(mydb)
mycur=mydb.cursor()

#This statements will Delete Record from table "student" where age = 23.
q1="delete from student where age=23"
mycur.execute(q1)
mydb.commit()
print(mycur.rowcount, " Record(s) Deleted...")
Ln: 2 Col: 76

```

Observe the above code, one SQL query is prepared in variable *q1* to delete the student record whose age is 23. This query will delete record/s from table where age of any student is 23. As of now current table has only one student whose age is 23 and name is ravi.

Now on execution of *mycur.execute(q1)* statement, the record of student/s will delete whose age is 23 in student table. In the second last line, we used *mydb.commit()* statement to save updated data permanently into database.

In the last we have used *mycur.rowcount* statement to display the number of records deleted from the current result set.

Try to use some other select statement to get different result.

In Example 5, First record is for ravi and age is 23. But after performing delete command in this example, record of ravi where age was 23 get deleted. See the output at SQL command prompt carefully.

```
File Edit Shell Debug Options Window Help
e Connectivity/Example 6 Python Code.py
<mysql.connector.connection_cext.CMySQLConnection object at 0x7facf9ec1c60>
1 Record(s) Deleted....
>>>
```

```
mysql> select * from student;
+-----+-----+-----+-----+
| rollno | name  | age  | city  |
+-----+-----+-----+-----+
|      102 | Gagan | 31  | New Delhi |
|      103 | Veena | 40  | Nanital |
|      104 | Zaid  | 34  | Mumbai |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

**Example 7:** Write a python code to Update data into table “student” using Update command.

Suppose Veena comes and requests to update her age by 30 as her age in table is 40. One more request from Head Clerk is to change city for all students to “Mathura”.

Now for this we need to use two update query commands for both the request. Now observe the code of update query given below.

```
File Edit Format Run Options Window Help
import mysql.connector
mydb=mysql.connector.connect(host="127.0.0.1", user="root", passwd="password", database="school")
print(mydb)
mycur=mydb.cursor()

#The below query will age of veena to 30 in table "student".
q1="update student set age=30 where rollno=103"

#This below query will update city for all students to mathura in table "student".
q2="update student set city='Mathura'"

mycur.execute(q1)
mycur.execute(q2)
mydb.commit()

print(mycur.rowcount, " Record(s) Updated....")
```

Here we have prepared two SQL statements as q1, q2 for python string which are used to update required record(s). The first statement q1 is to update age to 30 where rollno is 103. The second statement q2 is to update city to “Mathura” for all students.

Using mycur.execute() statement, we can execute both SQL queries and by using mydb.commit(), we can update and save data permanently in the database.

```
File Edit Shell Debug Options Window Help
code.py
<mysql.connector.connection_cext.CMySQLConnection object at 0x7f41fae71c60>
3 Record(s) Updated....
>>>
```

In the output, observe the difference in two figures, Veena’s age is updated by 30 and all students city are updated to Mathura.

```
mysql> select * from student;
+-----+-----+-----+-----+
| rollno | name  | age  | city  |
+-----+-----+-----+-----+
|      102 | Gagan | 31  | Mathura |
|      103 | Veena | 30  | Mathura |
|      104 | Zaid  | 34  | Mathura |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

## Summary

- SQL-connector is used to connect Python with MySQL. So download My SQL-connector-python with the help of “pip install mysql-connector”.
- fetchall() method is used to fetch multiple values from a database table.
- rowcount() is a read-only attribute and return the number of rows that were affected by an execute() method.
- To disconnect from database while working with python, use close () function.
- A Database Cursor is a special control structure that facilitates the row by row processing of records in the result set.
- You can use connect() method for establishing database connection, cursor() to create a cursor and execute() to execute an SQL query.
- To fetch records from a result set, You can use fetchone() method to read one record at a time and fetchall() method to read all records at a time.
- For INSERT, UPDATE and DELETE queries, You must run commit() method with connection object.

## Solved Programs

Consider database “school” having “student” table.

1. Write a Python-MySQL Connectivity code to retrieve data, one record at a time, for student rollno is less than 200.

### Solution:

```
import mysql.connector
mydb=mysql.connector.connect(host="127.0.0.1", user="root", passwd=" Admin@123",
database="school")
print(mydb)
mycur=mydb.cursor()
q1="select * from student where rollno <200"
mycur.execute(q1)
r1=mycur.fetchall()
for x in r1:
    print(x)
```

2. Write a Python-MySQL Connectivity code to insert a student record into table at a time.

### Solution:

```
import mysql.connector
mydb=mysql.connector.connect(host="127.0.0.1", user="root", passwd=" Admin@123",
database="school")
print(mydb)
mycur=mydb.cursor()
r1="insert into student(rollno,name,age,city) values(%s,%s,%s,%s)"
v1=(101,"Geeta Sharma",33,"kolkata")
mycur.execute(r1,v1)
mydb.commit()
```

3. Write a Python-MySQL Connectivity code to update city with Agra where rollno is 101.

### Solution:

```
import mysql.connector
```

```

mydb=mysql.connector.connect(host="127.0.0.1", user="root", passwd="Admin@123",
database="school")
print(mydb)
mycur=mydb.cursor()
q1="update student set city="agra: where rollno=101"
mycur.execute(q1)
mydb.commit()
print(mycur.rowcount, " Record(s) Updated....")

```

## CHECK YOUR PROGRESS

### A. Multiple choice questions

1. Identify the name of connector to establish bridge between Python and MySQL (a) mysql.connection (b) execute() (c) mysql.cursor() (d) mysql.connector.connect()
  2. Which of the following component act as a container to hold all the data returned from the query and from there we can fetch data one at a time? (a) database (b) Cursor (c) Container (d) Table
  3. Identify the correct statement to create cursor: (a) con.cursor() (b) con.create\_cursor() (c) con.open\_cursor() (d) con.get\_cursor()
- ```

