Computer Application in Business PYQ 2019

Q1 a) State True/False with reasons:

(i)Third generation computers were based on transistors

(ii) Multiprocessing operating system may run on the system having two or more processors.

(iii) A network is interconnection of computers that enables the users to share network resources

(w) A derived attribute is directly stored in the table.

Ans.

(**i) True.** Third generation computers, which emerged in the 1960s, were based on transistors, which replaced vacuum tubes used in earlier generations of computers. Transistors were smaller, faster, and more reliable, leading to significant advancements in computer technology.

(ii**) True.** A multiprocessing operating system is designed to handle two or more processors simultaneously, allowing for parallel processing and increased efficiency in handling tasks and workload. This enables multiple processors to work together to execute tasks, leading to improved performance and faster processing times.

(**iii) True.** A network is an interconnection of computers that allows users to share resources such as files, printers, and applications. Networks can be local area networks (LANs) or wide area networks (WANs), and they facilitate communication and data exchange among connected computers, enabling users to share resources and collaborate efficiently.

(iv) False. Derived attributes are not directly stored in a table. Derived attributes are calculated or derived from other attributes in the table, and their values are determined by applying operations or calculations on other attributes. They are not physically stored in the table but are instead calculated or derived as needed when queried from the database.

Q1 b. Choose appropriate words to fill in the blanks:

(i) Operating system is..... (a system / an application) software.

(in) Hard drive is a ……. storage device.(primary /secondary)

(iii) In a .....(ring/star) topology all the computers are connected to each other through a central network hub.

Ans.

(i) Operating system is an application software.

(ii) Hard drive is a secondary storage device.

(iii) In a star topology, all the computers are connected to each other through a central network hub.

Q2 How do the different functional components of a computer system interact with cach other for data processing?

Ans. A computer system consists of several functional components that work together to process data. These components typically include the following:

Central Processing Unit (CPU): The CPU is often referred to as the "brain" of the computer. It is responsible for executing instructions and performing calculations. The CPU interacts with other components by fetching instructions and data from memory, decoding the instructions, and executing them.

Memory (RAM): Memory, or Random Access Memory, is used to temporarily store data and instructions that the CPU needs to access quickly during data processing. The CPU reads and writes data to and from memory to perform operations on the data.

Input devices: Input devices such as keyboards, mice, and other sensors allow users to input data into the computer system. The data is then processed by the CPU, which may involve storing the data in memory or performing calculations on it.

Output devices: Output devices such as monitors, printers, and speakers display or produce the results of data processing for users to view or use.

Storage devices: Storage devices such as hard drives, solid-state drives, and optical drives are used for long-term storage of data and programs. The CPU interacts with storage devices to read and write data as needed during data processing.

System bus: The system bus is a communication pathway that connects the CPU, memory, and other components of the computer system. It allows data and instructions to be transferred between components for processing.

Operating system: The operating system is a software component that manages and controls the overall operation of the computer system. It coordinates the interactions between different components, manages memory, schedules tasks, and handles input and output operations.

The interactions between these functional components are typically orchestrated by the operating system, which manages the flow of data and instructions between the CPU, memory, storage devices, input devices, and output devices. The CPU fetches instructions and data from memory, performs operations on the data, and writes the results back to memory or sends them to output devices. Input devices allow users to provide data for processing, and output devices display or produce the results of data processing. Storage devices provide long-term storage for data and programs, which can be read from and written to by the CPU as needed. The system bus facilitates the communication between different components by transferring data and instructions between

them. Overall, the interactions between these functional components are tightly coordinated to enable the computer system to process data effectively and perform various tasks and operations.

OR

Q2. Application software and system software are two different types of software that serve distinct purposes in a computer system.

Ans. Here are some key differences between the two:

Function: Application software is designed to perform specific tasks or applications for end users. It includes programs like word processors, spreadsheets, web browsers, games, and multimedia players. Application software is typically created for specific purposes and is used by end users to accomplish their specific tasks or goals.

On the other hand, system software is responsible for managing and controlling the overall operation of the computer system. It includes programs like operating systems, device drivers, utility programs, and system libraries. System software provides an interface between the hardware and the application software, manages resources such as memory and file systems, and provides essential services and functions to enable the operation of other software and hardware components.

User Interaction: Application software is typically designed to be used directly by end users to perform specific tasks. It provides a user-friendly interface that allows users to interact with the software and accomplish their goals. Application software is often designed with features and functionalities that are specific to the tasks it is intended to perform, and it is typically installed and used by individual users or organizations.

System software, on the other hand, is not directly used by end users. It operates in the background and provides services and functions that are essential for the overall operation of the computer system. System software is typically installed and managed by system administrators or IT professionals, and it is responsible for managing and controlling the computer system as a whole.

