DCS SHEET#1

1. Define Distributed Computer System DCS? What are the pros and cons of Distributed system over centralized system?

   - A: **DCS is a collection of autonomous computer systems that are physically separated but are connected by a centralized computer network that is equipped with distributed system software**.

2. What are the purposes of distributed systems? What are the fundamental components of distributed computing systems?
   A: **A distributed system is a collection of independent computers that appear to the users of the system as a single computer**

3. Draw example of centralized, decentralized, and distributed systems and compare between their advantages and disadvantages.
   A: **Centralized systems rely on a single point of control, providing simplicity but risking a single point of failure.**

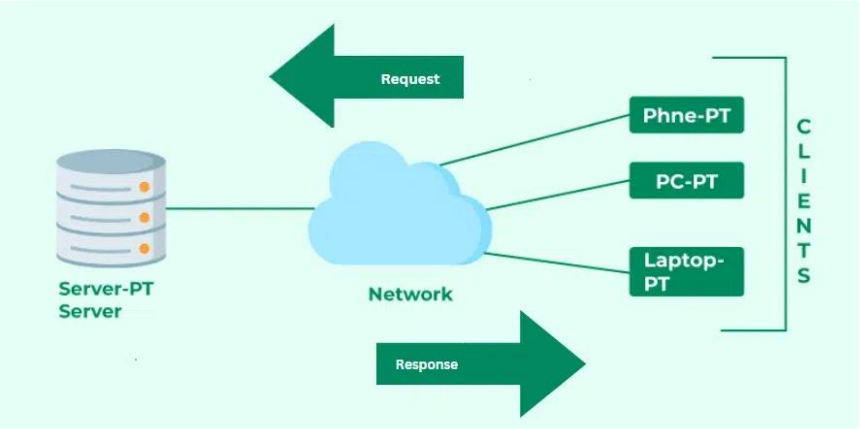
   - **Decentralized systems distribute control among multiple nodes, enhancing fault tolerance and scalability.**
   - **Distributed systems further spread resources across multiple locations, optimizing performance and reliability**.

4. Explain the technologies that move the computer system systems from centralized to distributed.

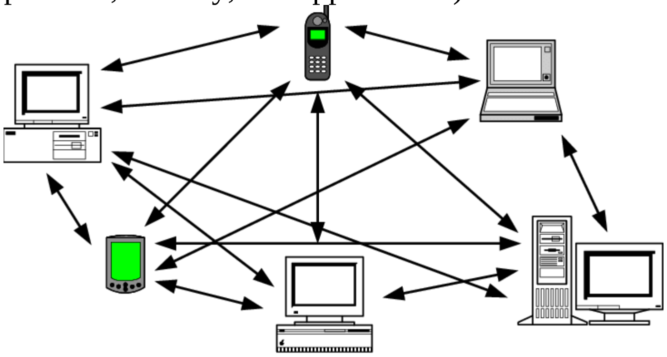
A: 1- **Microcomputer and microprocessor technology**
**2- Evolution of Local Area Network LAN Technology**
**3- PCs Technology**
**4- Need for multimedia for decision making**

5. Draw a sketch of client-server model? What key elements (client, server, services, network protocol, security, and applications) characterize client-server model?

A:

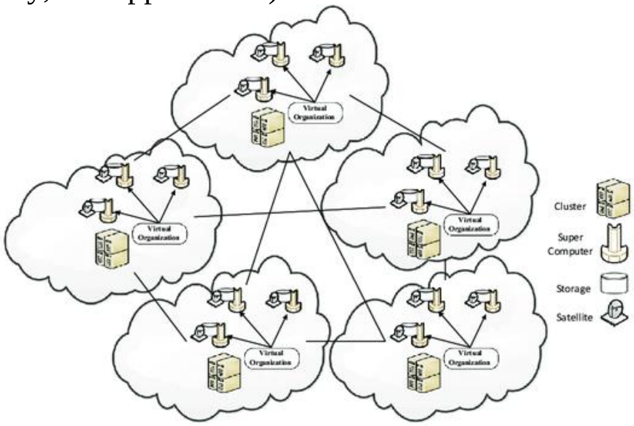


6. Draw a sketch of peer to peer model? What key elements (client, server, services, network protocol, security, and applications) characterize this model?



