NAME: ABRAR ZAHIN ZAWAD
ID: 21201577
COURSE: CSE449

# CRITICAL REVIEW REPORT
# DISTRIBUTED DATABASES

## ABSTRACT

Distributed database systems increase communication and processing by distributing data over multiple network nodes. Data access is faster, there is less risk of failure, and users have local control over their data.

# INTRODUCTION

In today's information-dependent world, companies need access to their databases for employees, customers, suppliers, and vendors. A centralized database can be managed well but poses problems like a centralized site downtime and expensive communication costs. A distributed database design is proposed, where data is distributed among cities with their own computer and data storage facilities. This allows location transparency, where users can query the database

without knowing its location. This complexity can be managed by a distributed database management system (DBMS).

## DEFINITION

A distributed database (DDB) is a collection of interconnected databases distributed over a computer network, managed by a distributed database management system (DDBMS). DDBS integrates DDB and DDBMS, merging database and networking technologies, creating a system that appears as a single machine to the user.

1. A distributed database management system (DDBMS) is a software that manages and makes the distribution of a database transparent to users. It is the combination of a distributed database (DDB) and a distributed database management system (DDBS), merging database and networking technologies. DDBs are similar to distributed file systems in that they facilitate access to distributed data but have distinct differences in structure and functionality.

2. Distributed file systems allow users to access files on machines other than their own, with no explicit structure or relationships between data. DDBs are organized according to a schema,

providing a simple interface for users to access, read, write, and close files. They also offer high-level query capability, transaction management, and integrity enforcement.

3. Distributed DBMS provides transparent access to data, extending the concept of data independence to environments where data is distributed and replicated over multiple machines connected by a network.

# DDBS ARCHITECTURE

1. THE HARDWARE
2. THE SOFTWARE

## THE HARDWARE

DBBS design becomes more complex and sophisticated due to its extended functionality. At the physical level, centralized and distributed systems have multiple computers called sites connected via a communication network. Networks can have several types of topologies that define how nodes are physically and logically connected. One popular topology used in DDBS is the client-server architecture, which defines specialized servers with specific functions such as printer server, mail server, file server, etc. These servers are connected to a network of clients that can access the services of these servers.

The server-client architecture requires some kind of function definition for servers and clients. The DBMS functions are divided between

servers and clients using different approaches. A common approach used with relational DDBS is centralized DMBS at the server level. The client refers to a data distribution dictionary to decompose the global query into multiple local queries.

There are different architectures of DDBS for the two main types: the file server approach, the database server or DBMS server approach, the two-tier client/server, and the three-tier approach. The file server approach involves sending the entire file from the server to the client computer, while the DBMS server approach splits processing between the client and the server, reducing network data traffic and handling security and concurrency control.

The three-tier approach involves client PCs, application servers, and database servers. Clients handle local screen and keyboard interaction, but application servers rely on database servers and their databases to supply the necessary data.

One simple idea in distributing data is to disperse six tables among the five sites, which benefits include local autonomy, efficient local transactions, and cost-effective joins. However, problems include expensive joins and security concerns.

Replication of databases at each site offers better availability and minimizes telecommunications costs. However, it also has issues with security, concurrency, and consistency. Selective replication involves replicating all tables at headquarters and only replicating each table once in the network. This approach has downsides, such as frequent table usage and a bottleneck at headquarters. To avoid these issues, heuristics include placing copies of tables at heavily used sites, ensuring at least two copies of important tables, limiting table copies, and avoiding bottlenecks.

## THE SOFTWARE

A DDBS consists of three software modules: server software for local data management, client software for distribution functions, and communications software for communication. Advantages of this architecture include efficient division of labor, resource scaling, better price/performance, and familiar tools. Disadvantages include server bottlenecks, single point of failure, and difficulty in database scaling. A preferred feature is distribution transparency, allowing users to issue global queries without worrying about global distribution.

## FRAGMENTATION, REPLICATION

In DDBS, logical unit distribution and allocation may be more efficient by splitting tables into smaller fragments and allocating them in different sites, with three different types available-

## HORIZONTAL FRAGMENTATION

Figure shows the addition of a horizontal fragmentation table, G, to demonstrate the operation of splitting G into partitions based on employees, enhancing efficiency in management, queries, and transactions, but requiring a union on all partitions.


## VERTICAL FRAGMENTATION

Vertical partitioning divides a table's columns among multiple cities, each containing the primary key attribute. This arrangement is useful for processing different functions, like salary and skills attributes. However, it requires a query involving the entire table G, which requires requesting all portions from all sites and joining them.

## HYBRID FRAGMENTATION

This fragmentation scheme divides a table into blocks based on requirements, allocating each fragment to a specific site, making it complex and requiring more management.

## QUERY PROCESSING

DDBS adds processing costs to conventional centralized databases due to additional hardware and software design for data transfer over the network. Database designers focus on query optimization to minimize data transfer costs. One method is simijoin, where a relation sends the entire join-column to a target relation, reducing data transfer costs and improving query efficiency.

