Unit V

What is Database

- The database is a collection of inter-related data which is used to retrieve, insert and delete the data efficiently. It is also used to organize the data in the form of a table, schema, views, and reports, etc.
- **For example:** The college Database organizes the data about the admin, staff, students and faculty etc.
- Using the database, you can easily retrieve, insert, and delete the information.

Database Management System

- Database management system is a software which is used to manage the database. For example: MySQL, Oracle, etc are a very popular commercial database which is used in different applications.
- DBMS provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more.
- It provides protection and security to the database. In the case of multiple users, it also maintains data consistency.

Continue..

- Database is a collection of related data and data is a collection of facts and figures that can be processed to produce information.
- Mostly data represents recordable facts. Data aids in producing information, which is based on facts. For example, if we have data about marks obtained by all students, we can then conclude about toppers and average marks.
- A database management system stores data in such a way that it becomes easier to retrieve, manipulate, and produce information.

Characteristics of DBMS

- It uses a digital repository established on a server to store and manage the information.
- It can provide a clear and logical view of the process that manipulates data.
- DBMS contains automatic backup and recovery procedures.
- It contains ACID properties which maintain data in a healthy state in case of failure.
- It can reduce the complex relationship between data.
- It is used to support manipulation and processing of data.
- It is used to provide security of data.
- It can view the database from different viewpoints according to the requirements of the user.