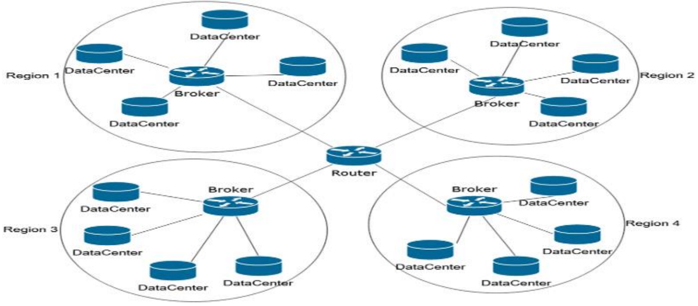
A:

7. Draw a sketch of grid computing model? What key elements (client, server, services, network protocol, security, and applications) characterize this model?



A:

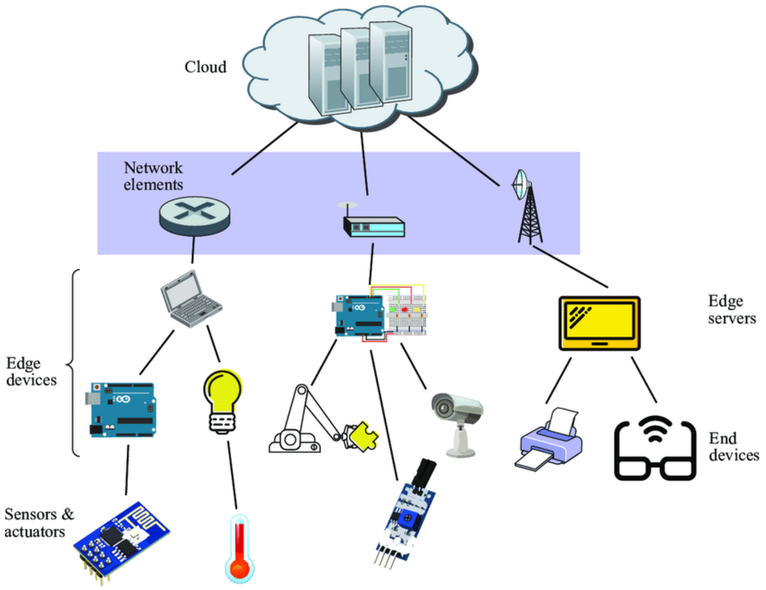
8. Draw a sketch of data center and cloud computing model? What key elements (client, server, services, network protocol, security, and applications) characterize this model?



A:

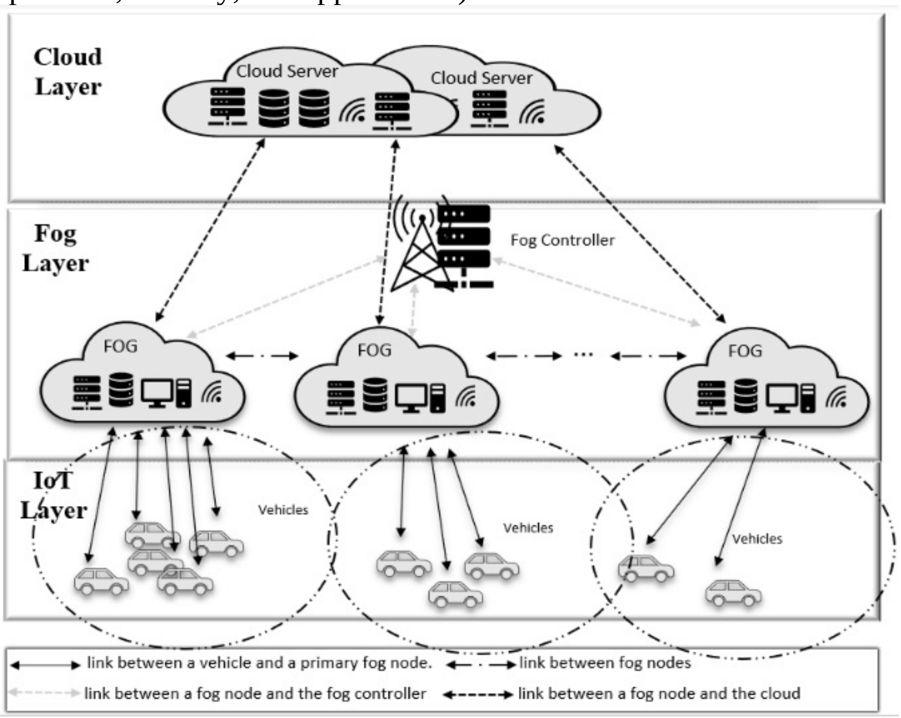
9. Draw a sketch of edge computing model? What key elements (client, server, services, network protocol, security, and applications) characterize this model?