Scope: Application software is focused on specific tasks or applications and is typically limited in its scope. It is designed to perform a specific function or set of functions, such as word processing, spreadsheet calculations, or photo editing.

System software, on the other hand, has a broader scope and is responsible for managing and controlling the entire computer system. It provides services and functions that are essential for the operation of the computer system as a whole, including managing hardware resources, providing an interface for application software, managing system configurations, and handling system-level tasks.

Customizability: Application software is often designed to be customizable by end users. It may provide options for users to configure settings, personalize preferences, and adapt the software to their specific needs or requirements.

System software, on the other hand, is typically not customizable by end users. It is designed to provide essential services and functions that are required for the overall operation of the computer system, and its configuration and management are typically handled by system administrators or IT professionals.

In summary, application software and system software are distinct types of software that serve different purposes in a computer system. Application software is focused on specific tasks or applications, is used directly by end users, and is typically customizable. System software, on the other hand, manages and controls the overall operation of the computer system, is not used directly by end users, and is typically not customizable.

Q3. A tree topology connects a number of star networks through a central cable." Do you agree? Explain advantages and disadvantages of trec topology.

Ans. No, the statement "A tree topology connects a number of star networks through a central cable" is not accurate. Tree topology, also known as hierarchical topology, is a network topology in which nodes are organized in a hierarchical structure with a central root node and multiple levels of branching nodes. It does not connect star networks through a central cable.

In a tree topology, the root node is connected to one or more child nodes, and each child node can have its own child nodes, forming a hierarchical structure resembling a tree. Data flows from the root node to the leaf nodes (end nodes) and vice versa, following the hierarchical structure.

Advantages of Tree Topology:

Scalability: Tree topology allows for easy expansion by adding or removing nodes as needed, making it scalable for growing networks.

Centralized control: The root node in a tree topology provides a centralized control point, allowing for efficient management and control of the network.

Point-to-point connections: Each node in a tree topology has a dedicated point-to-point connection with its parent node, which can provide reliable and efficient data transmission.

Disadvantages of Tree Topology:

Single point of failure: The root node in a tree topology serves as a single point of failure. If the root node fails, the entire network may become inaccessible.

Limited flexibility: Tree topology may not be as flexible as other topologies like mesh or ring, as data flow is restricted to follow the hierarchical structure, and adding or removing nodes may require reconfiguration of the entire network.

Cost: Tree topology may require additional cabling and infrastructure to connect multiple levels of nodes to the root node, which can increase the overall cost of the network.

Performance: Tree topology may suffer from decreased performance if the network becomes too deep or if there is high traffic between nodes at different levels, as data has to pass through multiple levels of nodes before reaching its destination.

In conclusion, while tree topology offers scalability and centralized control, it also has limitations such as a single point of failure, limited flexibility, cost implications, and potential performance issues. The choice of network topology should be based on the specific requirements and constraints of the network environment.

OR

Q3. Compare client-server computing architecture with peer-to-peer computing architecture.

Ans. **Client-Server Computing Architecture:**

Client-server computing architecture is a model where clients, typically end-user devices such as computers or mobile devices, request services or resources from servers, which are powerful computers or systems that provide those services or resources. In a client-server architecture, the client and server have distinct roles and responsibilities, and they communicate over a network to exchange data and perform tasks.

Key features of client-server computing architecture:

Centralized control: Servers are responsible for providing services or resources, and clients request and utilize those services. Servers are typically centrally managed and control access to resources.

Specialized roles: Servers have specialized roles such as file servers, database servers, web servers, or application servers, and clients rely on these servers to fulfill their requests.

Scalability: Client-server architecture allows for scalability, as servers can be added or upgraded to handle increasing demands from clients.

Reliability: Servers are typically designed to be highly reliable with redundancy and backup mechanisms to ensure availability and data integrity.

Security: Client-server architecture often includes security measures such as authentication, access controls, and data encryption to protect data and resources.

Peer-to-Peer Computing Architecture:

Peer-to-peer (P2P) computing architecture is a model where nodes, typically end-user devices, are both clients and servers, and they share resources and services directly with each other without relying on a central server. Each node in a P2P network can act as both a client and a server, and they can initiate requests and respond to requests from other nodes in the network.

Key features of peer-to-peer computing architecture:

Decentralized control: P2P architecture does not rely on a central server for resource or service provisioning. Each node can directly access and share resources with other nodes in the network.

No specialized roles: In P2P architecture, all nodes are equal and have similar capabilities, and they can act as both clients and servers.

Dynamic network: P2P networks can be dynamic, with nodes joining or leaving the network at any time without affecting the overall operation of the network.

