MAJOR PROJECT 1

MID-TERM REPORT

ON

Relational Database With Minimal Functionality

Submitted By

Prakash Tiwari	Akshit Chauhan	Gaurav Singh
500062116	**500062444**	**500062611**

Under the guidance of

Amit Singh

Assistant Professor, SoCSE

Department of Cybernetics,

School of Computer Science

UNIVERSITY OF PETROLEUM AND ENERGY STUDIES

Dehradun-248007

October-2020

Abstract:

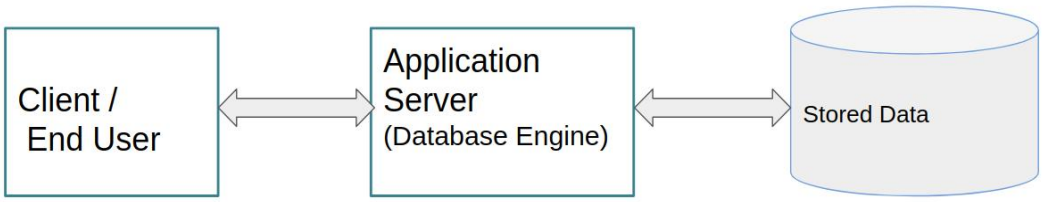
There are many applications out there today that use some sort of local storage format to keep data collocated with the application. However, without a database, most of the proposed solutions cannot handle the atomicity of multiple changes (all or nothing, and the management of transactions that may impact each other), and there is no built-in recovery mechanisms in case of a failure during the transaction. Here we propose a project to design a relational database with minimal functionality.

A relational database is a collection of data items with pre-defined relationships between them. These items are organized as a set of tables with columns and rows. Tables are used to hold information about the objects to be represented in the database. Advantages of relational model are simplicity, structural independence, ease of use,query capability, data independence, scalability.

Introduction:

A collected information which is in an organized form for easier access, management, and various updating is known as a database. The relational database was invented by E. F. Codd at IBM in 1970. The data in relational databases is present in tabular form, i.e. as a *collection* of tables with each table consisting of a set of rows and columns) and it provides relational operators to manipulate the data in tabular form.

1. **Database Engine:** A database engine (or storage engine) is the underlying software component that a database management system (DBMS) uses to create, read, update and delete (CRUD) data from a database. Most database management systems include their own application programming interface (API) that allows the user to interact with their underlying engine without going through the user interface of the DBMS.



2. Database storage structures:

Databases may store data in many data structure types. Common examples are the following: ordered/unordered, flat files, hash tables, B+ trees, ISAM and heaps.

In our project we are using B+ tree. These are the most commonly used in practice. A B+ tree is an m-ary tree with a variable but often large number of children per node. A B+ tree consists of a root, internal nodes and leaves. The root may be either a leaf or a node with two or more children. The primary value of a B+ tree is in storing data for efficient retrieval in a block-oriented storage context in particular, file systems. This is primarily because unlike binary search trees, B+ trees have very high fanout (number of pointers to child nodes in a node,typically on the order of 100 or more), which reduces the number of I/O operations required to find an element in the tree. Time taken to access any record is the same because the same number of nodes is searched. Access in B+ tree is fast.

Problem Statement:

There is always a need to store, retrieve and update data efficiently. Also remove data redundancy and inconsistency. In this project Relational Database will solve the problem of redundant data and inconsistency. For performing efficient operation in the database it is designed using B+ tree.

Objective:

Creating A Relational Database to perform CRUD operations.

Sub Objective:

- Store data internally using B+ tree and creating caching mechanism for fast retrieval.
- Apply hashing and indexing in the database engine to find data quickly.

Objective Achieved:

- We have achieved validation of query(Create, Delete, Insert, Select) and extraction of information from query that user enters.
- We are able to store the information that user enters in a cache from which execution of query will take place.

Literature Review:

Since the 90's, databases have shown tremendous growth. This growth can be determined in different aspects. Different demands of every era give databases a new bunch of challenges. To achieve those challenges, researchers come up with different ideas and combinations. These various combinations enhance features of databases and this way databases start evolving from one period to another. Database that we had in 1960 is entirely, absolutely different from what we have now.[3] In this phase of evaluation The relational database was invented by E. F. Codd at IBM in 1970. In a relational database, all data is held in tables, which are made up of rows and columns. Each table has one or more columns, and each column is assigned a specific datatype, such as an integer number, a sequence of characters (for text), or a date. Each row in the table has a value for each column. A relational database usually contains many relations, with tuples in relations that are related in various ways,A *relational database* is a type of database. It uses a structure that allows us to identify and access data *in relation* to another piece of data in the database[1]. The main advantage of relational databases is that they enable users to easily categorize and store data that can later be queried and filtered to extract specific information for reports. Relational databases are also easy to extend and aren't reliant on physical organization. After the original database creation, a new data category can be added without all existing applications being modified. [2]

Methodology:

Software Development Model:

In our project we are using the Waterfall model to build our project. Waterfall model is the oldest software development model and all other software development models are formed on the basis of the Waterfall model. There are six phases in Waterfall model

1) Feasibility Study: The feasibility study involves understanding the problem and then determines the various possible strategies to solve the problem.

