CSCI 5408

FEASIBILITY STUDY

Distributed Database Management System

Professor:

Dr. Saurabh Dey

Submitted on:

February 10, 2022

Submitted by:

Group DPG 4

Hardee Rameshchandra Garala (B00869195)

Manali Shah (B00890746)

Parvish Vijay Gajjar (B00912090)

Shaik Asaduddin (B00894318)

Shiva Shankar Pandillapalli (B00880049)

Table of Content

Introduction

Building our own Distributed database management system gives us a better understanding of how storage systems, synchronization of tasks, and queries work in the existing database management system like MySQL workbench. The following feasibility studies highlight the feasibility of building a lightweight distributed database management system. The study includes the amount of time it will take to build this project; Design and description of the architecture to be used for the project; Development environment, Testing strategy and Deployment environment for the project; Individual strengths in the team; Personal learning that we are expecting from this project; Challenges we will face while developing this project; And, additional features that we think would help users have a better experience at managing their distributed database.

High-level Architecture

Design architecture of the project is the basic requirement before any project can be developed, designed, or managed. It acts as a road map for the development of software or project. The project of the Distributed database management system has different modules to be developed and interactions between these modules can be understood by the design architecture. There are three major components in this architecture:

1. Database Management System/Server
2. Database/Persistent Storage
3. Virtual machines/Users

The server is composed of numerous components that all work together to provide the best data management system possible.

The first component is registration - to access and use the system's features, the user must first register. The user profile is stored in both virtual machines once registration is complete. All registered users can then log in to the system and view their information, increasing the system's legitimacy and security.

If verified users successfully log in, they will be provided with a set of features to choose from. These options include writing queries to access specific data, exporting the data in the form of the data dump, obtaining a data model in the form of an entity-relationship diagram based on the tables in the schema, and performing analysis on query searches.

When the query option is selected, the user can enter a query for data insertion, retrieval, or deletion. When you enter a query, the parser checks it for semantic errors and syntax errors. If it

encounters one, it throws an error; otherwise, depending on the syntax, this query is interpreted as a query or a transaction and processed accordingly.

The logger will record the duration of various query processing operations as well as database metadata at that time. It will also log any updates or modifications made to the database, as well as any database events. For this project, we will consider 2 virtual machines initially, which can retrieve data from two different synchronous databases.

The high-level architecture for the distributed database management system is below:

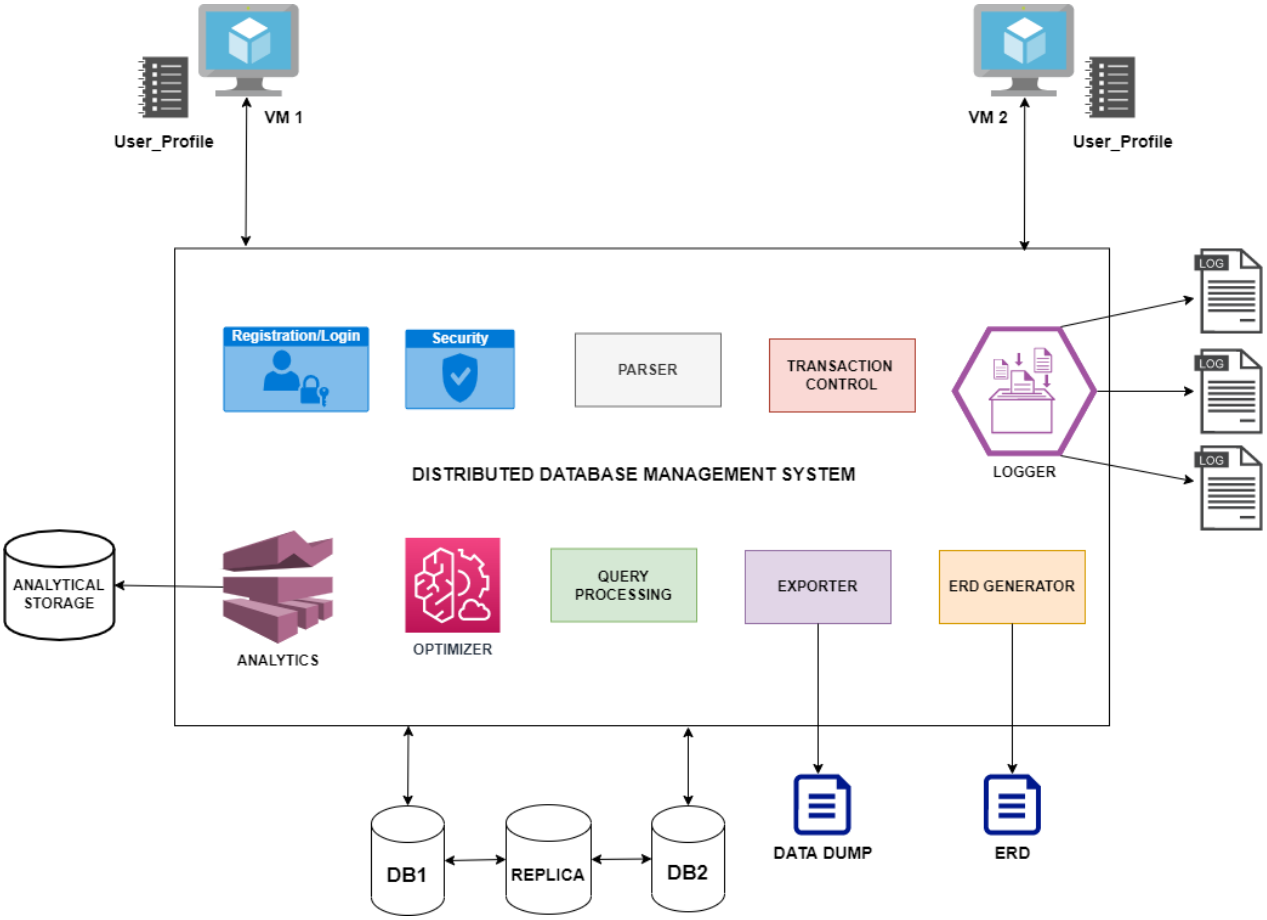

Figure 1 High-level architectural diagram for distributed DBMS

Feasibility of Project and Timeframe

Building a distributed database system that supports multiple users is a complex task. However, it is achievable if planned carefully and worked properly on the tasks created for the whole project. For such an enormous project, it is a bit difficult to build and deploy all the modules of the project in a single go. Hence, the entire team has decided to build the entire project in 3 phases by segregating the whole modules into three parts which increase the focus. The description and plan for the entire project which is divided into phases are explained below.

Phase-1: In this phase, we majorly focus on the major functionality of the project. The major functionality includes the following items listed below.

- **Query processing manager** – is the module that should be given priority as the database system is accessed and interacted with the data by the end-user via the query processing manager. Without a query processing manager, we cannot imagine a database system.
- **User Interface and Security** – The user interface is one of the main modules of the product as it is the face for the whole project where the end-user interacts with the product. Hence, we need to start building the project from the user Interface module and spread it across all other important modules mentioned in this list.
- **Overall Database Design** – The overall design is the next important part of this phase as we are building a distributed database system, we need to implement the core of the database that can support the functionality in a distributed computing model without compromising on the security of the data.

Because phase 1 focuses primarily on core functionality, we decided to implement these three modules because all of the other modules rely on them, and without the core functionality, we cannot implement the other modules even if we rush to develop them.

Phase-2: In this phase, we focus on the intermediate functionality of the project where implement all the functionalities that are must for a database to work as a distributed database system with almost all the functionalities and we also plan to fix the bugs found during the testing of phase-1. This will give us an ample amount of time in fixing the errors and changing the strategies if something goes wrong with the implementation of phase 1. The following are the modules we decided to develop in this phase.

- **Transaction Processing manager** – is a module that deals with the queries that could be converted into a transaction which needs to be executed as one query at a time without committing the data to the files. Without the transaction processing manager, it is extremely hard to perform many tasks on the database that needs the help of transactions.

- **Data Modelling (Reverse Engineering)** – is a feature of the project that can represent the entire database in a plain text file with all the cardinality and relations between the columns and the tables. Implementing this module requires the implementation of the relationships between the tables and the columns at the back. Implementing the relationships takes a considerable amount of time as we will be dealing with the raw data directly in the file system.
- **Exporting Structure and Values** – is like the data modeling module. However, the major changes come in this module is that here we do not export only the structure of the data, we also get our hands on the data as well as structure. The user should be able to import the database file in entirely in another database that is exported from our distributed database. That sets our goal for phase 2.
- **Another round of Regression & Bug fixes found while testing the project during phase-1** – Fixing the problems found in earlier stage makes our project more robust and will reduce the rework in most of the cases. Hence, we spend some time fixing the bugs found during the complete regression test performed at the end of phase 1.

Phase-3: In this phase, we focus on the remaining modules as we are almost done with most of the functionality, and we need to focus on the features part of the project. For any project, features are very crucial as they get the impression from the end user. For example, able to import the data from a standalone database to distributed database system etc.

