МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра защиты информации

*Изображение выглядит как текст, Шрифт, снимок экрана, логотип

Автоматически созданное описание*

**ПРАКТИЧЕСКАЯ РАБОТА №2**

**«**Разработка сетевого интерфейса системы управления базой данных.**»**

**по дисциплине: «*Программирование*»**

|  |  |
| --- | --- |
| Выполнил:  Студент гр. «АБс-324», «АВТФ»  *Деревянкин Дмитрий Александрович*  «17» января 2025г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | Проверил:  *доцент кафедры ЗИ*  *Архипова Анастасия Борисовна*  «\_\_\_» \_\_\_\_\_\_ 2025г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |

Новосибирск 2025

**Задание:** требуется реализовать сетевой интерфейс для СУБД из практики 1.

При запуске приложения из первой практики СУБД должна ожидать соединение по протоколу tcp на порту 7432. При подключении требуется обработать запрос либо в отдельном потоке, либо в отдельном процессе, либо асинхронной задачей и отдать результат в ответ на запрос.

Также необходимо позаботится о блокировках на структуре БД если используются потоки или процессы.

Модули программы

## headers:

includes.h

#pragma once

using namespace std;

#include <iostream>

#include <fstream>

#include <sstream>

#include <cstring>

#include <thread>

#include <mutex>

#include <chrono>

#include <unistd.h>

#include <arpa/inet.h>

commands.h

#ifndef commands\_h

#define commands\_h

#include "includes.h"

#include "list.h"

#include "json.hpp"

int countingLine (string& fin); // ф-ия подсчёта строк в файле

string fileread (string& filename); // Производим чтение из файла // чтение из файла

void filerec (string& filename, string data); // запись в файл

struct BaseDate {

string BD; // название БД

int rowLimits; // лимит строк

SinglyLinkedList<string> nametables; // названия таблиц

SinglyLinkedList<string> stlb; // столбцы таблиц

SinglyLinkedList<int> fileindex; // количество файлов таблиц

struct Filter { // структура для фильтрации

string table;

string colona;

string value;

string logicOP;

bool check;

};

string checkcommand(string& command); // Функция ввода команд

void parser();

void createdirect();

/// Функии для INSERT ///

string checkInsert(string& table, string& values); // Проверка ввода команды инсерта

string Insert(string& command); // Функция инсерта

/// Функции для DELETE ///

string delAll(string& table); // Функция очистки всей таблицы

string delZnach(string& table, string& stolbec, string& values); // Функция удаления строк по значению

string delYslov(SinglyLinkedList<Filter>& conditions, string& table); //Функция удаления по условию

string Delete(string& command); // Проверка синтаксиса команды

bool parseConditions(string& conditions, string& table, SinglyLinkedList<Filter>& cond);

bool isValidColumn(string& table, string& colona);

/// Функции для SELECT ///

string selectall(SinglyLinkedList<Filter>& conditions); // ф-ия обычного селекта

string selectWithValue(SinglyLinkedList<Filter>& conditions, string& table, string& stolbec, struct Filter value); // ф-ия селекта с where для обычного условия

string selectWithLogic(SinglyLinkedList<Filter>& conditions, SinglyLinkedList<string>& table, SinglyLinkedList<string>& stolbec, SinglyLinkedList<Filter>& value);

// Вспомогательные ф-ии, чтобы избежать повтора кода в основных ф-иях

bool checkLockTable(string table); // ф-ия проверки, закрыта ли таблица

string lockTable(string& table, bool open);

SinglyLinkedList<int> findIndexColona(SinglyLinkedList<Filter>& conditions); // ф-ия нахождения индекса столбцов(для select)

int findIndexStlbCond(string table, string stolbec); // ф-ия нахождения индекса столбца условия(для select)

SinglyLinkedList<string> textInFile(SinglyLinkedList<Filter>& conditions); // ф-ия инпута текста из таблиц(для select)

SinglyLinkedList<string> InputTable(SinglyLinkedList<Filter>& conditions, SinglyLinkedList<string>& tables, int stlbindexvalnext, string table); // ф-ия инпута нужных колонок из таблиц для условиястолбец(для select)

string selection(SinglyLinkedList<int>& stlbindex, SinglyLinkedList<string>& tables); // ф-ия выборки(для select)

string Select(string& command);

};

#include "../src/commands.cpp"

#endif // COMMANDS\_H

networks.h

#pragma once

#include "includes.h"

#include "commands.h"

#define PORT 7432

int server, new\_socket; // идентификатор сокетов сервера и нового(для взаимодействия)

struct sockaddr\_in server\_address; // информация о адресе сервера

int addrlen = sizeof(server\_address);

int opt = 1; // переменная для настройки сокета

void createServer(BaseDate& shopNet);

void createSocket(); // ф-ия создания сокета сервера

void connectClient(BaseDate& shopNet); // ф-ия прослушивания и принятия входящих соединений

void procOfReq(int client\_socket, BaseDate& shopNet, mutex& mx); // ф-ия обработки запроса от клиента

#include "../src/networks.cpp"

## файлы cpp:

main.cpp

#include "../include/networks.h"

int countingLine (string& fin) { // ф-ия подсчёта строк в файле

ifstream file;

file.open(fin);

int countline = 0;

string line;

while(getline(file, line)) {

countline++;

}

file.close();

return countline;

}

string fileread (string& filename) { // чтение из файла

string result, str;

ifstream fileinput;

fileinput.open(filename);

while (getline(fileinput, str)) {

result += str + '\n';

}

result.pop\_back();

fileinput.close();

return result;

}

void filerec (string& filename, string data) { // запись в файл

ofstream fileoutput;

fileoutput.open(filename);

fileoutput << data;

fileoutput.close();

}

string BaseDate::checkcommand(string& command) {

if (command.substr(0, 11) == "INSERT INTO") {

command.erase(0, 12);

return Insert(command);

} else if (command.substr(0, 11) == "DELETE FROM") {

command.erase(0, 12);

return Delete(command);

} else if (command.substr(0, 6) == "SELECT") {

command.erase(0, 7);

return Select(command);

} else if (command == "STOP") {

exit(0);

} else {

return "Ошибка, неизвестная команда!";

}

}

int main() {

BaseDate shopNet;

shopNet.parser();

shopNet.createdirect();

createServer(shopNet);

return 0;

}

Вывод

В процессе выполнения второй практической работы я изучил понятие сетевого подключения и интернет-протоколов. Также я освоил навыки реализации собственного сетевого интерфейса.

Ссылка на репозиторий : <https://github.com/WarpMiner/pract2>