A:



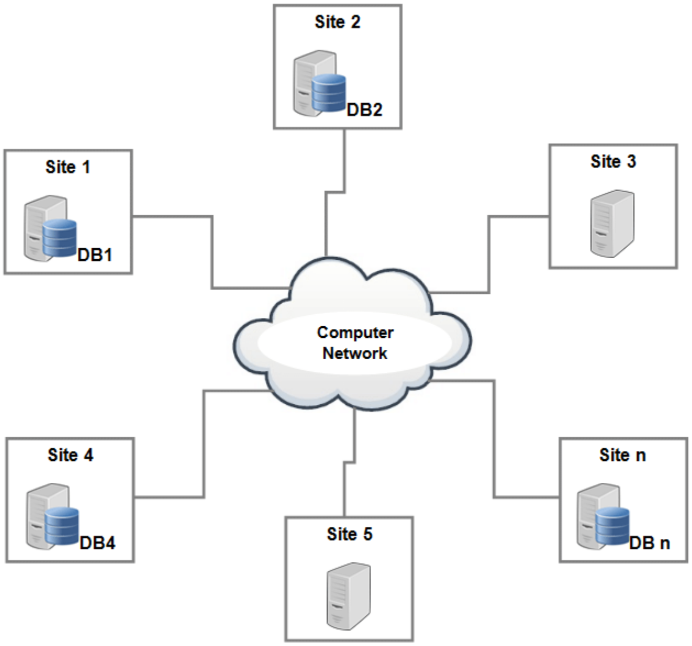
10. Draw a sketch of fog computing model? What key elements (client, server, services, network protocol, security, and applications) characterize this model?

A:

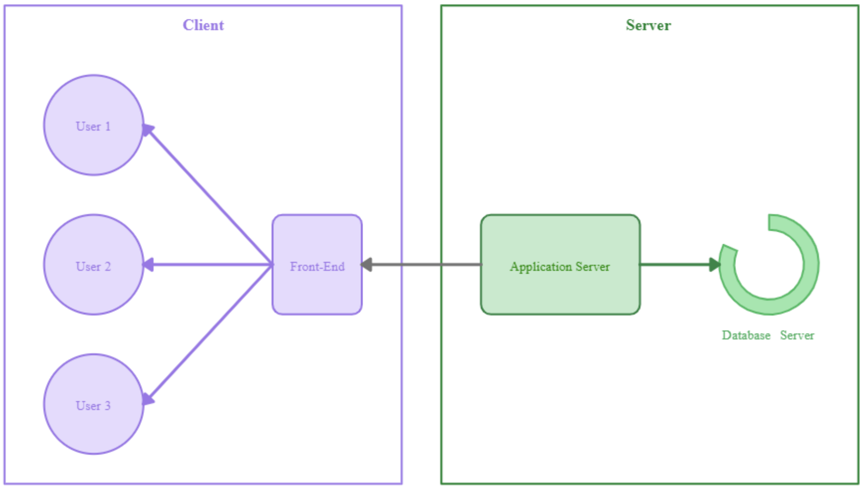

Cloud Layer
Cloud Server
Cloud Server
Fog Layer
Fog Controller
FOG
FOG
FOG
IoT Layer
Vehicles
Vehicles
Vehicles
link between a vehicle and a primary fog node.
link between fog nodes
link between a fog node and the fog controller
link between a fog node and the cloud

11. Draw a sketch of distributed database computing model? What key elements (client, server, services, network protocol, security, and applications) characterize this model?