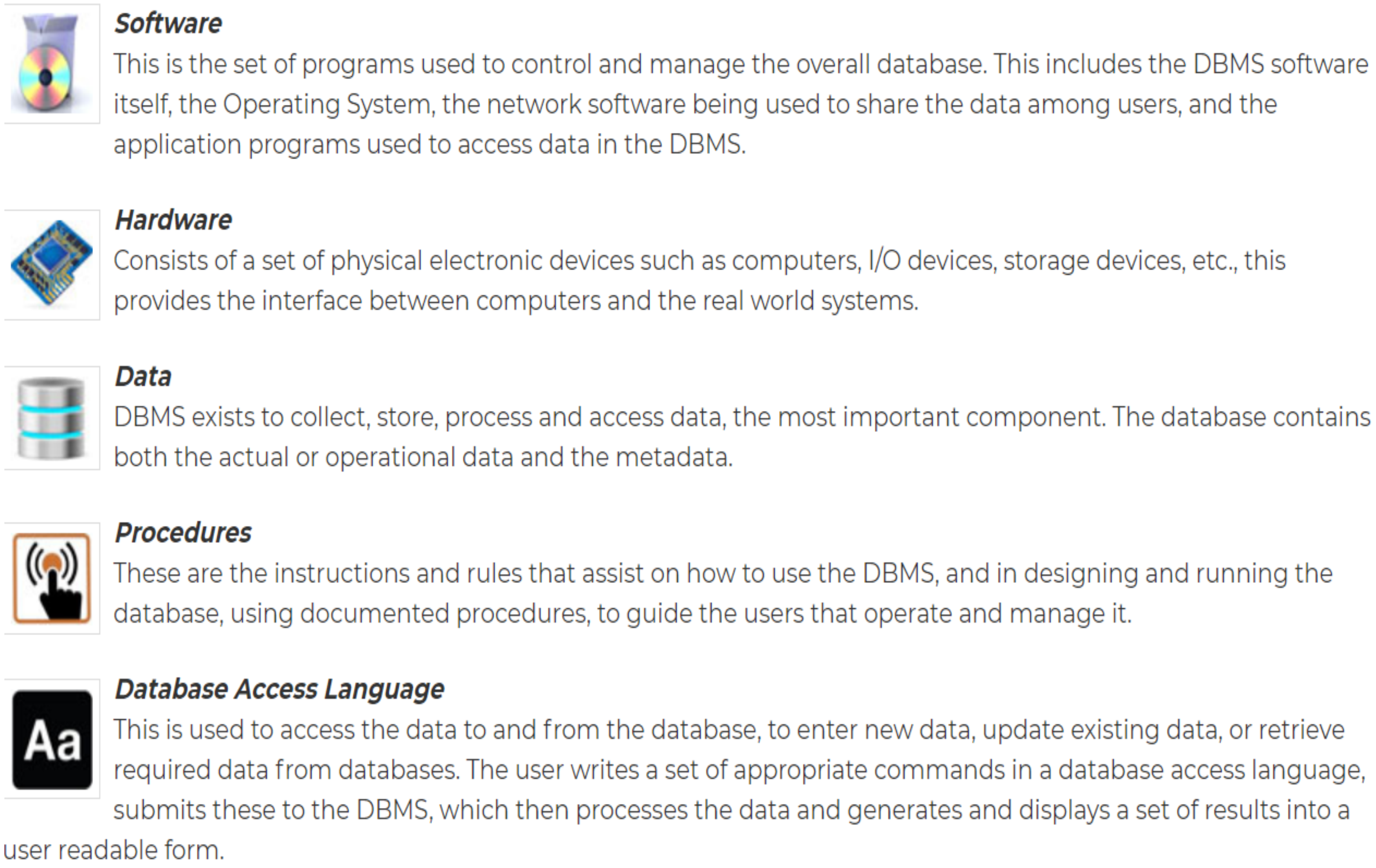
Advantages of DBMS

- **Controls database redundancy:** It can control data redundancy because it stores all the data in one single database file and that recorded data is placed in the database.
- **Data sharing:** In DBMS, the authorized users of an organization can share the data among multiple users.
- **Easily Maintenance:** It can be easily maintainable due to the centralized nature of the database system.
- **Reduce time:** It reduces development time and maintenance need.
- **Backup:** It provides backup and recovery subsystems which create automatic backup of data from hardware and software failures and restores the data if required.
- **Multiple User Interface:** It provides different types of user interfaces like graphical user interfaces, application program interfaces.

Disadvantages of DBMS

- **Cost of Hardware and Software:** It requires a high speed of data processor and large memory size to run DBMS software.
- **Size:** It occupies a large space of disks and large memory to run them efficiently.
- **Complexity:** Database system creates additional complexity and requirements.
- **Higher impact of failure:** Failure is highly impacted the database because in most of the organization, all the data stored in a single database and if the database is damaged due to electric failure or database corruption then the data may be lost forever.

Components of DBMS



Software

This is the set of programs used to control and manage the overall database. This includes the DBMS software itself, the Operating System, the network software being used to share the data among users, and the application programs used to access data in the DBMS.

Hardware

Consists of a set of physical electronic devices such as computers, I/O devices, storage devices, etc., this provides the interface between computers and the real world systems.

Data

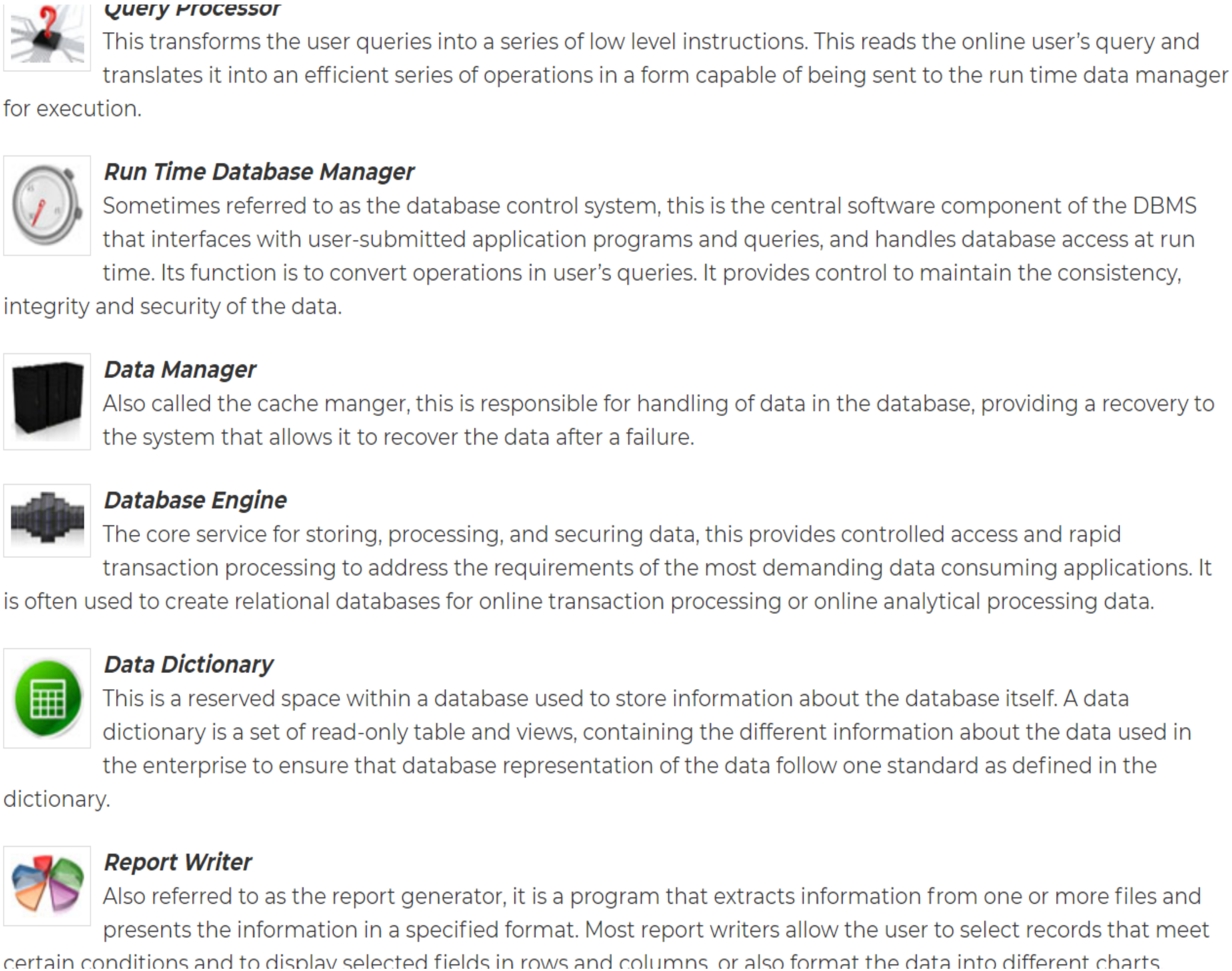
DBMS exists to collect, store, process and access data, the most important component. The database contains both the actual or operational data and the metadata.

