Факультет прикладной математики и информатики

Цао Юаня

ОТЧЕТ ПО ТЕХНОЛОГИЯМ ПРОГРАММИРОВАНИЯ

студента 2 курса 16 группы

Лабораторная работа №4

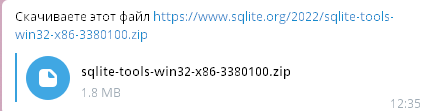
Преподаватель

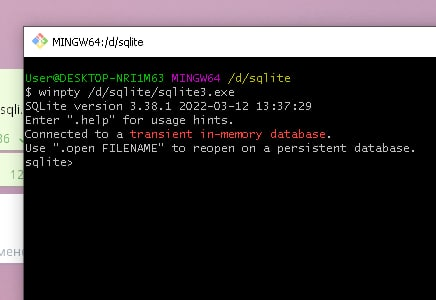
Давидовская М.И.

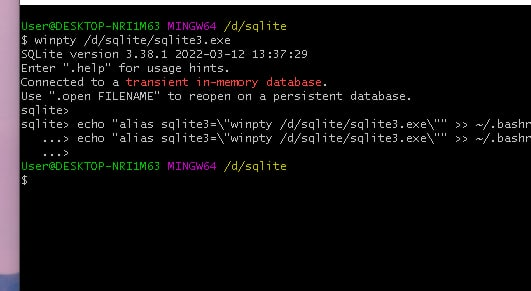
Минск 2022

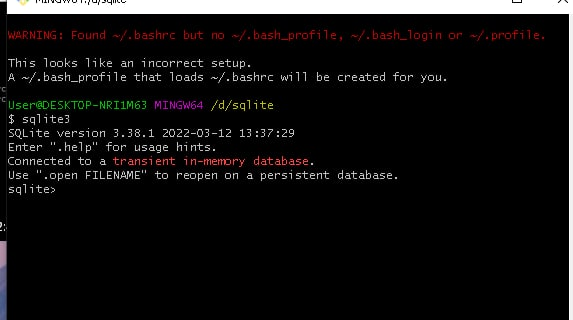
**Задание\_1**

Установил базу данных  SQLite и подключил к гиту.



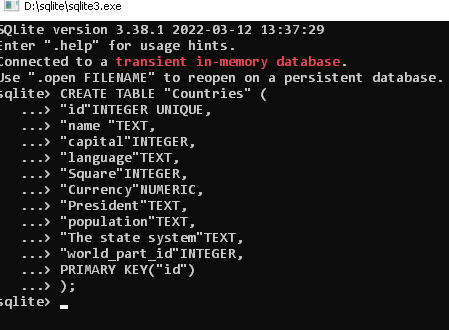






**Задание\_2 (25 вариант)**

создал БД с консоли и выполнить вопросы



https://lh4.googleusercontent.com/GBYNwsua1JW-fDE7mN6C_HS-MR_UZ27QnyXRkqccp9kVvK5RNVbNGCfNWX1TyjbgJjIUnuj7yyao7sCTGox4AfzVk0EUz3X8vGCG93ICg7D7Wg9JhOBAM2me7mo0JzRWUlt_mzxfByWIxmwblw

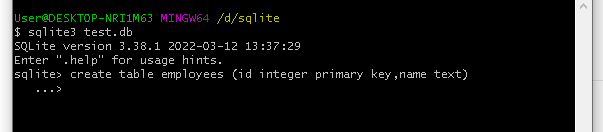
https://lh4.googleusercontent.com/ysEEzsXFbkEFoLIR8xQZIim_AuO9ViLaDG4G2Ef2fKEvLJBlBOwxmOnIZNfKhddNtaNzVD4V_Zv86XrNfs_8_2n_rAYFMO1Hmcs3nZ8S_dybt9CF9MMrx5JFqxDClBBdHezDWiy9ks0j4q2PLw

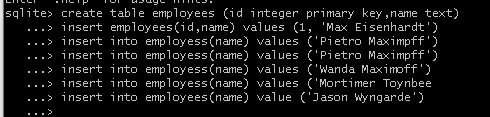
**Задание\_3 (25 вариант)**

Создал свою БД по примеру из примеров с инетернета.

