CSE 344
SYSTEM PROGRAMMING
FINAL REPORT

OĞUZHAN SEZGİN
1801042005

General Process

Program works as a desired. In server, main thread takes the argument from command line, then do error check. After that it creates the temporary file for prevent the double instantiation. Then it reads the data file and load this data to memory, then creates the pool threads. After the thread creation main thread creates a socket and waits the connection. When a client connect to the server it takes the connection and gives an empty thread. Main threads do this process until the SIGINT signal arrives. When this signal arrives, it wait for worker thread termination and after that free all resource and terminate itself.

In worker thread, they take the connection from queue and reads the clients query from socket. Then it determines they are reader(select) or writer(update). After the determine the role, they execute the queries and sends response to corresponding client with using same socket. It repeats this processes until clients queries end. If the client queries end, client write exit to socket and worker read this comment and then it waits for the new connection.

In client side, they just read query file and takes corresponding queries then put this queries to queue. Then they connect the server with socket and sends the queries to server and then reads the servers response and print response to screen.

Giving Connection To Threads

Main thread accepts the connection then waits to until a thread become empty. This is producer consumer paradigm. When a thread empty they decrements the counter and unlock the mutex and waits the new connection. Then main thread put the connection descriptor to the queue and waits a new connection or the empty thread.

Execute Queries

The threads which taken the connection they checks the query type. If it is the update query then thread become a writer this means that this thread become alone in the critical section, if query is the select query then they become the reader this means that they can read with other thread. So for this situation I use reader writer paradigm. The reader user shared locks, they can read together but the writer can not update anything together.

Double Instantiation

I used the temporary file for prevent double instantiation. I create a temp file with O_EXCL flag so if the file already exist the program give error and terminate itself this means that if any server is awake, there is temp file and if any other server program wants to execute this file returns the new server program and it wont start.

Data Structures

For database I use linked list. Because I create two pointer for database, one of them points head node node and the other points the tail node. Each node is represents the a row. With this way new row insertion when program read the data file will be O(1) so program will be fast.

The response of the query also is the linked list and I also use same technique with the database insertion, when I scan the database if the data required then program adds the tail of the response. Each row represents the one row(record) so response size can be determine easy.

Main thread and worker threads uses the queue for connection descriptor, main thread add new connection to queue and workers take the first added connection with this way clients waits less.

Clients saves the their queries in the queue also, when they connect the socket they take query from the queue and sends it to the server.

Writing Big Data to Socket

I solve this problem with determine response data size. As I said before each node represent a row and this row numbers are the response size. Worker thread sends the size of response before the sending response. With this way client knows how many record will he/she read then go in to a loop with worker thread. Worker write response row by row and client read the the row by row with use socket.