Scalability: P2P architecture can be highly scalable as the addition of new nodes can potentially increase the resources and services available in the network.

Q4. What do you mean by a database system? How is it different from traditional file system?

Ans. A database system refers to a software system that is designed to store, organize, manage, and retrieve data in a structured and efficient manner. It consists of a database management system (DBMS) that provides tools and services for creating, modifying, and managing databases, and an associated set of application programs that interact with the DBMS to perform various data-related tasks.

A traditional file system, on the other hand, is a method of organizing and storing data in files and folders without the use of a centralized DBMS. In a file system, data is typically stored in separate files, and there may be limited or no standardization in terms of data structure, data relationships, or data integrity. File systems are commonly used in personal computers, where data is stored in files and folders on the local disk or external storage devices.

Here are some key differences between a database system and a traditional file system:

Data Structure: In a database system, data is organized into structured formats such as tables, with defined relationships between tables and enforced data integrity rules. In a file system, data is stored in files and folders, without any standardized structure or relationship between files.

Data Integration and Consistency: A database system allows for integration of data from multiple sources, and provides mechanisms to ensure consistency and integrity of the data across the database. In a file system, data is typically stored in separate files, and integrating and maintaining consistency across files may require manual effort.

Data Access and Retrieval: A database system provides powerful and efficient methods for querying, retrieving, and manipulating data, such as SQL (Structured Query Language) for relational databases. In a file system, data retrieval and manipulation may require custom file operations or programming, and may be less efficient and flexible compared to a database system.

Data Security: Database systems typically provide robust mechanisms for data security, such as authentication, authorization, and encryption, to protect data from unauthorized access or modification. File systems may have limited or no built-in security features, and may rely on operating system or file-level permissions for data security.

Scalability and Performance: Database systems are designed to handle large amounts of data and concurrent users, and provide performance optimization features such as indexing, caching, and query optimization. File systems may have limitations in handling large datasets and concurrent access, and may not provide similar performance optimization features.

Data Redundancy and Backup: Database systems often provide mechanisms for data redundancy, replication, and backup to ensure data durability and availability. File systems may require manual efforts for data backup and redundancy, and may have limitations in terms of data durability and availability.

In summary, a database system is a centralized software system that provides structured data management with standardized data organization, integrity, retrieval, and security features, while a traditional file system is a less organized method of storing and managing data in separate files and folders. Database systems are typically more robust, scalable, and efficient for handling large amounts of data and concurrent users, compared to traditional file systems.

OR

Q4. Explain the following types of relationship with example:

(a) One-to-one

(b) One-to-many

(c) Many-to-many.

Ans. (a) One-to-one relationship: In a one-to-one relationship, one entity in a relationship is uniquely associated with only one entity in another relationship, and vice versa. It means that each entity in one relationship corresponds to exactly one entity in the other relationship.

Example: A person and their passport can have a one-to-one relationship. Each person can have only one passport, and each passport can belong to only one person. This relationship is unique, and each person's passport is associated with only one person, and vice versa.

(b) One-to-many relationship: In a one-to-many relationship, one entity in a relationship is associated with multiple entities in another relationship, but each entity in the other relationship is associated with only one entity in the first relationship. It means that each entity in one relationship corresponds to multiple entities in the other relationship.

Example: A university and its students can have a one-to-many relationship. One university can have multiple students enrolled, but each student can be enrolled in only one university. In this case, the university is associated with multiple students, but each student is associated with only one university.

(c) Many-to-many relationship: In a many-to-many relationship, multiple entities in one relationship are associated with multiple entities in another relationship. It means that each entity in one relationship corresponds to multiple entities in the other relationship, and vice versa.

Example: A bookstore and its customers can have a many-to-many relationship. A customer can purchase multiple books from the bookstore, and each book can be purchased by multiple customers. In this case, multiple customers are associated with multiple books, and multiple books are associated with multiple customers, creating a many-to-many relationship between the two entities. To represent a many-to-many relationship in a database, a junction table or an intermediary table is often used to keep track of the relationships between the entities.

Security challenges: P2P architecture can present security challenges, as nodes directly share resources with each other, and it may require additional measures to ensure data integrity, authentication, and access control.

Limited reliability: P2P architecture may not be as reliable as client-server architecture, as it relies on the availability and reliability of individual nodes in the network.

In conclusion, client-server and peer-to-peer computing architectures have different approaches to resource sharing, control, scalability, and security. Client-server architecture is characterized by centralized control and specialized roles for servers, while peer-to-peer architecture is characterized by decentralized control and equal capabilities for all nodes. Both architectures have their advantages and disadvantages, and the choice between them depends on the specific requirements and constraints of the system or network being designed.