Procedures

These are the instructions and rules that assist on how to use the DBMS, and in designing and running the database, using documented procedures, to guide the users that operate and manage it.

Database Access Language

This is used to access the data to and from the database, to enter new data, update existing data, or retrieve required data from databases. The user writes a set of appropriate commands in a database access language, submits these to the DBMS, which then processes the data and generates and displays a set of results into a user readable form.



Query Processor

This transforms the user queries into a series of low level instructions. This reads the online user's query and translates it into an efficient series of operations in a form capable of being sent to the run time data manager for execution.

Run Time Database Manager

Sometimes referred to as the database control system, this is the central software component of the DBMS that interfaces with user-submitted application programs and queries, and handles database access at run time. Its function is to convert operations in user's queries. It provides control to maintain the consistency, integrity and security of the data.

Data Manager

Also called the cache manger, this is responsible for handling of data in the database, providing a recovery to the system that allows it to recover the data after a failure.

Database Engine

The core service for storing, processing, and securing data, this provides controlled access and rapid transaction processing to address the requirements of the most demanding data consuming applications. It is often used to create relational databases for online transaction processing or online analytical processing data.

Data Dictionary

This is a reserved space within a database used to store information about the database itself. A data dictionary is a set of read-only table and views, containing the different information about the data used in the enterprise to ensure that database representation of the data follow one standard as defined in the dictionary.

Report Writer

Also referred to as the report generator, it is a program that extracts information from one or more files and presents the information in a specified format. Most report writers allow the user to select records that meet certain conditions and to display selected fields in rows and columns, or also format the data into different charts

DATA INDEPENDENCE

- A database system normally contains a lot of data in addition to users' data. For example, it stores data about data, known as metadata, to locate and retrieve data easily. It is rather difficult to modify or update a set of metadata once it is stored in the database.
- But as a DBMS expands, it needs to change over time to satisfy the requirements of the users. If the entire data is dependent, it would become a tedious and highly complex job.