import mysql.connector as msq
con = msq.connect( #Connection String )
mycursor = _____

```
4. Which attribute of cursor is used to get number of records stored in cursor (a) mycursor.count (b) mycursor.row\_count (c) mycursor.records (d) mycursor.rowcount
  5. Which of the Symbols are used for passing parameterized sql query for execution to cursor? (a) % (b) {} (c) \$ (d) Both a and b
  6. Which function is used to fetch all records from cursor? (a) fetch() (b) fetchone() (c) fetchmany() (d) fetchall()
  7. Which cursor function is used to send query to database? (a) query() (b) execute() (c) run() (d) send()

### B. Fill in the blank

1. \_\_\_\_\_ method will return one row from the result set.
2. \_\_\_\_\_ method will return all row from the result set.
3. To import MySQL connector \_\_\_\_\_ code is used.
4. Command \_\_\_\_\_ is used for counting the number of rows in the database.
5. Function \_\_\_\_\_ is used for execute SQL statement.

### C. State whether True or False

1. Can we connect python with MySQL.
2. Is python open source software.
3. Is MySQL open source software.
4. We can store data in Front end form
5. JAVA can be use as database.

### D. Answer the following questions in short

1. What is the difference between fetchall() and fetchone() function?
2. What is commit() function used for ?
3. What is command to install mysql connector?
4. What is resultset?
5. Write is database connectivity?
6. Which function/method do you use for establishing connection to database?
6. Which function/method do you use for executing an SQL query?
7. Which method do you use to fetch records from the resultset?

### Practical Exercise

1. Write a python program that fetches all records from student table.
2. Write a python program to modify/update age as 20 to all record (use example of this book
3. Write a python program to delete all record from table student (use example of your book).
4. Write a python program to display all record from table student whose age is 40 (use example of your book).
5. Write a python program to display all record from table student whose city is nanital (use example of this book).

## Module 3

## Software Engineering

### Module Overview

Software engineering concepts are essential for the development of reliable, and high-quality software products. Software engineers create applications by applying the principles of software engineering. The Software Development Life Cycle (SDLC) is a structured process that enables the production of high-quality, low-cost software, in the shortest possible production time. SDLC is a process followed for software development. It consists of a detailed plan describing how to develop, maintain, replace, and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process. There are more than 50 recognized SDLC models, each of them having its advantages and disadvantages. Some of the most useful models are discussed in this unit.

There are 7 Phases of SDLC that include planning, analysis, design, development, testing, implementation, and maintenance. In this unit, these phases are discussed in detail. The implementation of sample project by using the concepts of software engineering is also demonstrated in this unit.

### Learning Outcomes

After completing this module, you will be able to:

- Describe the concepts of software engineering
- Describe the software development process flow
- Demonstrate to implement minor software project using python

### Module Structure

Session 1: Software Engineering Concepts

## Session 2: Software Development Process

**Session 1: Software Engineering Concepts**

Software is an essential component to make effective use of a computer system. The first and essential software that every computer must have is its operating system.

A software is a large collection of executable programs that are associated with libraries and documentation. Every software is written for a specific purpose. Software engineers apply principles of Software Engineering to develop robust software applications for improving quality, budget, time and efficiency.

In this chapter, you will understand the concept of software engineering used for software development. The software development life cycle is explained with various phases of software development. The most commonly used models of software engineering are also explained.

**1.1 SOFTWARE ENGINEERING**

A computer program and software are related terms but have different scope and purpose.

**Program** – A program is a set of instructions written in a programming language that performs a specific task. Programs are coded for software and are not directly used by end-users. Examples of programs are, program to find ASCII value of a character, program to compute quotient and remainder, program to find the size of int, float, double and char.

**Software** – Software is a broader term that refers to a collection of programs, data, and other supporting elements that work together to perform a specific function. Software can be thought of as a complete package that provides a solution to a particular problem. Software includes programs, documentation, and other components required to support the program.

Software is categorised into different types, such as system software, application software, and utility software.

**Software engineering** is the technological and managerial discipline concerned with systematic production and maintenance of software products that are developed and modified on time and within cost estimates.

Software engineering is defined as a process of analyzing user requirements and then designing, building, and testing software applications which will satisfy those requirements.

IEEE defines Software Engineering (SE) as a knowledge area of computing that defines systematic, disciplined and quantifiable approaches for the development, operation and maintenance of software.

**1.1.1 Need and Importance of Software Engineering in Software Development**

Software is required in almost every industry. The working of software can dramatically affect our day to work. So it becomes important to produce good quality workable software. It is essential to apply the principles of software engineering to produce the quality software because of the following features.

**1. Complexity** – It becomes difficult to build big and complex software with a large number of programs. It is possible to reduce the complexity of the project by applying the principles of software engineering for the development of software.

**2. Cost effectiveness** – The costs of any software depends upon the required man hours for its development. It is possible to break up the long project into small components. This helps to optimize the code and hence reduces the cost.

**3. Time optimization** – The software development process is complex and it may require a lot of time to get error free executable code. It is possible to decrease the development time by applying the principles of software engineering.



**4. Effectiveness** – The commercial software needs to comply with the standards available with the company. These standards can be achieved by applying the scientific method of software engineering.

**5. Reliability** – The software developed should be reliable in nature, i.e. it should be executable under cross platforms and every time when it is executed should serve the purpose. With the help of methods of software engineering we can achieve high reliability for the software products.

### 10.1.2 Characteristics Of Good Software

A good software is characterised by its functionality, reliability, usability, efficiency, maintainability, security, and scalability. By adhering to these characteristics, software developers can create high-quality software that meets the needs of users and businesses. Some of the characteristics of good software are listed below:

**1. Functionality** – Good software should fulfil its intended purpose and meet the requirements of its users. It should perform the functions efficiently and accurately for which it was designed.

**2. Reliability** – Good software should work consistently and without errors, even under varying conditions.

**3. Usability** – The user-friendly interface makes the software easy to use for users to accomplish tasks quickly and efficiently.

**4. Efficiency** – Good software should be efficient and fast, using minimal system resources while performing its functions. It should run smoothly without causing slowdowns or crashes.

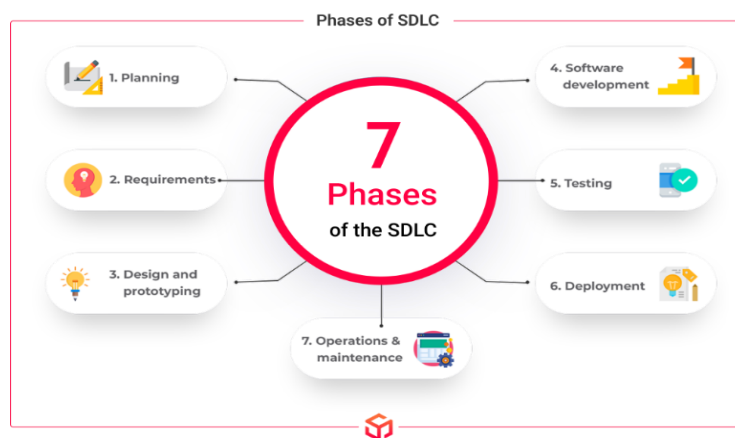
**5. Maintainability** – The software with modular design is easy to maintain and update. It allows you to make changes without affecting the entire system.

**6. Security** – Good software should be secure, protecting against unauthorised access and malicious attacks. The security measures such as encryption, authentication, and access control should be implemented in software.

**7. Scalability** – Good software should be able to handle increasing demands as the user base grows. It should be designed to scale up or down depending on the workload.

### 1.2 SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)

A software development is a complex process and hence it has different phases to complete the process. It is also called a software life cycle or software process model. It provides a systematic management framework with specific deliverables at every stage. It comprises a detailed plan that describes how to develop, maintain, and replace the software. The Specific SDLC models have different ways of implementing these steps. SDLC comprises seven different stages: planning, analysis, design, development, testing, implementation, and maintenance. Figure 1.1 shows these phases of SDLC.



**Fig. 1.1 Phases of software development**

Let's take a closer look at the general steps of all SDLC approaches.

**1. Planning and Requirement Analysis**

Planning is the core and first phase of SDLC. It acts as the foundation of the whole SDLC for the successful execution of further stages of the project. In this phase the problem statement for the project and scope is defined. The development team takes input from the users, customer, and stakeholders to plan the outline of the project. The stakeholders may include internal departments and industry experts. The planning process helps to identify how a specific problem can be solved with a certain software solution. This stage estimates the overall cost of the software and analyses the resources and costs needed to complete the project. This stage clearly defines the outline of system development with deadlines and time frames for each phase, ensuring timely completion of the project.

**2. System Analysis**

In this phase, the feasibility of the system is analysed technologically, practically, and financially. This includes system prototype drafts, market research, and an evaluation of competitors. It specifies the software process flow, inputs and deliverables, hardware and software tools and techniques and security issues. Market research is conducted to define the needs of end users and customers.

A detailed project report, Software Requirement Specification (SRS) is prepared in this phase. The SRS clarifies what needs to be designed and developed during the software development process. The SRS is explained to the customer and approval is obtained with modifications if any.

**3. System Design**

In this phase the system is designed by using the appropriate design approach, as per the requirements mentioned in requirement analysis and system analysis phases. The design models are built depending on the type, complexity, size of the project and the development method. The various activities of system design includes: design the environment, design application architecture and software, design user interfaces, design system interfaces, design the database, and design system controls & security. The system design is validated by the stakeholders and even a small mistake will be resolved before proceeding to develop the software. A suitable platform, programming languages and tools are identified for development of the software to start with the actual coding.

**4. System Development**

In this phase the developers build the software according to the design documents and outlined specifications. They write the code in the selected programming languages, debug and execute the programs to build the entire system. It includes both front and back-end development. They use multiple tools, programming environments, and languages such as C++, PHP, Python, as per the specifications and requirements outlined in the SRS document. This stage is considered to be one of the longest phases in software development. The system analyst will continuously monitor the work done by the developers and will suggest the modifications and enhancement in the work.

**5. System Testing**

The testing stage ensures that the software developed should be error free and work properly as per the expectation of the user. This process involves detecting the possible bugs, defects, and errors, searching for vulnerabilities. The software should be tested for different data sets to see that it delivers the specific functionality. The software should meet the quality standards as defined in the SRS document. The software should also be tested to work properly under the cross platforms i.e. it should work properly in various platforms and browsers. The appropriate testing ensures the quality assurance of the software. Many times the several iterations of software testing are carried out to ensure the quality of the product. Testing phase of the software can take time depending on the skill of the developers, the complexity of the software, and the requirements for the end-user.

## 6. System Implementation

After finishing the testing and fixing errors, the software is ready for deployment and implementation. At this stage, the software undergoes final testing through the training or pre-production environment, after which it's ready for presentation on the market. A software is released to the customer after its testing. System performance is compared to performance established during the planning phase. Implementation includes user notification, user training, installation of hardware and software, and integration of the system into daily work processes. This phase continues until the system is operating in production in accordance with the defined user requirements. The users are then provided with the training or documentation that will help them to operate the software.

## 7. System Maintenance

The system is monitored for continued performance in accordance with user requirements. Many times the system requirements are changed over a period of time. So it is necessary to update the developed software at regular intervals of time. Maintenance is a process of modifying software in order to keep it working over a time. It is possible that sometimes the bugs may arrive or the security issue may arise due to various reasons. This is particularly important for large systems, which usually are more difficult to test in the debugging stage. In such a case, in the software maintenance these issues are addressed to get the solution. When modifications are identified, the system may reenter the planning phase and the software development life cycle repeats.

### 1.3 SOFTWARE DEVELOPMENT METHODOLOGIES (SDLC MODELS)

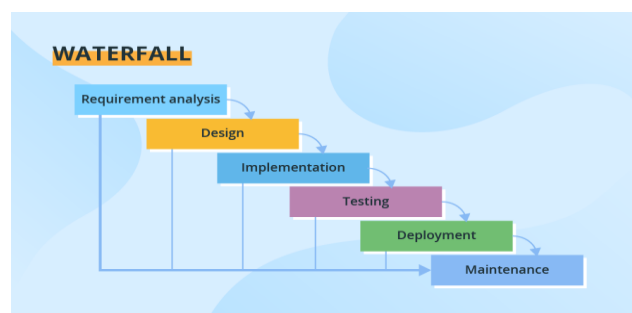
Software development methodologies are called SDLC models. SDLC is a process followed for software development. It consists of a detailed plan describing how to develop, maintain, replace, and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process. There are more than 50 recognized SDLC models, each of them having its advantages and disadvantages. Some of the most useful models are discussed below.

#### 1.3.1 Classical waterfall model

The Waterfall model is the first process model to be used for software development. It illustrates the process in a linear sequential flow and hence it is also called a linear-sequential model. In this model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. A drawback of this model is that even the small details left incomplete can hold an entire process.

Figure 1.2 shows the several consecutive phases of the classical waterfall model in software engineering. You can see that there are six distinct stages in this model, namely requirement analysis, system design, implementation, testing, deployment, and maintenance. The activities of each stage can begin only after completing the previous step and all activities are properly documented. The output of one phase becomes the input of the next. Thus, the development process can be viewed as a sequential flow in a waterfall.

**Note: Redraw all figures consisting of similar phases of software development and look.**



**Fig. 1.2 Waterfall Model of Software Development**

### 1.3.2. V-model (Validation and Verification model)

V model is an extension of the Waterfall model, with the improvement that each development phase will go through a testing phase before moving forward. Thus, this model is also known as the Verification and Validation model. So, with all these validations and verifications, the risk of errors is much less. The mistakes in requirements specifications, code and architecture errors can be detected early. It follows a systematic and well-defined process for developing high-quality software. It provides a comprehensive view of the software development lifecycle, incorporating requirements gathering, design, implementation, testing, and maintenance. In the V model each step executes in a sequential manner with parallel testing for each development stage.

Figure 1.3 shows the process work flow of the V model, where each development phase is associated with the testing phase. The left half of the V shape depicts Verification, and the right half depicts Validation, and both halves are joined by a coding phase which gives it a V shape.

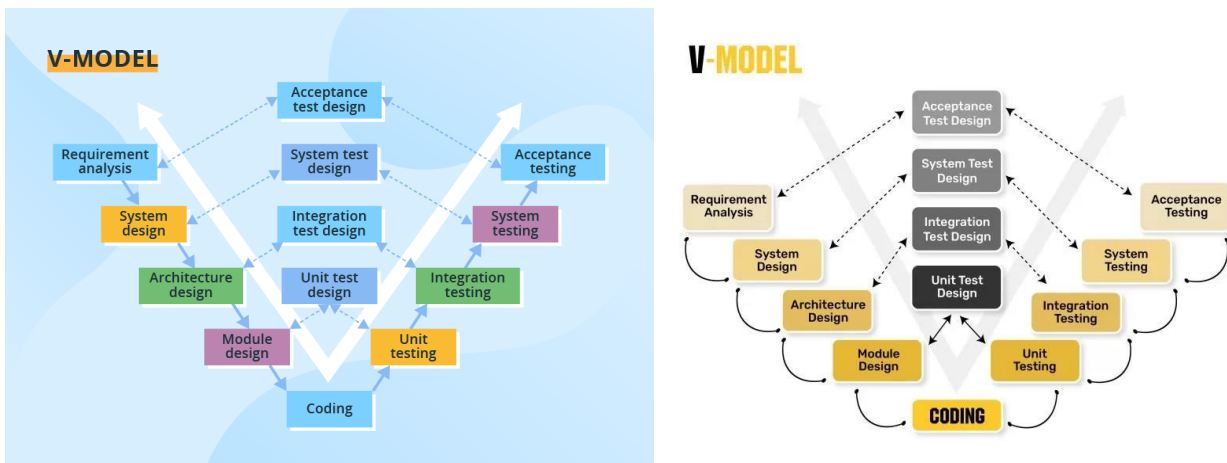
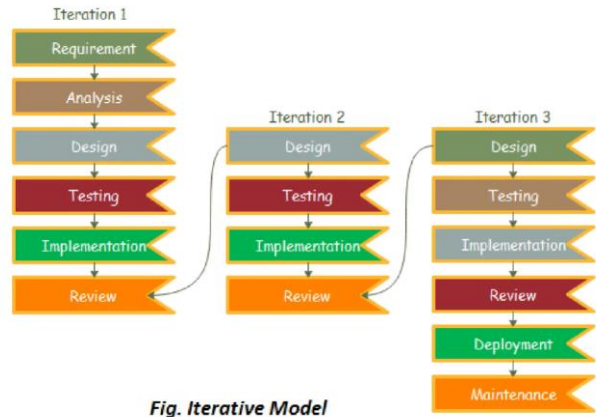
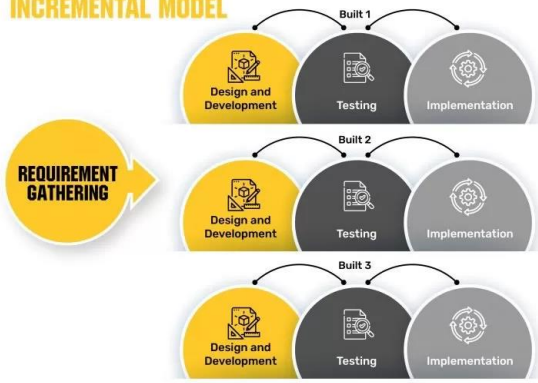


Fig. 1.3 V Model of Software Development

### 1.3.3. Incremental and Iterative model

The waterfall model has several limitations and it cannot be used practically for software development because the handling of development errors are not considered in the waterfall model. Hence the incremental and iterative model is suggested where there is a provision of feedback path for handling of errors. It takes an evolutionary approach to software development. In the initial stage the software development starts with a limited set of considerations and requirements. The feature code is designed, developed, and tested in repeated cycles. With each iteration, additional features can be designed, developed, and tested until there is a fully functional software application ready to be deployed to customers. The initial working software is ready in the early stage. The iterative model begins with a simple execution, which iteratively improves until the entire system is executed and ready to be redistributed. Bugs and errors from the previous iteration do not propagate to the next iteration. This model is flexible to incorporate customer feedback in every iteration. Figure 1.4 shows the approach of an incremental and iterative model.

**ITERATIVE AND INCREMENTAL MODEL**



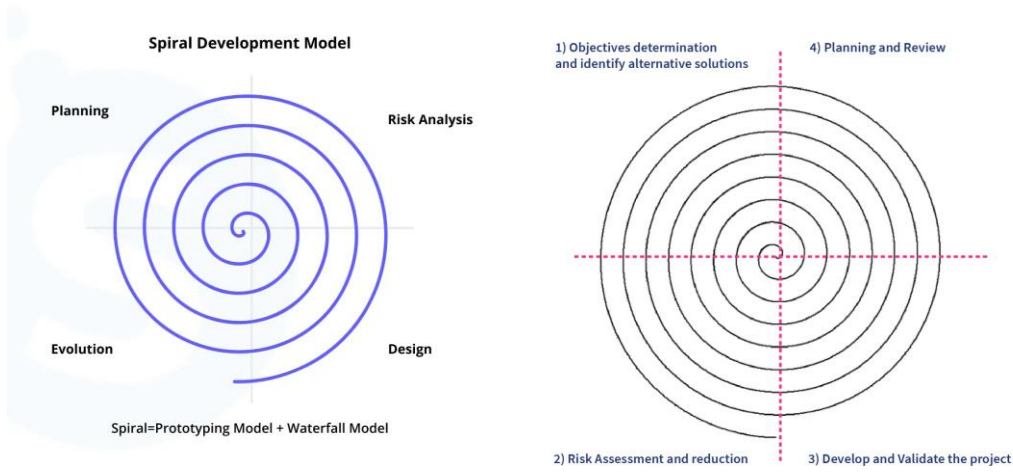
*Fig. Iterative Model*

**Fig. 1.4 Incremental and Iterative Model of Software Development**

**1.3.4. Spiral model**

It is a combination of an iterative and waterfall model. It is similar to the incremental model, with more emphasis placed on risk analysis. The spiral model is one of the best models due to risk management, ability to adapt to changes and frequent feedback from the client. It has the capability of rapid production of new software versions. The program is developed in a series of incremental releases using the spiral methodology. The development process begins with a limited set of needs and progresses through each development phase for that set of criteria. The software engineers keep on adding the functionality as per the requirement in every increasing spiral until the application is ready for production.

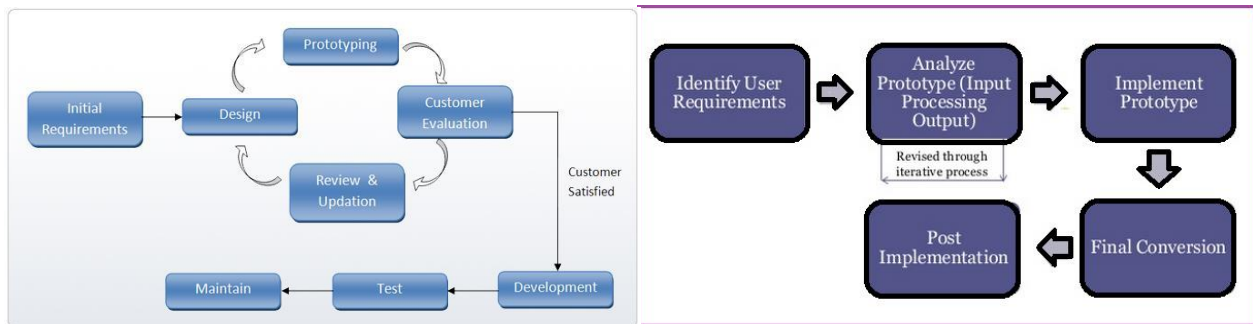
The spiral model has four phases: Planning, risk analysis, product development and planning or Evaluation. A software project repeatedly passes through these phases in iterations as shown in Figure 1.5.



**Fig. 1.5 Spiral Model of Software Development**

### 1.3.5. Prototyping Model

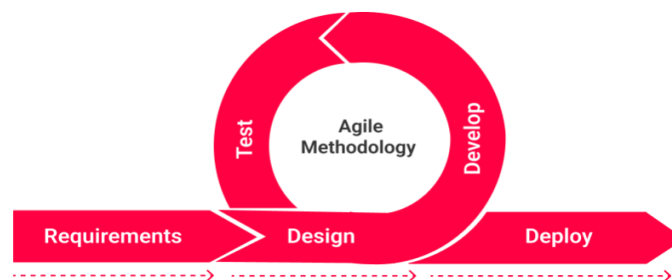
The prototype model provides a replica of the desired software for consumer feedback. A prototype helps to examine the technical issues associated with the software system. Prototyping is a way to receive input on requirements, functionality, and operability, so that the final development of the software can be achieved quickly and efficiently. A prototype of the software is created, tested, and refined based on customer feedback. Once the prototype is ready, developers can seek feedback from their clients to understand if any other stages of development are required. The process is repeated until the user approves the prototype and is satisfied with the working model. Once all the changes are made, and the clients are satisfied with the prototype, developers can convert it into the end product straight away. Figure 1.6 shows the process flow in the prototype model.



**Fig. 1.6 Prototype Model of Software Development**

### 1.3.6. Agile Model

Agile methodologies have a modern approach to software development. It focuses on flexibility, collaboration, and *delivery of software in short increments*. It is used for rapid delivery of working software. It is a combination of incremental and iterative processes. It is more efficient than other process models. It is fast and a more responsive development process which is essential for creating high-quality software that meets the needs of the end users. It ensures that the projects are delivered on time and within budget. The most significant aspect of the Agile model is determining the project scope, requirements, number, and duration of iterations at the start of the development process. Figure 1.7 shows agile methodology.



**Fig. 1.7 Agile Model of Software Development**

There are several types of Agile models. The two most popular types being Scrum and Kanban.

**Scrum** – This is the most popular Agile methodology. It involves a small team of developers working together in short sprints to deliver a working product incrementally. Each sprint typically lasts for 2–4 weeks. For each iteration under the Scrum model, developers need proper planning and necessary assessments from the previous sprint. Also, once a sprint is complete, it is not allowed to make any further changes.

**Kanban** – This methodology focuses on continuous delivery and improving workflow efficiency. Work is broken down into smaller pieces and tracked on a visual board, and team members pull work items as they are ready to work on them.

| SDLC Model                         | Advantages                                                                                                           | Disadvantages                                                                      | Best Use Case                                                                               |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------|
| <b>Waterfall Model</b>             | Easy to understand and use. Clear structure with defined stages.                                                     | Cannot move back to a previous stage once it's done. Doesn't handle changes well.  | Small projects where requirements are very well understood.                                 |
| <b>V-Shaped Model</b>              | Emphasizes rigorous testing and validation. Clear and simple structure.                                              | Like Waterfall, it's inflexible to changes.                                        | Projects with clear and fixed requirements, where high reliability is important.            |
| <b>Iterative Incremental Model</b> | Progressive elaboration of the product. Allows for refinement with each increment.                                   | Requires careful planning to make sure increments are meaningful.                  | Projects where it's beneficial to get basic functionality out quickly and refine over time. |
| <b>Spiral Model</b>                | High degree of risk management and flexibility. Allows for repeated iterations.                                      | Can be complex to follow/understand. Needs careful management.                     | Large, complex, and high-risk projects.                                                     |
| <b>Prototype Model</b>             | Reduces risk of failure, as a working model is seen early. Helps in getting user feedback and refining requirements. | May lead to too much focus on a limited prototype, not the full system.            | Projects where user requirements are unclear or complex.                                    |
| <b>Agile Model</b>                 | High flexibility and adaptability. Emphasizes customer satisfaction and team collaboration.                          | Can be difficult to estimate time and cost. Requires customer and team engagement. | Projects where requirements can change and quick, incremental delivery is desired.          |

#### 1.4 SELECTION OF SDLC MODELS

The selection of the software models for the development of software depends on various factors. The type of software is one of the probable aspects on which the software engineer can select the software model. Following are some of the examples of software for selecting the software engineering models.

For a **simple data processing project**, it is suggested to go for a waterfall **model**. In such projects, the requirements are fixed and there is no chance of changing it. Also for processing data, the basic operations are fixed and the software can be delivered. In the waterfall model, it is possible to implement all specified requirements and deliver the whole product at a time.

For a **new system comparing fingerprints**, it is suggested to go for a spiral model. In this project the requirement of fingerprint data will be added continuously. Even customer feedback i.e. usefulness of the project has to be checked every time. In the spiral model, for each set of requirements you need to follow six activities and in turn that spiral way will be continuing till the project is delivered.

For an **online inventory management system of the automobile industry** it is suggested to go for an incremental **model**. In this project, the new services are required to be delivered in every increment. It is not possible to deliver the entire module and even cannot delay the delivery. The first version of the software needs to be delivered with limited features. Later on, in each increment the new features can be added and delivered increment by increment.

For a **new missile tracking system**, it is suggested to go for a waterfall **model**. In this project the frequency range for tracking missiles is fixed. All data will be provided at start only. Using this data, the entire system has to deliver at a time only, so the waterfall model is useful here.

