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 №1**

**«**Разработка СУБД**»**

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

|  |  |
| --- | --- |
| Выполнил:  Студент гр. «АБс-324», «АВТФ»  *Петров Максим Игоревич*  «20» декабря 2024г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | Проверил:  *доцент кафедры ЗИ*  *Архипова Анастасия Борисовна*  «\_\_\_» \_\_\_\_\_\_ 2024г  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |

Новосибирск 2024

**Задание:** требуется реализовать реляционную СУБД с поддержкой запросов на языке SQL. СУБД использует единственный тип данных - строка.

Требуется поддержать следующие SQL выражения:

SELECT <> FROM <> - выборка

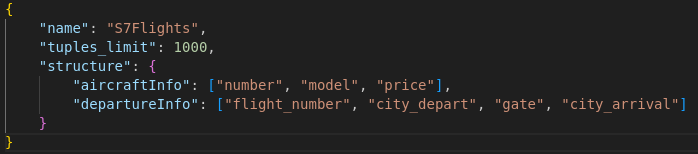
WHERE и операторы OR , AND - фильтрация

INSERT INTO - вставка данных в таблицы

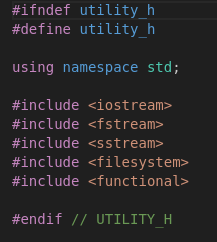
DELETE FROM - удаление данных из таблицы

Модули программы

base\_date.json:



utility.h:



list.h:

#ifndef list\_h

#define list\_h

#include "utility.h"

template <typename T>

struct Node {

T data;

Node\* next; //Указатель на след. элемент

Node(T value); //Конструктор узла

};

template <typename T>

struct SinglyLinkedList {

Node<T>\* head;

size\_t elementCount = 0;

SinglyLinkedList(); //Конструктор

~SinglyLinkedList(); //Деконструктор

bool isEmpty() const;

void print(); // ф-ия вывода списка

void pushFront(T value); //Добавление в начало списка

void pushBack(T value); //Добавление в конец списка

void popFront(); //Удаление в начале списка

void popBack(); //Удаление в конце списка

void removeAt(T value); //Удаление по индексу

bool find(T value); //Поиск значений в списке

void clearSList();

T getElementAt(int index) const;

int getIndex(T value);

void replace(int index, T newValue);

Node<T>\* getHead() const;

size\_t size();

};

#include "../src/list.cpp"

#endif // LIST\_H

сommands.h

#ifndef commands\_h

#define commands\_h

#include "utility.h"

#include "list.h"

#include "json.hpp"

int countingLine (string& fin);

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

void filerec (string& filename, string data); // Производим 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; // кол-во файлов таблиц

struct Filter { // структура для фильтрации

string table;

string colona;

string value;

string logicOP;

bool check;

};

void parser();

void createdirect();

/// Функии для INSERT ///

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

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

/// Функции для DELETE ///

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

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

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

void Delete(string& command); // Поверка синтаксиса команды

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

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

/// Функции для SELECT ///

void selectall(SinglyLinkedList<Filter>& conditions); // Функция команды select

void selectWithValue(SinglyLinkedList<Filter>& conditions, string& table, string& stolbec, struct Filter value);

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

bool isValidColumn(const string& table, const string& column);

void Select(string& command);

/// Вспомогательные функции ///

bool checkLockTable(string table); // Функция проверки открытия таблицы

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

void selection(SinglyLinkedList<int>& stlbindex, SinglyLinkedList<string>& tables); // Функция выбора

SinglyLinkedList<int> findIndexColona(SinglyLinkedList<Filter>& conditions);

int findIndexStlbCond(string table, string stolbec);

SinglyLinkedList<string> InputTable(SinglyLinkedList<Filter>& conditions, SinglyLinkedList<string>& tables, int stlbindexvalnext, string table);

SinglyLinkedList<string> textInFile(SinglyLinkedList<Filter>& conditions); // Функция инпута текста из таблиц

void lockTable(string& table, bool open);

};