# CONCURRENCY AND RECOVERY

The design of a Distributed Database System (DDBS) must consider various aspects beyond centralized DBS, such as multiple data copies, communication link failures, individual site failures, distributed commits, and deadlocks on multiple sites. Two suggestions are provided for managing concurrency control.

## DISTINGUISHED COPY OF A DATA ITEM

Three variations of the data deduplication system (DDBS) are the primary site technique, primary site with backup site, and primary copy technique. The primary site technique controls all data unit locks and unlocks, but has downsides like overloading and single point failure. The primary site with backup site designates a backup site for failure, and the primary copy technique distributes load to designated sites.

## VOTING

This method does not assign a specific coordinator for data unit locking. When a site attempts to lock a copy, requests must be sent to all sites, and if not granted, the transaction fails.

## RECOVERY

Identifying the failure type and site, managing distributed recovery involves database logs, update protocols, and transaction failure recovery protocols.

# RESEARCH IN DDBS

This section presents current research in DDBS, specifically focused on distributed query optimization.

## DISTRIBUTED QUERY PROCESSING USING ACTIVE NETWORKS

This paper presents an efficient method for implementing query processing on high-speed active WANs. It studies traditional criteria for distributed query optimization and devises a new criteria for approaching DDBS query optimization based on the different characteristics of low and high-speed networks. Transmission delay is considered the dominant factor in the communication cost function in conventional networks, and many distributed query processing algorithms are devised to minimize the amount of data transmitted over the network. In high-speed networks, latency (as well as local processing time and disk I/O) becomes significant cost factors.

The paper proposes a new algorithm for executing distributed queries, which minimizes not only the amount of transferred data but also the number of messages on the network (latency). The algorithm involves

a query Q requiring a join among relations R1 at site S1, R2 at site S2, …, and Rn at site Sn on a common attribute A. The query Q involves retrieving DV (R1.A) from disk, setting up a point-to-multipoint unidirectional connection, sending Q together with DV (R1.A) to site Si, then tearing down this connection. At each site Si, logically ANDing DV(Ri.A) to create JV, using DVI(Ri.A) to read participating tuples, Ri.A', in JV order, shipping Ri.A' to site R1 along the same connection set up used in step c, then tear down each connection, and merging join Ri.A' to get the final result.

## DISTRIBUTION OPTIMIZATION

This paper presents two problems in Distributed Database Systems (DDBS) data distribution optimization and introduces their solutions. The first problem is one-to-many database segmentation, where a large database is partitioned into multiple segments, ensuring each table appears in one and only one segment. The problem can be solved by minimizing the number of possible pairwise dependencies and the number of segments possible, such as one 4-table segment and two 3-table segments.

The second problem is many-to-one database segmentation, where data is distributed across a network and the designer is interested in a subset of the data. A decision point occurs when a preferred subset of n data sources meets the criteria of containing all desired data elements, containing the smallest number of data sources needed to provide all data elements, and belonging to the set of non-inferior solutions on the Pareto design frontier.

A decision problem is formulated via mathematical programming (MP) where the design variables become decision variables. Adding a set of constraints, the MP technique proceeds to find a combination of data elements and data sources that identify an optimal solution, such as a smallest-cost solution. Multiple criteria in distributed database design can include cost, performance, reuse/sharing of data sources, and flexibility of configuration.

The different waits can be varied to get different possible solutions, then to compare and select the best fit the designer criteria, given that the selected design is feasible and applicable.

## FRAGMENT ALLOCATION IN DDBS

This paper focuses on the fragment allocation problem in a distributed database. It aims to find the optimal number of copies per fragment and the optimal allocation of fragments into different sites. The optimality measures include minimal cost, performance, and CPU and I/O time. The ultimate goal is to allocate fragment copies to sites with minimal total communication cost. The study uses heuristic algorithms to approach the optimization problem, considering factors such as storage and processing capacity, network information, and transaction frequency.

## A PROBE-BASED TO OPTIMIZE JOIN QUERIES IN INTERNET DDB

This paper addresses the optimization problem of join operation in Internet DDBS by proposing an adaptive solution. The paper uses a data transfer cost analysis to compare the two alternatives, comparing fixed transfer cost and transfer cost per byte. The RTO algorithm

estimates the term C1.Sj0r and C1.Sr, dividing the join operation into sub-operations and tracking the statistics of the current sub-operations. The algorithm adjusts the plan if needed, as some queries can take a long time due to changing internet conditions. The paper also presents an adaptive solution for dynamically optimizing the join operation, ensuring accurate comparisons between the two alternatives.

## FUTURE WORK

Our current DBMS is centralized, but we plan to upgrade to the new DDBMS HDBE_net® using peer-to-peer architecture, simplified concurrency control, communication protocol, and semi-joint.

## CONCLUSION

This study explores distributed database design (DDBS) fundamentals and alternatives, addressing issues like architecture, distribution, query processing, and concurrency control. It also discusses research on query optimization, distribution optimization, fragmentation, optimization, and join optimization on the internet. The paper aims to attract readers to the advantageous side of DDBS, discuss software architecture, fragmentation, replication, and recovery aspects, and attract researchers for new scope.