For a **satellite launching system**, a **prototype model** is suggested. Satellite launching system is a very costly system and the current hardware and software technology may not match the requirement. So instead of freezing is the requirement before any design or coding, it is suggested

to build a prototype. This will help to understand the requirements with minimal design, coding and testing.

For an **Online inventory management system**, a **waterfall model** is suggested. This software is not so costly. The software team can gather all the information from the user followed by analysis and development.

For a data **entry system for office staff** that have never used computers before, it is suggested to go for an incremental **model**. In this project the user interface and user friendliness are extremely important. The basic software would be developed and delivered. And in each increment some functional capability may be added to the system until the system is implemented. At each step, extensions and design modifications can be made and the testing can be done at each increment.

## CHECK YOUR PROGRESS

### A. Multiple choice questions

- Which of the following is not the characteristic of software (a) functionality (b) reliability (c) maintainability (d) probability
- Which of the following is not the task of system design (a) architecture design (b) graphics design (c) interfaces design (d) database design
- SDLC stands for (a) Software Development Life Cycle (b) System Development Life cycle (c) Software Design Life Cycle (d) System Design Life Cycle
- Which of the following SDLC model has parallel testing for each development stage (a) Waterfall model (b) V model (c) Prototype model (c) Agile model
- Which of the following is the most important feature of the spiral model? (a) Efficiency (b) Time (c) Risk (d) Quality
- Which of the following is the first step of SDLC? (a) Design (b) Testing (c) Development (d) Analysis
- Choose the advantage of the Iterative model among the following. (a) Simple to manage (b) Divided workload (c) Early revenue generation (d) All of the above
- When is the 'risk analysis' In the spiral model performed? (a) In first loop (b) Before using spiral model (c) Every loop (d) In first and second loop

### B. Fill in the blanks

- In \_\_\_\_\_ the code is designed, developed, and tested in repeated cycles.
- The Waterfall model is also called as a \_\_\_\_\_
- V model is also known as the \_\_\_\_\_
- The Scrum and Kanban are the most popular types of \_\_\_\_\_ methodology.
- The \_\_\_\_\_ model is the combination of iterative and waterfall models.
- The \_\_\_\_\_ model facilitates the reusability of components.
- The \_\_\_\_\_ model is built based on incremental and iterative development.
- In \_\_\_\_\_ stage of SDLC, training of the user staff, system documentation and implementation is done.

### C. State whether True or False

- The Agile model is the most recent and has a modern approach to software development.
- A working model of a system is a prototype.
- Design is the first step of coding.
- A waterfall model is the most suitable model for smaller projects.



5. Software development cost is higher than that of the software maintenance cost.
6. Testing approach changes based on the life cycle applied for development of a software.
7. In an incremental model, requirements do not need to be prioritized.
8. The selection of the software models for the development of software depends on various factors

**D. Answer the following questions in short.**

1. What is the difference between a computer program and software?
2. What are the different categories of software?
3. What is software engineering?
4. List the characteristics of software.
5. What are the phases of software development life cycle?
6. List the various software development methodologies.
7. What are the main types of Agile methodology?

## Session 2: Software Development Process

The software development process involves the activities related to the production of the software such as design, coding, and testing. The Software Development Life Cycle (SDLC) refers to a methodology with clearly defined processes for creating high-quality software. In this chapter you will understand in detail the phases of software development.

### 2.1. REQUIREMENT ANALYSIS

Requirement analysis is the process of determining user expectations for a new or modified system. It includes gathering, documenting, and analyzing the needs and constraints of stakeholders.

The following four steps are involved in this process.

1. Feasibility study,
2. Requirement elicitation and analysis,
3. Requirement specification,
4. Requirement validation

#### 2.1.1 Feasibility study

A feasibility study is conducted to determine the viability of the project. It examines all aspects of a proposed project, including technical, economic, financial, legal and environmental considerations. It explores various aspects such as usability, maintainability, productivity and integration. The report of the feasibility study contains recommendations whether or not the project should be developed or not.

There are five types of feasibility study.

1. Technical feasibility – It ensures the availability of technical resources such as hardware and software. The technical team converts the ideas into a working system.
2. Economic feasibility – It determines the cost and benefits analyses and suggests the potential economic benefits to the organization.
3. Operational feasibility – It analyses how the system being developed will meet the operations needs of the organisation.
4. Legal feasibility – It analyses the legal aspects such as zoning laws, data protection acts or social media laws.
5. Schedule feasibility – It estimates how much time a team needs to complete the project.

### **2.1.2 Requirement elicitation and analysis (Requirement Gathering)**

It is the process of gathering and defining the requirements for a software system. It is based on a clear and comprehensive understanding of the customer's needs and requirements. It involves the identification, collection, analysis, and refinement of the requirements. It involves the stakeholders including business owners, technical experts and end-users.

The various activities involved in requirement elicitation are as follows.

**a. Requirement discovery** – It is the process of interacting with stakeholders in the system to collect their requirements. Domain requirements from stakeholders and documentation are also discovered during this activity.

**b. Requirement classification and organisation** – This activity takes the unstructured collection of requirements, group related requirements and organizes them into coherent clusters.

**c. Requirement prioritization and negotiation** – This activity is concerned with prioritizing requirements, finding and resolving requirements conflicts through negotiation.

**d. Requirement documentation** – The requirements are documented and input into the next round of the spiral. Formal and informal requirements documents may be produced.

#### **Requirements Elicitation Techniques**

There are many techniques to obtain critical information from stakeholders. The most commonly used techniques are Brainstorming, Interview, Focus Group, Observation, Document Analysis/Review, Prototyping and Survey/Questionnaire.

**a. Brainstorming** – The subject matter experts discuss in group and generate new ideas to find a solution for a specific issue. Each member is given time to share their ideas.

**b. Interview** – In this technique, the interviewer asks the questions to stakeholders to obtain information. It can be structured or unstructured. The structured interview consists of the questions to get the answer is Yes or No form. In unstructured interviews, open-ended questions are used to get the detailed information.

**c. Document Analysis/Review** – This technique is used to gather the information through available documents. It includes business plans, technical documents, problem reports, and existing requirement documents. This is useful to update or migrate an existing system.

**d. Focus Group** – A focus group consists of 6 to 12 subject matter experts. They discuss the topic in the group and the moderator manages the discussion to analyze the results and provide findings to the stakeholders.

**e. Observation** – The necessary information can be obtained through observation. The observer records all the activities and the time taken to perform the work. Observation can be either active or passive. In active observation information is obtained by asking the questions while passive observation is silent and information can be obtained by observing the work.

**f. Prototyping** – In this technique, a prototype is created and demonstrated to the client to give an idea of the product. Prototypes can be used to create a mock-up of sites, and describe the process using diagrams.

**g. Survey/Questionnaire** – In this technique, a set of questions is given to stakeholders. Their responses are collected and analyzed to identify the area of interest. The question can be open ended or close ended. An open-ended question results in a descriptive answer and close ended questions has a predefined set of answers just as multiple choice questions.

### **2.1.3. Requirement specification**

Requirement specification is a process of jotting down all the system and user requirements in the form of a document called Software Requirement Specifications (SRS). These requirements must be clear, complete, comprehensive, and consistent. It is an important document for software development, technical architecture design, Software testing, project management, training, and writing user manuals.

### **2.1.4. Requirement validation**

Requirements validation is the process of checking the defined requirements. It helps to detect errors at an early stage of development. It is important to cross check the requirements from the customer to avoid the extensive rework. The cost of fixing a requirement is much greater than repairing design or coding errors. The change in requirements usually changes the system design, implementation and testing.

Let us have a detailed look at Software Requirements Specifications (SRS).

## **Software Requirements Specifications (SRS)**

A software requirements specification (SRS) document describes the overall functionality of the software as per the requirements. There are four types of requirements as listed below.

### **1. Functional Requirements**

Functional requirements describes the behavior of the software that fulfills the user requirements. It can be a calculation, technical details, data manipulation, processing, and other specific functionality. Development, manufacturing, verification, deployment, training, operations, support, and disposal are the eight generic functions that most systems must complete over their life cycle. Each of them is important when identifying all the functional requirements for a system.

### **2. Performance Requirements**

Performance requirements describe how well the software system accomplishes certain functions under specific conditions. For example, the software speed of response, throughput, execution time and storage capacity.

### **3. System Technical Requirements**

It specifies the technical characteristics of the software such as its architecture, hardware and software requirements, the programming language, operating system and interface. It describes the technical specifications and constraints to fulfill the user requirement.

### **4. Specifications**

The specification document clearly states the necessary features of the software product and what is expected to perform the desired task. The specifications must be clear, concise, complete, correct, and consistent.

### **2.2.1 Characteristics of Good SRS**

There are many characteristics of good software. Some of them are:

**Complete** – The SRS is said to be complete when it includes all the essential requirements related to functionality, performance, design, constraints, attributes, and external interfaces.

**Consistent** – It means that there are no conflicts between any set of requirements. It should not happen that processes produce different outputs for inputs coming from different sources.

**Correct** – SRS is said to be correct if it covers all the requirements that are actually expected from the system.

**Modifiable** – SRS should be capable of easily accepting changes to the system. To be modifiable, requirements documents must have a logical structure. Modifications should be properly indexed and cross-referenced.

**Ranked** – The organization and structure of the requirements document establish a ranking of specification statements based on stability and importance. It becomes difficult to create a document for large and complex problems.

**Testable** – A requirement defined in SRS can be tested and validated. For example the requirement, “*The system is user-friendly*” is not testable. It should be written as, “*The user interface should be menu driven with a tooltip for all the text boxes*”.

**Traceable** – The SRS is traceable if the origin of each of the requirements is clear and if it facilitates the referencing of each condition in future development or enhancement documentation.

**Unambiguous** – A requirement statement is unambiguous if it can only be interpreted in one way. The use of weak phrases or poor sentence structure will lead to misunderstanding in the specification statement.

**Valid** – To validate requirement specification, all project participants, including managers, engineers, and customer representatives, should be able to comprehend, analyze, and accept or reject it.

**Verifiable** – It means that the requirements mentioned in SRS ensure that it is being met by the system. The requirements are verified with the help of reviews. For example, a requirement stating that the system must be user-friendly is not verifiable then it should not be mentioned in SRS.

### 2.2.2 Structure of SRS document

The type of information included in SRS is determined by a number of factors, including the type of software being developed and the approach used in its development.

The general structure of SRS as proposed by IEEE standard is given below.

1. Introduction
  - 1.1 Purpose
  - 1.2 Scope
  - 1.3 Definitions, Acronyms and Abbreviations
  - 1.4 Reference
  - 1.5 Overview
2. Overall Description
  - 2.1 Product Perspective
  - 2.2 Product Functions
  - 2.3 User Characteristics
  - 2.4 General Constraints
  - 2.5 Assumptions and Dependencies
3. Specific Requirements
  - 3.1 External Interface Requirements
    - 3.1.1 User Interfaces
    - 3.1.2 Hardware Interfaces
    - 3.1.3 Software Interfaces
    - 3.1.4 Communication Interfaces
  - 3.2 Functional Requirements
    - 3.2.1 Mode 1
      - 3.2.1.1 Functional Requirement 1.1
      - 3.2.1.2 Functional Requirement 1.2
    - 3.2.2 Mode 2
      - 3.2.2.1 Functional Requirement 2.1

## 3.2.2.2 Functional Requirement 2.2

- 3.3 Performance Requirements
- 3.4 Design Constraints
- 3.5 Attributes
- 3.6 Other Requirements

The three main sections of the SRS document are described as follows.

**1. Introduction** – This provides an overview of the entire information described in SRS. It involves the purpose and the scope of SRS, which states the functions to be performed by the system. In addition, it describes definitions, abbreviations, and the acronyms used. The references used in SRS provide a list of documents that are referenced in the document.

**2. Overall description** – It describes the general factors that affect the product and its requirements. A general overview is presented to understand the specific requirements. A general abstract description of the functions to be performed by the product is given. Product perspective is essentially the relationship of the product to other products; defining if the product is independent or is a part of a larger product, and what the principal interfaces of the product are.

**3. Specific requirements** – It describes all the details that the software developer needs to know for designing and developing the system. This is the largest and most important part of the document. The external interface requirements section specifies all the interfaces of the software: to people, other software, hardware, and other systems. In the functional

## 2.2 SYSTEM ANALYSIS

System analysis is one of the initial stages of a software development life cycle. Analysis is a detailed study of the various operations performed by a system and their relationships within and outside the system. It is an in-depth evaluation study of the system that consists of the processes involved in analysing and modeling the system. In the analysis process the system is broken down into components and analysed how each component interacts with the other components to accomplish the system's overall goal. System analysts having a good knowledge of software products and its requirements conduct the system analysis. They collect the requirements of the system and document them.

### 2.2.1 Structured Analysis Tools

Structured Analysis is a systematic approach to understand the system and its activities in a logical way. It uses graphical diagrams to develop the system specifications. It mainly focuses on logical systems and functions to convert the requirements into computer programs. Some of the commonly used structured analysis tools are explained below.

#### 1. Data Flow Diagram (DFD)

It is a technique developed by Larry Constantine to represent the system in a graphical form. It is a graphical representation that shows the flow of data between various functions of a system and specifies how the current system is implemented. It gives an overview of system processes, transformations performed, data stored, results produced and where they flow.

There is a difference between DFD and Flowchart. The flowchart depicts flow of control in program modules. DFDs depict flow of data in the system at various levels. DFD does not contain any control or branch elements.




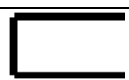
#### Types of DFD

Data Flow Diagrams are either Logical or Physical.

**Logical DFD** – This type of DFD concentrates on the system process, and flow of data in the system. For example in a Banking software system, how data is moved between different entities.

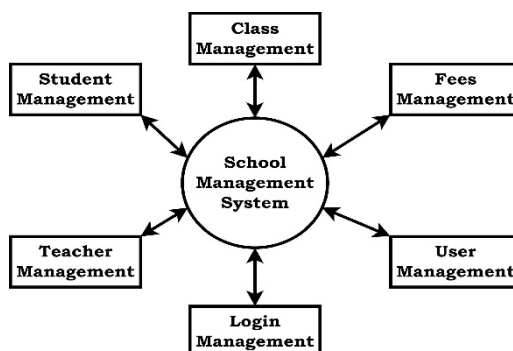
**Physical DFD** – This type of DFD shows how the data flow is actually implemented in the system. It is more specific and close to the implementation

#### DFD Symbols

| Symbol                                                                            | Meaning                                                                                                 |
|-----------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------|
|  | Square defines a source or destination of data.                                                         |
|  | Arrow identifies data flow, means the data in motion. It is a pipeline through which information flows. |
|  | Circle or a bubble represents a process that transforms incoming data flow into outgoing data           |
|  | Open rectangle is a data store, or data at rest, or a temporary repository of data                      |

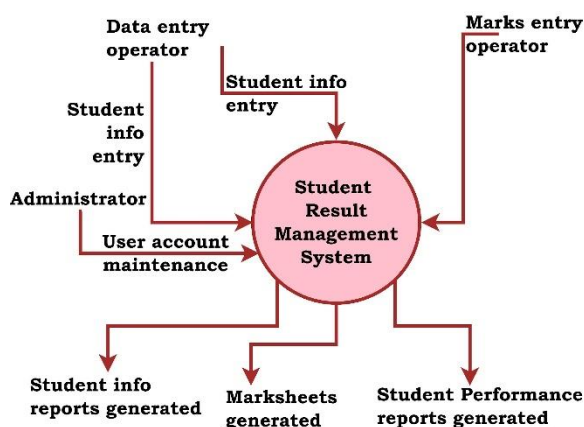
**Levels of DFD**

**Level 0 or Context diagram** – Level 0 DFDs are also known as *context diagrams*. It is the highest abstraction level, which depicts the entire information system as one diagram. It starts with mentioning major processes with little details and then goes on giving more details of the processes with the top-down approach. It establishes the context in which the system operates such as who are the users, what data do they input to the system, and what data they received by the system. The data input to the system and the data output from the system are represented as incoming and outgoing arrows. Figure 2.1 shows the context diagram of the school management system.



**Fig. 2.1 Context diagram of school management system**

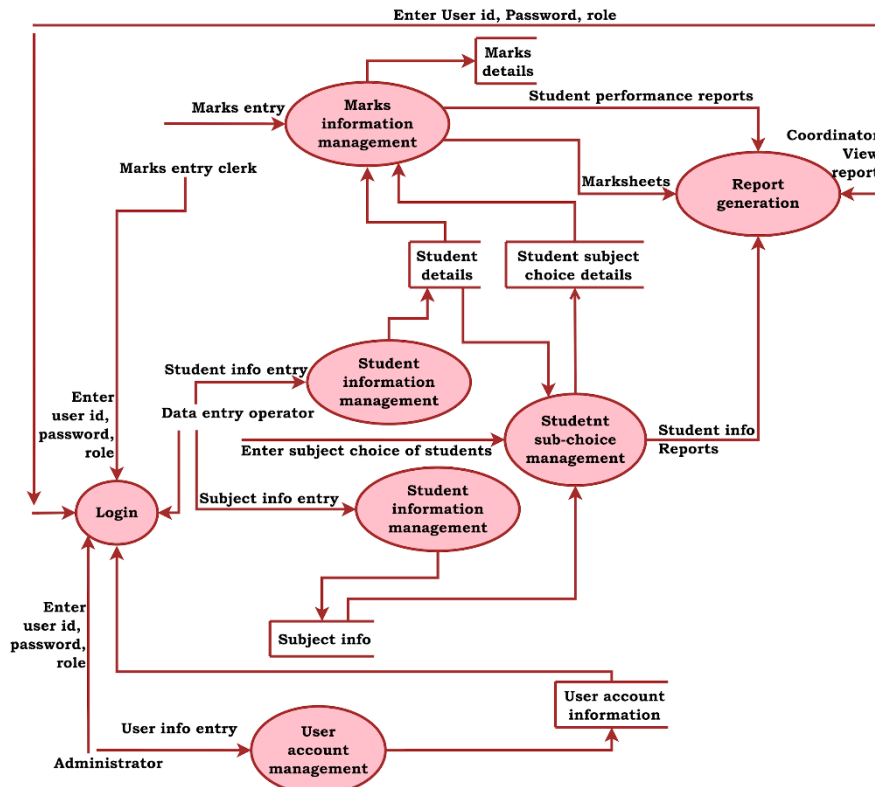
The Level 0 DFD or context diagram for the result management system is shown in Figure 2.2. The bubbles are decomposed into less and less abstract bubbles, the corresponding data flow may also be needed to be decomposed.



**Fig. 2.2 Level 0 DFD of result management system**

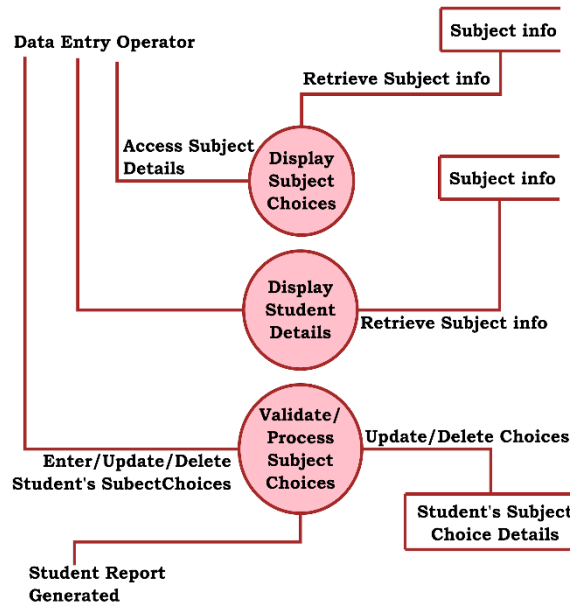
**Level 1** – It is the more specific level of DFD. Level 1 DFD depicts basic modules in the system and flow of data among various modules. Level 1 DFD also mentions basic processes and sources

of information. Figure 2.3 shows Level 1 DFD of the result management system, where high level processes of Level 0 are further broken down into subprocesses.



**Fig. 2.3 Level 1 DFD of result management system**

**Level 2** – At this level, DFD shows how data flows inside the modules mentioned in Level 1. Higher level DFDs can be transformed into more specific lower level DFDs with deeper levels of understanding unless the desired level of specification is achieved. Figure 2.4 shows Level 1 DFD.



**Fig. 2.4 Level 2 DFD Student subject choice management**

**2. Data Dictionary**

A data dictionary is also called meta-data (data about data). It is a structured repository of data elements in the system. It stores the descriptions of all DFD data elements. Data dictionary

provides a way of documentation for the complete database system in one place. Validation of DFD is carried out using a data dictionary. Data dictionary contains the following items.

**Data Elements** – It is the smallest unit of data that provides for no further decomposition. For example, DATE consists of day, month and year

**Data Structure** – It is a group of data elements handled as a unit. For example, a phone is a data structure consisting of four data elements: area-code-exchange-number-extension.

**Data Flows and Data Stores** – data flows are data structures in motion, whereas data stores are data structures at rest. A data store is a location where data structures are temporarily located

A typical data dictionary for the book is as follows.

| Data Field | Description              | Data Type | Length |
|------------|--------------------------|-----------|--------|
| STUD_NAME  | Name of the student      | Character | 20     |
| DOB        | Date of Birth of student | Date      | 8      |
| SEX        | Sex of the student       | Boolean   | 1      |
| AGE        | Age of the student       | Number    | 2      |

### 3. Decision Trees

A Decision Tree is a graph that uses a branching method to display all the possible outcomes of any decision. It helps in processing logic involved in decision-making. It is a diagram that shows conditions and their alternative actions within a horizontal tree framework. Decision trees depict the relationship of each condition and their permissible actions. A square node indicates an action and a circle indicates a condition.

For example, Bookstores get a trade discount of 25%; for orders from libraries and individuals, 5% allowed on orders of 6-19 copies per book title; 10% on orders for 20-49 copies per book title; 15% on orders for 50 copies or more per book title. A decision tree for this is shown in Figure 2.5.

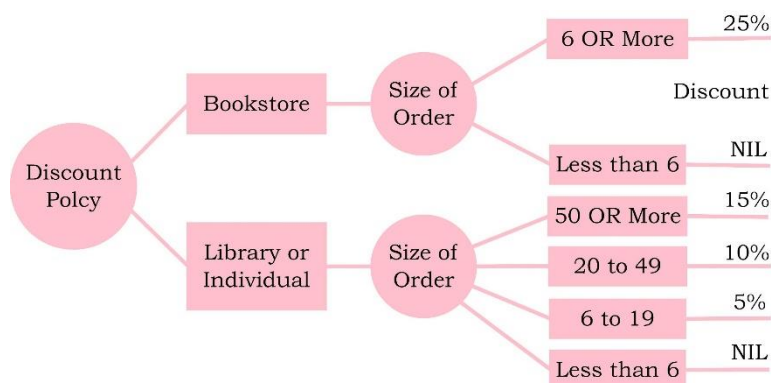


Fig. 2.5 Decision tree

### 4. Decision Tables

Decision table is a structured tabular format that represents conditions and actions. It is an effective debugging tool that assists in grouping similar information into a single table. A decision table contains condition entries, condition stubs, action entries, and action stubs.

Decision tables are a precise yet compact method of modeling complex logic. A decision table is a table with rows and columns divided into four quadrants.

|            |                         |
|------------|-------------------------|
| Conditions | Conditions Alternatives |
| Actions    | Action Entries          |

Decision tables are precise and compact methods of modeling complex logic. It links conditions to actions like if-then-else and switch-case statements.

The entries in the decision table are given by Decision Rules which define the relationships between combinations of conditions and action. In rules section,



Y shows the existence of a condition.

N represents the condition is not satisfied.

A blank against action states it is to be ignored.

X (or a check mark will do) against action states it is to be carried out.

**Example :** The Decision table for the problem mentioned in the Decision tree can be represented as below.

| Condition Stub |                                  | Condition Entry     |   |   |   |   |   |
|----------------|----------------------------------|---------------------|---|---|---|---|---|
|                |                                  | 1                   | 2 | 3 | 4 | 5 | 6 |
| If             | Customer is Bookstore            | Y                   | Y | N | N | N | N |
| (Condition)    | Order size 6 copies or more ?    | Y                   | N | N | N | N | N |
|                | Customer Librarian or Individual |                     |   | Y | Y | Y | Y |
|                | Order-size 50 copies or more ?   |                     |   | Y | N | N | N |
|                | Order-size 20-49 copies ?        |                     |   |   | Y | N | N |
|                | Order-size 6-19 copies ?         |                     |   |   |   | Y | N |
| Then           | Allow 25% Discount               | X                   |   |   |   |   |   |
| (Action)       | Allow 15% Discount               |                     |   | X |   |   |   |
|                | Allow 10% Discount               |                     |   |   | X |   |   |
|                | Allow 5% Discount                |                     |   |   |   | X |   |
|                | No Discount allowed              |                     |   |   |   |   | X |
|                | <b>Action Stub</b>               | <b>Action Entry</b> |   |   |   |   |   |

### 5. Structured English

Structured English is the description of the programming code in simple English. It uses common verbs such as IF-THEN-ELSE and DO WHILE-ENDDO. Structured English is based on structured logic, used to express all logic in terms of sequential structures, decision structures, iterations and case structures. It assists programmers to write error-free code.

**Example:** The Structured English notation for the problem mentioned in the Decision tree can be represented as below.

IF order is from Bookstore

and-IF order is for 6 copies or more per book title

THEN: Discount is 25%

ELSE (order is for fewer than 6 copies per book title)

SO: no discount is allowed

ELSE (order is from libraries or individuals)

ELSE (order is from libraries or individuals)

SO-IF order is for 50 copies or more per book title

Discount is 15%

ELSE IF order is for 20 to 49 copies per book title

Discount is 10%

ELSE IF order is for 6 to 19 copies per book title

Discount is 5%

ELSE (order is for less than 6 copies per book order)

SO: no discount is allowed

### 6. Pseudocode

Pseudocode is normally produced before the target code is generated for the software application. It is written in the programming language used. It can be thought of as an augmented programming language, with lots of comments and descriptions.

The Pseudocode to check if the given integer is even or odd, can be written as below.

