Distributed Database Management System Architecture

College Name: Dr.Sudhir Chandra Sur Institue of Technology & Sports Complex

Name: Arijit Bera

Roll No: 25500123046

Sec: A

Branch: CSE

Year: 3rd

Sem: 6th

Sub: Distributed Database Management System(PECIT601B)

Introduction to Distributed DBMS

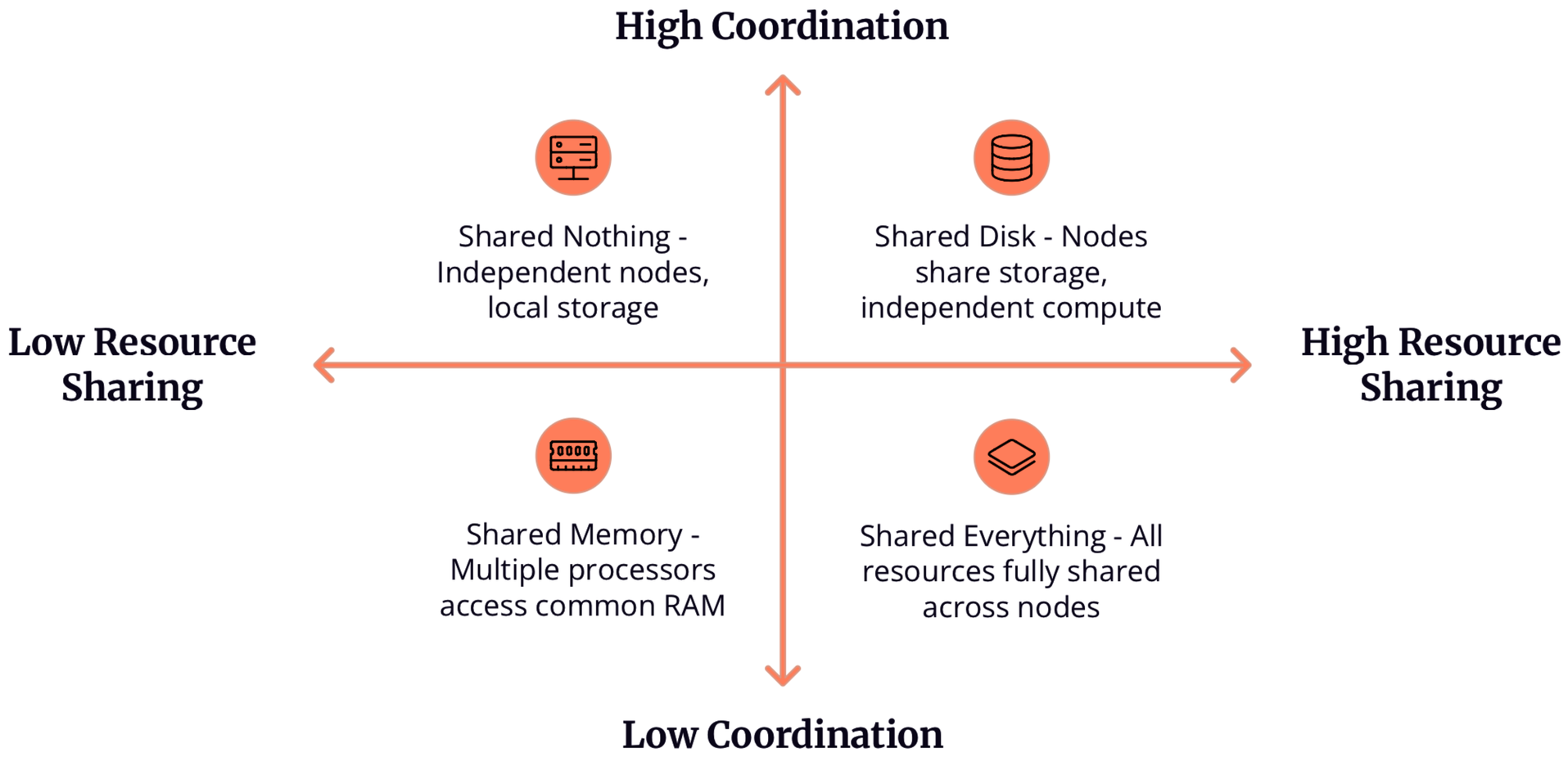
A **Distributed Database** is a collection of multiple, logically interrelated databases physically distributed across different sites. These sites are interconnected via a robust network, allowing seamless data access.