упражнялся с (create); (insert); (select); (условие where);(инструкция like);

 (alter);

****

****

**https://lh6.googleusercontent.com/nTper_xTWwkVYku4zwU4FyCLz9AsV1U1ebVEZ0r3foLqVo4svNOW6OJSWvdvRv44wSuoW73bLLdvdnzeX-yhmATiarzfroXlVoHVMMqep7C6QsdUDihWQvfifJ5l6kNiup-V24Ev2MZZknahHg**

**https://lh5.googleusercontent.com/vHdWSyGXKey-qWEBTLanLV8bVwwvDlL_ZcXCiNMNY3J3w03DFKNPoqrJiKpokqrUTPdh905P7MLlttPhbKQFzscG3yzFRpnWy9XmhQwIp5coCDul9ypFDm7caAo4r-xouIcVjJiZ-TutDr3ClQ**

**https://lh3.googleusercontent.com/CD4_LnV3fNXUJpo4tXQI-yvPoRYNg2c0Yqx1EO7xNTB8V1-q6HepSGfZizqHHPKZz9juwr51tU53aYhagiKFOgUUn_bGJZUMFrzxQCqnNiqw8x_wqVAwZOFvVpSbwI1lg22XmXck9rXN9tUvNw**

**https://lh4.googleusercontent.com/crYyRrivk6UjFJkKCu1B-8g8wkip1WPvlaObqIjbyke1V4CzYh_WckJNeEs7pmNT8qHtb6Oqh-kvHRvGfGB46Ax5kfpsbE2_C9nTqPXA4nQykWyKLMCewE5vU3-JkG-ojkDOrCrFJEsybz7WGQ**

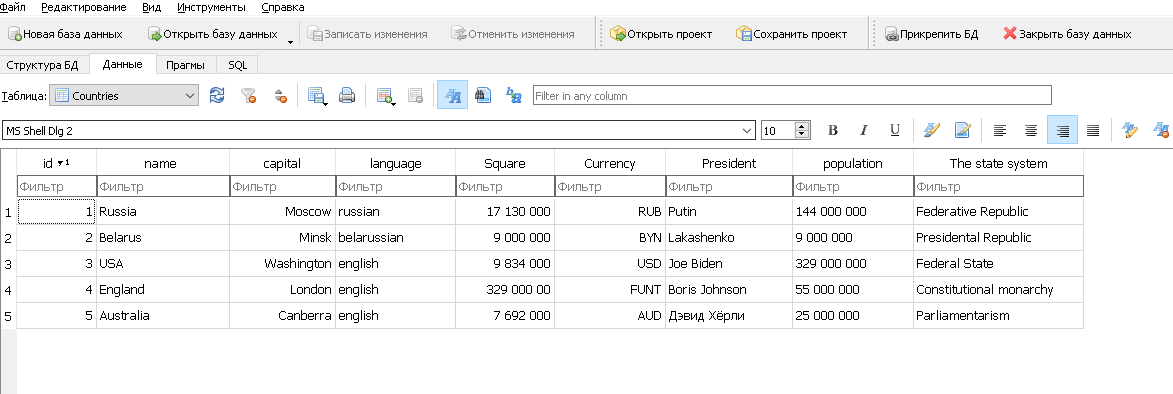
**https://lh5.googleusercontent.com/p5V6YY7aZAsIAumX470nNm5bQXYk7uLUToMclQh4bnnPzGIRzdbflmDHB591yD-6tUWcANdxgqiwzg0Xbxw_gIh7YPKuu9JhYxPcdy7idba7FKyjSykg6zxSPj9-t3-ldi5NhlaA8MF5_m-rpg**

**На верхних фотографии я показа работу по созданию таблицы с консоли, заполнению ее разными полями с различными идентификаторами и типами данных, поработал с запросами create, insert, select, where, like, alter, update, экспорт базы данных , и удаление таблицы TRUNCATE TABLE [name\_of\_DB].**