```

READ X
COMPUTE x%2
IF x%2 == 0
    PRINT "Even Number"
ELSE
    PRINT "Odd Number"
EXIT

```

### 2.2.3 Guidelines for Selecting Appropriate Tools

As you have seen the various tools for structured analysis. It is the choice of the analyst to choose the appropriate tool for the system. The following guidelines are suggested for selecting the most appropriate tool to suit the requirements.

1. DFD is used at high or low level analysis for providing good system documentations.
2. Data dictionary is used to simplify the structure for meeting the data requirement of the system.
3. Structured English is used if there are many loops and actions are complex.
4. Decision tables are used when there are a large number of conditions to check and logic is complex.
5. Decision trees are used when sequencing of conditions is important and if there are few conditions to be tested.

## 2.3 SYSTEM DESIGN

Systems design is the process of defining elements of a system like modules, architecture, components and their interfaces and data for a system based on the specified requirements. The interfaces, designs, data, and modules must all meet the system requirements. In this phase the SRS document is converted into an implementable format that decides how the system will operate. The overall system design process includes output design, input design, database design, program construction, processing, testing and documentation.

### 2.3.1 Types of System Design

There are different types of system design, each of which has its own unique approach. Some common types of system design are:

**1. Conceptual Design** – Conceptual design is the first stage of system design and focuses on developing a high-level conceptual model of the system. This includes defining the system's goals and objectives, identifying key components and their interactions, exploring alternative design options. Conceptual design involves brainstorming, research, and stakeholder input to create a broad understanding of the system.

**2. Logical Design** – Logical design involves creating a more detailed model of the system. It includes the processes, data flow, and functional requirements needed to achieve the system's goals. It describes the inputs, outputs, databases, procedures in a format that meets the user requirements. This involves creating flowcharts, data models, and other visual representations of the system's structure and processes. Data flow diagrams, E-R diagram modeling are used.

**3. Physical Design** – Physical design relates to the actual input and output processes of the system. It focuses on how data is entered into a system, verified, processed, and displayed as output. Physical design includes testing and prototyping to ensure that the system will function as intended. It is concerned with user interface design, process design, and data design.

**4. User Interface Design** – User interface design focuses specifically on designing the visual and interactive components of a system, such as menus, buttons, and screens. This involves creating wireframes and prototypes, as well as defining the specific user requirements and preferences.

User interface design is essential for ensuring that the system is user-friendly and easy to navigate, which can greatly enhance its overall usability.

### 2.3.2 Low Level and High Level System Design

System design is conducted at two levels that are Low-Level Design and High-Level Design.

#### 1. Low-Level System Design

Low-Level Design, refers to the component-level design process, where the large system is broken down into smaller and manageable components that define how these components will interact with each other. This may involve creating detailed system diagrams, flowcharts, and other visual representations of the system's architecture. It also includes the detailed design of software algorithms, data structures, interfaces, specific programming languages, software libraries, and hardware components that are required to implement the system's functionality. Low-level system design follows the higher-level conceptual and logical design phases, and is focused on the specific implementation details of the system. The low-level system design includes defining the database architecture and ER-diagrams, creating tables and defining relationships between them, deciding a design pattern, classes structure and models.

#### 2. High Level System Design

This is the first phase of system design, and is focused on creating a comprehensive blueprint for the system that can guide the detailed design and implementation phases that follow. It refers to the process of designing the overall architecture and components of a system at conceptual level. It involves creating a broad understanding of the system's goals and objectives, identifying key components and their interactions, and exploring alternative design options. The goal of high-level system design is to provide a clear and complete picture of the structure and functionality.

### 2.3.3 Software Design Strategies

Software design is a process to conceptualize the software requirements into actual design. The design strategies help to conceptualize a plan into the best possible design for implementing the intended solution.

#### Structured Design

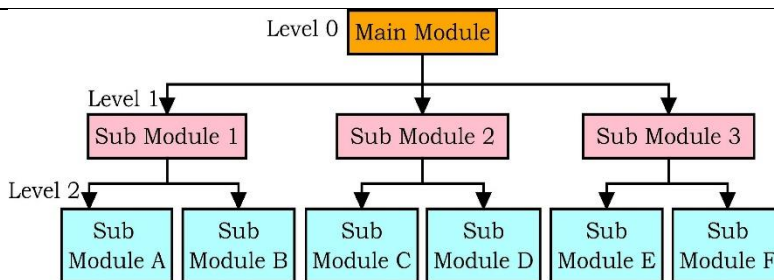
The main objective of structured design is to minimize the complexity and increase the modularity of a program. Structured design is based on the *divide and conquer* technique, in which a large problem is divided into several small tasks. The small pieces of the problem are solved by means of solution modules. Solution modules are used to address the individual problems. In structured design these modules are arranged in hierarchy to communicate with each other to produce exact results.

#### Modularization

Structured design uses the modularization approach to minimize the complexity of the project. In this approach the program is divided into small and independent modules in a top down manner. There are two types of modular design strategy – *top down* and *bottom up strategy*.

#### 1. Top-Down Strategy

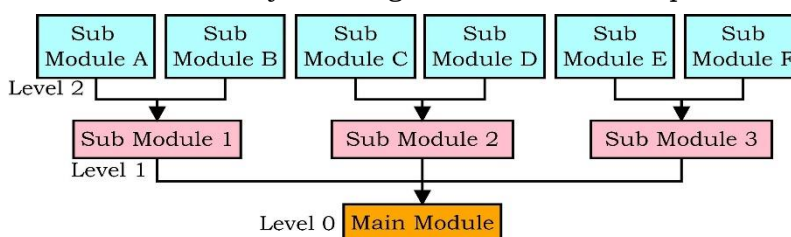
The top-down strategy uses the modular approach to design the system. It starts from the highest-level or topmost module and moves towards the lowest level or bottom modules. In this technique, the main module is divided into several small modules or segments based on the task performed by each module. Then, each module is further subdivided into several submodules of the next lower level. Figure 2.6 shows the top down design strategy.



**Fig. 2.6 Top down design strategy**

**2. Bottom-Up Strategy**

In this technique, the modules at the most basic or the lowest level are grouped together based on the function performed by each module to form the next higher-level modules. Then, these modules are further combined to form the next higher-level modules. Bottom-up strategy is more suitable when a system needs to be created from some existing system, where the basic primitives can be used in the newer system. Figure 2.7 shows the top down design strategy.

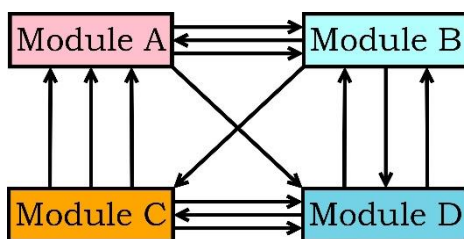


**Fig. 2.7 Bottom up design strategy**

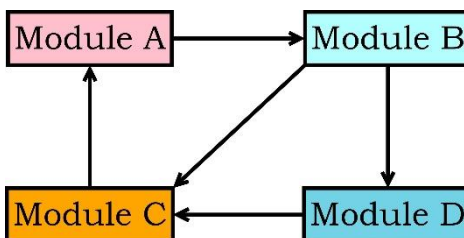
**Coupling and Cohesion**

These are the two important concepts related to the system development that help in determining the complexity. Structured design follows the rules of *cohesion* and *coupling*. The high cohesion and low coupling arrangements result in good structured design.

Figure 2.8 (a) shows that in high coupling all the four modules are highly dependent on each other, while in low coupling all the modules are not much more dependent on each other as shown in Figure 2.8 (b).



**Fig. 2.8 (a) High coupling**



**Fig. 2.8 (b) Low coupling**

**2.3.4 Structured Design Tools**

There are several tools used in system design. Following are some of them.

**a) Structure Charts**

The structure chart illustrates the hierarchical layout of modules. In the Hierarchical structure, the components are read from left to right and from top to bottom. A structure chart is a top

down and modular design method where a complete problem is divided into many modules till each module becomes manageable. It is a design tool that displays the relationship between program modules. It consists of diagram consisting of rectangular boxes that represent the modules, connecting arrows, or lines as shown in the Figure 2.9.

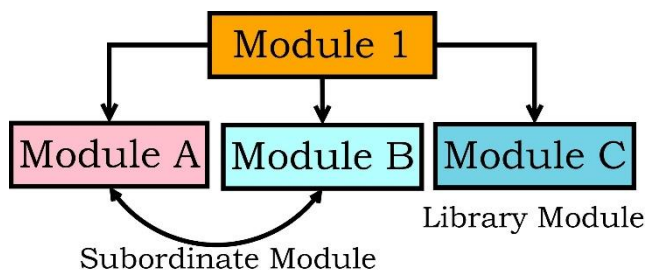


Fig. 2.9 Structured chart

**b) HIPO Diagram**

HIPO (Hierarchical Input Process Output) diagram is a combination of two organized methods for system analysis and documentation. It organizes the software modules into a hierarchy. It emphasizes on functions of the system rather than structure, logic or organisation. This is a commonly used tool for developing system software. They are also used for documenting information. Their graphical representation allows one to get a visual representation of the system structure. The HIPO diagram is used by the analyst to obtain a high-level view of system functions. The functions are further divided into sub-functions as shown in Figure 2.10.

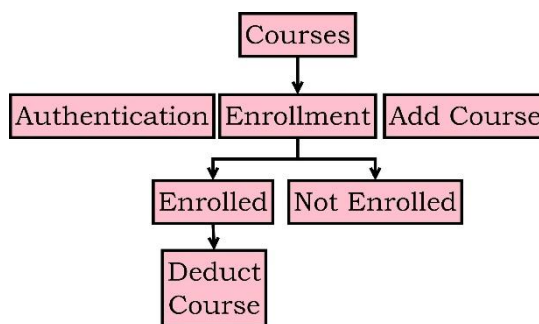
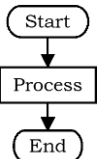
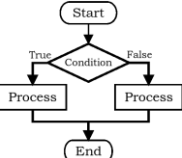


Fig. 2.10 HIPO diagram

**c) Structured Flowchart**

These are graphic tools used to represent the structure of the software in modular and top down fashion. It creates an illustration that visually analyzes the workings of a system or process. The titles may indicate process stages or actions that attend the process and its various stages. All of the processes and decisions fit into one of a few basic structured elements. The structured flowchart helps to create new algorithms by encapsulating a range of data points inside an interlinked illustration. The structures used in flowchart include: Sequence, Decision, Loop, Case.

| Flowchart structure                                                                                                                                                                                           | Example                                                                              |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| <p><b>Sequence</b> – A sequence is the simple flowchart to depicts the steps in sequential manner one after another as shown in the exemplar flowchart.</p>                                                   |  |
| <p><b>Decision</b> – This structure is used to make the decision on the specified condition. The control is transferred to the next step after checking the condition as shown in the exemplar flowchart.</p> |  |

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |  |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|
| <p><b>Loop</b> – There are two types of structure as shown in the exemplar flowchart that allows to repeat a task over and over. In the first type the task is performed first then the condition is checked. If the condition is true the task is repeated and if the condition is false the loop is terminated. In the second type the condition is checked initially. If the condition is true the task is repeated and if the condition is false the loop is terminated. In the first type the loop is executed at least once while in the second type the loop may not be executed at all.</p> |  |
| <p><b>For loop</b> – It is used if the task is to be performed for a specified number of times. For that the condition is checked as shown in the exemplar flowchart. Depending on the condition, the task may not be performed at all. There is also a "For Each" structure that is like the for loop, but has no counter. It will go through each item of a collection and do the task.</p>                                                                                                                                                                                                       |  |
| <p><b>Case</b> – It is used to perform several tasks based on the generated output. The case structure makes a flowchart more readable and saves space on the paper.</p>                                                                                                                                                                                                                                                                                                                                                                                                                            |  |
| <p><b>Start and End</b> – Each flowchart must have one starting point. The terminal shape is used for the start and end.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |  |
| <p><b>Connector Block</b> – It is used to transfer the control to another point which may be another procedure or located in another page. It is shown a letter enclosing in a circle.</p>                                                                                                                                                                                                                                                                                                                                                                                                          |  |

**2.3.5 Entity Relationship Model**

Entity-Relationship Model (ER Model) is a graphical approach to database design. It defines the data elements and their relationship within a specified system or software. ER Model is used in database design to visualize database tables and their relationships between them. ER models are used through a defined set of symbols such as ovals, diamonds, rectangles, and connecting lines to display the interconnectedness between them.

Entity Relationship Diagram (ERD) is a way of representing ER Model graphically. It is a type of flowchart used for identifying different system elements and how they interact with each other. It helps to understand how different entities, objects, or concepts are related to each other within a system.

**Components of ER Diagram**

There are three principal components of the ER diagram as shown in Figure 2.11. ER diagrams are created using three components: *entities*, *attributes*, and *relationships*.

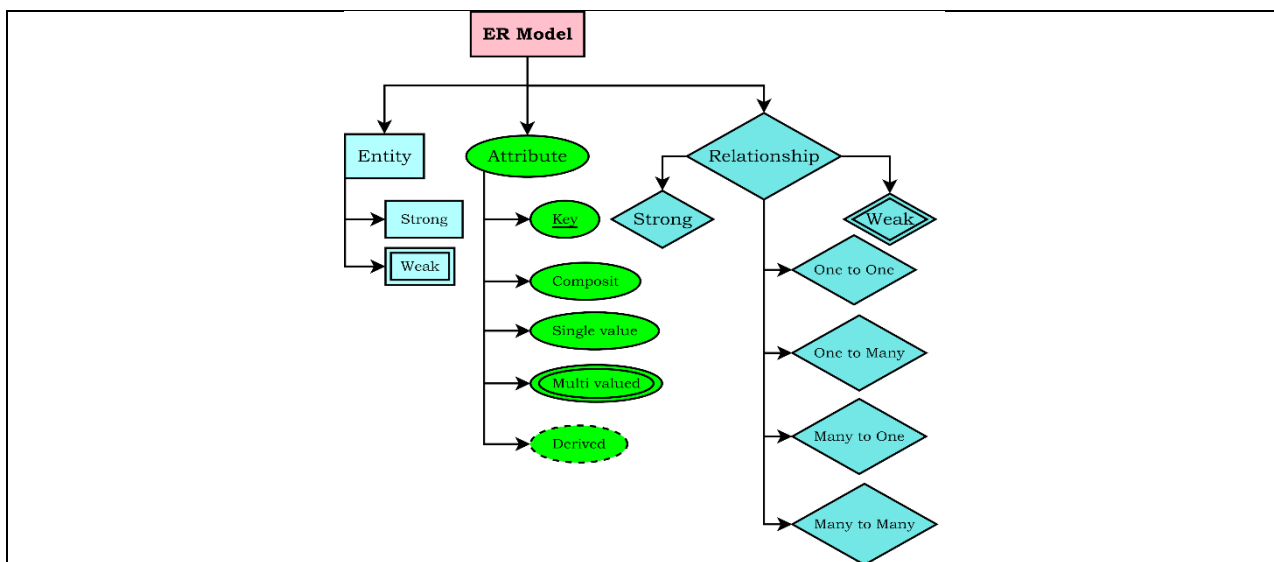

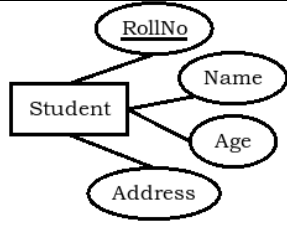
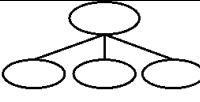
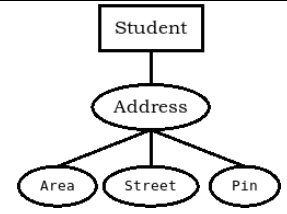




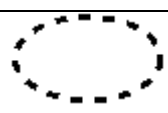

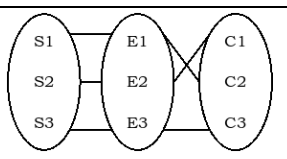
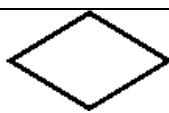
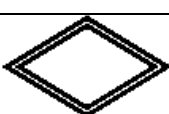


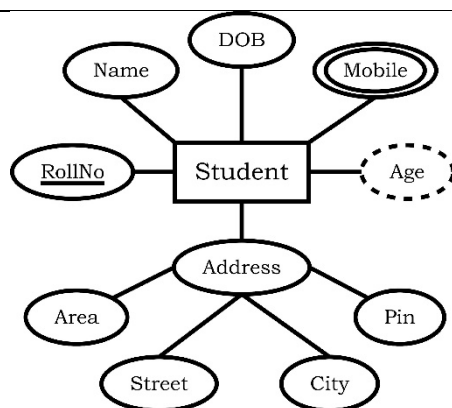
Fig. 2.11 ER Model

| Terms                                                                                                                                                                                                    | Symbol | Example |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|---------|
| <b>Entity</b> – It is an object that can have attributes associated with it. It is represented by a rectangle in the ER diagram. For example, a Student is an entity.                                    |        |         |
| <b>Attribute</b> – It represents the characteristic or property of an entity. It is represented by an oval in the ER diagram. For example, Name, Age and Address are the attributes of a Student entity. |        |         |
| <b>Relationships</b> – It defines how the entities are related to each other. It is represented by a diamond in the ER diagram. For example, Teaches are the relationships between student and teacher.  |        |         |
| <b>Entity Set</b> – An entity set is a collection of related types of entities. For example, a Student set may contain all the students of a school.                                                     |        |         |
| <b>Strong Entity</b> – It does not depend on other Entity. It has a key attribute as primary key to identify it uniquely. It is represented by a rectangle in the ER diagram.                            |        |         |
| <b>Weak Entity</b> – It is dependent on a strong entity for existence. It does not have key attributes such as primary key. It is represented by a double rectangle in the ER diagram.                   |        |         |

|                                                                                                                                                                                                                                    |                                                                                      |                                                                                       |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| <p><b>Key attribute</b> – It is an attribute that uniquely identifies an entity among the entity set. For example, The RollNo uniquely identifies the student.</p>                                                                 |    |    |
| <p><b>Composite attribute</b> – It is a combination of other attributes. For example, In student entity, the student address is a composite attribute as it is composed of Area, Street, Pin.</p>                                  |    |    |
| <p><b>Single-value attribute</b> – It is an attribute having only a single value. For example, the Aadhar number of an individual is a single value attribute.</p>                                                                 |    |    |
| <p><b>Multi-valued Attribute</b> – It is an attribute having more than one value. For example, a person can have more than one phone number, email-address.</p>                                                                    |    |    |
| <p><b>Derived attribute</b> – It is an attribute that does not exist in the physical database, but their values are derived from other attributes present in the database. For example, age can be derived from date_of_birth.</p> |   |   |
| <p><b>Relationship set</b> – A set of relationships of a similar type is known as a relationship set. Like entities, a relationship too can have attributes. These attributes are called descriptive attributes.</p>               |                                                                                      |  |
| <p><b>Strong Relationship</b> – A strong or identifying relationship is when the primary key of the related entity contains the primary key of the “parent”.</p>                                                                   |  |                                                                                       |
| <p><b>Weak Relationship</b> – A weak or non-identifying relationship exists between two entities when the primary key of one of the related entities does not contain a primary key component of the other related entities.</p>   |  |                                                                                       |

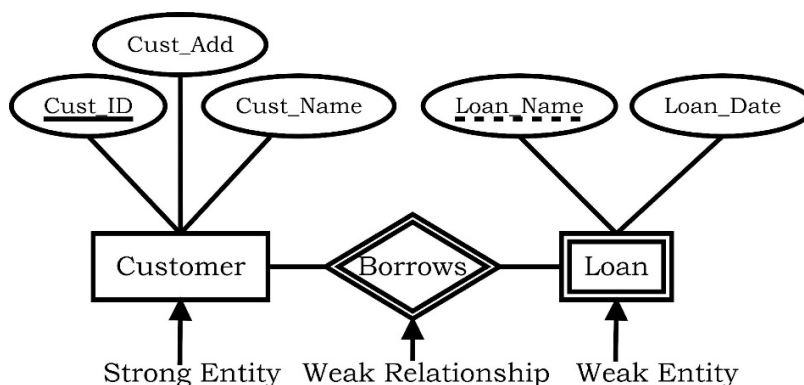
The complete entity type Student with its attributes can be represented as shown in Figure 2.12. In this ERD, Student is entity, RollNo is a key attribute, Mobile is multivalued attribute, Age is derived attribute and Address is composite attribute.





**Fig. 2.12 Student entity with attributes**

A strong relationship exists between two strong entities, denoted by a single diamond and strong entity holds the weak relationship with the weak entity, denoted by double diamond in the ER diagram. Figure 2.13 shows the weak relationship between the strong entity Customer and weak entity Loan.



**Fig. 2.13 Weak relationship between strong and weak entity**





**Degree of a relationship set**

The *degree of relationship* depends on the number of different entity sets participating in a relationship.

|                                                                                                                                                                                                                                                                                                                                                                                                                                                            |  |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|
| <p><b>Unary relationship</b> – If there is only one entity in the relationship then it is called a unary relationship. For example, one person is married to only one person.</p>                                                                                                                                                                                                                                                                          |  |
| <p><b>Binary relationship</b> – It is a relationship between the instances of two entity types. For example, the Teacher teaches the subject.</p>                                                                                                                                                                                                                                                                                                          |  |
| <p><b>Ternary relationship</b> – It is a relationship amongst instances of three entity types. The relationships “<i>may have</i>” provide the association of three entities, i.e., Teacher, Student, and Subject. All three entities are many-to-many participants. There may be one or many participants in a ternary relationship. In general, <b>n</b> entities can be related by the same relationship and is known as <b>n-ary</b> relationship.</p> |  |

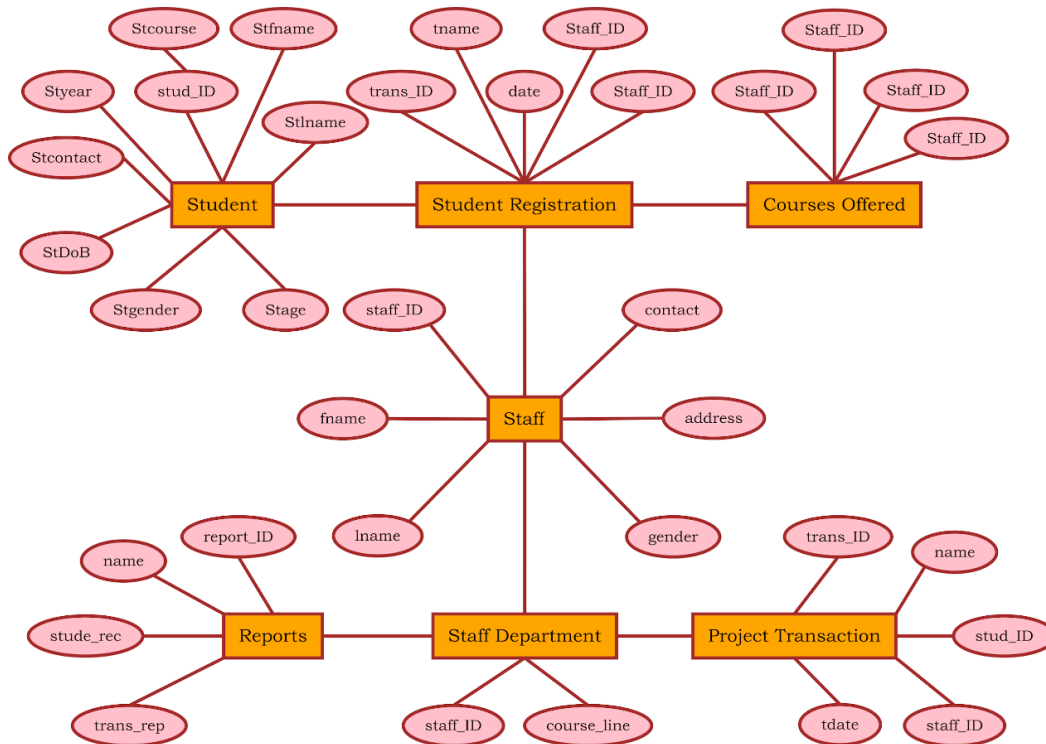
**Cardinality and its Types**

The number of times an entity of an entity set participates in a relationship set is known as **cardinality**. There are four types of cardinalities.

|                                                                                                                                                                                                                                                                                                                                                          |                                                                                     |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|
| <p><b>1. One to One</b> – One entity from entity set A can be contained with at most one entity of entity set B and vice versa. Let us assume that each student has only one student ID, and each student ID is assigned to only one person. So, the relationship will be one to one.</p>                                                                |   |
| <p><b>2. One to many</b> – When a single instance of an entity is associated with more than one instance of another entity then it is called one to many relationships. For example, a client can place many orders; an order cannot be placed by many customers.</p>                                                                                    |   |
| <p><b>3. Many to One</b> – More than one entity from entity set A can be associated with at most one entity of entity set B, however an entity from entity set B can be associated with more than one entity from entity set A. For example, many students can study in a single school, but a student cannot study in many school at the same time.</p> |   |
| <p><b>4. Many to Many</b> – One entity from A can be associated with more than one entity from B and vice-versa. For example, the student can be assigned to many projects, and a project can be assigned to many students.</p>                                                                                                                          |  |

**ER Diagram for Student Management System**

A sample ER diagram for the Student management system is shown in Figure 2.14 with entities and their associated attributes.



**Fig. 2.14 Student management system**

## 2.4 SYSTEM DEVELOPMENT

In this phase the actual coding and building of the software takes place based on the requirement and design specifications. The activities involved in the development phase are:

1. **Coding** – Developers write the actual code for the software based on the software's requirements and design documents. It ensures all planned features and functionality are implemented.
2. **Integration** – The developers write code and integrate different components. These components can be different modules or features of the software, and integration is critical to ensure that they work together seamlessly.
3. **Component Testing phase** – Component testing is an ongoing activity during the development phase. Each part of the software is tested individually to ensure it works correctly. Testing is done through unit tests that check small pieces of code for correctness.
4. **Review and Revision** – The code is regularly checked and corrected to improve quality and fix any errors to keep standards, and to identify potential problems. It helps to maintain code quality and consistency throughout the project.
5. **Version Control** – Systems are used to manage changes in the code, helping to organize and track different versions. Version control systems play a critical role to track changes, manage different versions of the software, and facilitate collaboration among team members. These systems are especially important in projects where multiple developers are working on different parts of the software at the same time.
6. **Documentation** – Developers write documents and comments within the code to explain how it works. It is essential for future maintenance, updates, and for new team members who may join the project at a later stage.

### Key principles of the software development phase

The development phase of the software development process is guided by several key principles, each of which plays a critical role in ensuring the creation of a successful and efficient software product. Here are five key principles:

#### 1. Code Quality and Cleanliness

The code that the developers are writing should be functional, readable, maintainable and well-documented. It should be according to coding standards, using meaningful names for variables and functions, and logically organizing the code.

#### 2. Modularity and Scalability

The code should be modular. The separate modules can be independently developed, tested, and maintained. This approach enhances the scalability, and expansion of the software as per the evolving requirements.

#### 3. Efficient Problem Solving

The ability to effectively tackle and solve problems is essential. This includes debugging skills, the ability to think logically and algorithmically, and the creativity to find innovative solutions to complex challenges.

#### 4. Collaboration and Communication

Software development is typically a collaborative effort, requiring developers to work closely with each other, as well as with other stakeholders like project managers, quality assurance teams, and clients.

#### 5. Adherence to Requirements and Design

Development should be closely aligned with predefined requirements and design specifications. This ensures that the software meets the need and expectations of users and stakeholders to perform as intended in the real-world environment.

Together, these principles ensure that the software developed is not only technically sound, but also well aligned with user needs and business goals.

## 2.5 SYSTEM TESTING

Software testing is an important stage in the software development process to ensure that it works properly. Software testing is the process of evaluating and verifying the functionality of the software that matches the expected requirements, specification, functionality, and performance of a software. It helps to enhance the quality of the software in terms of accuracy, reliability, scalability, practicability, usability, portability and reusability.

The process of software testing involves running the software under controlled conditions at various levels.

### Verification and validation testing

Verification and validation are two essential processes in software testing.

**Verification** is the static testing process that checks the software functioning against the specified requirements. The main activities involved in the verification process are – inspection, review, walkthrough, and desk-checking.

**Validation** is the dynamic testing process that focuses on evaluating the software at the end of the development process to determine whether the software product meets the customer's expectations and requirements or not.

### Types of Software Testing

Software testing is a complex process carried out through various types of software testing, each designed to meet specific objectives and address different aspects of the software. The various types of testing can be divided into various levels as shown in Figure 2.15.

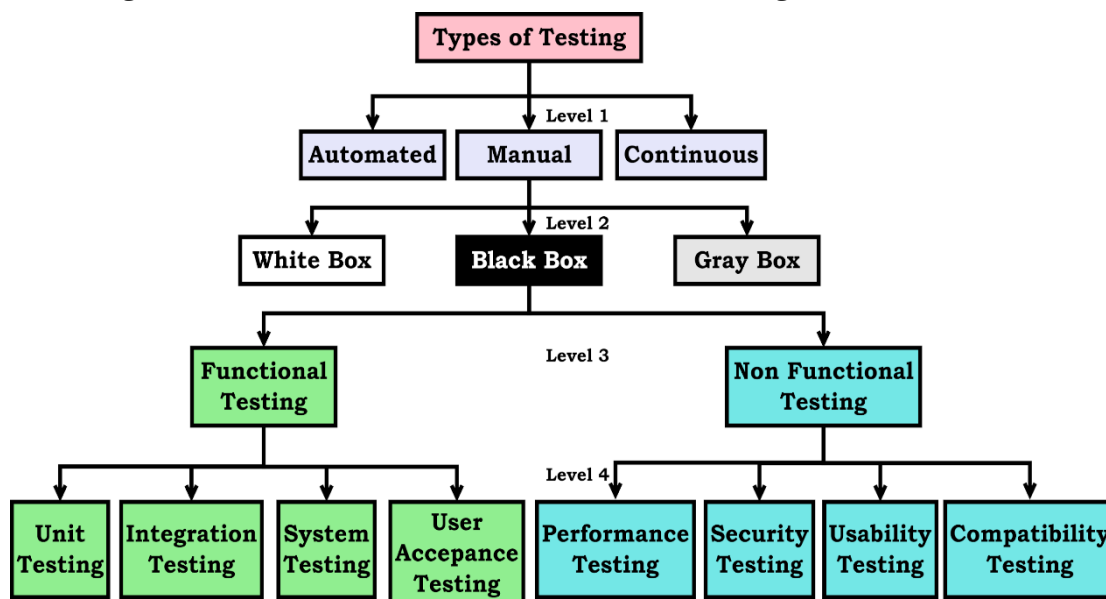


Fig. 2.15 Types of Testing

### Testing Types

At the first level there are three types of testing – manual, automatic and continuous.

**Manual Testing** – In manual testing the functionality of an application is checked manually without using any automation tool or script. It uses test plans, test cases, or test scenarios to test software.

**Automated Testing** – Automated testing uses automated software testing tools to run tests repeatedly and quickly. It increases the test coverage, improves accuracy, and saves time and money as compared to manual testing.

**Continuous testing** – Continuous testing integrates testing into every phase of the software development lifecycle, providing continuous quality assurance and rapid feedback on potential problems.

### Testing approaches

There are three types of testing approaches – black-box, white-box and gray box testing.

**Black Box Testing** – In this type of testing the tester does not have access to the source code of the system that's being tested. It considers the system's external behavior and does not require programming skills. It can be used for functional tests, as well as for non-functional tests such as performance testing, usability, and accessibility. It is done by the Quality Assurance (QA) team at higher levels. It takes less time to perform.

**White-Box Testing** – It is also known as clear box testing, glass box testing, code-based testing, or structural testing. In this type of testing the tester is aware of the internal workings of the system and has access to its source code. It focuses on the logic and the implementation of the software. Is usually done through automation testing. Is based on a good understanding of the system's code and usually done by the developers.

**Grey Box Testing** – It is a combination of white-box testing and black-box testing. Its aim is to search for the defects, if any, due to improper structure or usage of applications. The gray box tester may not have complete knowledge of an application's source code but may have partial knowledge of it and/or access to design documentation.

### Functional Testing and Non-Functional Testing

In the next level, software testing is classified as functional testing and non-functional testing.

#### A. Functional Testing

Functional testing tests the specific actions and features of the software against the functional requirements. The various types of functional testing are Unit testing, Integration testing, System testing and User Acceptance Testing.