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

#endif // COMMANDS\_H

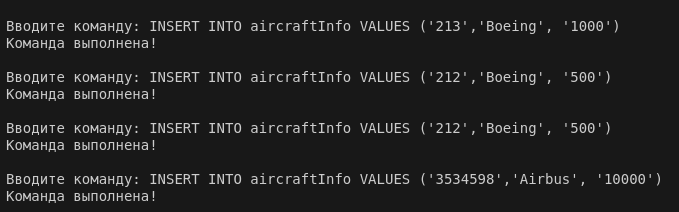
Работа программы

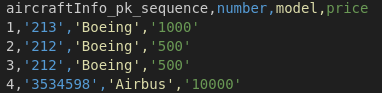
В таблицу aircraftInfo мы делаем запрос INSERT INTO aircraftInfo VALUES ('213','Boeing', '1000'), чтобы заполнить таблицу данными:

Изображение выглядит как снимок экрана, текст, программное обеспечение, Мультимедийное программное обеспечение

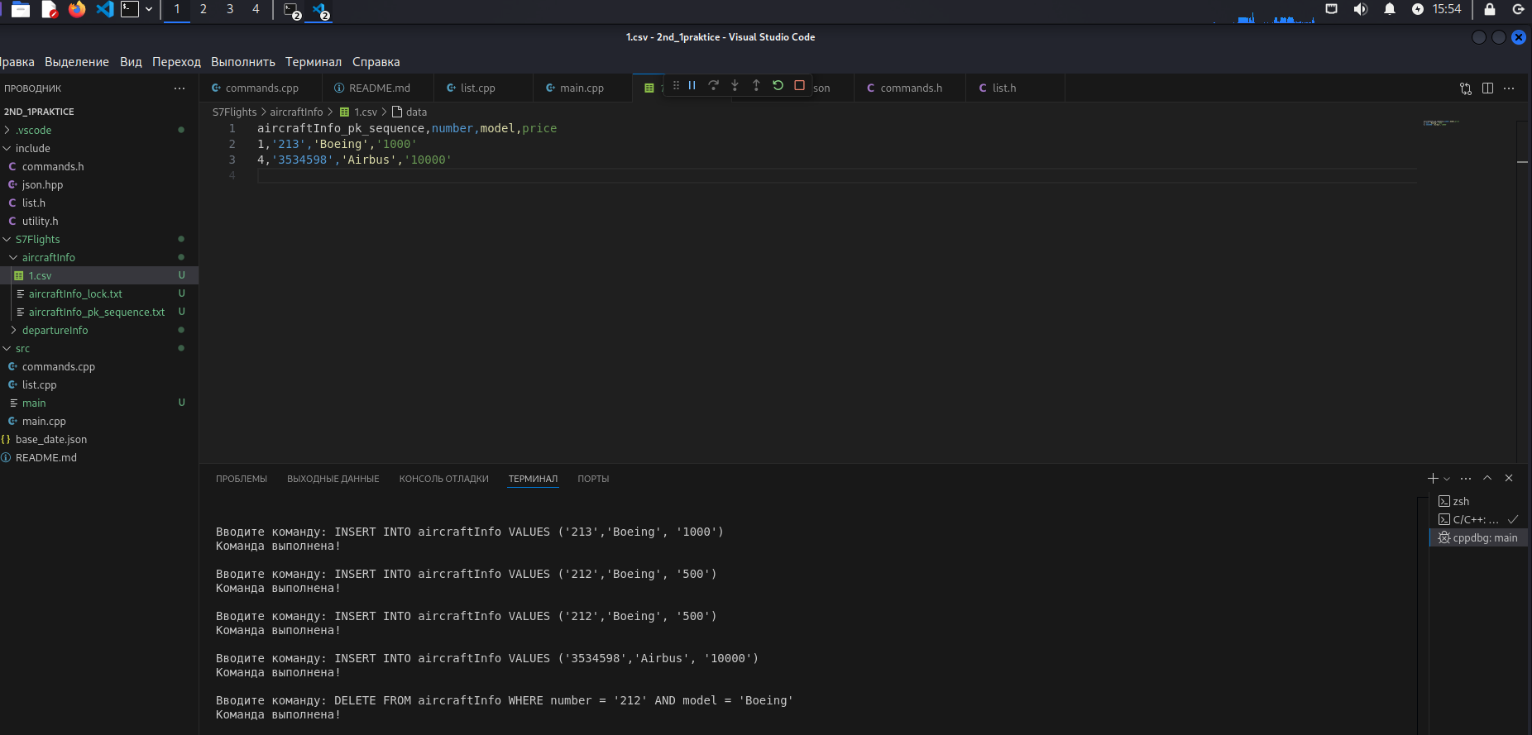
Автоматически созданное описание

Далее заполняем таблицу другими значениями:

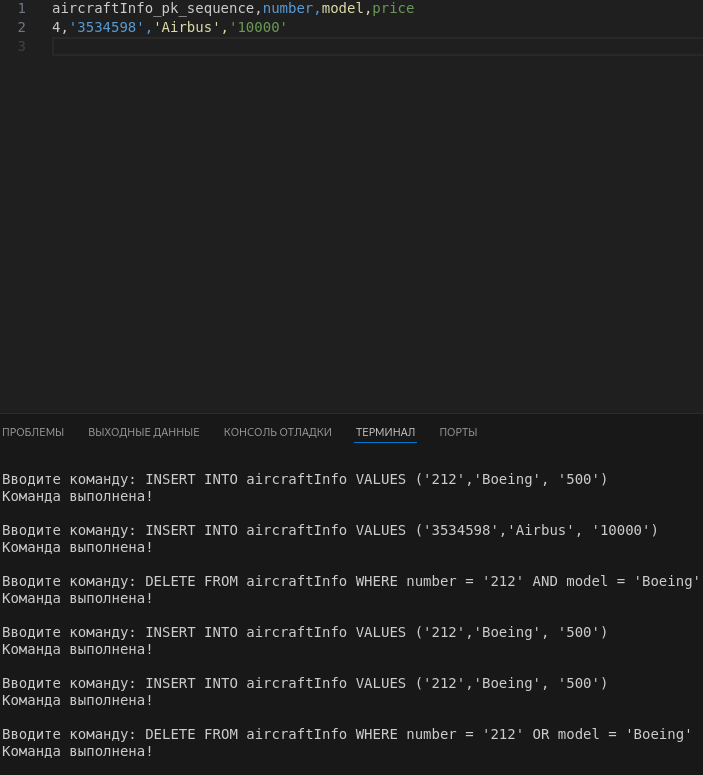




После чего проведем удаление конкретной строки. Используем запрос DELETE FROM aircraftInfo WHERE number = '212' AND model = 'Boeing', который должен удалить только 2 строчки, которые имеют 212 и Boeing:

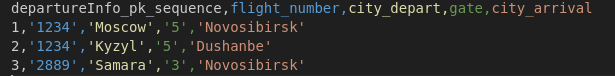


После проведения данного удаления проверим другой запрос на удаление DELETE FROM aircraftInfo WHERE number = '212' OR model = 'Boeing':

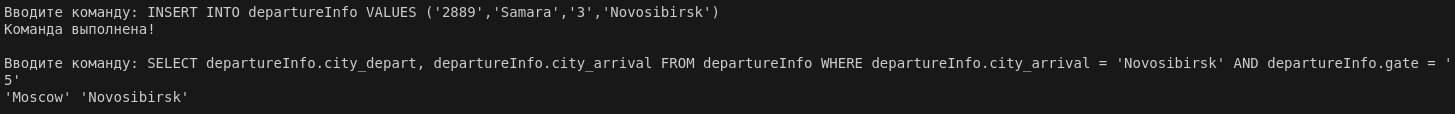


У нас удалились те строки, которые имели в себе значения 212 или Boeing.

Далее проведем запрос SELECT. У нас есть таблица departureInfo:



И мы используем запрос SELECT departureInfo.city\_depart, departureInfo.city\_arrival FROM departureInfo WHERE departureInfo.city\_arrival = 'Novosibirsk' AND departureInfo.gate = '5' который работает, как город прилета должен быть Новосибирск и выход должен быть 5.



Вывод

В ходе выполнения практической работы №1 я подробно познакомился с СУБД, SQL запросами и изучил базы данных. Научился реализовывать свою собственную реляционную СУБД при поддержки некоторых запросов на языке SQL.

Ссылка на репозиторий: <https://github.com/Mixassss/2nd_1praktice.git>