A **Distributed DBMS (DDBMS)** is the software system that manages this complex arrangement, providing a transparent view to users, making it appear as a single, unified database. This transparency is crucial for ease of use.

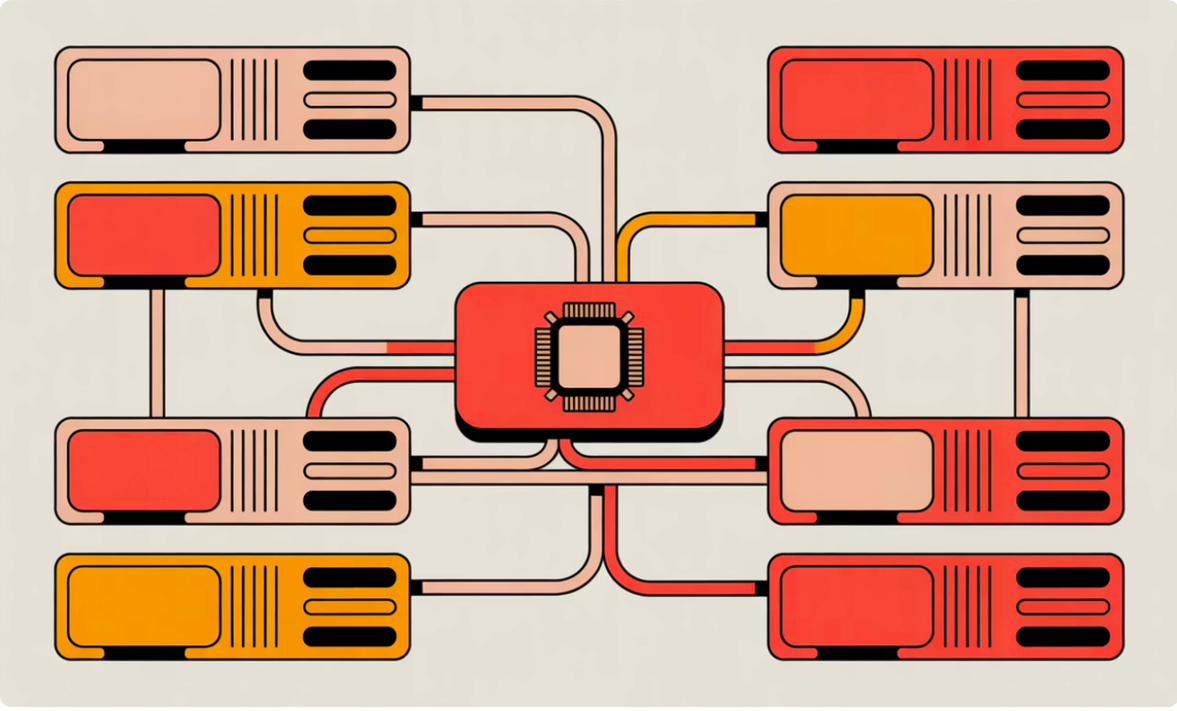
The primary goals of a DDBMS include ensuring fault tolerance, enabling high scalability, and guaranteeing data availability across geographically dispersed locations. Imagine banking branches spread across Jaipur and Amritsar, all accessing the same virtual database seamlessly.

Architectural Models Overview

The architecture of a DDBMS dictates how processing units (CPUs) access shared resources and coordinate their operations. This choice significantly impacts performance, scalability, and complexity.



Shared Nothing Architecture



In a **Shared Nothing Architecture**, each node operates with complete autonomy, possessing its own dedicated CPU, memory, and disk storage. Communication between nodes occurs exclusively over the network, making each unit independent.

This model offers significant advantages, including high scalability and superior performance, as adding new nodes directly expands processing and storage capabilities. It is easier to scale out horizontally.