**1. Unit Testing** – Unit testing is a method in which individual units or the components of the software application are tested. Unit testing helps to identify bugs at an early stage and improves the overall quality of the software. It is mainly executed at the early stage of software development. It is seen as a function, procedure, or method. For example, in unit testing, the login button is tested to ensure it can route to the correct page link.

**2. Integration Testing** – In this method the different units or modules of the software application are integrated to test the system as a whole. The advantage of this method is that it helps to identify errors when the different units of the software work together. In integration testing, errors about performance, requirements, and functional level are investigated. In unit testing, individual units are tested, however, in integration testing, such units' performance is checked when they are integrated.

#### 3. System Testing

System testing is mainly executed to investigate the behavior, architecture, and design of the software. In system testing, all the integrated modules of the complete system are tested to verify and validate the system requirements. It involves a different test that includes validating output in terms of particular input and the user's experience. Here, performance and quality standards are tested in compliance with the technical and functional specifications.

#### 4. User acceptance testing

In this type of testing the end users test the software to verify that it performs the required tasks in real-world scenarios according to the specifications. There are two main subtypes: alpha and beta testing.

**Alpha Testing** – It is conducted by internal staff in a controlled environment, to identify bugs and issues before the software is released to external users.

**Beta Testing** – In this testing the software is distributed to the external users to test its functionality in real-world conditions. It collects feedback and user experience to make the necessary changes before the official release of the software.

### **B. Non-functional Testing**

Non-functional testing considers the non-functional aspect of the software like performance, usability, reliability, portability, efficiency, security, and others.

The various types of functional testing are Performance testing, Security testing, Usability testing and Compatibility testing.

#### **1. Performance Testing**

This test evaluates the performance of the system on the parameters such as speed, responsiveness, and stability under various conditions. It is essential to ensure that the software performs well in terms of speed and response time under expected workload. Load testing, Stress testing, Spike testing, Endurance testing, Scalability testing are the subtypes of performance testing.

#### **2. Security testing**

Security testing focuses on identifying vulnerabilities, threats, and risks in a software application to prevent malicious attacks. It involves two crucial aspects of testing-authentication and authorization. Security testing makes the application secure and able to store confidential information when required. It also checks the behavior of the software related to attacks from hackers and how it should be maintained for data security upon noticing such attacks. The different types of security testing includes: Penetration Testing, Vulnerability Scanning, Security Auditing, Security Scanning, Ethical Hacking, Portability Testing.

#### **3. Usability Testing**

Usability testing is done to ensure the quality and easiness of application usage. A gaming application's usability testing checks whether it is operated by both hands, the color of the background, the vertical scroll, and others. The type of usability testing includes the Cross-Browser Testing, Accessibility Testing, Exploratory Testing.

#### **4. Compatibility testing**

Compatibility testing focuses on evaluating whether a software application works as intended across different browsers, databases, hardware, operating systems, mobile devices, and networks. It ensures the compatibility of the software with different environments and configurations.

### **2.6. SYSTEM IMPLEMENTATION**

It is the process of installing, configuring, and integrating systems. It involves various tasks, such as setting up hardware and software components, migrating data, customizing features, training users, and providing support. Software implementers use different techniques, standards, and best practices to ensure the software works properly and meets the needs and goals of the organization. System implementation allows access to the latest technology by replacing old applications with new software. New applications increase customer satisfaction with a more user-friendly experience. System implementation requires technical, operational, and organizational skills.

#### **Software implementation methodologies**

Software implementation methodologies are frameworks or models that provide the principles, practices, and procedures to achieve the desired outcomes. These are the software development methodologies that include Waterfall, Agile, Iterative and Incremental approach. Each methodology has its own advantages and disadvantages.

The best software implementation methodology depends on various factors, such as the size, complexity, and nature of the project, user requirement and preferences. Therefore, it is

important to evaluate and compare the pros and cons of each software implementation methodology, and choose the one that suits the project's goals, requirements, and context.

### **Elements of a successful software implementation**

The following are the elements that are considered for the successful implementation of software.

#### **Defining the organization's needs**

It is necessary to understand the organization's needs to choose the appropriate software in terms of its deliverables, number of users, platform, functionality and compatibility with the organization's existing systems and security features.

#### **Choosing the appropriate software**

After understanding the organization's needs, it is easy to look for the appropriate software. Search for the latest system with innovative features that meet the organisation requirement.

#### **Installing the application**

Generally, the vendors provide the installation support with no extra charge. The vendors may provide the support for installation with the collaboration with the IT department. This collaboration can ensure all necessary devices get the application and facilitate seamless integration with existing systems. If the IT department installs the new application independently, try providing the vendor's instructional manual or contact information to troubleshoot potential issues.

#### **Configuring features**

Once the application is installed, configure the most basic features first. Configuring simple features typically involves using the program's default settings. Keeping the initial configuration process simple also allows us to troubleshoot underlying issues before adding more complicated features.

#### **Customizing features**

Initially the default settings can help to start the system with basic features. To offer more flexibility, it is necessary to customize the advanced features of the system. Additional customization helps employees to understand their progress and encourage them to meet their goals.

#### **Integrating with existing systems**

During the selection process, it's important to choose an application that can integrate with the organization's systems. Compatibility allows multiple features to work together and prevent errors. As your team integrates the new application, they might consider how to transfer data from systems that the organization is no longer using. Automatic data migration can help your team save time while protecting sensitive information, including customer payment details.

#### **Training employees**

A good training program can ensure employees understand how to use the new application. It may emphasize how the software differs from old systems and how employees can optimize the various features. As part of the training program, consider providing employees with their account login information and establishing the appropriate permissions.

#### **Testing the software**

It is obvious that some errors may be encountered while using new software. Application testing allows us to evaluate the effectiveness of each feature and identify bugs that affect multiple users. Identify the features that require improvement, and make the appropriate adjustments and conduct more testing. Testing also allows us to identify issues that the vendor is responsible for addressing.

#### **Software Implementation Challenges**

There are some challenges faced by the development team while implementing the software. Some of them are mentioned below.

**Code-reuse** – The interfaces of modern programming languages are very sophisticated and are equipped with huge library functions. Still, to bring down the cost, the organization management prefers to re-use the code, created earlier for some other software. There are huge issues faced by programmers for compatibility checks and deciding how much code to reuse.

**Version Management** – Every time a new software is issued to the customer, developers have to maintain version and configuration related documentation. This documentation needs to be highly accurate and available on time.

**Target-Host** – The software program, which is being developed in the organization, needs to be designed for host machines at the customer's end. But at times, it is impossible to design a software that works on the target machines.

### **Software Documentation**

Software documentation is a comprehensive collection of written materials that describe and explain a software system. It includes various documents that provide the design, functionality, architecture, and use of the software. The documentation serves as a critical resource for developers, stakeholders, and end users to help them understand, use, and maintain the software effectively.

Types of software documentation

There are five major types of software documentation addressing different aspects of the development lifecycle.

#### **1. Process documentation**

Process documentation focuses on capturing and describing the workflows, procedures, and software development methodologies involved in a software development life cycle. It serves as a comprehensive guide for project managers, system administrators, software developers, and stakeholders, to understand the sequence of tasks, dependencies, and responsibilities throughout the software lifecycle. There are two types of documents in this category.

**Process Flowcharts** – Visual representations of the step-by-step sequence of activities and decisions in a software development process, providing a clear overview of the workflow and potential branching paths.

- a) **Standard Operating Procedures (SOPs)** – Detailed written instructions that outline the specific tasks, roles, and responsibilities for each phase of the software development lifecycle, ensuring consistency to established processes.

#### **2. Requirement documentation**

It contains all the functional, non-functional and behavioural descriptions of the software. It includes a detailed outline of the software's intended features, functionality, and performance expectations. This documentation serves as the foundation for the entire development process, providing a clear roadmap for creating the software and guiding the work of developers, designers, and testers. There are five key types of requirements documentation.

**Software Requirements Specification (SRS)** – It provides a detailed description of the software's functional and non-functional requirements. It includes user needs, system capabilities, constraints, and interfaces, serving as the foundation for the entire development process.

- a) **Use Cases** – Descriptions of interactions between users and the software, illustrating how the software responds to various user actions. Use cases help in identifying system behaviour and understanding user interactions with the application.
- b) **Functional Requirements** – It includes statements that specify the software's expected behaviour and the actions it must perform in response to certain inputs. In other words, functional requirements define the features and functionalities that the software should deliver.



- c) **Non-Functional Requirements** – It specifies system qualities, such as performance, security, usability, scalability, and reliability.
- d) **User Stories** – These are short, user-centric descriptions of software features from an end-user’s perspective. It focuses on capturing specific functionalities and benefits that users expect from the software.

### 3. User documentation

This documentation focuses on providing information and instructions to end-users on how to use and interact with a software product. It may include software installation and uninstallation procedures, user-guides, and special references to get more information like license updation. It bridges the gap between the technical complexities of the software and the users’ perspectives, making the software accessible and usable. It plays an important role in enhancing the overall user experience by knowing the software’s functionalities and features.

User documentation comes in various forms for different levels of user expertise. Some of the main examples of user documentation are as follows.

**User Manuals** – It provides detailed instructions on how to use a software, including setup, navigation, and feature explanations.

1. **Tutorials** – Step-by-step instructional materials that walk users through specific tasks or workflows, helping them gain hands-on experience with the software.
2. **Knowledge Bases** – Extensive repositories of articles, guides, and troubleshooting resources, empowering users to explore and resolve more complex software development problems independently.
3. **Release Notes** – Documents highlighting new features, bug fixes, and changes in software updates, keeping users informed about the latest improvements.
4. **User Guides** – Targeted documentation to specific user roles or use cases, providing guidance based on user needs.
5. **Video Tutorials** – These are visual guides in the form of videos, demonstrating software functionalities and guiding users through specific tasks.

### 4. Architecture design documentation

This type of documentation deals with the structural design and organization of a software system. It provides a comprehensive view of the software’s high-level components, their interactions, and the architecture of behavior of software. It is a detailed blueprint that helps developers, architects, and other stakeholders understand the complexity of the system and make informed decisions during development, maintenance, and future enhancements. Architecture documentation also plays a critical role in ensuring the scalability, maintainability, and robustness of software, as it outlines the rationale behind design decisions and helps teams identify potential bottlenecks or areas for optimization. There are three key types of architectural documentation.

**High-Level Architecture Diagram** – A visual representation of the software’s overall structure, showing major components, their interactions, and the flow of data. This diagram provides a high-level view of the system’s architecture and helps stakeholders understand its key building blocks.

1. **Data Flow Diagram (DFD)** – It is the graphical representation showing the data flow through the software, depicting inputs, outputs, and data processing steps.
2. **Class Diagram** – It depicts the software’s class structure, showing classes, their attributes, methods, and relationships. It helps to understand the software’s object-oriented design and inheritance relationships.

These documents give all necessary information that is required for coding and implementation.

### 5. Technical documentation

It is specifically designed for developers. It gives a detailed understanding of the codebase, algorithms, APIs, and other technical aspects. It mentions the objective of the code, information about the code, who wrote it, where it will be required, what it does and how it does, what are the other resources used in code. It increases the understanding between various programmers working on the same code. It enhances reusability of the code. It makes debugging easy and traceable.

There are various automated tools available and some come with the programming language itself. For example, Java comes with the JavaDoc tool to generate technical documentation of code.

Some examples of technical documentation include:

**Code comments** – Inline explanations within the source code that provide context, rationale, and explanations for specific code segments, making the code more understandable and maintainable.

1. **API documentation** – Comprehensive documentation that describes the usage, parameters, return values, and functionality of application programming interfaces (APIs), enabling developers to integrate external services or libraries seamlessly.
2. **Technical specifications** – Detailed descriptions of the software's technical requirements, design decisions, data formats, and protocols, aiding developers in understanding the overall system architecture and implementation.
3. **User guides for developers** – Instructional materials specifically tailored to help developers effectively use software libraries, frameworks, or SDKs, simplifying system integration and promoting best practices.
4. **Data models and database schema** – Documentation detailing the structure, relationships, and constraints of the database tables and data models, assisting developers in database management.
5. **Configuration guides** – Instructions on how to set up and configure the software environment, ensuring consistent deployment and optimal performance.

## 2.7. SYSTEM MAINTENANCE

Maintenance is the essential feature of any product. Software maintenance is a critical part of the software lifecycle. It is essential to keep software effective and up-to-date after it's initially released.

Software maintenance is the process of updating, improving and correcting a software application after its initial release to ensure its continued effectiveness and alignment with user needs and technological advancements. The main purpose of software maintenance is to modify and update software applications to correct faults and to improve performance.

Software maintenance is necessary for various purposes. It includes various tasks, such as correcting faults or bugs, improving the design, implementing enhancement, securing against new types of cyber threats, interfacing with the systems, adding or modifying hardware, software, system features to improve the overall performance of software.

Software maintenance is key to ensure that software continues to meet customer needs, remains compatible with other technologies, and complies with regulatory requirements.

Software Maintenance Types

There are four types of software maintenance – Corrective, Adaptive, Perfective, and Preventive. They all play an essential role in maintaining software systems.

### 1. Corrective Software Maintenance: Bug fixing and fault resolution

Corrective software maintenance is focused on identifying and correcting errors in software that are either reported by users or discovered through internal testing. It rectifies errors and faults in logic, code, and design or to enhance the performance of the system after the implementation. It is required when software doesn't function properly due to some faulty logic flow, wrong

implementation, invalid or incomplete tests. This type of problem needs immediate attention as it hampers the day to day work of the end user. Proper planning and interaction with the end user during the system development process can minimize the occurrence of corrective maintenance.

### **2. Adaptive Software Maintenance: Keeping pace with change**

The software system once developed needs the changes over a period of time due to change in environment, organisation functioning, policy changes. The client may require the system with modified functionality, platform, or new operating environment as per the requirement of new hardware and software. Maintenance of the software to adapt to this kind of changes is called adaptive maintenance. This activity is not as urgent as corrective maintenance as these changes are gradual and allow sufficient time to the system group to make changes to the software. Adaptive maintenance is necessary to ensure the functionality of software in a changing technological landscape.

### **3. Perfective Software Maintenance: Tuning for peak performance**

Perfective maintenance focuses on enhancing and improving the software to meet evolving user needs. It includes adding new features, improving user interfaces, and optimizing software performance. It makes the software more efficient, easier to use, and relevant in a competitive marketplace. It enhances the user experience and extends the useful life of software assets, leading to improved system performance.

### **4. Preventive Software Maintenance: Anticipating and preventing issues**

Preventive software maintenance helps to prevent the system from any future vulnerabilities. It includes tasks such as optimizing code, refining documentation, and updating systems to improve overall performance and maintainability. It addresses problems, which are not significant at this moment but may cause serious issues in future. It reduces the need for corrective maintenance. It is done when the system is least used or not used at all. It does not add value to the system, but lowers the cost of corrective maintenance.

### **Cost of Maintenance of Software**

The cost of software maintenance is very high to about 60% to 67% of the total cost of the software development process. The standard lifespan of any software is upto 10-15 years. With advancement in software technology, the cost of maintenance of old software becomes very high. The most common method to correct any software problem used by engineers is the trial and error method and the changes are usually left undocumented which in return causes conflicts in future. This process changes the structure of the original software structure. Other factors at the software-end are programming language used for the development of software, structure of the software, reliability and availability of staff, dependence on the external software.

The cost of software maintenance is influenced by several factors such as, size and complexity of the software, program lifetime, dependency on external environment, hardware stability and frequency of change in software system.

## **CHECK YOUR PROGRESS**

### **A. Multiple choice questions**

1. Which of the following is the functional testing. (a) Unit testing (b) Performance testing (c) Security testing (d) Compatibility testing.
2. Which of the following is the non-functional testing. (a) Unit testing (b) System testing (c) Usability testing (d) User acceptance testing
3. Requirement elicitation is the process of (a) gathering the requirements (b) jotting down the requirements (c) checking the requirements (d) noting the requirements

4. Requirements validation is the process of (a) gathering the requirements (b) jotting down the requirements (c) checking the requirements (d) noting the requirements
5. Requirement specification is a process of (a) gathering the requirements (b) jotting down the requirements (c) checking the requirements (d) noting the requirements
6. Which of the following is not the type of attribute in ERD (a) key attribute (b) derived attribute (c) strong attribute (d) multivalued attribute
7. Which of the following is not the type of entity in ERD (a) Entity set (b) Strong entity (c) Weak entity (d) Key entity
8. Which of the following is not the type of relationship in ERD (a) Unary relationship (b) Binary relationship (c) Ternary relationship (d) Null relationship
9. Which of the following is not a structured analysis tool (a) DFD (b) Data dictionary (c) Structured chart (d) Structured English
10. Which of the following is not a structured design tool (a) HIPO diagram (b) Structured flowchart (c) Structured chart (d) Structured English

### B. Fill in the blanks

1. The number of times an entity of an entity set participates in a relationship set is known as \_\_\_\_\_.
2. Observation can be either \_\_\_\_\_ or \_\_\_\_\_.
3. Close ended questions have a \_\_\_\_\_ set of answers.
4. Technical requirements describes the technical specifications and \_\_\_\_\_.
5. The requirements are verified with the help of \_\_\_\_\_.
6. Level 0 DFD is also called as \_\_\_\_\_.
7. Structured analysis helps to understand the system in a \_\_\_\_\_.
8. DFD can be \_\_\_\_\_ and \_\_\_\_\_.
9. Validation of DFD is carried out using \_\_\_\_\_.
10. Structured English is based on \_\_\_\_\_.
11. \_\_\_\_\_ Structured design follows the rules of \_\_\_\_\_ and \_\_\_\_\_.
12. Structured design is based on the \_\_\_\_\_ and \_\_\_\_\_ technique.
13. Two types of modular design strategy are \_\_\_\_\_ and \_\_\_\_\_.
14. \_\_\_\_\_ Preventive maintenance reduces the need of \_\_\_\_\_ maintenance.
15. Strong entity has a \_\_\_\_\_ attribute as primary key.

### C. State whether True or False

1. Unit testing helps to identify bugs at an early stage.
2. Integration testing identifies errors when the different units of the software work together.
3. The structured interview consists of the open ended questions.
4. Functional requirements describe the behavior of the system.
5. The information in active observation can be obtained just by observing.
6. Data dictionary is the structured repository of data elements.
7. Decision tree depicts the relationship of each condition and their permissible actions.
8. Structured English is the description of the programming code in simple English.
9. Pseudocode is normally produced after the target code is generated.
10. Logical design relates to the actual input and output processes of the system.
11. High Level Design, refers to the component-level design process.
12. The low cohesion and high coupling arrangements result into good structured design.
13. Structure chart is a top down and modular design method.
14. HIPO diagram organizes the software modules into a hierarchy.
15. The For loop may or may not execute at all depending on the result of the condition.
16. The ER Model is a graphical approach to database design.
17. A strong relationship exists between two strong entities.
18. Strong entity holds a weak relationship with the weak entity.
19. The *degree of relationship* depends on the number of different entity sets participating in a relationship.

20. Validation is the static testing process and Verification is the dynamic testing process.

**D. Answer the following questions in short.**

1. What are the types of feasibility studies?
2. What are the types of documentation?
3. What are the four types of software maintenance?
4. What are the various activities involved in requirement elicitation?
5. What are the various required elicitation techniques?
6. What is the difference between DFD and Flowchart?
7. What is the difference between logical and physical DFD?
8. List the elements of a data dictionary?
9. What is meta data? Give examples of metadata.
10. What are the different types of system design?
11. What is the difference between a strong entity and a weak entity?
12. What is the difference between white box testing and black box testing?
13. What is gray box testing?

## Assignment

### Project Implementation using Concepts of Software Engineering

Let us consider a prototype software project, details of which are as given below.

**Project Title:** Banking Management System

#### **Purpose of the Project**

The main purpose of the software project is to simplify the tedious task of banking by providing a user friendly environment. It also aims at increasing the efficiency and reducing the drawbacks of existing manual banking processes. This makes it a more convenient banking tool for the customers.

#### **Scope**

In this system, only certain banking operations are permitted. It includes opening of a new account, updating of account, deposit amount in account, withdrawal and applying for the loan.

#### **Definitions**

The abbreviations used in this project are as below.

BMS – Banking Management system

UI - User Interface

DBMS – Database Management System

#### **Process Model**

The first step in the project development is to decide the appropriate software development model. In this project we have selected the Waterfall model because it is simple and easy to implement. Also the project requirements are well known. This model provides us with a structured approach for software development.

#### **Software Requirement Specification**

##### **Overall Description**

The manual banking system has a lot of paperwork and the data is stored on paper only. It is a very sluggish and time consuming process that is inconvenient to the customers. The existing system cannot tolerate the increasing number of customers. Hence the automated banking system is proposed. The proposed system will decrease the amount of paperwork and automate certain tasks.

This system will provide the user-ID validation and hence unauthorized access is prevented.

##### **Product functions**

The “BMS” software is an independent web based application. There are various user interfaces related to this software. These interfaces help the user to interact with the software and provide the necessary information for the online banking system. In order to achieve the automatic banking system it is necessary to divide the entire functionality of the system into different modules. The modular approach is used as a design strategy.

The software is divided into following modules. These are

1. Customer Management
2. Loan System
3. Transaction System

#### **Module 1. Customer Management**

In this module, the user can open a new account or can update an existing account by providing the details such as name, father name, address, phone number and Email id.

#### **Module 2. Loan System**

In this module the customer can apply for a loan by providing required documents and details like time period of loan, amount of loan and get the detailed description of EMIs.

#### **Module 3: Transaction System**

This module allows the customers to deposit and withdraw money from their account using some private information like signature and OTP.

#### **Programming Platform for Software**

This Banking Management System (BMS) can be implemented by using the Python programming language.

#### **User characteristics**

The software user interface will be in English language and hence the user should be comfortable in English language and basic usage of computer and Internet.

#### **General Constraints**

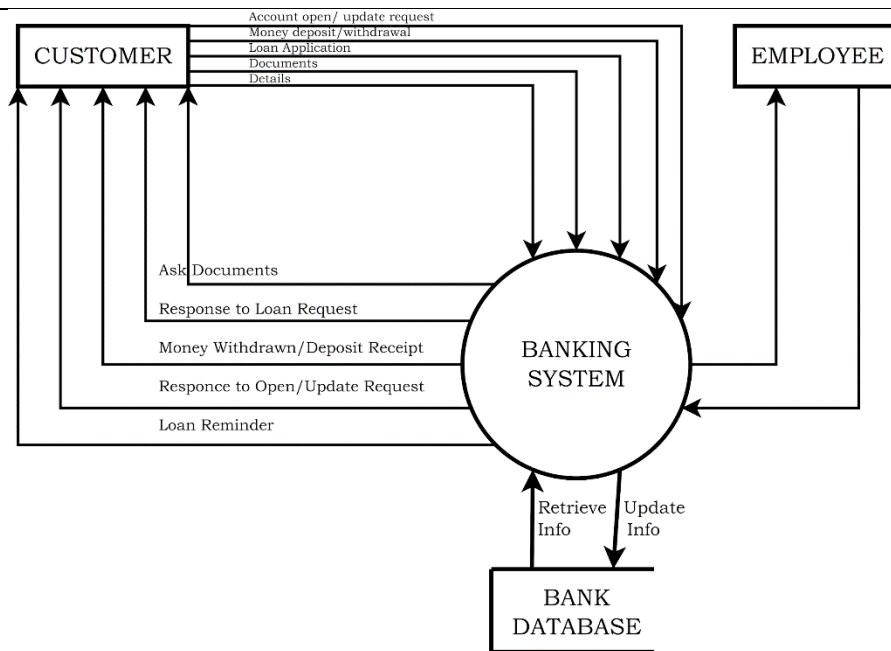
The database of the system should be accessible only by the authorised person. Any other person should not be able to access it.

#### **Data Flow Diagram (DFD)**

The data flow diagram indicates the flow of the data of software systems to the different nodes as shown below.

#### **DFD for system and system users**

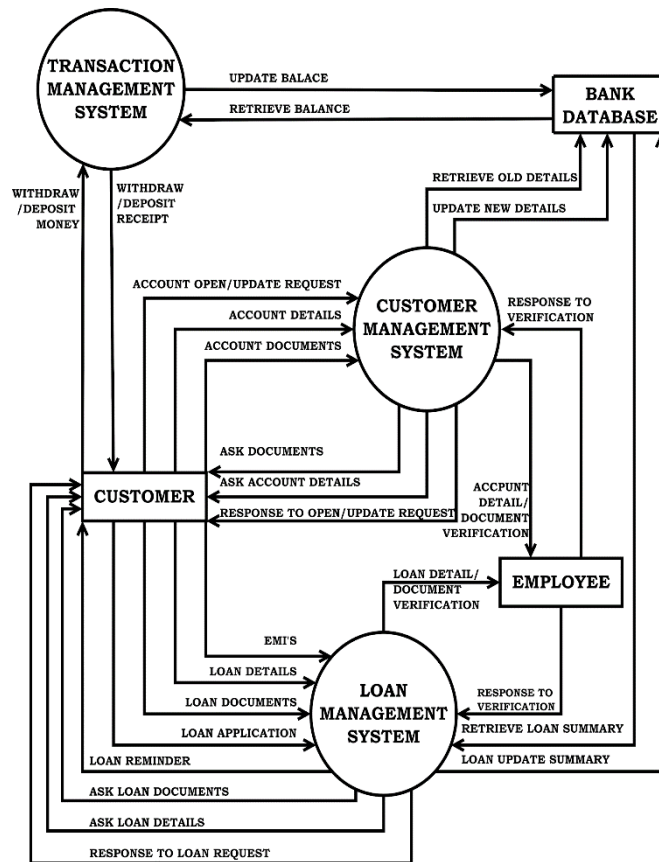
The DFD shown in Figure 2.16 shows that the users of the system can be employees of bank and customers are registered account holders.