2) Requirement Analysis: The aim of the requirement analysis and specification phase is to understand the exact requirements of the customer and document them properly.

3) Designing: The aim of the design phase is to transform the requirements specified in the SRS document into a structure that is suitable for implementation in some programming language.

4) Coding and Unit Testing: In coding phase software design is translated into source code using any suitable programming language. Thus each designed module is coded.

5) Integration and System Testing: Integration of various modules is carried out incrementally over a number of steps.

6) Maintenance: It includes corrective maintenance, perfective maintenance, Adaptive maintenance.

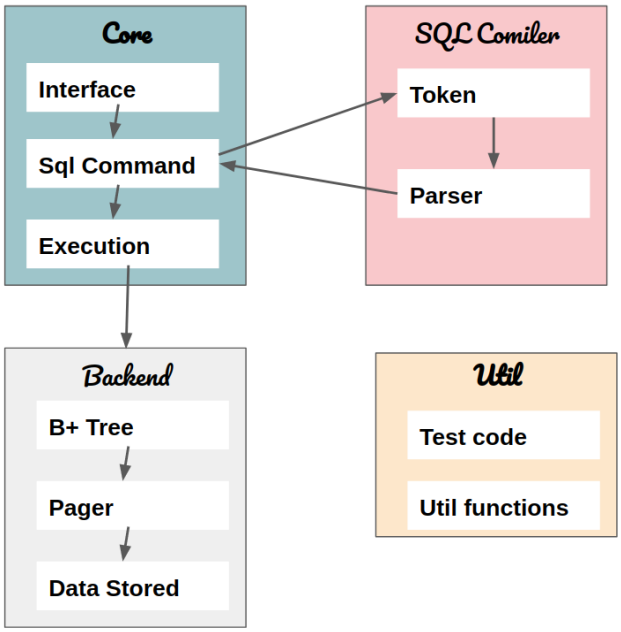
Implementation:

The query entered by the user is firstly checked if it is valid or not. For that the object of ValidateQuery is created and its method isValid() called. Internally for the parsing of the query we are using Hyrise c++ Sql Parser. Which checks the grammar of the query and generates the parsing tree. If the query is valid then we extract the information of the query entered by the user. That includes query type (READ or WRITE), name of table, table field information and where clauses. That info is stored into a cache for the validation of the name of the table if it is Create command. (If table name already exist).

We are using B+ tree to store data. *B+ tree* is an algorithm (i.e. a set of rules) allowing not just to quickly find the entries in a database, but also add / edit and delete these entries (assuming you can use an eraser). We manage a separate tree for each table and a master tree which will help to find the table info. The information of the table will be stored in the leaf of the tree. In which each row will be stored in a single node. By using this we can perform all operation quickly.

Let's assume each table has a primary *key (PK)* - a set of columns uniquely identifying any row there. If some table doesn't, we can always add a so-called *surrogate primary key* - a column storing e.g. a very precise time stamp for any particular row showing the moment it was added. Since any table has its own primary key (PK), we can store its data as a dictionary sorted by its primary key in B+ tree structure.

Architecture of DBPlus:



Commands for run the project:

```
1. mkdir build
2. cd build
3. cmake ..
4. make
5. ./DBPlus
```