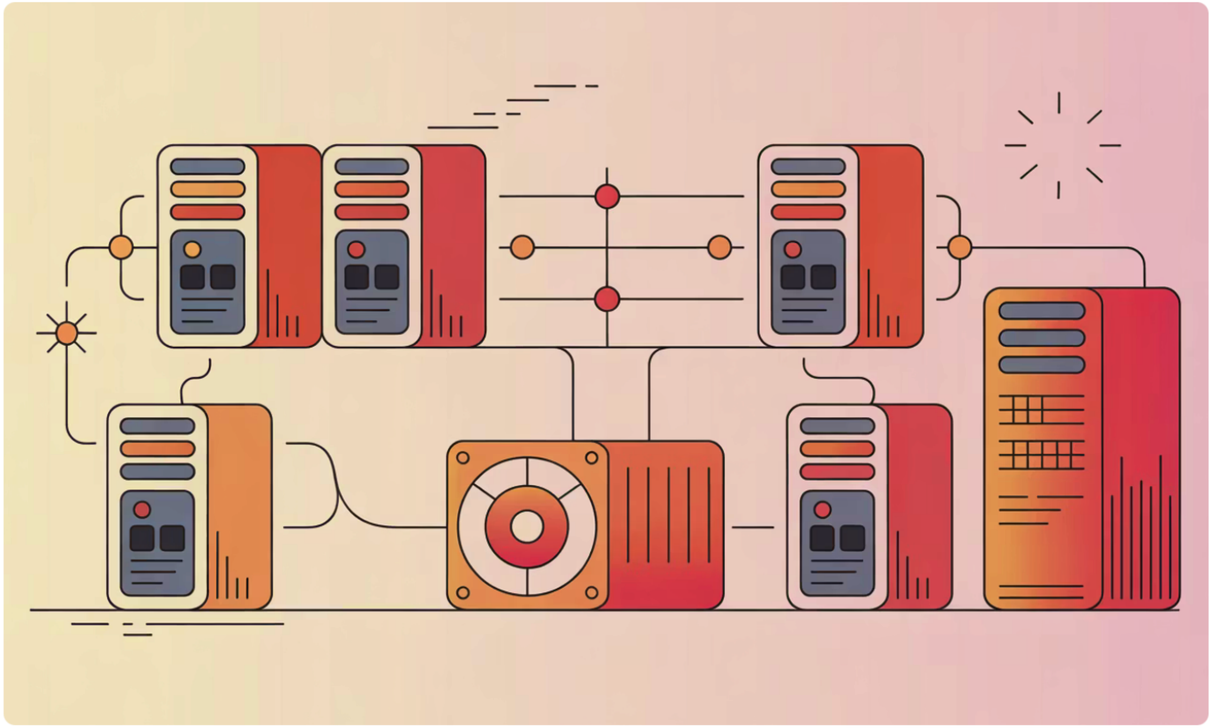
However, maintaining data consistency and coordinating complex transactions across independent nodes presents considerable challenges. This architecture is common in large-scale distributed DBMS and cloud platforms, where data is often partitioned by ranges or hash across various nodes.

Shared Disk Architecture

The **Shared Disk Architecture** allows all processing nodes to access a single, shared logical disk over a network, while each node retains its private memory. This separation enables independent scaling of compute and storage layers.

This model facilitates the creation of cloud-based data lakes and serverless DBMS, where compute resources can be dynamically scaled based on demand without affecting data storage. An example includes multiple servers accessing a shared Storage Area Network (SAN).

A crucial requirement in this architecture is robust coordination mechanisms to maintain cache coherence and ensure data consistency across all nodes, preventing data anomalies.