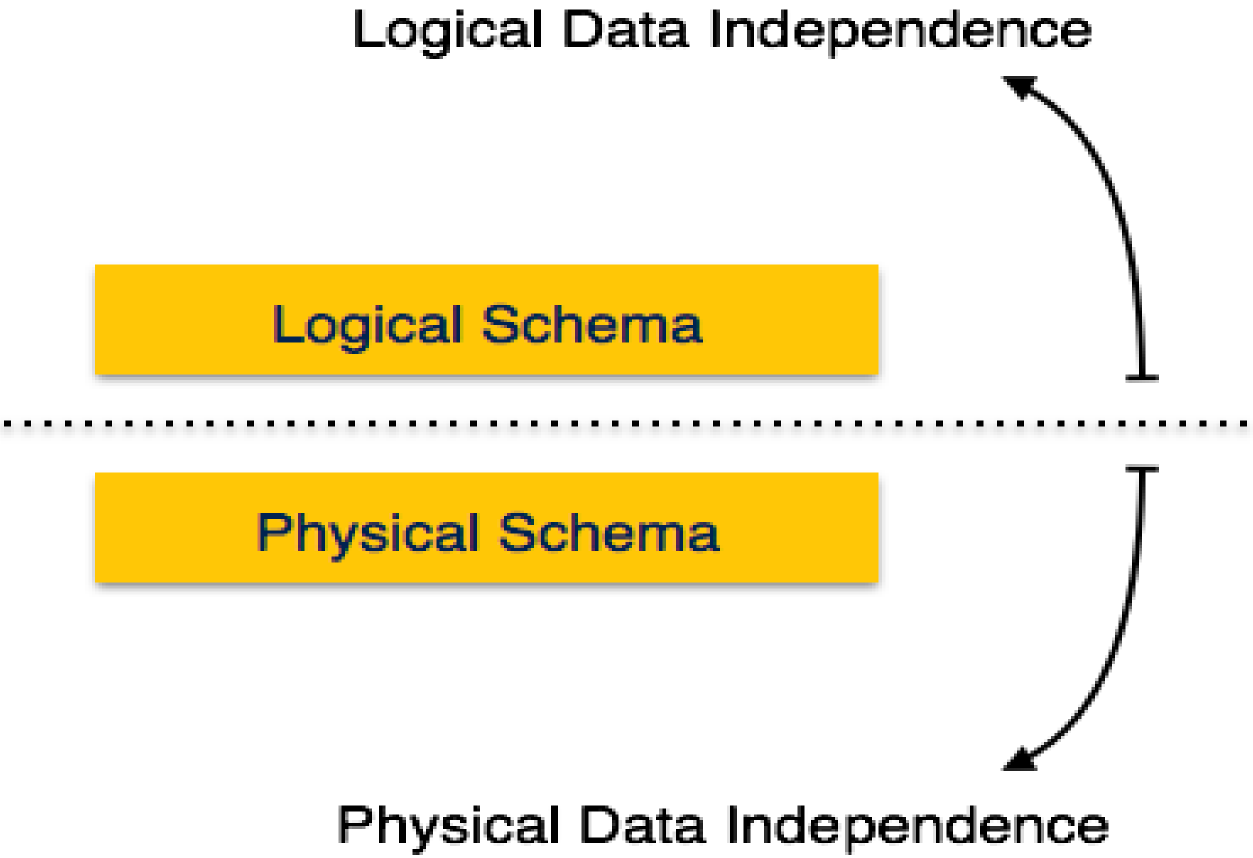
Logical Data Independence

- Logical data is data about database, that is, it stores information about how data is managed inside. For example, a table (relation) stored in the database and all its constraints, applied on that relation.
- Logical data independence is a kind of mechanism, which liberalizes itself from actual data stored on the disk. If we do some changes on table format, it should not change the data residing on the disk.

Physical Data Independence

- All the schemas are logical, and the actual data is stored in bit format on the disk. Physical data independence is the power to change the physical data without impacting the schema or logical data.
- For example, in case we want to change or upgrade the storage system itself − suppose we want to replace hard-disks with SSD − it should not have any impact on the logical data or schemas.

Continue..


Logical Data Independence
Logical Schema
Physical Schema
Physical Data Independence

Introduction to Computer Programming

- If you understood what a computer program is, then we will say: the act of writing computer programs is called computer programming.
- As we mentioned earlier, there are hundreds of programming languages, which can be used to write computer programs and following are a few of them –
  - Java
  - C
  - C++
  - Python
  - PHP
  - Perl
  - Ruby

CONTINUE..

- A program is a set of instructions given to a computer to perform a specific operation. or computer is a computational device which is used to process the data under the control of a computer program. While executing the program, raw data is processed into a desired output format. These computer programs are written in a programming language which are high level languages.
- The computer only understands binary language (the language of 0's and 1's) also called machine-understandable language or low-level language but the programs we are going to write are in a high-level language which is almost similar to human language.