**Fig. 2.16 DFD of users of the system**

**DFD for working of online banking system**

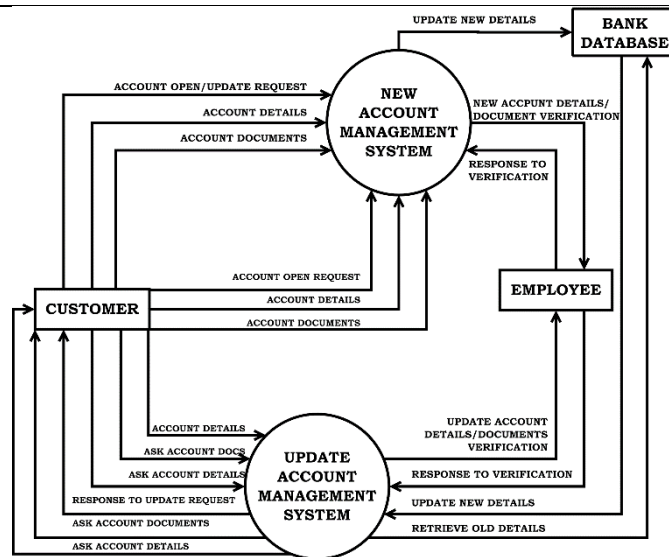
The DFD in Figure 2.17 shows the system with a Bank database and Transaction Management System. The Customer Management System and Loan Management System which has the access to Customer and Employee of the bank.



**Fig. 2.17 DFD for working of online banking system**

**DFD of Customer Management System**

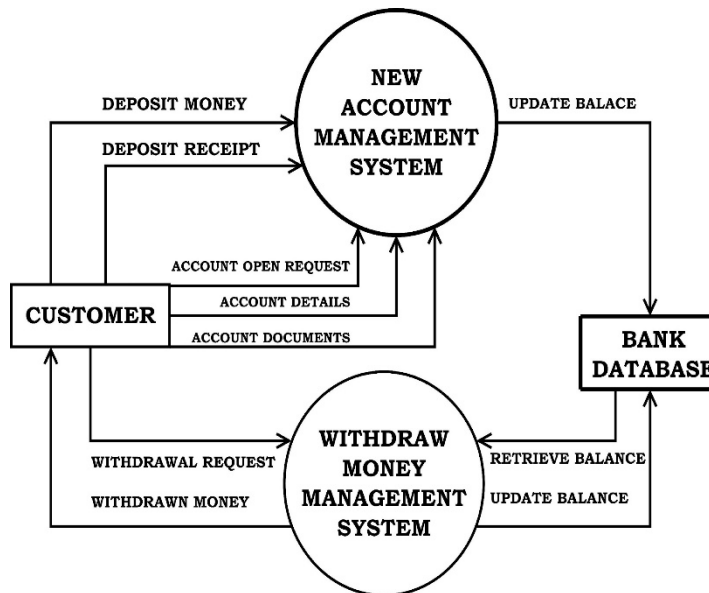
The DFD for Customer Management System is shown in Figure 2.18.



**Fig. 2.18 DFD of Customer Management System**

**DFD for Transaction System**

The DFD for Transaction System is shown in Figure 2.19.



**Fig. 2.19 DFD for Transaction System**

**DFD for Loan System**

The DFD for Loan System is shown in Figure 2.20.



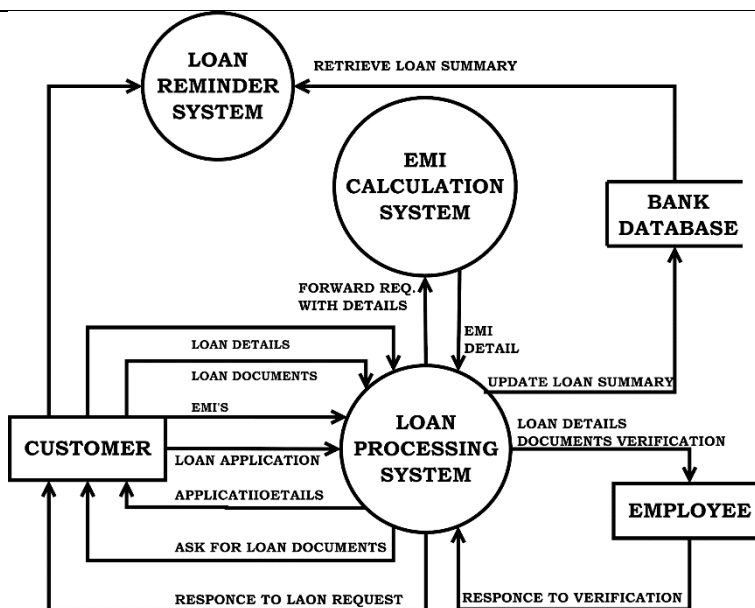


Fig. 2.20 DFD for Loan System

**Data Dictionary**

The data dictionary, or Metadata repository, is a “centralized repository of information about data such as meaning, relationships to other data, origin, usage, and format.”

The term may have one of several closely related meanings pertaining to databases and database management systems (DBMS). A document describing a database or collection of databases; an integral component of DBMS is required to determine its structure; and a piece of middleware that extends the native data dictionary of DBMS.

The data dictionary for this system can be represented as shown in Table 2.1.

Table 2.1: Data dictionary for the banking system

| Data              | Description                                                                          |
|-------------------|--------------------------------------------------------------------------------------|
| Documents         | Account documents   Loan documents                                                   |
| Details           | Account details   Loan details                                                       |
| Account documents | ID Proof + Residence Proof + Date of Birth Proof                                     |
| Loan documents    | ID Proof + Residence Proof + Birth Proof + Salary slip + PAN Card + Loan application |
| Account details   | Name + Address + Phone Number + Account number + Signature                           |
| Loan details      | Loan time period + Loan amount + Loan type                                           |
| Name              | First name + Middle name + Last name                                                 |
| Address           | House No. + Village name + State name + PIN code                                     |
| Phone number      | Digit+Digit+Digit+Digit+Digit+Digit+Digit+Digit+Digit+Digit                          |
| Account number    | Digit + Digit + Digit + Digit + Digit + Digit                                        |
| Deposit Receipt   | Date of deposit + Amount deposited                                                   |
| Response          | Approval   Rejection                                                                 |
| Loan Reminder     | Reminder message + Last date to pay EMI                                              |

**Cost Estimations**

Let us estimate the cost for this system by using Functional Point Estimation. The Table 2.2 shows the grade value points for various parameters.

Table 2.2 Grade value points for various parameters.

| SN | Questions                                             | Grade value |
|----|-------------------------------------------------------|-------------|
| 1  | Does the system require reliable backup and recovery? | 5           |

|    |                                                                                                            |   |
|----|------------------------------------------------------------------------------------------------------------|---|
| 2  | Are specialized data communications required to transfer information to or from the application?           | 3 |
| 3  | Are there distributed processing functions?                                                                | 3 |
| 4  | Is performance critical?                                                                                   | 0 |
| 5  | Will the system run in an existing, heavily utilized operational environment?                              | 5 |
| 6  | Does the system require on-line data entry?                                                                | 5 |
| 7  | Does the on-line data entry require the input transaction to be built over multiple screens or operations? | 5 |
| 8  | Is the data updated online?                                                                                | 5 |
| 9  | Are the inputs, outputs, files, or inquiries complex?                                                      | 2 |
| 10 | Is the internal processing complex?                                                                        | 2 |
| 11 | Is the code designed to be reusable?                                                                       | 3 |
| 12 | Are conversions and installations included in the design?                                                  | 0 |
| 13 | Is the system designed for multiple installations in different organizations?                              | 5 |
| 14 | Is the application designed to facilitate change and for ease of use by user?                              | 5 |

**Value Adjustment Factors,  $\Sigma f_i = 48$**

Table 2.3 Value Adjustment Factors

| Information Domain Value | Est. Count | Weighing Factor | Weighing Count |
|--------------------------|------------|-----------------|----------------|
| External Input           | 4          | 4               | 16             |
| External Output          | 4          | 5               | 20             |
| External Inquiries       | 1          | 4               | 4              |
| Number Of Logical Files  | 1          | 10              | 10             |
| External Interface Files | 0          | 7               | 0              |
| <b>Total</b>             |            |                 | 50             |

**Computing Function Points:**

$$FP = \text{COUNT TOTAL} * (0.65 + 0.01 * \Sigma f_i)$$

$$= 50 * (0.65 + 0.01 * 48)$$

$$= 56.50$$

**Efforts**

The average productivity for this kind of system = 6.5 FP/pm

Considering the labour rate = 8000

Cost per FP = 230

Total efforts = FP (calculated) / average productivity

$$= 56.50 / 6.5$$

$$= 8.69 \text{ person months}$$

Total cost for the project = Total efforts \* labour rate

$$= 8.69 * 8000$$

$$= 69,520$$

**3.2 Risk Table**

The Risk table for the banking system can be shown in Table 2.4.

Table 2.4 Risk table for the banking system

| Risks                     | Category | Probability | Impact | Mitigation                                                                |
|---------------------------|----------|-------------|--------|---------------------------------------------------------------------------|
| Quality not maintained    | DE       | 60%         | 3      | Take up steps to maintain quality at each stage of development.           |
| Size estimates may be low | PS       | 50%         | 2      | Past experiences must be considered and a similar task on a smaller scale |

|                                                     |    |     |   |                                                                                                                                        |
|-----------------------------------------------------|----|-----|---|----------------------------------------------------------------------------------------------------------------------------------------|
|                                                     |    |     |   | may be attempted                                                                                                                       |
| Requirements not properly documented and understood | CU | 50% | 1 | Regular interaction with the customer and getting the requirements verified before finalising them                                     |
| Delivery deadline will be tightened                 | BU | 40% | 2 | Review the progress from time to time and take appropriate steps to keep up with the schedule                                          |
| Lack of skill                                       | ST | 40% | 2 | External resources might help                                                                                                          |
| Building the wrong product                          | CU | 20% | 1 | Early and continuous validation is critically important. You need to establish a clear vision and solid justification for the product. |

Impact levels used in the above table are 1 for Catastrophic, 2 for Critical, 3 Marginal, 4 Negligible. Acronyms used for the Categories are PD for Process Definition, ST for Staff-size and experience, DE for Development Environment, CU for Customer Characteristics, BU for Business Impact, and PS for Product Size.

**Timeline**

Timeline for the Banking system project is shown in Table 2.5.

Table 2.5: Timeline for the Banking system project

| Timeline in weeks                    | WK 1 | WK 2 | WK 3 | WK 4 | WK 5 | WK 6 | WK 7 | WK 8 | WK 9 | WK 10 | WK 11 | WK 12 |
|--------------------------------------|------|------|------|------|------|------|------|------|------|-------|-------|-------|
| 1. Identify Customer Requirements    |      |      |      |      |      |      |      |      |      |       |       |       |
| Meet with customers                  |      |      |      |      |      |      |      |      |      |       |       |       |
| Identify needs and constraints       |      |      |      |      |      |      |      |      |      |       |       |       |
| Establish Problem Statement          |      |      |      |      |      |      |      |      |      |       |       |       |
| MILESTONE: Problem statement defined |      |      |      |      |      |      |      |      |      |       |       |       |
| 2. Define function behaviour         |      |      |      |      |      |      |      |      |      |       |       |       |
| Identify modules                     |      |      |      |      |      |      |      |      |      |       |       |       |
| Define DFDs                          |      |      |      |      |      |      |      |      |      |       |       |       |
| Develop ERDs                         |      |      |      |      |      |      |      |      |      |       |       |       |
| MILESTONE: System Functions defined  |      |      |      |      |      |      |      |      |      |       |       |       |
| 3. Estimation                        |      |      |      |      |      |      |      |      |      |       |       |       |
| Function Point                       |      |      |      |      |      |      |      |      |      |       |       |       |

|                                    |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|------------------------------------|--|--|--|--|--|--|--|--|--|--|--|--|--|--|
| Estimation                         |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Efforts Calculation                |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| MILESTONE: Cost Estimated          |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4. Design Development              |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Formulate System architecture      |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Generate Code                      |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| MILESTONE: System Design developed |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5. Testing                         |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Develop test cases                 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Calculate cyclomatic complexity    |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Develop flow graph                 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| MILESTONE: Testing Complete        |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Architectural Design for the banking system is shown in Figure 2.21.

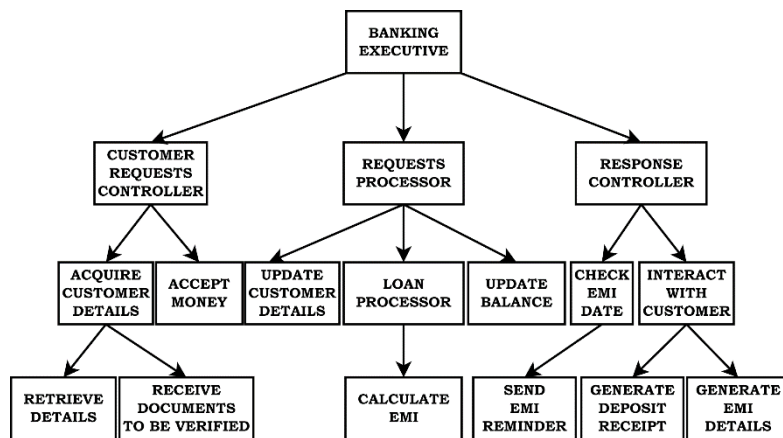


Fig. 2.21 Architectural Design for the banking system

**Data Design**

Data Design for the banking system is shown below.

**OPEN ACCOUNT DETAILS**

|                |             |
|----------------|-------------|
| FIRST NAME     | VARCHAR(15) |
| MIDDLE NAME    | VARCHAR(15) |
| LAST NAME      | VARCHAR(15) |
| HOUSE NUMBER   | VARCHAR(4)  |
| VILLAGE        | VARCHAR(15) |
| STATE          | VARCHAR(15) |
| PIN            | VARCHAR(10) |
| PHONE NUMBER   | VARCHAR(10) |
| INITIAL AMOUNT | INT         |

**UPDATE ACCOUNT DETAILS**

|                |             |
|----------------|-------------|
| FIRST NAME     | VARCHAR(15) |
| MIDDLE NAME    | VARCHAR(15) |
| LAST NAME      | VARCHAR(15) |
| HOUSE NUMBER   | VARCHAR(4)  |
| VILLAGE        | VARCHAR(15) |
| STATE          | VARCHAR(15) |
| PIN            | VARCHAR(10) |
| PHONE NUMBER   | VARCHAR(10) |
| ACCOUNT NUMBER | INT         |

**LOAN DETAILS**

|                |             |
|----------------|-------------|
| TYPE           | VARCHAR(30) |
| AMOUNT         | INT         |
| DURATION       | INT         |
| ACCOUNT NUMBER | INT         |

**DEPOSIT RECEIPT**

|                |      |
|----------------|------|
| ACCOUNT NUMBER | INT  |
| AMOUNT         | INT  |
| DATE           | DATE |

**TESTING****Test Cases**

| S<br>n | Test<br>Case<br>Suit | Descriptio<br>n                                                         | INPUT       |                                                          |            |                 | Expecte<br>d Error                 | Remar<br>k |
|--------|----------------------|-------------------------------------------------------------------------|-------------|----------------------------------------------------------|------------|-----------------|------------------------------------|------------|
|        |                      |                                                                         | Name        | Address                                                  | Phone No.  | Accoun<br>t No. |                                    |            |
| 1      | Open                 | Check the details provided by the customer while opening a new account. | Mayank      | 76 Mangkapuri, Delhi - 110058                            | 9575364500 | ....            | ....                               | PASS       |
| 2      | Open                 |                                                                         |             | 97 Karnal, Punjab - 11542                                | 66973612   | ....            | Name Field can not be empty.       | FAIL       |
| 3      | Open                 |                                                                         | Nitin       | 100 Kapashera, Delhi                                     | 76173361   | ....            | PIN not provided in address field. | FAIL       |
| 4      | Open                 |                                                                         | Nikhil Rana | 420 Thane, Mumbai - 54212                                | 7899159745 | ....            | Wrong Phone number.                | FAIL       |
| 5      | Open                 |                                                                         | Pankaj      | Bijwasan, Road From Bhabbar Chowk to Palam, Delhi 110061 | 95764500   | ....            | Address too long.                  | FAIL       |
| 6      | Update               | Checks the details                                                      | Rani Kumari | B220 AnandVihar,                                         | 78991597   | 9875354         | ....                               | PASS       |

|    |        |                                                  |               |                            |            |         |                                   |      |
|----|--------|--------------------------------------------------|---------------|----------------------------|------------|---------|-----------------------------------|------|
|    |        | provided by                                      |               | Delhi 1165                 |            |         |                                   |      |
| 7  | Update | the customer while updating an existing account. | Kirti         | 65 Mahipalpur Delhi 110037 | 7619873361 | AHGSJK  | No such account exists.           | FAIL |
| 8  | Update |                                                  | Ramn Sharma   | Rangpuri Delhi 110041      |            | 3654895 | Phone no. field can not be empty. | FAIL |
| 9  | Update |                                                  | Riya          |                            | 9919078531 | 244261  | Address field can not be empty.   | FAIL |
| 10 | Update |                                                  | Nikita Bansal | 420 Thane, Mumbai-54212    | 9575364500 | ....    | Account number not provided.      | FAIL |

**Test case Suits**

**Open** – For new account open requests.

**Update** – For update account requests.

**Assignment**

Develop the small project on the following software systems, by using Python Programming Language.

1. Library Management System
1. Petrol Pump Management System
2. Departmental Store
3. Hospital Management System
4. School Management System

**Module 4****Emerging Trends and Social Impact****Module Overview**

Many new technologies are introduced almost every day. Some of these new technologies prosper and persist over time, gaining attention from users. Emerging trends in information technology is the primary catalyst for change to survive in the competitive market. Technological innovations brings the progress in the corporate industry. The competition requires to stay with technologies and pursue digital transformation. It becomes important to incorporate the new software or hardware technology quickly. IT professional are constantly learning, unlearning, and relearning. Adapting new trends is necessary to deliver quality services, reduce spending, and boost user experience. It is a long-term strategy that asks for time, effort, and expertise. However, it is better to learn about emerging trends in information technology to understand the business needs.

The increasing use of '*Digital Technologies*', have made the significant impact on our lives, making things more convenient, faster, and easier to handle. Applications of digital technologies have redefined and evolved all spheres of human activities to the common man. These technologies can also be misused. Best practices can ensure a productive and safe digital environment.

Social impact is the positive change that individuals, organizations, and movements create for society and the environment. Social impact reporting is the process of measuring and communicating the social and environmental effects of an organization's activities. As the world faces complex and urgent challenges, social impact becomes more important than ever. It is a key tool for accountability, transparency, and learning in the social sector.

In this unit, you will learn about some of the emerging trends and innovations in social impact to keep up with the emerging trends and follow the best practices.

**Learning Outcomes**

After completing this module, you will be able to:

- Describe Emerging Trends and New Technologies
- Describe the Societal Impact of Technology

**Module Structure**

Session 1: Emerging Trends and Technologies

Session 2: Societal Impact

**Session 1: Emerging Trends and Technologies**

Sajid was visiting Mumbai for first time. When he was standing on road, he got surprised to see that there was a car moving on road without any driver. Later on, he understands that using the new technology of Artificial Intelligence (AI), it is possible to drive a car automatically without a driver. (Figure 1.1) In AI technology there is no need of driver to drive a car.



**Fig. 1.1 Driver-less car**

### **1.1 New Technology Trends and Emerging Technologies**

Many new technologies are introduced almost every day. Some of these new technologies prosper and persist over time, gaining attention from users. Emerging technologies are the state-of-the-art technologies, which gain popularity and set a new trend among users. IT professional will constantly be learning, unlearning, and relearning. In this chapter, you will learn the top emerging technology trends that will make a huge impact on digital economy and interaction in digital societies. It will possibly secure one of the jobs that will be created by these new technology trends.

### **1.2 Artificial Intelligence (AI)**

Have you ever wondered how maps in your smartphone are able to guide you to take the fastest route to your destination by analyzing real time data, such as traffic congestion? On uploading a photo on a social networking site, has it ever happened that your friends in the photograph were recognized and tagged automatically? These are some of the examples of application of Artificial Intelligence. AI is already known for image and speech recognition, navigation apps, smartphone personal assistants, ride-sharing apps and so much more.

The intelligent digital personal assistants like Siri, Google Now, Cortana, Alexa are all powered by AI. Artificial Intelligence endeavors to simulate the natural intelligence of human beings into machines, thus making them behave intelligently. An intelligent machine is supposed to imitate some of the cognitive functions of humans like learning, decision making and problem solving. In order to make machines perform tasks with minimum human intervention, they are programmed to create a knowledge base and make decisions based on it. AI system can also learn from past experiences or outcomes to make new decisions. With AI spreading its wings across sectors, new jobs will be created in development, programming, testing, support and maintenance.

In simple words, Artificial Intelligence is the ability of computers to perform tasks which are commonly performed by human beings such as writing, driving, and so on. Artificial intelligence as a field is concerned with building systems which are capable of human-level thinking.

#### **Advantages of AI**

Now people have started using AI at home and at industry as well. Many software and other industries are using AI based algorithms. In Google Assistant, Apple's *Siri* and Amazon *Alexa*, AI is used.

There are certain advantages of using AI. These are,

1. As humans require break, AI based machines can work 24x7 without break.
2. AI based machines are the digital assistance by which our performance gets enhanced.
3. Repetitive jobs can be easily handled with AI technology.
4. Errors are reduced when job is performed by using AI based machines.



5. Humans cannot work in hazardous environment, where AI based machines can work.

### **Disadvantages of AI**

There are certain disadvantages of AI.

1. Machine cost increases with the use of AI technology.
2. AI is under research. Still It is not possible to replace human by AI based machine.
3. AI is not creative like humans. It cannot create any original thing to some extent.

### **1.3 Fields of AI**

AI is a vast topic constituting many fields such as knowledge representation, planning, learning, natural language processing, reasoning and perception. Machine learning, neural network, deep learning, Cognitive computing, Computer vision and Natural language processing are the areas that widely uses artificial intelligence. Some of these emerging fields of AI are discussed in this section.

#### **1.3.1 Machine Learning**

Machine Learning is a subsystem of Artificial Intelligence, wherein computers have the ability to learn from data using statistical techniques, without being explicitly programmed by a human being. It comprises algorithms that use data to learn on their own and make predictions. These algorithms, called models, are first trained and tested using a training data and testing data, respectively. After successive trainings, once these models are able to give results to an acceptable level of accuracy, they are used to make predictions about new and unknown data.

Machine learning is an application area of AI. In machine learning, algorithms are developed such that the computing machine, can learn from experience or data. Machine learning is widely used in image processing, medical diagnosis, prediction and classification.

A neural network is a network of artificial neurons that recognizes relationship between set of data in similar way to that of human brain. Such network can be trained and it generates best possible result without any redesign. Neural Network algorithms can be used in rules forecasting, risk management and data validation.

Deep learning is a part of machine learning. Like human beings learn naturally, deep learning algorithms teaches computers. This technology is used for “*driver-less cars*”.

#### **1.3.2 Computer Vision**

Computer vision is a technology in which computers are enabled to see, identify and process images in the same way as humans. It is the subset of AI which makes use of statistical models to aid computer systems in understanding and interpreting visual information in the environment. This technology is used responsible for creating efficient self-driving cars, drones, medical diagnosis and monitoring health of crops.

#### **1.3.3 Expert Systems**

Expert Systems are perhaps the most rigid subset of AI due to their use of rules. This area involves the use of explicitly stated rules and knowledge bases in an attempt to imitate the decision-making of an expert in a certain field. In other words, it is the use of explicitly stated rules and inference techniques to make informed decisions in specific fields, such as medicine.

#### **1.3.4 Robotics**

Robotics is essentially the integration of all the above-mentioned concepts. It is the sub-field responsible for making AI systems perceive, process, and act in the physical world. Robotics involves using algorithms which can recognize objects in their immediate environment and interpret how interactions with these objects can alter their current state and that of the environment plus the people in it. Robots are used in fields such as medicine, manufacturing, e-commerce (warehouses), and many more.

#### **1.3.5 Natural Language Processing (NLP)**

Natural Language Processing is the subset of AI which is responsible for enabling AI systems to interact using Natural Human Language such as English and Hindi. The predictive typing feature of search engine that helps us by suggesting the next word in the sentence while typing keywords and the spell checking features are examples of Natural Language Processing (NLP). NLP involves using statistical models to understand, interpret, and generate human language in a way that is meaningful to human beings. Human language includes all languages spoken by humans. It is the technology behind chatbots like ChatGPT, Siri, Alexa, and others.

In fact it is possible to search the web or operate or control the devices using voice. NLP system can perform text-to-speech and speech-to-text conversion as depicted in Figure 1.2. Machine translation is a rapidly emerging field where machines are able to translate texts from one language to another with fair amount of correctness. Another emerging application area is automated customer service where a computer software can interact with customers to serve their queries or complaints.



**Fig 1.2: Use of natural language processing**

Computers cannot understand human languages such as English, Hindi, Marathi and other regional languages. Natural Language Processing (NLP) is a technology in which computers are enabled to understand human languages. NLP is used in machine translation, speech processing and automatic summarization.

### **1.3.6 Generative-AI**

Generative AI, has revolutionized various industries by enabling machines to create content that resembles human-generated work. It encompasses a wide range of applications, from text generation to image synthesis and even music composition. After mastering generative AI, individuals can pursue exciting job roles in fields such as artificial intelligence research, data science, and creative industries. The ever-expanding applications of generative AI promise a bright future, offering opportunities to interact and create content in the digital age.

### **1.3.7 ChatGPT**

ChatGPT is an artificial intelligence chatbot developed by OpenAI and launched in November 2022. A chatbot is a software application used to conduct an online chat conversation via text or text-to-speech. GPT is a Generative Pre-trained Transformer. It is an autoregressive language model released in 2020 that uses deep learning to produce human-like text.

#### **Capabilities of ChatGPT**

1. Remembers what user said earlier in the conversation.
1. Allows user to provide follow-up corrections.
2. Trained to decline inappropriate requests
3. It can write and debug computer programs, compose music, teleplays, fairy tales, and student essays
4. It can answer test questions, write poetry and song lyrics

5. It can emulate a Linux system; simulate an entire chat room; play games like tic-tac-toe; and simulate an ATM

ChatGPT's training data includes man pages and information about internet phenomena and programming languages, such as bulletin board systems and the Python programming language.

By using ChatGPT we can perform various activities such as topic searching, poem or essay creation and help for programming. The steps to use ChatGPT are,

Step 1. Google ChatGPT and click on OpenAI link.


Step 2. Login with your email ID


Step 3. Click on + to start a new chat.

Step 4. Write your question.

Following are the some of the activities that can be performed by using ChatGPT.

### 1. Topic Searching on ChatGPT

 Explain quantum computing in simple terms


 Quantum computing is a type of computing that uses the principles of quantum mechanics, which is a branch of physics that studies the behavior of matter and energy at a very small scale.


In traditional computing, information is represented using bits, which are either 0 or 1. In quantum computing, information is represented using quantum bits, or qubits, which can be both 0 and 1 at the same time. This is called superposition.

This allows quantum computers to perform certain types of calculations much faster than traditional computers, such as factoring large numbers or searching large databases. Quantum computers can also perform multiple calculations at the same time, which is called parallelism.

However, quantum computing is still in the early stages of development and is not yet

### 2. Poem Creation

 poem on friend

 A friend is someone who's always there,  
To lift you up and show they care.  
They make you laugh, they wipe your tears,  
And help you overcome your fears.

A friend will listen without a word,  
And give you comfort that's assured.  
They're a shining light in darkest times,  
And help you see life's brighter signs.

A friend is there through thick and thin,  
A lifelong bond that can begin.  
With a smile, a hug, a simple nod,  
They can turn your day from bad to odd.

### 3. Help For Programming

C program on factorial of number

Sure, here's a C program to calculate the factorial of a given number:

```

c
Copy code

#include <stdio.h>

int main() {
    int num, i;
    unsigned long long fact = 1;

    printf("Enter a number: ");
    scanf("%d", &num);

    // calculate factorial
    for (i = 1; i <= num; ++i) {
        fact *= i;
    }

    printf("Factorial of %d = %llu", num, fact);
    return 0;
}

```

### Limitations of ChatGPT

1. May occasionally generate incorrect information
1. May occasionally produce harmful instructions or biased content
2. Limited knowledge of world and events after 2021
3. Sometimes cannot draw or show photo as it is a linguistic model

It is not sure that ChatGPT will give correct answer every time, so one needs to recheck it. Do not blindly follow answers given by ChatGPT.

Demonstrate the activity based on use of Google Assistant, Siri and Alexa.

### 1.4 Immersive Experiences

With the three-dimensional (3D) videography, the joy of watching movies in theatres has reached to a new level. Video games are also being developed to provide immersive experiences to the player. Immersive experiences allow us to visualise, feel and react by stimulating our senses. It enhances our interaction and involvement, making them more realistic and engaging. Immersive experiences have been used in the field of training, such as driving simulators as shown in Figure 1.3, flight simulator and so on. Immersive experience can be achieved using virtual reality and augmented reality.



**Fig. 1.3: Driving Simulator**

**(a) Virtual Reality** – Everything that we experience in our reality is perceived through our senses. From this came the idea that if we can present our senses with made-up or non-real information, our perception of reality would also alter in response to that. Virtual Reality (VR) is a three-dimensional, computer-generated situation that simulates the real world. The user can interact with and explore that environment by getting immersed in it while interacting with the

objects and other actions of the user. At present, it is achieved with the help of VR Headsets. In order to make the experience of VR more realistic, it promotes other sensory information like sound, smell, motion, and temperature. It is a comparatively new field and has found its applications in gaming (Figure 1.4), military training, medical procedures, entertainment, social science and psychology, engineering and other areas where simulation is needed for a better understanding and learning.



**Fig. 1.4 : VR Headset**

**(b) Augmented Reality (AR)** – AR is an enhanced version of the real physical world that is achieved through the use of digital visual elements, sound, or other sensory stimuli and delivered via technology. It adds components of the digital world to the physical world, along with the associated tactile and other sensory requirements, thereby making the environment interactive and digitally manipulable. Users can access information about the nearest places with reference to their current location. They can get information about places and choose on the basis of user reviews. With the help of location-based AR App, travellers can access real-time information of historical places just by pointing their camera viewfinder to subjects as depicted in Figure 1.5. Location-based AR apps are major forms of AR apps.



**Fig. 1.5: Location based Augmented Reality**

### **1.5 Internet of Things (IoT)**

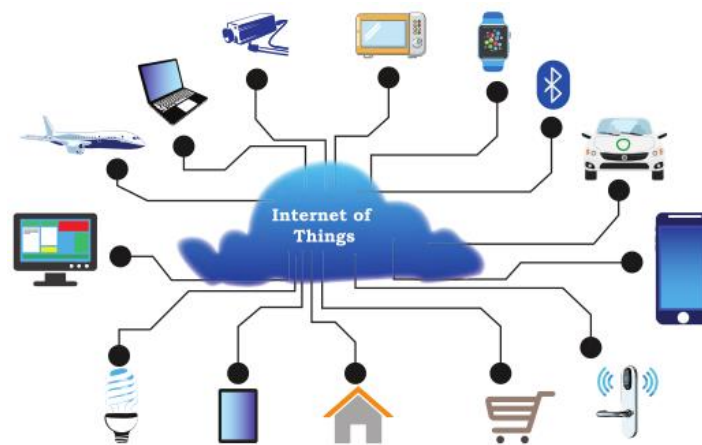
Today many “things” are easily connected to the Internet—and to each other with WiFi connectivity. Hence, the Internet of Things, or IoT is the future, and has already enabled devices, home appliances, fan, refrigerator, air conditioner, cars and much more to be connected to and exchange data over the Internet.

The network of interconnected objects that are able to collect and exchange data is called as Internet of Things (IoT). In IoT, there is a system of interconnected computing devices, mechanical and electronic devices, objects, animals and peoples. Each one is provided Unique Identifiers (UIDs). They have ability to collect and to transfer data over a network without human interaction.

The 'Internet of Things' is a network of devices that have an embedded hardware and software to communicate (connect and exchange data) with other devices on the same network as shown in Figure 1.6. In IoT, our mobile phones, computers, car, TVs, Fridges, Radios, Watches, Tablets can all be connected together. All these devices can talk to each other i.e. they can exchange data.

The IoT can enable better safety, efficiency and decision making for businesses as data is collected and analyzed. It can enable predictive maintenance, speed up medical care, improve customer service, and offer benefits.

It is the beginning of this new technology. Forecasts suggest that by 2030 around 50 billion of these IoT devices will be in use around the world. There will be a massive web of interconnected devices, everything from smartphones to kitchen appliances. You will have to learn about Information security, AI and machine learning fundamentals, networking, hardware interfacing, data analytics, automation, understanding of embedded systems, and must have device and design knowledge.



**Fig. 1.6: Internet of Things (IoT)**

### 1.5.1 Web of Things (WoT)

Internet of Things allows us to interact with different devices through Internet with the help of smartphones or computers, thus creating a personal network. But to interact with number of different devices, we need to install different apps. The web is already being used as a system to communicate with each other. Web of Things (WoT) allows the use of web services to connect anything in the physical world, besides human identities on web. It will pave way for creating smart homes, smart offices, smart cities and so on.

### 1.5.2 Sensors

What happens when you hold your mobile vertically or horizontally? The display also changes to vertical or horizontal with respect to the way we hold our mobile. This is possible with the help of two sensors, namely accelerometer and gyroscope (gyro). The accelerometer sensor in the mobile phones detects the orientation of the phone. The gyroscope sensors tracks rotation or twist of your hand and add to the information supplied by the accelerometer. Sensors are very commonly used for monitoring and observing elements in real world applications. The evolution of smart electronic sensors is contributing in a large way to the evolution of IoT. It will lead to creation of new sensor-based, intelligent systems. A smart sensor is a device that takes input from the physical environment and uses built-in computing resources to perform predefined functions upon detection of specific input and then process data before passing it on.

### 1.5.3 Smart Cities

With rapid urbanization, the load on our cities is increasing day-by-day There are challenges in management of resources like land water, waste, air pollution, health and sanitation, traffic congestions, public safety and security. These challenges are forcing many city planners to the

idea of a smart city as shown in Figure 1.7. It makes use of IoT and WoT to manage and distribute resources efficiently. The smart building shown here uses sensors to detect earthquake and then warn nearby buildings so that they can prepare themselves accordingly. The smart bridge uses wireless sensors to detect any loose bolt, cable or crack. It alerts concerned authorities through SMS. The smart tunnel also uses wireless sensors to detect any leakage or congestion in the tunnel. This information can be sent as wireless signals across the network of sensor nodes to a centralized computer for further analysis.



**Fig. 1.7 Smart City**

### Advantages of IOT

There are several advantages of IOT as given below:

1. *Efficient resource utilization:* All natural and artificial resources are utilized efficiently once we know how it is to be used.
2. *Reduction in Human Effort:* IOT works without any human involvement. So, human efforts are saved.
3. *Saves Time:* IOT is an automated technology. It saves human time to perform task.
4. *Rich Data Collection:* A lot of data is generated in IOT. Hence data collection is enhanced.
5. *Improved security:* IOT system is secure and efficient.

### Disadvantages of IOT

There are significant disadvantages or challenges faced IOT as it is a new technology getting evolved over a time. Following are disadvantages of IOT.

1. *IOT is a Complex System:* In IOT, a large variety of devices are connected together. So it is very complex system to design, develop and implement. It is also difficult to maintain.
2. *No Privacy in IOT:* In IOT, every user data is collected without its active participation. So, there is no privacy to any user. A personal data may be available over the network.
3. *Threats to security:* In IOT, there are always chances that hackers may attack on the system. In spite of control measures, there are chances of data leakages or braches of security to the network.

### Applications of IOT

There are various applications of IOT such as,

1. *Smart Home:* An automatic Home tent act as per user requirements.
2. *Smart Wearable:* Such as smart clothing smart watches and smart shoes.
3. *Smart Cities:* Traffic management and water distribution is smartly controlled.
4. *Smart Grids:* Efficiently of electricity grids will be improved.
5. *Industrial Internet:* All industries will be connected to each other to understand their requirements.

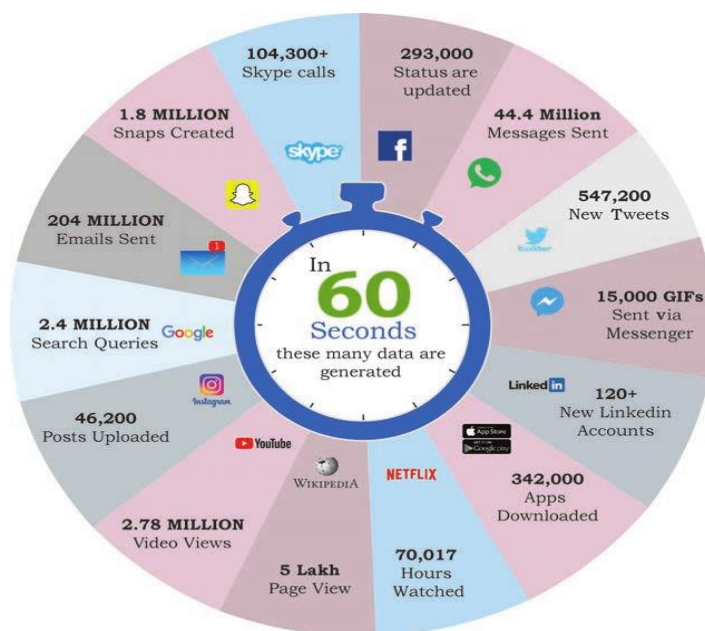
6. Connected Cars: All cars in the city will be connected to each other to avoid accidents.
7. Digital Health Service: All health services will be in electronic form.
8. Smart Retail: Retail will be automated as per requirement of the user.

### Practical Activity

1. Explore and list a few IoT devices available in the market.
2. Demonstrate the controlling AC through mobile phones
3. Demonstrate the lighting system using IoT

### 1.6 Big Data

With technology making an inroad into almost every sphere of our lives, data is being produced at a colossal rate. Today, there are over a billion Internet users, and a majority of the world's web traffic is coming from smartphones. Figure 1.8 shows that at the current pace, around 2.5 quintillion bytes of data are created each day, and the pace is increasing with the continuous evolution of the Internet of Things (IoT). This results in the generation of data sets of enormous volume and complexity called Big Data. Such data cannot be processed and analyzed using traditional data processing tools as the data is not only voluminous, but also unstructured like our posts, instant messages and chats, photographs that we share through various sites, our tweets, blog articles, news items, opinion polls and their comments, audio/video chats, etc. Big data not only represents voluminous data, it also involves various challenges like integration, storage, analysis, searching, processing, transfer, querying and visualization of such data. Big data sometimes hold rich information and knowledge which is of high business value, and therefore there is a keen effort in developing software and methods to process and analyze big data.



**Fig. 1.8 Sources of big data (numbers are approximate)**

#### 1.6.1 Characteristics of Big Data

Big data exhibits following five characteristics that distinguish it from traditional data.

**(a) Volume** – The most prominent characteristic of big data is its enormous size. If a particular data set is of such large size that it is difficult to process it with traditional DBMS tools, it can be termed as big data.

**(b) Velocity** – It represents the rate at which the data under consideration is being generated and stored. Big data has an exponentially higher rate of generation than traditional data sets.



**(c) Variety** – It asserts that a data set has varied data, such as structured, semi-structured and unstructured data. Some examples are text, images, videos, web pages and so on.

**(d) Veracity** – Big data can be sometimes inconsistent, biased, and noisy or there can be abnormality in the data or issues with the data collection methods. Veracity refers to the trustworthiness of the data because processing such incorrect data can give wrong results or mislead the interpretations.

**(e) Value** – Big data is not only just a big pile of data, but also possess to have hidden patterns and useful knowledge which can be of high business value. But as there is cost of investment of resources in processing big data, we should make a preliminary enquiry to see the potential of the big data in terms of value discovery or else our efforts could be in vain.

### 1.6.2 Data Analytics

Data analytics is the process of examining data sets to draw conclusions about the information they contain, with the aid of specialized systems and software. Data analytics technologies and techniques are becoming popular day-by-day. They are used in commercial industries to enable organizations to make more informed business decisions. In the field of science and technology, it can be useful for researchers to verify or disprove scientific models, theories and hypotheses. Pandas is a library of the programming language Python that can be used as a tool to make data analysis much simpler.

There are various data analysis tools such as Microsoft Excel, Python, R, Jupyter Notebook, Apache Spark, SAS, Microsoft Power BI, Tableau and KNIME.

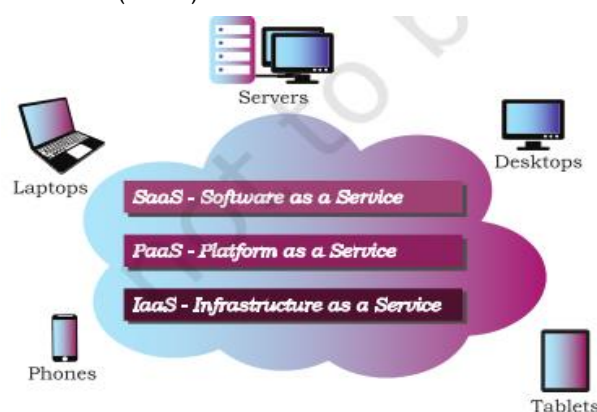
### 1.7 Cloud Computing

Cloud computing refers to the availability of large computing and storage facility delivered over the Internet or the cloud. It does not require active management by the user. Data centers available on the internet are examples of cloud computing. Google's Gmail service, Facebook and WhatsApp services are examples of cloud computing.

The services comprise software, hardware (servers), databases, and storage. These resources are provided by companies called cloud service providers and usually charge on use basis. We already use cloud services while storing our pictures and files as backup on Internet, or host a website on the Internet. Through cloud computing, a user can run a big application or process a large amount of data without having the required storage or processing power on their personal computer as long as they are connected to the Internet. Besides other numerous features, cloud computing offers cost-effective, on-demand resources. A user can avail need-based resources from the cloud at a very reasonable cost.

#### 1.7.1 Cloud Services

There are three standard models to categorise different computing services delivered through cloud as shown in Figure 1.9. These are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).



**Fig. 1.9: Cloud computing and its services**

**(a) Infrastructure as a Service (IaaS)** – The IaaS providers can offer different kinds of computing infrastructure, such as servers, virtual machines (VM), storage and backup facility, network components, operating systems or any other hardware or software. Using IaaS from the cloud, a user can use the hardware infrastructure located at a remote location to configure, deploy and execute any software application on that cloud infrastructure. They can outsource the hardware and software on demand basis and pay as per the usage. Thus they can save the cost of software, hardware and other infrastructures as well as the cost of setting up, maintenance and security. Some popular IaaS platforms are Amazon EC2, Microsoft Azure, Google cloud platform, GoGrid, and Digital ocean.

**(b) Platform as a Service (PaaS)** – The PaaS provides a platform or environment to develop, test, and deliver software applications. Suppose we have developed a web application using MySQL and Python. To run this application online, you can avail a pre-configured Apache server from cloud having MySQL and Python pre-installed. In PaaS, the user has complete control over the deployed application and its configuration. It provides a deployment environment for developers at a much reduced cost without buying and managing the underlying hardware and software. Google App engine, Microsoft Azure, OpenShift and Oracle cloud are examples of PaaS.

**(c) Software as a Service (SaaS)** – SaaS provides on-demand access to application software, usually requiring a licensing or subscription by the user. SaaS from cloud can be used while using Google doc, Microsoft Office 365, Drop Box to edit a document online. Like PaaS, a user is provided access to the required configuration settings of the application software, that they are using at present. In all of the above standard service models, a user can use on-demand infrastructure or platform or software on charged basis. Microsoft office 360, Google G Suite, Zoho, Salesforce are common examples of SaaS. Government of India has embarked upon an ambitious initiative — ‘GI Cloud’ which has been named as ‘MeghRaj’ (<https://cloud.gov.in>).

#### **Advantages of cloud computing**

Cloud computing is an essential part of modern computing. There are several advantages of cloud computing as mentioned below.

**Cost saving** – Capital cost required for hardware and software can be with maintenance cost.

**Time saving** – Cloud computing offers competitive advantage to save time on installations.

**High speed** – Cloud computing services are deployed quickly and make available all the resources immediately.

**Backup and restore data** – The time consuming process of data back up and restoring can be easily done with cloud services.

**Software and hardware integration** – In cloud services, software and hardware are integrated with each other.

**Reliability** – Cloud services are highly reliable services.

**Mobility** – Employees can work at premises or at remote locations and they can access to cloud services.

**Unlimited storage** – Almost unlimited storage is offered under cloud services with a very nominal fees.

#### **Disadvantages**

Technical issues, downtime, security threat, internet connectivity and lower bandwidth are certain disadvantages of cloud services.

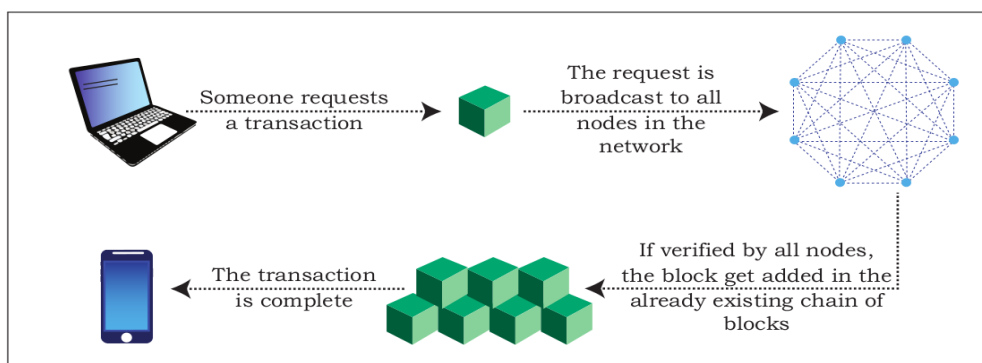
#### **Practical Activity**

Demonstration of uses of Gmail, WhatsApp and Facebook for sharing of photographs and videos of an event.

### **1.8 Blockchain**

Traditionally, we perform digital transactions by storing data in a centralized database and the transactions performed are updated one by one on the database. However, since all the data is stored on a central location, there are chances of data being hacked or lost. The blockchain technology works on the concept of decentralized and shared database where each computer has a copy of the database. A block can be thought as a secured chunk of data or valid transaction. Each block has some data called its header, which is visible to every other node, while only the owner has access to the private data of the block. Such blocks form a chain called blockchain as shown in Figure 1.10. We can define blockchain as a system that allows a group of connected computers to maintain a single updated and secure ledger. Each computer or node that participates in the blockchain receives a full copy of the database. It maintains an ‘*append only*’ open ledger which is updated only after all the nodes within the network authenticate the transaction. Safety and security of the transactions are ensured because all the members in the network keep a copy of the blockchain and so it is not possible for a single member of the network to make changes or alter data.

Several industries are involving and implementing blockchain, there is a demand for skilled professionals. It requires hands-on experience in programming languages, OOPS fundamentals, flat and relational databases, data structures, web app development, and networking.



**Fig. 1.10 Blockchain**

The most popular application of blockchain technology is in digital currency. However, due to its decentralized nature with openness and security, blockchains are being seen as one of the ways to ensure transparency, accountability and efficiency in business as well as in governance systems. For example, in healthcare, better data sharing between healthcare providers would result in a higher probability of accurate diagnosis, more effective treatments, and the overall increased ability of healthcare organizations to deliver cost-effective care. Another potential application can be for land registration records, to avoid various disputes arising out of land ownership claims and encroachments. A blockchain based voting system can solve the problem of vote alterations and other issues. Since everything gets stored in the ledger, voting can become more transparent and authentic. The blockchain technology can be used in diverse sectors, such as banking, media, telecom, travel and hospitality and other areas.

### Practical Activity

Prepare a list of areas where blockchain technology can be used to improve performance. Justify your answer.

### 1.9 Virtual Lab (VL)

Virtual Labs will provide to the students the result of an experiment by one of the following methods or possibly a combination:

1. Modeling the physical phenomenon by a set of equations and carrying out simulations to yield the result of the particular experiment. This can, at-the-best, provide an approximate version of the ‘*real-world*’ experiment
2. Providing measured data for virtual lab experiments corresponding to the data previously obtained by measurements on an actual system

3. Remotely triggering an experiment in an actual lab and providing the student the result of the experiment through the computer interface. This would entail carrying out the actual lab experiment remotely.

### **Need of Virtual Lab**

Physical distances and the lack of resources make us unable to perform experiments, especially when they involve sophisticated instruments. Also, good teachers are always a scarce resource. Web-based and video-based courses address the issue of teaching to some extent. Conducting joint experiments by two participating institutions and also sharing costly resources has always been a challenge.

With the present day internet and computer technologies the above limitations can no more hamper students and researchers in enhancing their skills and knowledge. Also, in our country, costly instruments and equipment need to be shared with fellow researchers to the extent possible. Web enabled experiments can be designed for remote operation and viewing so as to enthuse the curiosity and innovation into students. This would help in learning basic and advanced concepts through remote experimentation.

Internet-based experimentation further permits use of resources – knowledge, software, and data available on the web, apart from encouraging skillful experiments being simultaneously performed at points separated in space and possibly, time.

VL provide remote-access to Labs in various disciplines of Science and Engineering. These Virtual Labs would cater to students at the undergraduate level, post graduate level as well as to research scholars. VL enthuse students to conduct experiments by the arousing their curiosity. This would help them in learning basic and advanced concepts through remote experimentation.

VL provide a complete Learning Management System around the Virtual Labs where the students can avail the various tools for learning, including additional web-resources, video-lectures, animated demonstrations and self-evaluation. VL share costly equipment and resources, which are otherwise available to limited number of users due to constraints on time and geographical distances.

Virtual Labs will be made more effective and realistic by providing additional inputs to the students like accompanying audio and video streaming of an actual lab experiment and equipment. For the *'touch and feel'* part, the students can possibly visit an actual laboratory for a short duration. Virtual labs can be designed for subjects of various disciplines such as:

**Computer Science and Engineering** – ANN, Problem solving, Pattern Recognition.

**Electronics and Communication** – Digital Design, Network Technology, VLSI Design.

**Electrical Engineering** – Electrical machines, Industrial Lab.

**Mechanical Engineering** – Mechanics, Vibrations, Material Response.

**Physical Sciences** – Laser Optics, Modern Physics, Optics.

**Chemical Sciences** – Molecular Spectroscopy, Organic and Inorganic chemistry lab.

### **Practical Activity. 8051 Assembly Language Programme of addition of two numbers using Virtul Lab**

#### **Perform 8051 ALP using VL**

ALP1: Addition of two numbers

ALP:

ORG 0000H ; Set program counter 0000H

MOV A,#50H ; Load the number 50H into A

ADD A,#51H ; Add 51H with contents of A

MOV 52H,A ; Save the least significant byte of the result in location 52H

MOV A, #00 ; Load 00H into A

```

ADDC A, #00 ; Add the immediate data and the contents of carry flag to A
MOV 53H,A ; Save the most significant byte of the result in location 53
END

```

After execution we get [52H]=A1H and [53H]=00H

The screenshot displays a simulation interface for an 8051 microcontroller. It includes a Pin Diagram on the left, a Memory table in the center, a Peripherals diagram on the right, and an Assembly Code Editor on the far right. The memory table shows the following values:

| SFR   | Value | 0x50 | 0x00 |
|-------|-------|------|------|
| OP0.0 |       | 0x51 | 0x00 |
| OP0.1 | A     | 0x52 | 0xa1 |
| OP0.2 | B     | 0x53 | 0x00 |
| OP0.3 | PSW   | 0x54 | 0x00 |
| OP0.4 | P0    | 0x55 | 0x00 |
| OP0.5 | P1    | 0x56 | 0x00 |
| OP0.6 | P2    | 0x57 | 0x00 |
| OP0.7 | P3    | 0x58 | 0x00 |
| OP0.7 | SP    | 0x59 | 0x00 |
| OP0.7 | EA    | 0x5a | 0x00 |
| OP0.7 | DALE  | 0x5b | 0x00 |
| OP0.7 | OPSEN | 0x5c | 0x00 |
| OP2.0 | DPL   | 0x5d | 0x00 |
| OP2.1 | DPH   | 0x5e | 0x00 |
| OP2.2 | PCON  | 0x5f | 0x00 |
| OP2.2 | TCOD  | 0x60 | 0x00 |
| OP2.2 | TLO   | 0x61 | 0x00 |
| OP2.2 | TL1   | 0x62 | 0x00 |
| OP2.2 | TH0   | 0x63 | 0x00 |

The code editor shows the following assembly code:

```

1 ORG 0000H ; Set program counter 00
2 MOV A,#50H ; Load number 50H into
3 ADD A,#53H ; Add 53H with contents
4 MOV 52H,A ; Save the least significant
5 MOV A,#00 ; Load 00H into A
6 ADDC A,#00 ; Add the immediate dat
7 MOV 53H,A ; Save the most significant
8 END

```

## SUMMARY

- Artificial Intelligence (AI) endeavours to simulate the natural intelligence of human beings into machines thus making them intelligent.
- Machine learning comprises algorithms that use data to learn on their own and make predictions.
- Natural Language Processing (NLP) facilitates communicating with intelligent systems using a natural language.
- Virtual Reality (VR) allows a user to look at, explore and interact with the virtual surroundings, just like one can do in the real world.
- The superimposition of computer-generated perceptual information over the existing physical surroundings is called Augmented Reality.
- Big data holds rich information and knowledge which can be of high business value. Five characteristics of big data are: Volume, Velocity, Variety, Veracity and Value.
- Data analytics is the process of examining data sets in order to draw conclusions about the information they contain.
- The Internet of Things (IoT) is a network of devices that have an embedded hardware and software to communicate (connect and exchange data) with other devices on the same network.
- A sensor is a device that takes input from the physical environment and uses built-in computing resources to perform predefined functions upon detection of specific input and then processes data before passing it on.
- Cloud computing allows resources located at remote locations to be made available to anyone anywhere. Cloud services can be Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS).
- Blockchain is a system that allows a group of connected computers to maintain a single updated and secure ledger which is updated only after all the nodes in the network authenticate the transaction.
- Virtual lab allows to perform the physical lab experiments virtually with the help of simulations.

**CHECK YOUR PROGRESS****(A) Multiple choice questions**

1. The intelligent digital personal assistants like Siri, Google Now, Cortana, Alexa are powered by the technology called as (a) IoT (b) AI (c) Big Data (d) Cloud computing .
2. Which of the following is not true for AI based system. (a) AI based systems can work available 24 × 7 without any break. (b) AI based system can think like humans and create original thing. (c) Repetitive jobs can be handled with AI technology. (d) Errors are reduced when job is performed by using machines with AI.
3. Making the things smart is an application of (a) ChatGPT (b) Artificial Intelligence (c) Internet of Things (d) Cloud computing
4. Making computer to understand the human language is an application of (a) ChatGPT (b) Artificial Intelligence (c) Internet of Things (d) Natural Language Processing
5. Without owing the computer hardware and software, it is possible to use these services using the technology called as (a) Internet of Things (b) Cloud computing (c) Artificial Intelligence (d) Big Data
6. Which is not one of the features of IoT devices? (a) Remotely controllable (b) Programmable (c) Can turn themselves off if necessary (d) All of the above

**(B) Fill in the blanks**

1. Artificial Intelligence simulate the \_\_\_\_\_ of human beings into machines.
2. Immersive experiences are used in the field of training, as \_\_\_\_\_
3. In Natural Language Processing (NLP) computers are enabled to understand \_\_\_\_\_.
4. GPT is an \_\_\_\_\_ released that uses deep learning to produce human-like text.
5. The blockchain technology works on the concept of \_\_\_\_\_ and \_\_\_\_\_ database

**(C) State whether True or False**

1. Deep learning technology is used for “driver-less cars”.
2. Virtual Reality is a computer-generated situation that simulates the real world.
3. Internet of Things (IoT) is a secured network where the user’s data is secured.
4. Web of Things (WoT) allows the use of web services to connect anything in the physical world.
5. Virtual lab allows to perform the the physical lab experiment by using the equipment in physical lab.

**(D) Short answer questions**

1. List the various Emerging Technologies in IT.
2. List the various areas that widely uses artificial intelligence.
3. Differentiate between AR and VR.
4. State the limitations of ChatGPT.
5. State the advantages and disadvantages of Internet of Things.
6. State the characteristics of Big Data.
7. List the three different computing servies offered by the cloud.
8. List some of the cloud-based services that you are using at present.
9. What is ChatGPT?
10. How can you perform the lab experiment in virtual lab?

## Session 2: Societal Impact

In recent years, the increasing use of 'Digital Technologies', have made the significant impact on our lives, making things more convenient, faster, and easier to handle. In the past, sending a letter to some one take days to reach. Today, one can send and receive emails to more than one person at a time. The instantaneous nature of electronic communications has made us more efficient and productive. Applications of digital technologies have redefined and evolved all spheres of human activities to the common man. While we reap the benefits of digital technologies, these technologies can also be misused. Let's look at the impact of these technologies on our society and the best practices that can ensure a productive and safe digital environment for us.

### 2.1 DIGITAL FOOTPRINTS

While Internet surfing, some data trail behind which reflects the online activities performed by the user. This is the *digital footprint*. The digital footprint can be created and used with or without our knowledge. It includes websites visited, emails, and any information submitted online, along with the computer's IP address, location, and other device specific details. Such data can be misused or exploited. This awareness make us cautious about what we write, upload or download or even browse online.

There are two kinds of digital footprints – *active* and *passive*. Active digital footprints includes data intentionally submitted online. This may include emails, or responses or posts on different websites or mobile Apps. The digital data trail left online unintentionally is called *passive* digital footprints. This includes the data generated when visited a website, use a mobile App, browse Internet. (Figure 2.1)

Everyone who is connected to the Internet may have a digital footprint. With more usage, the trail grows. On examining the browser settings, we can see, how it stores browsing history, cookies, passwords, auto fills, and many other types of data. Besides browser, most of our digital footprints are stored in servers where the applications are hosted. We may not have access to remove or erase that data. Therefore, once a data trail is generated, even if we later try to erase data about our online activities, the digital footprints still remain. There is no guarantee that digital footprints will be fully eliminated from the Internet. Therefore, be more cautious while being online. All our online activities leave a data trace on the Internet as well as on the computing device that we use. This can be used to trace the user, his/her location, device and other usage details.



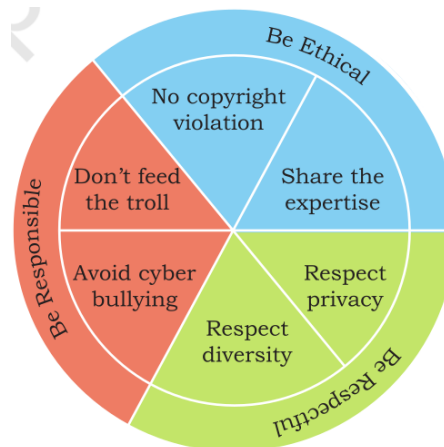
**Fig. 2.1: Exemplar web applications that result in digital footprints**

### 2.2 DIGITAL SOCIETY AND NETIZEN

In this era of digital society, our daily activities like communication, social networking, banking, shopping, entertainment, education, transportation, are increasingly being driven by online transactions. Anyone who uses digital technology along with Internet is a digital citizen or a netizen. A responsible netizen must abide by net etiquettes, communication etiquettes and social media etiquettes.

### 2.2.1 Net Etiquette

It is necessary to to exhibit proper manners and etiquette while being online as shown in Figure 2.2. One should be ethical, respectful and responsible while surfing the Internet.



**Fig. 2.2: Net Etiquette**

#### (A) Be Ethical

*No copyright violation* – do not use copyrighted materials without the permission of the creator or owner. As an ethical digital citizen, be careful while streaming audio or video or downloading images and files.

*Share the expertise* – It is good to share information and knowledge on Internet so that others can access it. The information shared should be true and unambiguous. In order to avoid redundant information, verify that the information is not available already on Internet.

#### (B) Be Respectful

*Respect privacy* – all the digital citizens have the right to privacy and the freedom of personal expression. Our personal communication with a digital citizen may include images, and documents, that are private to both. To respect this privacy, do not share these images and documents, with any other digital citizen without each others' consent.

*Respect diversity* – In a group or public forum, respect the diversity of the people in terms of knowledge, experience, culture and other aspects.

#### (C) Be Responsible

*Avoid Cyber bullying* – any insulting, threats, sexual harassment or comments aimed to publicly ridicule a victim is termed as Cyber bullying. Perhaps things done online have no effect in the real world. Bullying online can have very serious implications on the other person (victim). Also, remember our actions can be traced back using our digital footprints.

*Don't feed the troll* – an Internet troll is a person who deliberately sows discord on the Internet. Since trolls thrive on attention, the best way to discourage trolls is not to pay any attention to their comments.

### 2.2.2 Communication Etiquette

Digital communication includes email, texting, instant messaging, talking on the cell phone, audio or video conferencing, posting on forums, social networking sites. Good communication over email, chat room and other such forums require a digital citizen to abide by the communication etiquette as below.



**(A) Be Precise**

*Respect time* – do not waste precious time in responding to unnecessary emails or comments unless they have some relevance. Also, do not expect an instant response as the recipient may have other priorities.

*Respect data limits* – For concerns related to data and bandwidth, very large attachments may be avoided. Rather send compressed files or link of the files through cloud shared storage like Google Drive, Microsoft One Drive, Yahoo Dropbox.

**(B) Be Polite**

Whether the communication is synchronous (happening in real time like chat, audio/video calls) or asynchronous (like email, forum post or comments), be polite and non-aggressive. Avoid being abusive even if you don't agree with others' point of view.

**(C) Be Credible**

Always be cautious while making a comment, replying or writing an email or forum post as such acts decide your credibility over a period of time. On various discussion forums, we usually try to go through the previous comments of a person and judge their credibility before relying on that person's comments.

**2.2.3 Social Media Etiquette**

In the current digital era, we are familiar with different kinds social media and we may have an account on Facebook, Google+, Twitter, Instagram, Pinterest, or the YouTube channel. These platforms encourage users to share their thoughts and experiences through posts or pictures. In social media too, there are certain etiquette need to be followed are as below.

**Choose password wisely** – It is essential to choose the strong password and to change it frequently to safeguard social media accounts. Never share personal credentials like username and password with others.

**Know who you befriend** – social networks usually encourage connecting with users (making friends). But be careful while befriending unknown people as their intentions possibly could be malicious and unsafe.

**Beware of fake information** – A user should be aware of fake news, messages and posts are common in social networks. The user should apply their knowledge and experience to validate such news, message or post.

**Think before uploading** – it is possible to upload almost anything on social network. However, remember that once uploaded, it is always there in the remote server even after deleting the files. Hence, be cautious while uploading sensitive or confidential files.

**2.3 DATA PROTECTION**

Data or information protection is mainly about the privacy of data stored digitally. The sensitive data such as biometric information, health information, financial information, or other personal documents, images or audios or videos is accessible only to the authorised user. Privacy of sensitive data can be implemented by encryption, authentication, and other secure methods. The data protection policies (laws) provide guidelines to the user on processing, storage and transmission of sensitive information. The motive behind implementation of these policies is to ensure that sensitive information is appropriately protected from modification or disclosure.

**2.3.1 Intellectual Property Right (IPR)**

Intellectual Property refers to the inventions, literary and artistic expressions, designs and symbols, names and logos. The ownership of such concepts lies with the creator, or the holder of the intellectual property. This enables the creator or copyright owner to earn recognition or financial benefit by using their creation or invention. Intellectual Property is legally protected through copyrights, patents, trademarks.

**(A) Copyright**

Copyright grants legal rights to creators for their original works like writing, photograph, audio recordings, video, computer software, and other creative works like literary and artistic work. Copyrights are automatically granted to creators and authors. Copyright law gives the copyright holder a set of rights that they alone can avail legally. The rights include right to copy (reproduce) a work, right to create derivative works based upon it, right to distribute copies of the work to the public, and right to publicly display or perform the work. It prevents others from copying, using or selling the work.

**Executing IPR: say for a software**

- √ Code of the software will be protected by a copyright
- √ Functional expression of the idea will be protected by a patent
- √ The name and logo of the software will come under a registered trademark

**(B) Patent**

A patent is usually granted for inventions. Unlike copyright, the inventor needs to apply (file) for patenting the invention. When a patent is granted, the owner gets an exclusive right to prevent others from using, selling, or distributing the protected invention. Patent gives full control to the patentee to decide whether or how the invention can be used by others. Thus it encourages inventors to share their scientific or technological findings with others. A patent protects an invention for 20 years, after which it can be freely used.

**(C) Trademark**

Trademark includes any visual symbol, word, name, design, slogan, or label that distinguishes the brand or commercial enterprise, from other brands or commercial enterprises. It also prevents others from using a confusingly similar mark, including words or phrases.

**2.3.2 Violation of IPR**

Violation of intellectual property right may happen in one of the following ways:

**(A) Plagiarism**

It is easy to copy or share text, pictures and videos from internet. If we copy some contents from Internet, but do not mention the source or the original creator, then it is considered as an act of plagiarism. It is a serious ethical offense and sometimes considered as an act of fraud. Even for the contents that are open for public use, should cite the author or source to avoid plagiarism.

**(B) Copyright Infringement**

As per the Copyright Act, 1957, the use of a copyrighted work without the permission of the owner results in copyright infringement. Infringement occurs when a third person unintentionally or intentionally uses/copies the work of another without giving credit. It is usually classified into two categories, i.e. primary and secondary infringement. Primary infringement occurs when there is an actual act of copying, while secondary infringement occurs when unauthorised dealings take place, such as selling or importing pirated books.

**(C) Trademark Infringement**

Trademark Infringement means unauthorised use of other's trademark on products and services. An owner of a trademark may commence legal proceedings against someone who infringes its registered trademark.

**2.3.4 Public Access and Open Source Software**

Copyright sometimes put restriction on the usage of the copyrighted works by anyone else. If others are allowed to use and built upon the existing work, it will encourage collaboration and would result in new innovations in the same direction. Licenses provide rules and guidelines for others to use the existing work. When authors share their copyrighted works with others under public license, it allows others to use and even modify the content. Open source licenses help to contribute to existing work or project without seeking special individual permission to do so.

The GNU General public license (GPL) and the Creative Commons (CC) are two popular categories of public licenses. CC is used for all kind of creative works like websites, music, film, and

literature. CC enables the free distribution of an otherwise copyrighted work. It is used when an author wants to give people the right to share, use and build upon a work that they have created. GPL is primarily designed for providing public licence to a software. GNU GPL is another free software license, which provides end users the freedom to run, study, share and modify the software, besides getting regular updates.

Users or companies who distribute GPL license works may charge a fee for copies or give them free of charge. This distinguishes the GPL license from freeware software licenses like Skype, Adobe Acrobat reader, that allow copying for personal use but prohibit commercial distribution, or proprietary licenses where copying is prohibited by copyright law.

Many of the proprietary software that we use are sold commercially and their program code (source code) are not shared or distributed. However, there are certain software available freely for anyone and their source code is also open for anyone to access, modify, correct and improve. Free and open source software (FOSS) has a large community of users and developers who are contributing continuously towards adding new features or improving the existing features. For example, Linux kernel-based operating systems like Ubuntu and Fedora come under FOSS. Some of the popular FOSS tools are office packages, like Libre Office, browser like Mozilla Firefox.

Software piracy is the unauthorised use or distribution of software. Those who purchase a license for a copy of the software do not have the rights to make additional copies without the permission of the copyright owner. It amounts to copyright infringement regardless of whether it is done for sale, for free distribution or for copier's own use. One should avoid software piracy. Using a pirated software not only degrades the performance of a computer system, but also affects the software industry which in turn affects the economy of a country.

## 2.4 CYBER SECURITY

Cyber security is the application of technologies, processes, and controls to protect systems, networks, programs, devices and data from cyber-attacks. Cyber-attacks can be:

**1. Cyber fraud** – Including phishing, spear phishing, vishing and whaling. A technique carried out over the phone (vishing), email (phishing), text (smishing) or even social media with the goal being to trick you into providing information or clicking a link to install malware on your device.

Spear phishing is targeted phishing. This is even more effective as instead of targets being chosen at random, the attacker takes time to learn a bit about their target to make the wording more specific and relevant.

Whaling is going after executives or presidents. They're hoping for a bigger return on their phishing investment and will take time to craft specific messages in this case as well.

**2. Malware attacks** – Including viruses, worms, Trojans, spyware, rootkits. A malware attack is a common cyber-attack where malware i.e. a malicious software executes unauthorized actions on the victim's system.

**3. Ransomware attacks** – Ransomware is a type of malware attack that encrypts a victim's data and prevents access until a ransom payment is made. Ransomware attackers often use social engineering techniques, such as phishing, to gain access to a victim's environment.

The most common types include:

**Crypto Ransomware or Encryptors** – Encryptors are one of the most well-known and damaging variants. This type encrypts the files and data within a system, making the content inaccessible without a decryption key.

**Lockers** – Lockers completely lock you out of your system, so your files and applications are inaccessible. A lock screen displays the ransom demand, possibly with a countdown clock to increase urgency and drive victims to act.

**Scareware** – Scareware is fake software that claims to have detected a virus or other issue on your computer and directs you to pay to resolve the problem. Some types of scareware lock the

computer, while others simply flood the screen with pop-up alerts without actually damaging files.

**Drive-by downloads** – A drive-by download refers to the unintentional download of malicious code onto a computer or mobile device that exposes users to different types of threats.

**Hacking** – Including distributed denial-of-service attacks (DDoS), key logging- DDoS (Distributed Denial of Service) is a category of malicious cyber-attacks that hackers or Cyber criminals employ in order to make an online service, network resource or host machine unavailable to its intended users on the Internet.

**Password decryption** – Passwords are decrypted by hackers.

**Out-of-date, unpatched software** – Unpatched software refers to applications or systems that contain known vulnerabilities that have not yet been addressed through the implementation of updates or patches.

Various Tools such as Hunchly, Censys, Virtual Box, Kali Linux, 7 Zip, OSINT, Bitwarden, Password and Socint can be used to deal with cyber security issues.

### Practical Activity

Demonstrate to install and use Bitwarden on mobile phones to protect your passwords.

Demonstrate to generate VID for your Aadhar card.

## 2.5 CYBER CRIME

Cyber crime is defined as a crime is done through the use of Cyber space, in which computer is used as a tool to commit crimes. Hacking, phishing, spamming are the different forms of Cyber crime. Cyber crimes are carried out against either an individual, or a group, or an organisation or even against a country, with the intent to directly or indirectly cause physical harm, financial loss or mental harassment. A Cyber criminal attacks a computer or a network to reach other computers in order to disable or damage data or services. Apart from this, a Cyber criminal may spread viruses and other malware to steal private and confidential data for blackmailing and extortion. A computer virus is some lines of malicious code that can copy itself and can destroy data or corrupt the system. Similarly, malware is a software designed to specifically gain unauthorised access to computer systems. The nature of criminal activities are alarmingly increasing day-by-day, with frequent reports of hacking, ransomware attacks, denial-of-service, phishing, email fraud, banking fraud and identity theft.

Cybercrime has following impacts:

1. Involves national security – Threat to Nation.
2. No boundaries – No geographical boundaries, across all globe.
3. Shocking and crippling effects- Unexpected effects on people
4. Ubiquitous in nature – At all places at a time.
5. Remote indulgence – doing whatever wants.
6. Anonymity of the offenders – No identification of offenders.
7. Anyone can be a victim even inadvertently – You can be a victim by mistake also.
8. Lasting footprints – Significant impact on person.
9. Kids access internet – Kids may get involved.

### Forms of Cyber Crime

There are different forms of Cybercrime.

**Hacking** – Hacking is the act of identifying and then exploiting weaknesses in a computer system or network, usually to gain unauthorized access to personal or organizational data.

**Phishing** – Phishing is a technique for attempting to acquire sensitive data, such as bank account numbers, through a fraudulent solicitation in email or on a web site.

**Whaling** – Whaling is going after executives or presidents to acquire company

**Pornography / CSAM** – The meaning of pornography is the depiction of erotic behavior as in pictures or writing intended to cause sexual excitement. CSAM means Child sexual abuse material.

**Sexting / Sextortion** – It is the practice of extorting money or sexual favors from someone by threatening to reveal evidence of their sexual activity.

**DoS Attacks** – A Denial-of-Service (DoS) attack is an attack meant to shut down a machine or network, making it inaccessible to its intended users.

**Virus ingestion** – Virus are injected in machines.

**Vandalism** – Vandalism is the action involving deliberate destruction of or damage to public or private property.

**Privacy breach** – A person other than an authorized user accesses or potentially accesses data.

**Cyber stalking** – It is the repeated use of electronic communications to harass or frighten someone, for example by sending threatening emails.

**Cyber Squatting** – It is an unauthorized registration and use of Internet domain names that are identical or similar to trademarks, service marks, company names, or personal names

**Cyber grooming** – When some adult befriends a child online and builds an emotional connection with future intentions of sexual abuse, sexual exploitation or trafficking

**Cyber Bullying** – To use the internet to harm or frighten another person, especially by sending them unpleasant messages.

**Online job frauds** – It is a job scam that occurs when a scammer poses as an employer or recruiter, and offers attractive employment opportunities, which require the job seeker to pay some money in advance.

**Smishing** – Smishing is a form of phishing that uses mobile phones as the attack platform.

**SIM Swap Scam** – The fraud exploits a mobile phone service provider's ability to seamlessly port a phone number to a device containing a different subscriber identity module.

**Ransomware** – A type of malicious software designed to block access to a computer system until a sum of money is paid.

**Spamming** – Sending the same message indiscriminately to a large number of internet users

**Pharming** – The fraudulent practice of directing internet users to a bogus website that mimics the appearance of a legitimate one, in order to obtain personal information such as passwords, account numbers.

**Online drug trafficking** – An illegal trade involving the cultivation, manufacture, distribution, and sale of substances.

**Espionage, honeytraps** – An attractive person entices another person into revealing information or doing something unwise.

**Vishing** – The fraudulent practice of making phone calls or leaving voice messages purporting to be from reputable companies in order to induce individuals to reveal personal information, such as bank details and credit card numbers.

**Insurance** – It is a contract that an entity can purchase to help reduce the financial risks associated with doing business online. Fraud takes place in this process.

**OLX Like Apps** – Selling and purchasing online can lead to a fraud.

**Corona related** – Covid related misinformation is circulated for fraud.

**Cyber Terrorism** – Cyber-attacks on government agencies, defense and high tech companies.

#### **Examples of Cyber Crime**

1. A locks the computer of B with a password without the knowledge of B
2. Denies or causes denial of access to any person authorized to access any computer

3. A sends nude pictures to B; B sends them to C

### **Practical Activity**

Read your local newspaper for last one month and note various Cybercrime news spotted in newspaper.

Demonstrate the messages containing fraud links, emails, sharing of OTPs and fraud KYC. (Give some screen shots of fraud links, emails and messages along with explanation)

## **2.6 INDIAN INFORMATION TECHNOLOGY ACT (IT ACT)**

With the increasing use of Internet, many cases of Cyber crimes, frauds, Cyber attacks and Cyber bullying are reported. The nature of fraudulent activities and crimes keeps changing. To deal with such cases, many countries have come up with legal measures for protection of sensitive personal data and to safeguard the rights of Internet users. The Government of India's Information Technology Act, 2000 came into force on 17th October 2000, amended in 2008, provides guidelines to the user on the processing, storage and transmission of sensitive information. Act to provide legal recognition to transactions carried out by means of electronic data interchange and other electronic means E-Commerce. It is the primary law in India dealing with Cybercrime and electronic commerce. In many Indian states, there are Cyber cells in police stations where one can report any Cyber crime. The act provides legal framework for electronic governance by giving recognition to electronic records and digital signatures. The act outlines Cyber crimes and penalties for them.

Cyber Appellate Tribunal has been established to resolve disputes arising from cyber crime, such as tampering with computer source documents, hacking the computer system, using password of another person, publishing sensitive personal data of others without their consent. The act is needed so that people can perform transactions over the Internet through credit cards without fear of misuse. Not only people, the act empowers government departments also to accept filing, creation and storage of official documents in the digital format.

The original Act contained 94 sections, divided into 13 chapters and 4 schedules. The laws apply to the whole of India. If a crime involves a computer or network located in India, persons of other nationalities can also be indicted under the law,

Definitions for Digital Signature/ E-Signature/ E-Documents/ E-Storage of data, Electronic fund transfers and Electronic book-keeping are given in IT act.

### **IT Amendment Act 2008**

This amendment focus on data privacy, Info security, Defining cyber café, Reasonable security protections, Intermediaries and their role.

Additional crimes like pornography and Cyber terrorism are also described in this amendment Inspector to investigate not DSP is the modification in IT act.

Section 43 covers penalties and compensation. Section 43(a) deals with secures access including tracking computer, computer trespass, and violation of privacy. Banks are generally found liable under this Section. Section 43(b) deals with downloads, copies or extracts any data, data base or information. Section 43(c) deals with introduces or causes to be introduced any computer containment or virus. Section 43(d) deals with damages or causes to be damaged any computer, computer system, network, data, data base or any program. It includes physical or virtual means. Section 43(e) deals with disrupts or causes disruption. Section 43(f) deals with denies or causes denial of access to any authorized person. Section 43(g) provides any assistance to any person in contravention of this act. Charges the services availed off by a person to the account of another person by tampering with or manipulating. Online fraud and phishing are covered under this section. Section 43(i) deals with destroys, deletes or alters any information, including diminishing its value or effecting it injuriously by any means.

Section 65 deals with tampering with computer source documents for which there is a punishment for 3 years or fine of Rs. 2 lac or both. Section 66 deals with computer related

offenses. This section is related to Sec 43. For example, hacking. Any person who dishonestly or fraudulently does any act referred to in Sec 43. Dishonestly and fraudulently as defined in IPC. Section 66A deals with sending offensive message from any communication device.

Section 66B deals with dishonestly receiving or retaining any computer resource or communication device. It has a provision of imprisonment up to 3 years or fine up to Rs.1 lac or both. Example, purchasing stolen computer or cell phone is covered under this section. Section 66C deals with theft of identity. It has a provision of imprisonment up to 3 years and fine up to Rs.1 lac. Example, cloning of ATM cards is covered under this section. Most of the financial frauds are covered under this Act. Section 66D deals with cheating by personation. It has a provision imprisonment up to 3 years and fine up to Rs.1 lac.

Section 66E deals with violating privacy i.e. intentionally/knowingly captures, publishes or transmits the image of a private area of any person without his or her consent under circumstances violating the privacy of that person. It has a provision of Imprisonment up to 3 years and fine up to Rs. 1 lac.

Section 66F deals with Cyber Terrorism. It has a provision of Life imprisonment. Causing death or injuries with Intent to threaten the unity, integrity, security or sovereignty of India or to strike terror is covered under this section.

### **SUMMARY**

- Digital footprint is the trail of data we leave behind when we visit any website (or use any online application or portal) to fill-in data or perform any transaction.
- A user of digital technology needs to follow certain etiquette like net-etiquette, communication-etiquette and social media-etiquette.
- Net-etiquette includes avoiding copyright violations, respecting privacy and diversity of users, and avoiding Cyber bullies and Cyber trolls, besides sharing of expertise.
- Communication-etiquette requires us to be precise and polite in our conversation so that we remain credible through our remarks and comments.
- While using social media, one needs to take care of security through password, be aware of fake information and be careful while befriending unknowns. Care must be taken while sharing anything on social media as it may create havoc if being mishandled, particularly our personal, sensitive information.
- Intellectual Property Rights (IPR) help in data protection through copyrights, patents and trademarks. There are both ethical and legal aspects of violating IPR. A good digital citizen should avoid plagiarism, copyright infringement and trademark infringement.
- Certain software is made available for free public access. Free and Open Source Software (FOSS) allow users to not only access but also to modify (or improve) them.
- Cyber crimes include various criminal activities carried out to steal data or to break down important services. These include hacking, spreading viruses or malware, sending phishing or fraudulent emails, ransomware, etc.

**CHECK YOUR PROGRESS****(A) Multiple choice questions**

1. Digital footprints are stored in (a) browsing history, (b) cookies (c) servers (d) all of the above
2. A responsible netizen must abide by (a) net etiquette, (b) communication etiquette (c) social media etiquette (d) all of the above
3. Which of the following acts does not violate the intellectual property right (a) Plagiarism (b) Copyright Infringement (c) Trademark Infringement (d) Patent
4. Which of the following section deals with computer related offenses (a) section 66 (b) section 65 (c) section 66A (d) section 66B
5. Which of the following section deals with dishonestly receiving or retaining any computer resource or communication device. (a) section 66 (b) section 65 (c) section 66A (d) section 66B
6. Which of the following section deals with cheating by personation (a) section 66 (b) section 65 (c) section 66D (d) section 66B
7. Sending the same message indiscriminately to a large number of internet users refers to (a) Pharming (b) Spamming (c) Smishing (d) Ransomware
8. An illegal trade involving the cultivation, manufacture, distribution, and sale of substances refers to (a) Pharming (b) Spamming (c) Online drug trafficking (d) SIM Swap Scam .
9. The action involving deliberate destruction of or damage to public or private property refers to (a) Cyber stalking (b) Cyber Squatting (c) Cyber grooming (d) Vandalism
10. Cyber-attacks on government agencies, defense and high tech companies refers to (a) Cyber stalking (b) Cyber Squatting (c) Cyber grooming (d) Cyber Terrorism

**(B) Fill in the blanks**

1. The digital data trail left online unintentionally is called \_\_\_\_\_ digital footprints.
2. Infringement is classified into \_\_\_\_\_ and \_\_\_\_\_ infringement.
3. Software piracy is the \_\_\_\_\_ use or distribution of software.
4. In \_\_\_\_\_, a malicious software executes unauthorized actions on the victim's system.
5. Cyber Appellate Tribunal was established to resolve disputes arising from \_\_\_\_\_
6. Section \_\_\_ of IT Act covers penalties and compensation.
7. The Government of India's Information Technology Act, 2000 came into force on \_\_\_\_\_, amended in \_\_\_\_\_.
8. Section \_\_\_\_\_ deals with sending offensive message from any communication device. )
9. A \_\_\_\_\_ attack is an attack meant to shut down a machine or network, making it inaccessible to its intended users.
10. \_\_\_\_\_ is a form of phishing that uses mobile phones as the attack platform.

**(C) State whether True or False**

1. The digital footprint can be created and used with or without our knowledge.
2. The digital footprint once stored, cannot be deleted.
3. Data protection is related with the privacy of digital data.
4. The ownership of intellectual property lies with its distributor.
5. Online fraud and phishing are covered under Section 43(i)
6. The original IT Act contained 94 sections, divided into 13 chapters and 4 schedules.
7. The IT Act 2000, provides guidelines to the user on processing, storage and transmission of sensitive information.



8. Additional crimes like pornography and Cyber terrorism are also described in IT Act 2000
9. Passwords are encrypted by hackers.
10. Phishing and hacking are the same forms of Cybercrime.

**(D) Short answer questions**

1. What is the digital footprint? How it is stored?
2. What are the Net Etiquette?
3. What are the Communication Etiquette?
4. What are the Social Media Etiquette?
5. Write the long form of the acronym (a) GNU (b) GPL (c) FOSS (d) CC
6. Explain the term (a) vishing (b) phishing (c) smishing
7. List the different forms of Cybercrime.
8. What is Cyber security?
9. List the impacts of Cybercrime on the society.
10. What are the advantages of FOSS?

**(E) Match the following**

| Column A           | Column B                                                                                                            |
|--------------------|---------------------------------------------------------------------------------------------------------------------|
| Plagiarism         | Fakers, by offering special rewards or money prize asked for personal information, such as bank account information |
| Hacking            | Copy and paste information from the Internet into your report and then organise it                                  |
| Credit card fraud  | The trail that is created when a person uses the Internet.                                                          |
| Digital Foot Print | Breaking into computers to read private emails and other files                                                      |

## Answer

### Module 1. RDBMS Concepts in MySQL

#### Session 1. RDBMS Concepts

##### A. Multiple choice questions

1. (c) 2. (c) 3. (a) 4. (b) 5. (b) 6. (b) 7. (c) 8. (a) 9. (a) 10. (b) 11. (b) 12. (d)

##### B. Fill in the blanks

1. Relation, Tuple 2. DBMS 3. tables 4. show database 5. structure 6. foreign key 7. Composite Key 8. Primary Key 9. Primary Key 10. null, duplicate

##### C. State whether True or False

1. (T) 2. (T) 3. (T) 4. (T) 5. (T) 6. (F) 7. (F) 8. (T) 9. (F) 10. (F)

#### Session 2. Structured Query Language (SQL)

##### A. Multiple choice questions

1. (b) 2. (c) 3. (d) 4. (c) 5. (c) 6. (d) 7. (d) 8. (d) 9. (c) 10. (b) 11. (c) 12. (c) 13. (b) 14. (b) 15. (c) 16. (c)

##### B. Fill in the blanks

1. Five 2. Desc 3. Delete 4. alter 5. truncate 6. Drop 7. virtual 8. one 9. DCL 10. TCL 11. ORDER BY DESC 12. DISTINCT

##### C. State whether True or False

1. (F) 2. (T) 3. (T) 4. (F) 5. (F) 6. (T) 7. (T) 8. (F) 9. (T) 10. (F)

#### Session 3. Functions In SQL

##### A. Multiple choice questions

1. (d) 2. (c) 3. (a) 4. (d) 5. (a) 6. (b) 7. (d) 8. (d) 9. (c) 10. (a)

##### B. Fill in the blanks

1. value 2. (c) set of records 3. numeric 4. numeric 5. ascending 6. first occurrence 7. (d) 8. leading 9. leading, trailing 10. Intersect

##### C. State True or False

1. (F) 2. (F) 3. (T) 4. (T) 5. (F) 6. (F) 7. (T) 8. (T) 9. (T) 10. (T)

### Module 2. Advanced Python Programming

#### Session 1. Implementing Data Structure using Stack & Queue

##### A. Multiple Choice Questions

1. (b) 2. (d) 3. (c) 4. (a) 5. (a)

##### B. State whether True or False

1. (T) 2. (F) 3. (F) 4. (T) 5. (F)

##### C. Fill in the blanks

1. LIFO 2. Queue 3. FIFO 4. enqueue, dequeue 5. front

### Session 2. Exception Handling in Python

#### A. Multiple choice questions

1. (a) 2. (b) 3. (c) 4. (d) 5. (a)

#### B. Fill In the Blank

1. Interpreter 2. Exception 3. built-in 4. Raise 5. Assert

#### C. State whether True or False

1. (T) 2. (T) 3. (T) 4. (F) 5. (F)

### Session 3. File Handling in Python

#### A. Multiple choice questions

1. (a) 2. (b) 3. (b) 4. (c) 5. (c)

#### B. Fill in The Blank

1. (a) text file 2. a stream of 3. EOL (b) 4. open() 5. (c) newline character (\n)

#### C. State whether True or False

1. (F) 2. (F) 3. (F) 4. (F) 5. (T)

### Session 4. Numpy Array

#### A. Multiple Choice Questions

1. (c) 2. (d) 3. (c) 4. (a) 5. (c)

#### B. Fill in the Blank

1. colon 2. Arithmetic 3. (c) Data Analysis 4. 1D and 2D 5. array()

### Session 5. Pandas and Series in Python

#### A. Multiple Choice Questions

1. (c) 2. (b) 3. (d) 4. (b) 5. (d)

#### B. Fill in the blanks

1. data analysis 2. numpy array 3. data frame 4. mutable (b) 5. 2D

#### C. State whether True or False

1. (T) 2. (F) 3. (T) 4. (T) 5. (F)

### Session 6. Graphical Representation using Matplotlib

#### A. Multiple Choice Questions

1. (d) 2. (a) 3. (c) 4. (d) 5. (a) 6. (b) 7. (a)

#### B. Fill in the blanks

1. savefig 2. color 3. title 4. xlabel, ylabel 5. show()

#### C. State whether True or False

1. (T) 2. (T) 3. (F) 4. (T) 5. (F)

### Session 7. Database Connectivity with MySQL

**A. Multiple choice questions**

1. (d) 2. (b) 3. (a). 4. (d) 5. (b) 6. (d) 7. (b)

**B. Fill in the blank**

1. fetchone() 2. fetchall() 3. import mysql-connector 4. rowcount() 5. execute()

**C. State whether True or False**

1. (T) 2. (T) 3. (T) 4. (F) 5. (F)

**Module 3. Software Engineering****Session 1. Software Engineering Concepts****A. Multiple choice questions**

1. (d) 2. (b) 3. (a) 4. (b) 5. (c) 6. (d) 7. (d) 8. (c)

**B. Fill in the blanks**

1. Iterative and Incremental model 2. linear-sequential model 3. Verification and Validation model 4. Agile 5. Spiral 6. Prototype 7. agile 8. system implementation

**C. State whether True or False**

1. (T) 2. (T) 3. (T) 4. (T) 5. (F) 6. (T) 7. (F) 8. (T)

**Session 2. Software Development Process****A. Multiple choice questions**

1. (a) 2. (c) 3. (a) 4. (c) 5. (b) 6. (c) 7. (d) 8. (d) 9. (c) 10. (d)

**B. Fill in the blanks**

1. (cardinality) 2. (active, passive) 3. (predefined) 4. (constraints) 5. (reviews) 6. (context diagram) 7. (logical way) 8. (logical, physical) 9. (data dictionary) 10. (structured logic) 11 (cohesion, coupling) 12. (divide, conquer) 13. (top down, bottom up strategy) 14. (corrective) 15. (key)

**C. State whether True or False**

1. (T) 2. (T) 3. (F) 4. (T) 5. (F) 6. (T) 7. (T) 8. (T) 9. (F) 10. (F) 11. (F) 12. (F) 13. (T) 14. (T) 15. (T) 16. (T) 17. (T) 18. (T) 19. (T) 20. (F)

**Module 4. Emerging Trends and Social Impact****Session 1. Emerging Technologies****(A) Multiple choice questions**

1. (b) 2. (b) 3. (c) 4. (d) 5. (b) 6. (d)

**(B) Fill in the blanks**

1. (natural intelligence) 2. (driving simulators) 3. (human languages) 4. (autoregressive language model) 5. (decentralized, shared)

**(C) State whether True or False**

1. (T) 2. (T) 3. (F) 4. (T) 5. (F)

**Session 2. Societal Impact**

**(A) Multiple choice questions**

1. (d) 2. (d) 3. (d) 4. (a) 5. (d) 6. (c) 7. (b) 8. (c) 9. (d) 10. (d)

**(B) Fill in the blanks**

1. passive 2. primary, secondary 3. unauthorised 4. malware attack 5. cybercrime 6. (43) 7. 17th October 2000, 2008 8. 66A 9. Denial-of-Service 10. Smishing

**(C) State whether True or False**

1. (T) 2. (T) 3. (T) 4. (F) 5. (F) 6. (T) 7. (T) 8. (F) 9. (F) 10. (F)

PSSCIVE Draft Study Material © Not to be Published