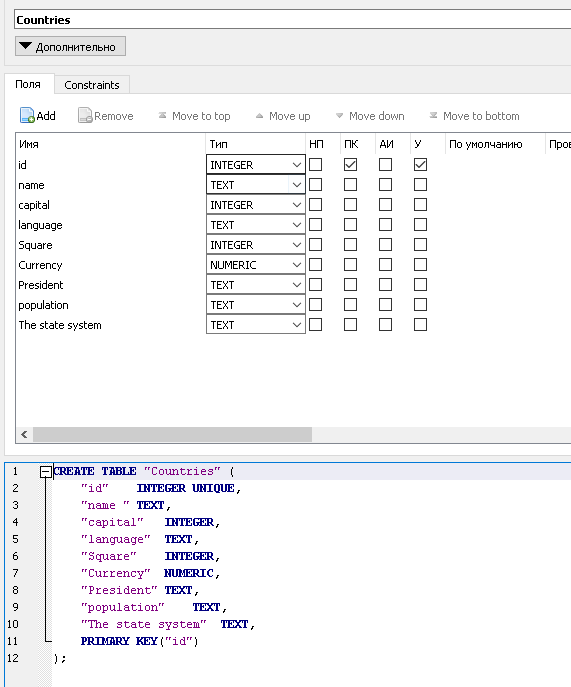
Скачал приложение для работы с БД: **DB Browser for SQLite**



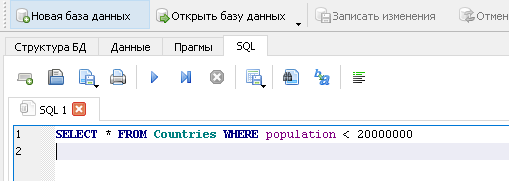
После создал свою БД из 5 стран и полей к ним по своему варианту: 25.



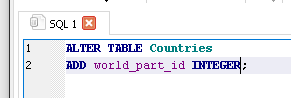
Создал все ключевые поля и прописал к ним фильтры:

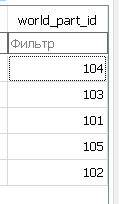


Поработал с запросами по своей БД

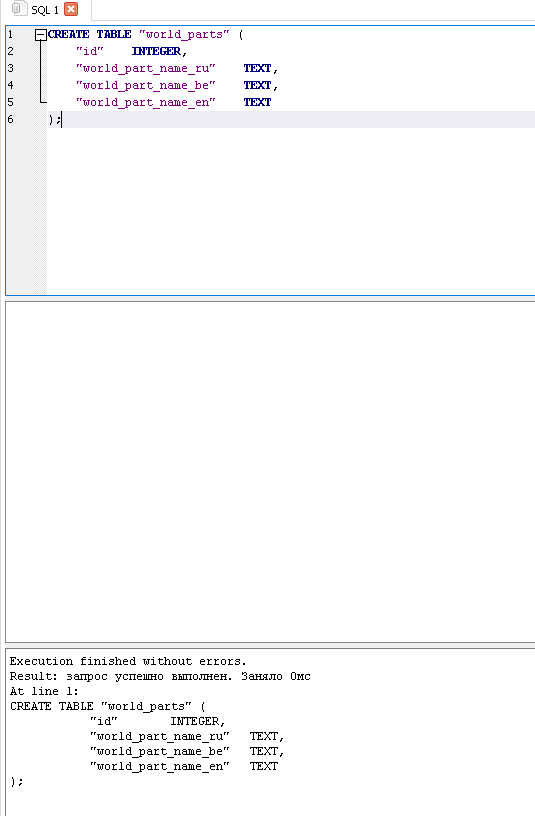


Добавление столбца world\_part\_id





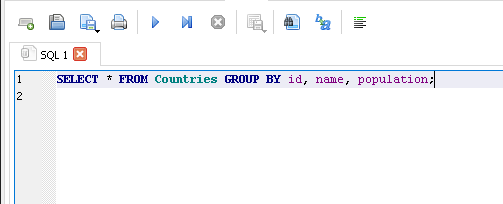
Создал с помощью **запросом**  новую таблицу “world\_parts”