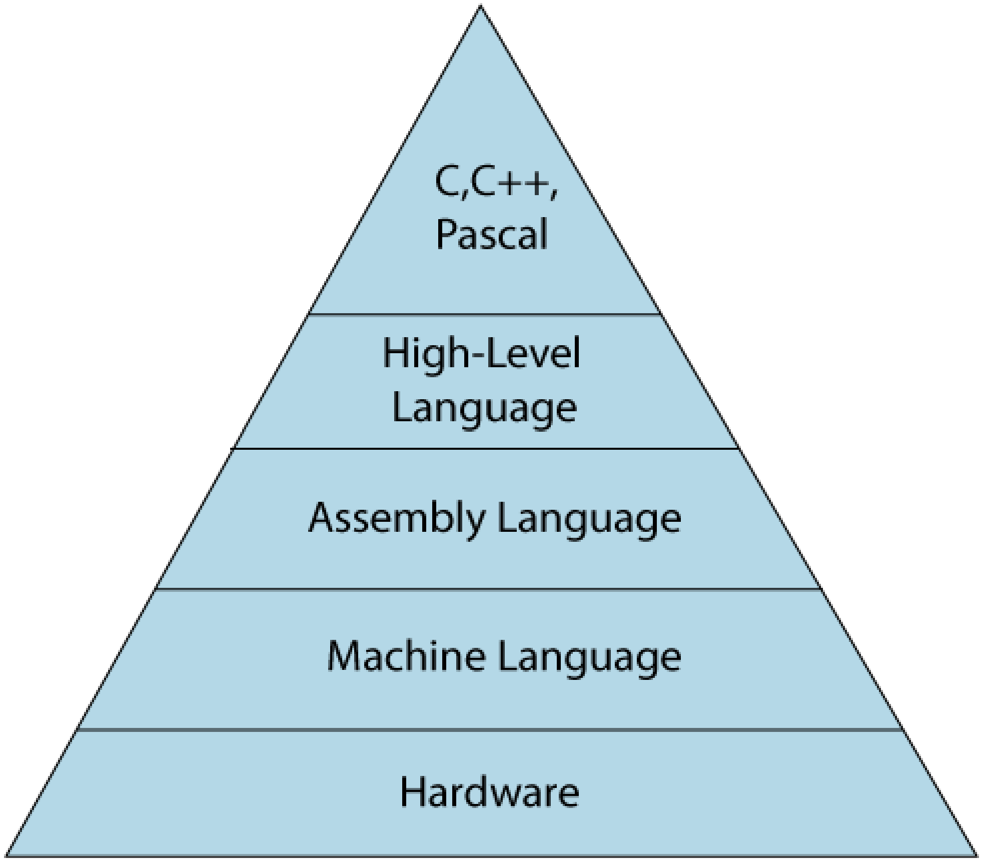
Continue..

- High level languages are nearly human languages which are more complex then the computer understandable language which are called machine language, or low level language. So after knowing the basics, we are ready to create a very simple and basic program.
- Like we have different languages to communicate with each other, likewise, we have different languages like C, C++, C#, Java, python, etc to communicate with the computers.

WHAT IS A PROGRAMMING LANGUAGE?

- A programming language defines a set of instructions that are compiled together to perform a specific task by the CPU (Central Processing Unit). The programming language mainly refers to high-level languages such as C, C++, Pascal, Ada, COBOL, etc.
- The image which is given below describes the abstraction level from hardware. As we can observe from the below image that the machine language provides no abstraction, assembly language provides less abstraction whereas high-level language provides a higher level of abstraction.

Continue..


C,C++,
Pascal
High-Level
Language
Assembly Language
Machine Language
Hardware

CONTINUE..

- Each programming language contains a unique set of keywords and syntax, which are used to create a set of instructions. Thousands of programming languages have been developed till now, but each language has its specific purpose. These languages vary in the level of abstraction they provide from the hardware. Some programming languages provide less or no abstraction while some provide higher abstraction. Based on the levels of abstraction, they can be classified into two categories:
  - Low-level language
  - High-level language

Low-level language

- The low-level language is a programming language that provides no abstraction from the hardware, and it is represented in 0 or 1 forms, which are the machine instructions. The languages that come under this category are the Machine level language and Assembly language.