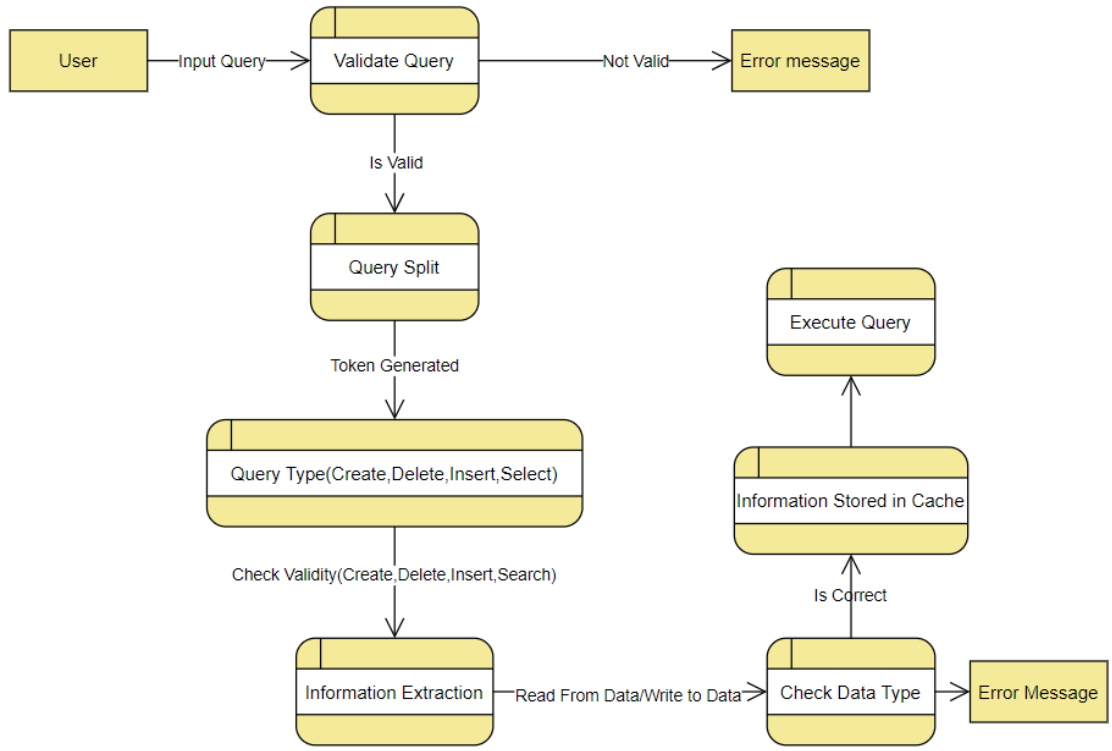
Project Dependencies:

1. Min version of CMAKE 3.0.1 or above

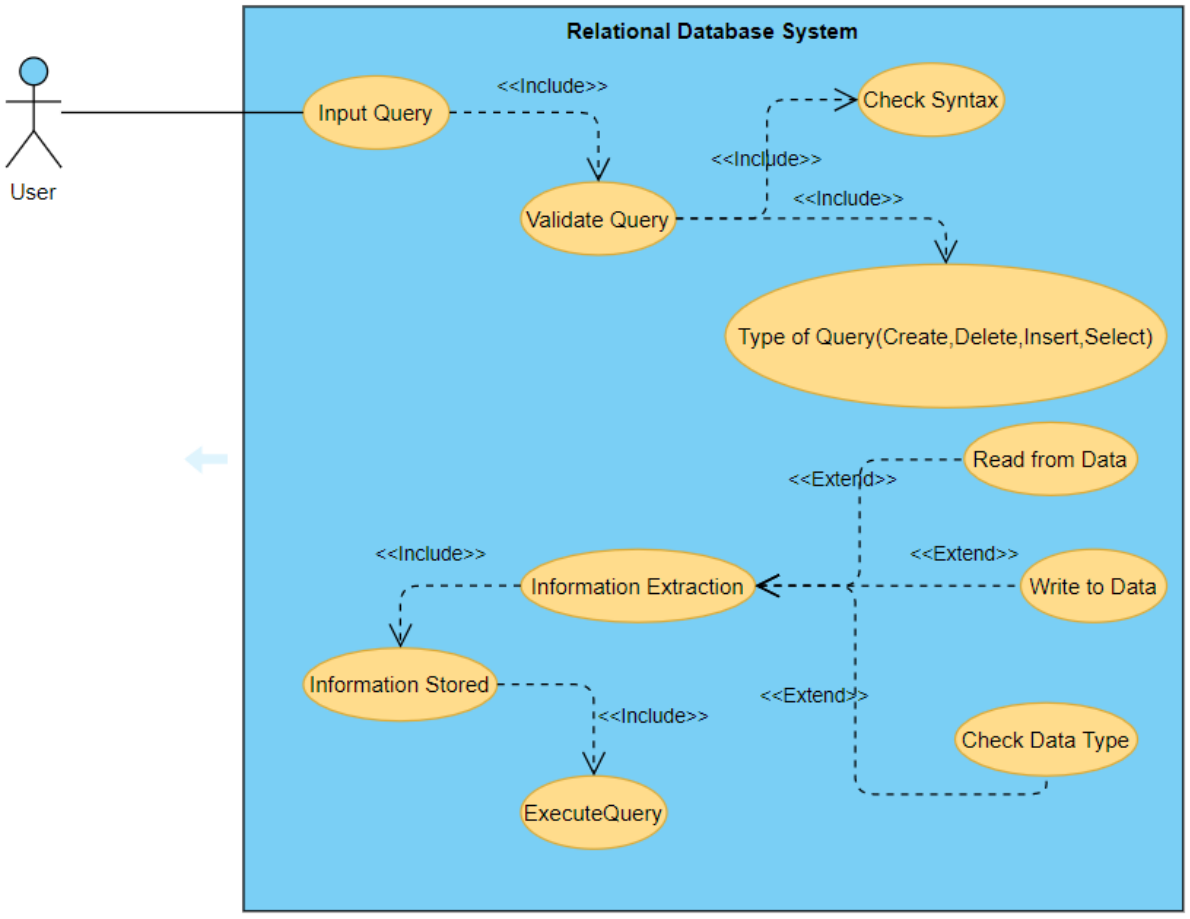
   cmake --version

2. Must be install **Hyrise c++ Sql Parser**
3 .g++ 11 or above

Data Flow Diagram:



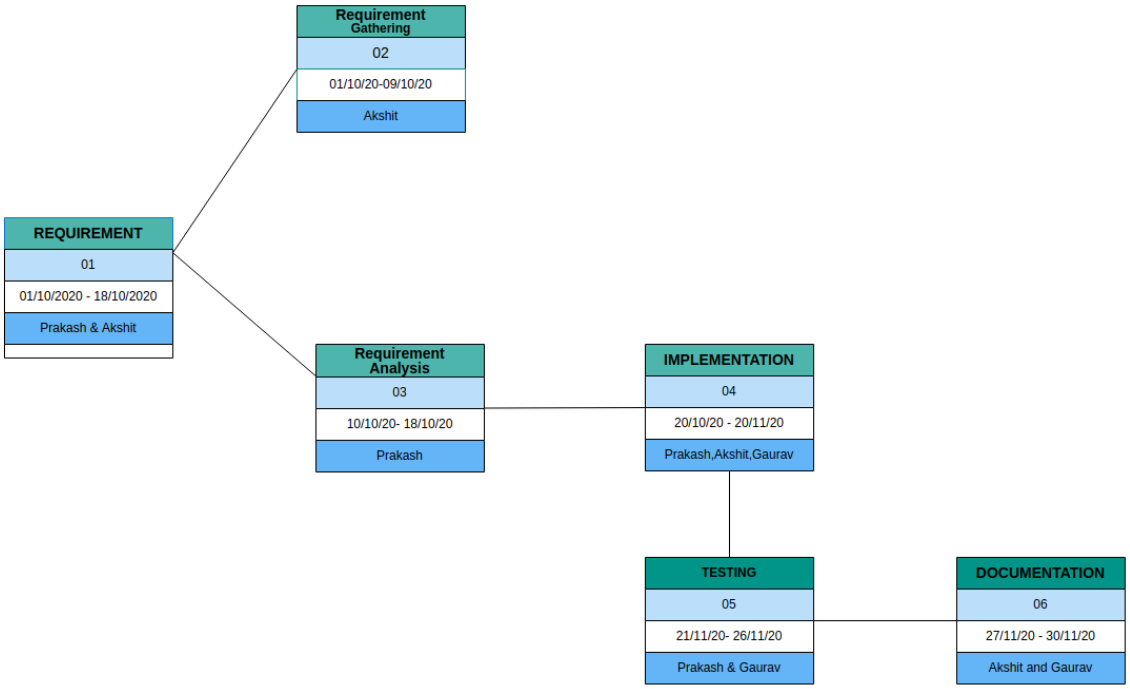
Use Case Diagram:



System Requirement:

Operating System	:	Ubuntu 18.04/Windows
Programming Language	:	C++
Running Environment	:	Terminal
Processor	:	Pentium IV
Disk Drive	:	Floppy or Hard Disk Drive
RAM	:	512 MB (min)

Schedule (PERT Chart):



References:

[1]. Navathe, Ramez Elmasri, Shamkant B. (2010). *Fundamentals of database systems* (6th ed.). Upper Saddle River, N.J.: Pearson Education. pp. 652–660. ISBN.

[2]. Lightstone, S.; Teorey, T.; Nadeau, T. (2007). *Physical Database Design: the database professional's guide to exploiting indexes, views, storage, and more*. Morgan Kaufmann Press. ISBN .

[3]. S Praveen,U Chandra,*A Literature Review on Evolving Database (March 2017)* International Journal of Computer Applications.

[4]. https://en.wikipedia.org/wiki/Database_engine

Approved By

Signature	**Signature**
Mr Amit Singh	**Dr. Monit Kapoor**
Mentor	**Head of Department**