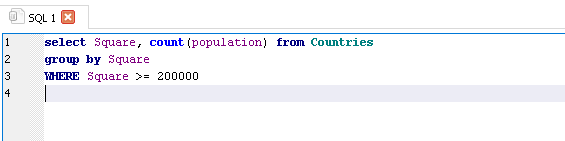
Заполнил эту же таблицу данными:



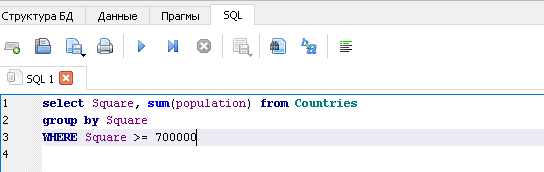
Вывожу данные обо всех странах в форме идентификатор страны, название, количество жителей страны



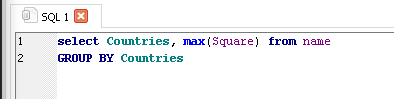
подсчет количества жителей с помощью count, если площадь территории >=200000 кв. км.

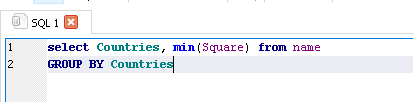


суммарная количество жителей с помощью sum, если площадь территории <700000 кв. км

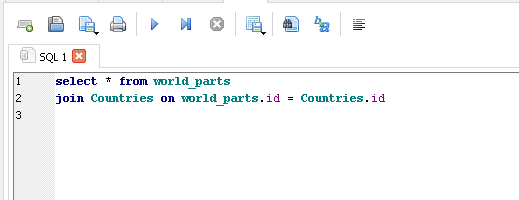


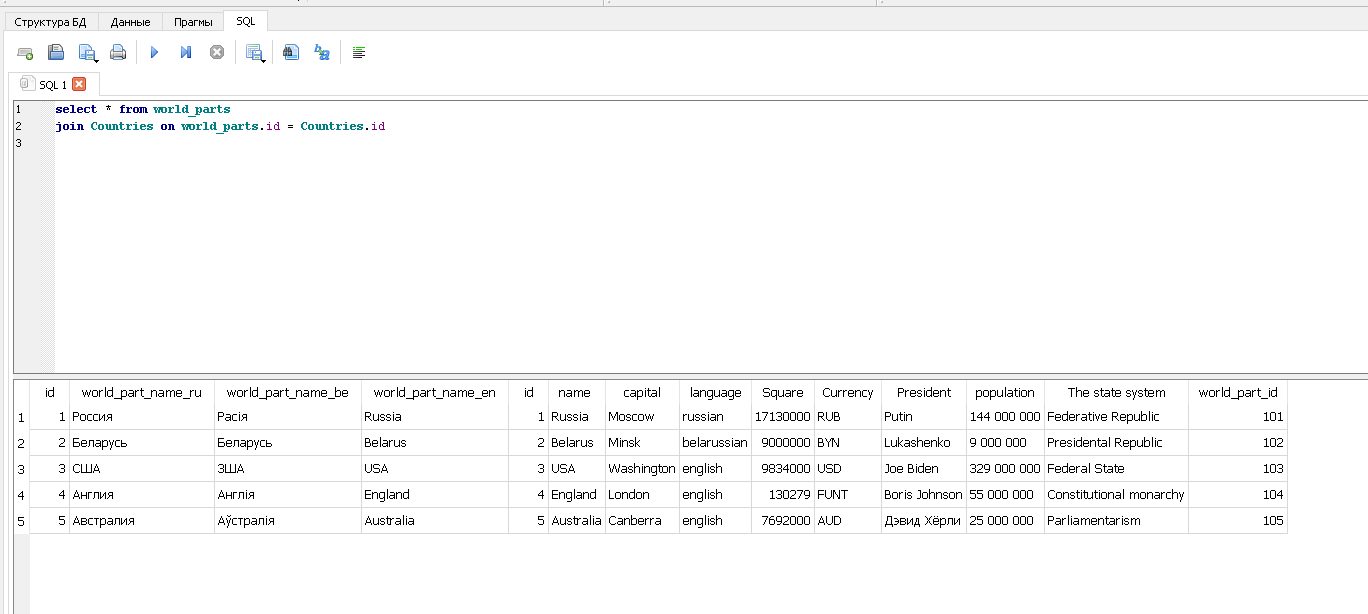
максимальная и минимальная площадь территории страны с помощью max и min