Machine-level language

- The machine-level language is a language that consists of a set of instructions that are in the binary form 0 or 1. As we know that computers can understand only machine instructions, which are in binary digits, i.e., 0 and 1, so the instructions given to the computer can be only in binary codes.
- Creating a program in a machine-level language is a very difficult task as it is not easy for the programmers to write the program in machine instructions.

Continue..

- It is error-prone as it is not easy to understand, and its maintenance is also very high.
- A machine-level language is not portable as each computer has its machine instructions, so if we write a program in one computer will no longer be valid in another computer.

Assembly Language

- The assembly language contains some human-readable commands such as mov, add, sub, etc. The problems which we were facing in machine-level language are reduced to some extent by using an extended form of machine-level language known as assembly language. Since assembly language instructions are written in English words like mov, add, sub, so it is easier to write and understand.
- As we know that computers can only understand the machine-level instructions, so we require a translator that converts the assembly code into machine code. The translator used for translating the code is known as an assembler.

Continue..

- The assembly language code is not portable because the data is stored in computer registers, and the computer has to know the different sets of registers.
- The assembly code is not faster than machine code because the assembly language comes above the machine language in the hierarchy, so it means that assembly language has some abstraction from the hardware while machine language has zero abstraction.

High-Level Language

- The high-level language is a programming language that allows a programmer to write the programs which are independent of a particular type of computer. The high-level languages are considered as high-level because they are closer to human languages than machine-level languages.
- Advantages of a high-level language
  - The high-level language is easy to read, write, and maintain as it is written in English like words.
  - The high-level languages are designed to overcome the limitation of low-level language, i.e., portability. The high-level language is portable; i.e., these languages are machine-independent.

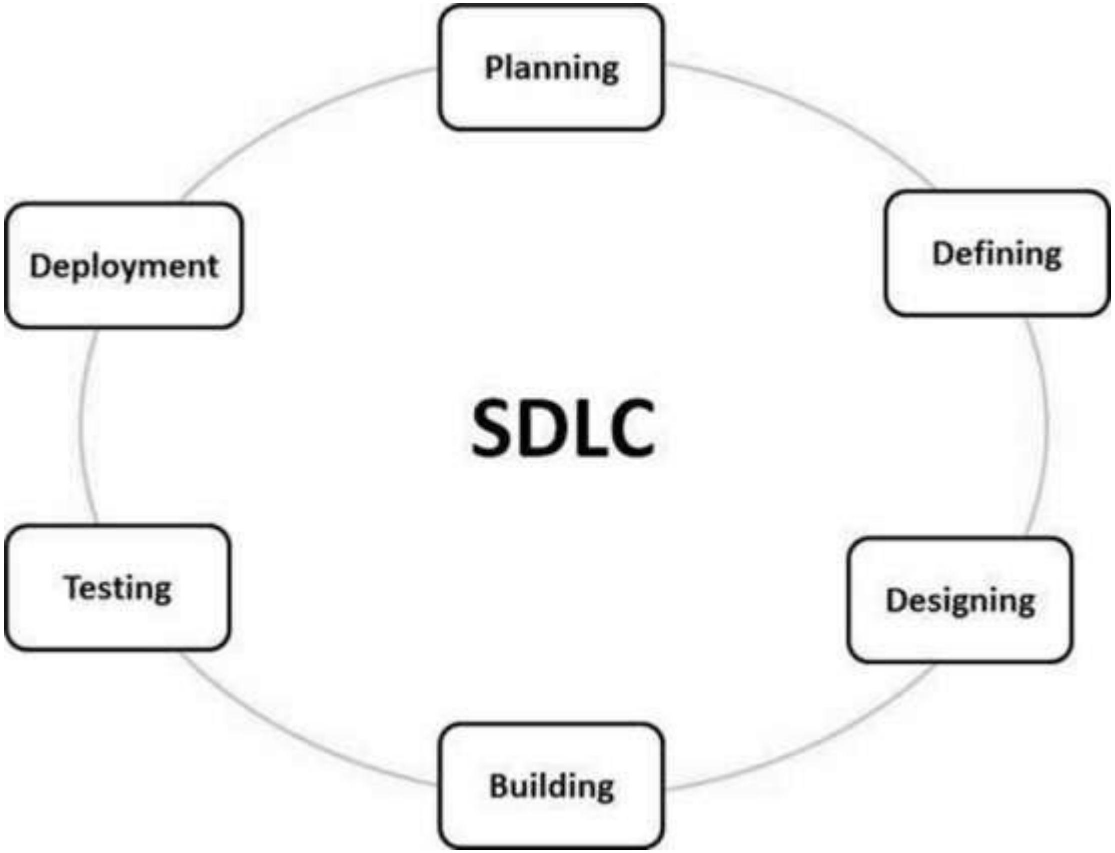
Software Development Life Cycle (SDLC)

- Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality softwares. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.
  - SDLC is the acronym of Software Development Life Cycle.
  - It is also called as Software Development Process.
  - SDLC is a framework defining tasks performed at each step in the software development process.
  - ISO/IEC 12207 is an international standard for software life-cycle processes. It aims to be the standard that defines all the tasks required for developing and maintaining software.