- **Log Management** – A log is particularly important for the database to know what is happening in the database. It captures almost all the events happening inside the database and gives us a clear understanding of what is happening and who is performing on what. Hence, we make it the primary focus during the development in phase 3.
- **Database Analytics** – Database analytics is a module that covers the analytics part of the database which gives the clear statistics of the queries and transaction performed on the database. It gives the clear-cut information about which user has interacted with the database and what operations/queries that the users have performed on the data. It talks even more clearly about on which table and which database has been modified etc., details. The information extracted from this module is important to take multiple decisions that might be helpful in making the database climb a step from the current version.
- **Final Regression Testing** – During this final regression, we focus on figuring out most of the problems not found during the first two phases and we focus on fixing those issues as soon as possible.
- **Implementation of Additional features** – Any additional feature added to this feasibility study could be taken into the backlog to work on it once the whole project is developed as per the requirements.

Development Environment

- IntelliJ (Version: 2021 3.2)
- Java Development Kit [Version 17 – Most recent yet stable version (link: https://openjdk.java.net/projects/jdk/17)]
- Test Library (JUnit 5)
- Supports all operating systems (Windows, Linux and Unix)
- Gitlab (Version Control Tool)
- Network capabilities with a minimum of 4MB of download and upload speed

Deployment Environment

The distributed database management system will be deployed on two separate Virtual machines on the Google cloud platform. The following configuration is for one machine which could be used for the other virtual instance as well. For now, the configuration described below is for the E2-mediumly configured instance.

- A minimum of 2 virtual CPU
- A minimum of 4 Gigabytes of RAM
- A minimum of 20 Gigabytes of Storage space
- Operating system: Either Windows, Linux, or Unix(mac).
- Network capabilities with a minimum of 4MB of download and upload speed
- Java 17

Since it is on cloud, we would like to go with CentOS, one of the light weight yet powerful operating system which have the capabilities of Fedora for high end processing yet light weight just like Red Hat Enterprise Linux.

Group Strengths

- Core Java
- Database Intermediate Queries
- Git (Version Control Tool)
- Java Collections Framework
- Data structures

Learnings

While building this project based on the given problem statement, we as a team will be developing individual proficiency in writing efficient and understandable code. Developing each module of the project would give us in depth knowledge about how distributed database is built, how data is stored in it, where it is stored, how to manage each queries synchronization while they are in transaction, and last but not the least how to keep the data secure.

Challenges

- Creating the core for the distributed databases
- Synchronization and consistency in transactions
- Security
- Developing Data dictionary

Testing Strategies

The goal will be to check whether the system delivers required specifications and to maintain system quality by finding defects and fixing them. Testing will include unit testing, integration testing and black box testing approach at the end

1. **Unit testing**

   Unit testing will be performed during the implementation phase and maximum coverage will be aimed at covering all the units in a module with a focus on code quality. JUnit 5 will be used to perform unit testing for the system.

2. **Integration testing**

   This will be an extension to the unit testing by combining multiple units and often mocking few dependencies. Like unit testing, JUnit5 will be used here to perform this type of testing.

3. **Black box testing**

   Each feature and module will be tested manually by the team with a focus on performance, acceptance criteria, latency, and security. This stage will not focus on the code and will be performed after implementation phase. The test cases will cover every step and feature of all the modules and will make sure that all the modules work as expected.

Meeting log

	Agenda	Participants
Date: February 01, 2022 **Duration:** 30 minutes 33 seconds	• Get to know teammates. • Initial and high level discussion on each module of the project. • Deliberation on the group's strengths. ***To-do:*** Each member to study in detail about each module.	1. Hardee Rameshchandra Garala 2. Manali Shah 3. Parvish Vijay Gajjar 4. Shaik Asaduddin 5. Shiva Shankar Pandillapalli
Date: February 08, 2022 **Duration:** 2 hours 17 minutes	• In-depth discussion of each module, as well as the collective perception of each member about that module. • Creating high level architectural design. • Discussion on development and deployment environments. • Discussion on testing strategies. • Discussion on learning and challenges. ***To-do:*** Sections of the feasibility document were divided for each member to write on.	1. Hardee Rameshchandra Garala 2. Manali Shah 3. Parvish Vijay Gajjar 4. Shaik Asaduddin 5. Shiva Shankar Pandillapalli
Date: February 10, 2022 **Duration:** 1 hour, 48 minutes	• Breaking down the project's development into phases. • A thorough discussion of how to approach each project module.	1. Hardee Rameshchandra Garala 2. Manali Shah 3. Parvish Vijay Gajjar 4. Shaik Asaduddin 5. Shiva Shankar Pandillapalli

References

[1] "Distributed Database Architecture," [Online]. Available: https://docs.oracle.com/cd/E18283_01/server.112/e17120/ds_concepts001.htm.

[2] "Distributed DBMS - Database Environments," [Online]. Available: https://www.tutorialspoint.com/distributed_dbms/distributed_dbms_database_environments.htm .