A:



12. Draw a sketch of client-server model? What key elements (client, server, services, network protocol, security, and applications) characterize client-server model?



A:

13. Identify the various challenges in the distributed system in detail.

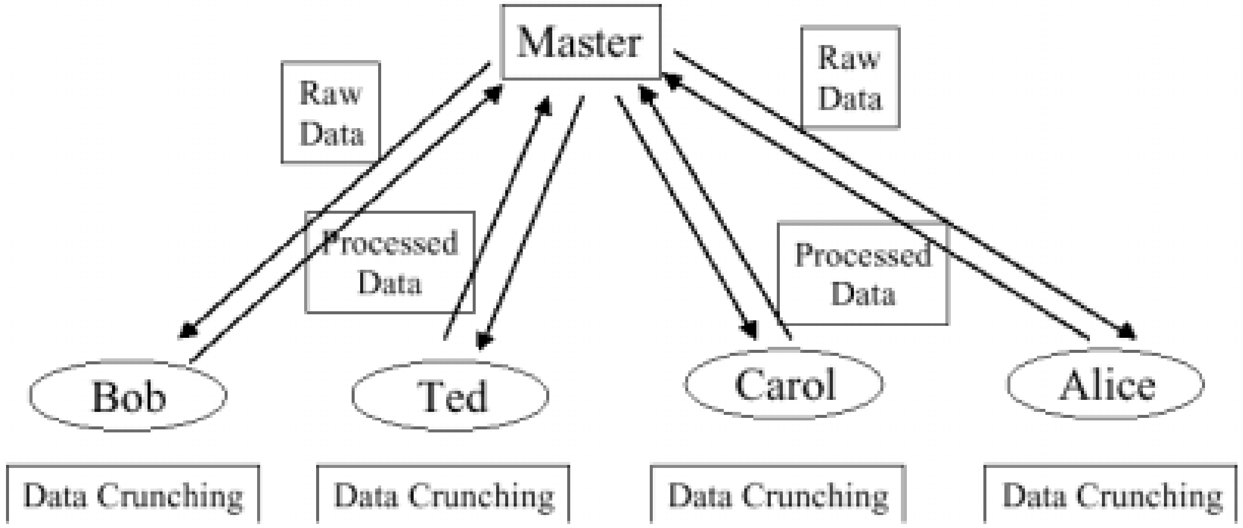
1. **A: Heterogeneity:**
    - **Imagine a party where everyone speaks different languages, dances to different beats, and wears mismatched shoes. That's heterogeneity in distributed systems.**
    - **What It Is: Differences in hardware, software, or network configurations among nodes.**
    - **Why It Matters: When computers don't speak the same language (literally or metaphorically), communication and coordination become a tango of confusion.**
    - **Solution: Common standards and agreed-upon protocols—like teaching everyone the same dance steps.**
    -
2. **Scalability:**
    - **Picture a restaurant that handles 10 customers smoothly but turns chaotic when 100 hungry souls arrive. Scalability is the art of keeping the kitchen running smoothly, no matter the crowd.**
    - **What It Is: Ensuring consistent performance as workload increases. A program shouldn't break a sweat whether it has 10 nodes or 100.**
    - **Why It Matters: As distributed systems grow, factors like size, geography, and administration come into play.**
    - **Solution: Clever load balancing, auto-scaling, and a dash of magic.**
3. **Openness:**
    - **Imagine a secret club where members refuse to share their secret handshake. In distributed systems, openness is about playing nice with others.**
    - **What It Is: Interoperability and openness to new components or services.**
    - **Why It Matters: Systems evolve, and new players join the dance floor. Being open ensures seamless integration.**
    - **Solution: APIs, standards, and a welcoming attitude.**
4. **Transparency:**
    - **Think of transparency as the invisible cloak worn by Harry Potter. Distributed systems should hide their complexity from users.**
    - **What It Is: Concealing the underlying complexity—users shouldn't know (or care) about the distributed magic.**
    - **Why It Matters: Users just want their cat videos; they don't need to see the server farm.**
    - **Solution: Abstraction layers and a sprinkle of user-friendly interfaces.**
    -
5. **Concurrency:**
    - **Imagine a crowded marketplace where everyone wants the same limited-edition sneakers. Concurrency is managing those shoe-hungry shoppers.**
    - **What It Is: Multiple actions happening simultaneously—reads, writes, updates—all jostling for attention.**