What is SDLC?

- SDLC is a process followed for a software project, within a software organization. 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.
- The following figure is a graphical representation of the various stages of a typical SDLC.

Continue..


Planning
Defining
Designing
Building
Testing
Deployment
SDLC

Stage 1: Planning and Requirement Analysis

- Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

CONTINUE..

- Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

Stage 2: Defining Requirements

- Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an SRS (Software Requirement Specification) document which consists of all the product requirements to be designed and developed during the project life cycle.

Stage 3: Designing the Product Architecture

- SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.
- This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product.

CONTINUE..

- A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

Stage 4: Building or Developing the Product

- In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.
- Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

Stage 5: Testing the Product

- This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

Stage 6: Deployment in the Market and Maintenance

- Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).
- Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

SDLC Models

- There are various software development life cycle models defined and designed which are followed during the software development process. These models are also referred as Software Development Process Models". Each process model follows a Series of steps unique to its type to ensure success in the process of software development.
- Following are the most important and popular SDLC models followed in the industry −
  - Waterfall Model
  - Iterative Model
  - Spiral Model
  - V-Model
  - Big Bang Model

Web Development Languages

- There is a saying about “right tool for the right job” and it very much applies to a programming language. Not all programming languages are designed the same and that’s why they are not equally good for everything.
- For example Java, is great for writing server-side code but I cannot use it for frontend development. There were attempts in the past in the form of writing dynamic code in Servlet and JSP using Java code but that created more problems than they solved.

CONTINUE..

- That’s where knowing about different programming languages helps so that you can choose the perfect programming language for your job. For example, Java is best for creating backends but JavaScript is best for both frontend and backend and no surprise it is the top programming language for web development.

1. JavaScript

- There is no doubt that JavaScript is the King of web development and probably the most popular language among web developers. It's also the only language that allows you to create web applications, both frontend, and backend as well as mobile applications (React Native).
- The strength of Javascript is not just it can run on browser and server using Nodejs but also the awesome frameworks and libraries it has for web development and app development.

Continue..

- For example, you can use React.js and Angular for frontend, Nodejs for backend, and React Native for creating cross-platform (both Android and iOS) mobile apps.
- Although JavaScript is primarily a front-end language run on the browser, it can also be used on the server-side through Node.js to build scalable network applications. Node.js is compatible with Linux, SunOS, Mac OS X, and Windows.
- Because JavaScript has a forgiving, flexible syntax and works across all major browsers, it is one of the friendliest programming languages for beginners.

2. PYTHON

- Python is another language which has gained a lot of popularity in recent year. While it's not exactly new and has passed the test of time, it's probably the most versatile programming language at the moment.
- You can use Python for web development, Data Science, Scripting, and Automation. Similar to JavaScript, Python also enjoys huge community support and has a lot of useful frameworks, libraries, and tools that can help you create a web application in quick time.

CONTINUE..

- For example, you can use Django to create full-stack web applications. If you want more control, you can also use Flask, a minimalist web development framework.
- Python also has packages such as NumPy and SciPy that are commonly used in the fields of scientific computing, mathematics, and engineering. Other Python libraries such as TensorFlow, PyTorch, scikit-learn, and OpenCV are used to build programs in data science, machine learning, image processing, and computer vision. Python's science and data applications make it a great choice for the academically inclined.

3. TypeScript

- TypeScript is my personal favorite and a modern programming language for web development. Just like we have C and C++, TypeScript can be considered as JavaScript++ though it’s not as popular as C++.
- The good part of TypeScript is that it adds type-safety in JavaScript code which means you can catch nasty JavaScript type-related errors in the development phase.
- It also makes developing object-oriented code easy for JavaScript and several inbuilt debugging tools of TypeScript make web development really easy.