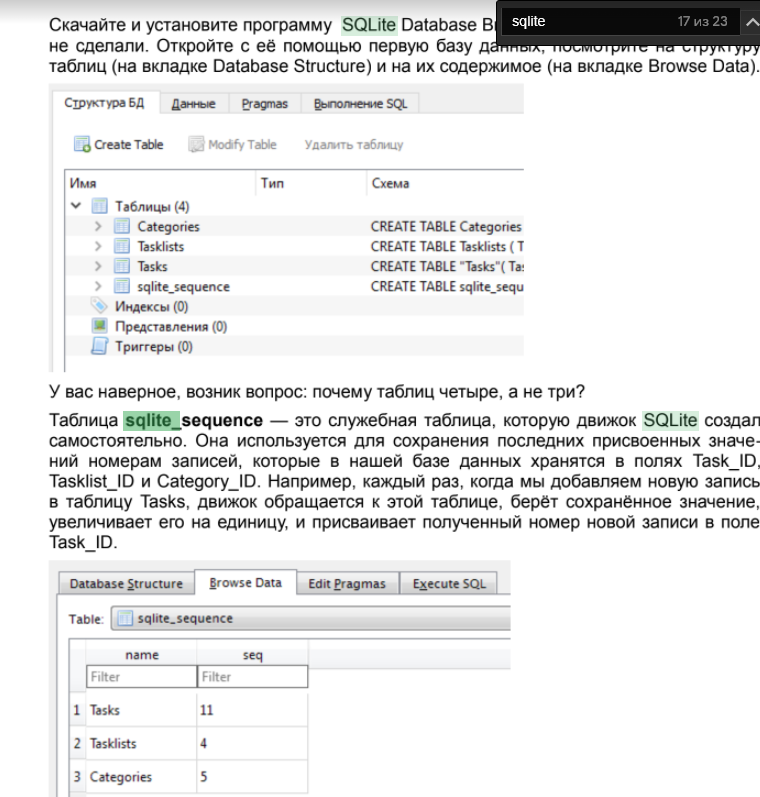


Используя инструкцию inner join вывел полные сведения о странах и части света для Европы

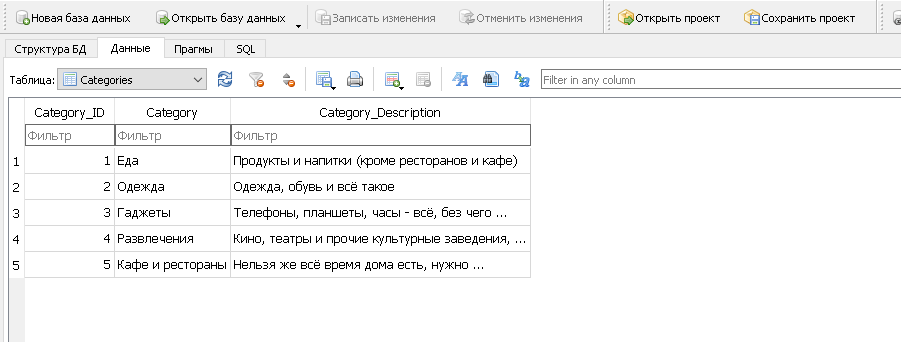




Изучил примеры работы с базой sqllite из документа Печеночкин Г. SQL для непрограммистов.

