Building a Distributed Database Using Go

Jessica Ayman Naeam,
Enas Ikram Girgis,
Marly Amgad Hamdy,
Moreen Mohsen Wagheem,
Abram Ashraf Abd-ElSayed,
Kerolos Nashaat Fakhry

May 14, 2025

1 Introduction

This project aimed to design and implement a basic distributed database system using the Go programming language. The primary objectives were to explore core concepts of distributed systems, such as data replication, fault tolerance, and master-slave architecture, and to provide a functional prototype demonstrating these principles.

2 System Architecture

The system follows a master-slave architecture, comprising one master node and multiple slave nodes.

2.1 Master Node

- Manages the creation of databases and tables.
- Executes queries and replicates data to slave nodes.
- Uses TCP for communication with slaves.

2.2 Slave Nodes

- Independently store data and execute queries.
- Receive replicated data from the master.
- Restricted from executing database creation or deletion operations.

Communication between nodes is handled via TCP, ensuring reliable message delivery.

3 Design Decisions

Data Replication: Implemented a mechanism where the master node sends executed queries to all registered slave nodes to maintain data consistency.
Operation Restrictions: Slave nodes are restricted from performing operations like CREATE and DROP to maintain centralized control over the database schema.
TCP Communication: Chose TCP over other protocols for its reliability in ensuring message delivery between nodes.

4 Challenges

Data Synchronization: Ensuring data consistency between the master and slave nodes, especially during network failures or message delays, was challenging. Implemented retry mechanisms and acknowledgment confirmations to address this.
Fault Tolerance: Ensuring system continuity in case of node failures required implementing data replication strategies and periodic health checks of nodes.
System Performance: As the number of nodes and queries increased, optimizing system performance became essential. Improved replication strategies and reduced the load on the master node to enhance performance.

5 Conclusion

The project successfully demonstrates a basic distributed database system using Go, incorporating essential features like data replication, fault tolerance, and a master-slave architecture. While the system is functional, future enhancements could include implementing consensus algorithms for better fault tolerance and scalability.