PHP

- PHP is one of the best programming languages when it comes to creating web applications.
- It's a dynamic, server-side scripting language that makes it really easy to create fully functional web applications. If that's not enough, half of the internet is running on PHP's shoulder, remember WordPress, the most popular web application software is made on PHP.
- PHP also has frameworks like Laraval, which is both powerful and allows you to swiftly create web applications using a model–view–controller architectural pattern.

5. Ruby

- Ruby is another programming language that is really great for web development. Similar to PHP and Python, Ruby is also easy to learn and goo for beginners.
- What makes Ruby special for web development is the Ruby on Rails framework which powers websites like Github, Shopify, Airbnb, Groupon, GoodReads, and Kickstarter.
- If you don't know, Rails is a model–view–controller (MVC) framework, providing default structures for a database, a web service, and web pages.

CONTINUE..

- It encourages and facilitates the use of web standards such as JSON or XML for data transfer and HTML, CSS, and JavaScript for user interfacing
- In addition to the active community and its straightforward syntax, Ruby is also a good language to pick up thanks to its association with great tech businesses. Twitter, Airbnb, Bloomberg, Shopify, and countless other startups have all built their websites using Ruby on Rails at some point.

6. Go

- One of the core languages favored by Google, Go is the little language that could. As a low-level language, Go is ideal for engineers who want to enter the field of systems programming. It encompasses much of the same functionality of C and C++ without the difficult syntax and steep learning curve. It's the perfect language for building web servers, data pipelines, and even machine-learning packages.
- As a compiled language, Go runs "close to the metal," allowing for a blazing-fast runtime. It's an open-source language, and ambitious developers can see their personal contributions adopted and enjoyed by programmers worldwide.

7. Swift

- If you’re interested in Apple products and mobile app development, Swift is a good place to start. First announced by Apple in 2014, Swift is a relatively new programming language used to develop iOS and macOS applications.
- Swift has been optimized for performance and built from the ground up to match the realities of modern iOS development. Not only does iOS run on every iPhone and iPad, but it’s also the basis for other operating systems such as watchOS (for Apple Watches) and tvOS (for Apple TVs). In addition, Apple isn't going anywhere as a tech industry leader, and iOS apps continue to be the most profitable in the mobile app marketplace.

8. Scala

- Scala combines the best features of Java (such as its Object-Oriented Structure and its lightning-fast JVM runtime environment) with a modern twist.
- As a functional programming language, Scala allows engineers to elevate the quality of their code to resemble pure math. Scala allows for concurrent programming, allowing complex procedures to be executed in parallel. Furthermore, it is a strongly typed language. Engineers can create and customize their own data types, allowing them to have the peace of mind knowing entire swaths of bugs are impossible at runtime.

9. Elm

- One of the youngest languages on our list, what began as a Harvard student's thesis has now grown to become a point of passion for front-end developers around the world.
- Elm compiles to JavaScript, making it ideal for building fast-executing UIs with zero errors at runtime. Elm is a functional programming language, allowing developers to create client-side interfaces without the declarative trappings of HTML and CSS.
- Furthermore, the Elm web architecture was the intellectual inspiration for Redux, the state-management library taught here at Fullstack.

10. C#

- C# a general-purpose, object-oriented language built on the foundations of C. It was originally designed by Microsoft as part of its .NET framework for building Windows applications.
- C# uses a syntax that's similar to other C-derived languages such as C++, so it's easy to pick up if you're coming from another language in the C family. C# is not only the go-to for Microsoft app development, but it's also the language mobile developers use to build cross-platform apps on the Xamarin platform.

CONTINUE..

- Additionally, anyone who is interested in VR development should consider learning C#. C# is the recommended language for building 3D and 2D video games using the popular Unity game engine, which produces one-third of the top games on the market.

11. Rust

- Rust is a bit of an upstart among the other languages on this list, but that doesn't mean it's not a valuable language to learn.
- Stack Overflow's 2020 Developer Survey found that Rust was the most loved programming language among developers for the fifth year in a row, with 86.1 percent of Rust developers saying that they want to continue working with it.

Continue..

- Developed by the Mozilla Corporation, Rust, like C and C++, is intended primarily for low-level systems programming.
- What Rust adds to the mix, however, is an emphasis on speed and security.
- Rust emphasizes writing “safe code” by preventing programs from accessing parts of memory that they shouldn’t, which can cause unexpected behavior and system crashes.