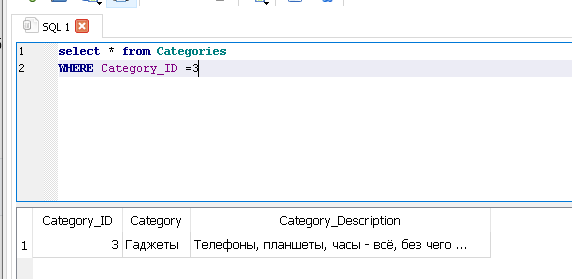


1. Скачал БД по ссылке



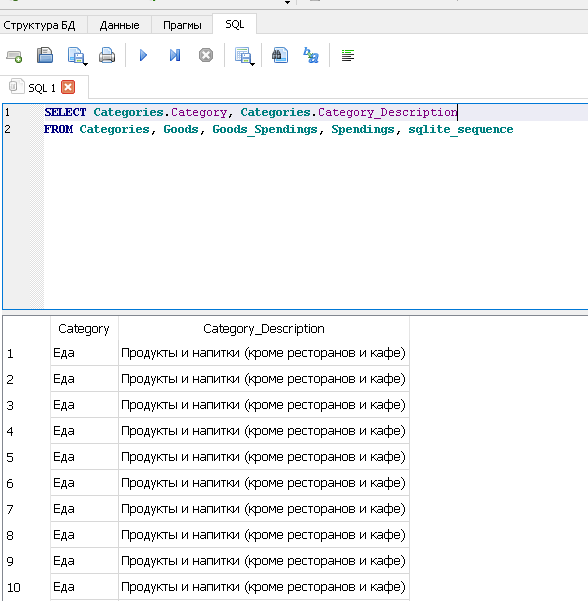
Выполняю запросы из БД по “Goods”:

1. Составил запрос, возвращающий описание категории с номером 3.



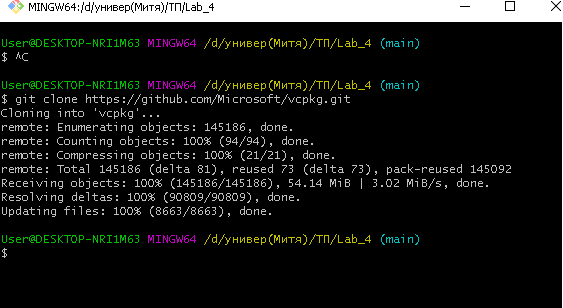
1. Составил запрос, обращающийся ко всем четырём таблицам (Categories, Spendings, Goods, Goods\_Spendings), возвращающий следующую информацию обо всех покупках отслеживаемых товаров: название категории, название

товара, название магазина и дату покупки.