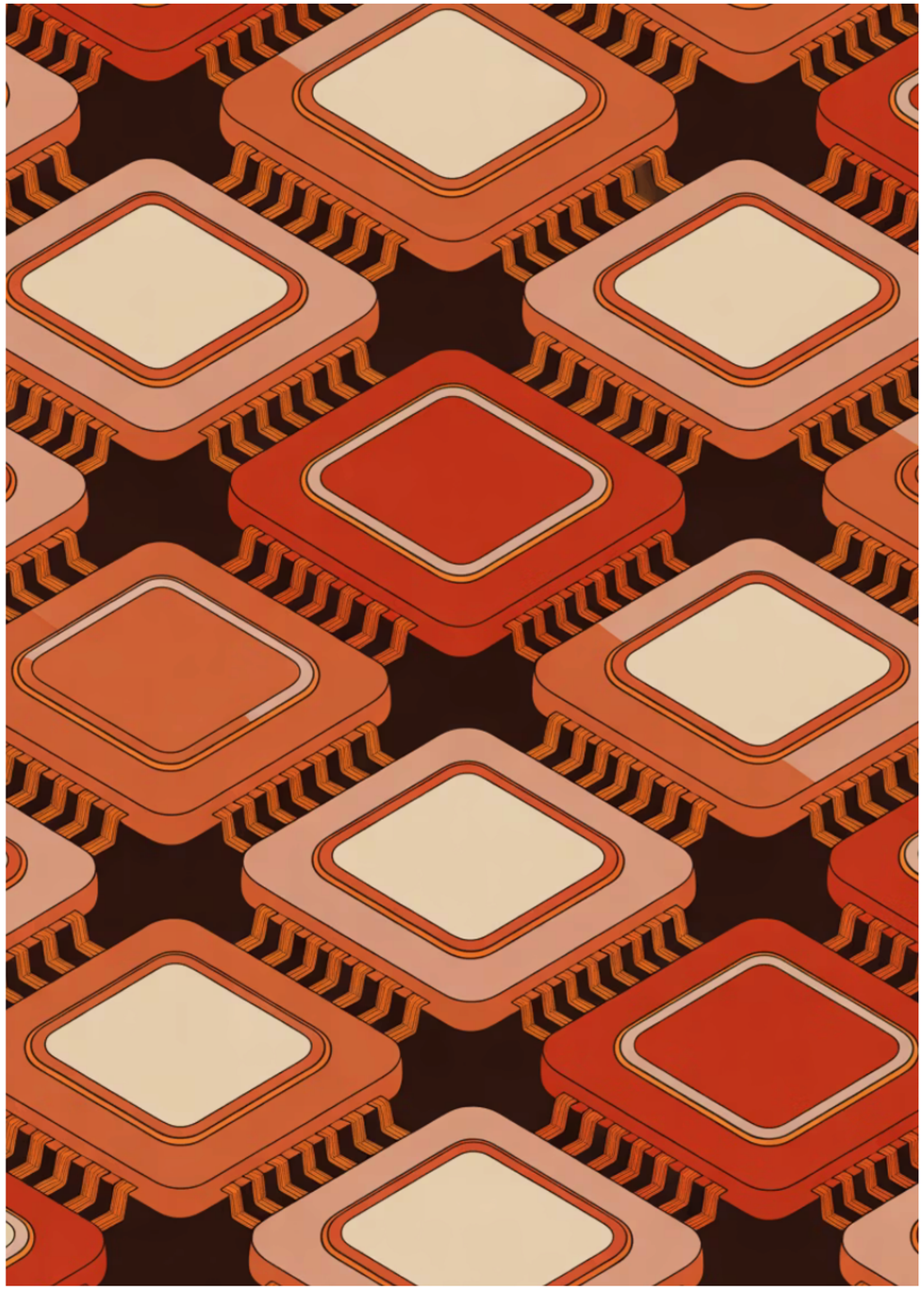


Shared Memory Architecture

In a **Shared Memory Architecture**, multiple CPUs share a common memory address space and disk. This configuration provides a global view of in-memory data structures, enabling rapid data access and inter-process communication.

However, this architecture is rarely employed in large-scale distributed DBMS due to the prohibitive cost and complexity associated with implementing fast, high-bandwidth interconnects required for efficient memory sharing across physically separate machines.

It is more commonly found in tightly coupled multiprocessor systems, such as symmetric multiprocessing (SMP) systems, rather than geographically distributed environments. Its limitations in scalability make it less suitable for truly distributed database solutions.



Design Issues and Conclusion

Key Design Challenges

- Data transparency
- Concurrency control
- Fault tolerance
- Query optimization

DDBMS must seamlessly hide data distribution from users.

Architecture Popularity

The Shared Nothing architecture is the most popular choice for modern distributed systems, primarily due to its inherent scalability and ability to handle massive datasets.

Application Selection

Understanding these architectural paradigms is crucial for selecting the right system for diverse applications like banking, e-commerce, and big data analytics, ensuring optimal performance and reliability.