- **Why It Matters: Without proper coordination, chaos ensues. Data consistency becomes a game of musical chairs.**
- **Solution: Clever locking mechanisms, transaction management, and a referee for fair play.**
-

6. **Security:**
   - **Picture a treasure map with X marking the spot. In distributed systems, security is guarding that treasure from pirates.**
   - **What It Is: Protecting data, ensuring authentication, and keeping the bad actors out.**
   - **Why It Matters: Distributed systems are juicy targets. Hackers love a good challenge.**
   - **Solution: Encryption, firewalls, and a trusty parrot (okay, maybe not the parrot).**
   - 
7. **Failure Handling:**
   - **Imagine a tightrope walker who occasionally slips. In distributed systems, things break—it's inevitable.**
   - **What It Is: Dealing gracefully with failures—whether it's a server hiccup or a network glitch.**
   - **Why It Matters: Users shouldn't see the circus behind the scenes. The show must go on!**
   - **Solution: Redundancy, fault tolerance, and a safety net.**

14. Prepare a neat sketch of resource sharing and explain with suitable example.

A:

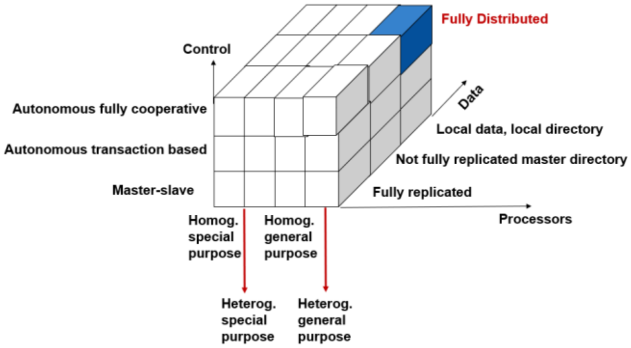


15. Provide some examples of distributed system.
A: **Finance and Commerce: Amazon, eBay, Online Banking, E-Commerce websites.**

**Information Society: Search Engines, Wikipedia, Social Networking, Cloud Computing.**
**Cloud Technologies: AWS, Salesforce, Microsoft Azure, SAP.**
**Entertainment: Online Gaming, Music, YouTube.**
**Healthcare: Online patient records, Health Informatics.**
**Education: E-learning.**
**Transport and logistics: GPS, Google Maps.**
**Environment Management: Sensor technologies.**

16. A definition for a distributed system is that of a collection of independent computers providing the view of being a single system, that is, it is completely hidden from users that there even multiple computers. Give two examples to explain this.
A: **1- Internet and Web Services (Wikipedia)**
**2- Cloud Computing (AWS)**

17. Draw and explain Enslow model for distributed processing systems
A:



18. What is the role of middleware in a distributed system?
A: **Middleware is a set of services that enables applications and end-user to interact with each other across a heterogeneous distributed system.**

19. Explain what is meant by (distribution) transparency and give examples of different types of transparency.
A: Transparency hides the fact that its processes and resources are physically distributed across multiple computers.