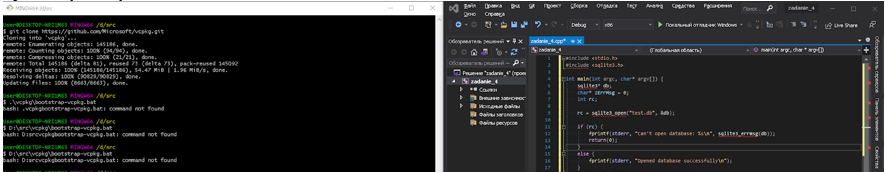


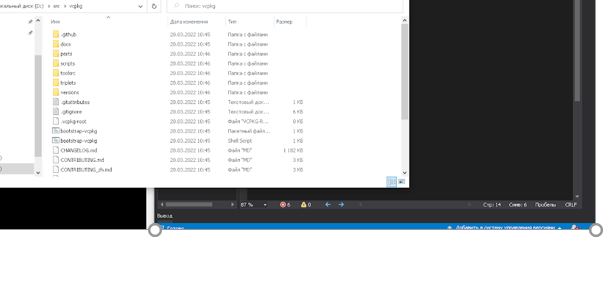
**Задание\_4**

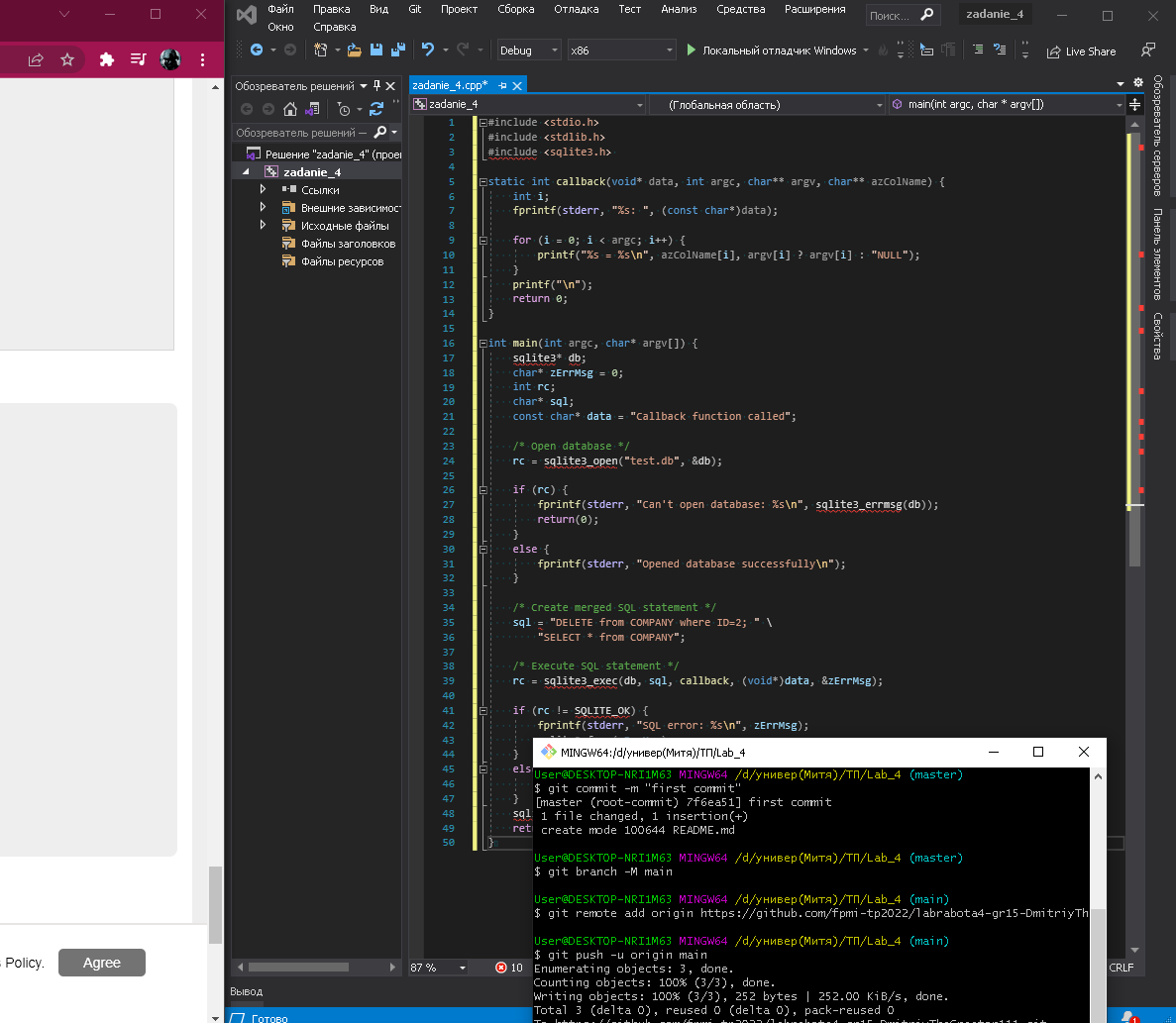
Создал проект в Visual Studio, клонировал туда библиотеку vcpkg для работы с базами данных посредством написания кода .



 Далее написал создал таблицу, потренировал на новых командах



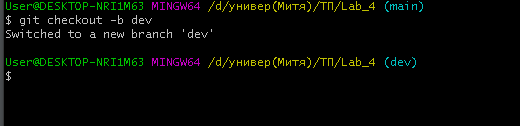




 В конце залил все на гитхаб.

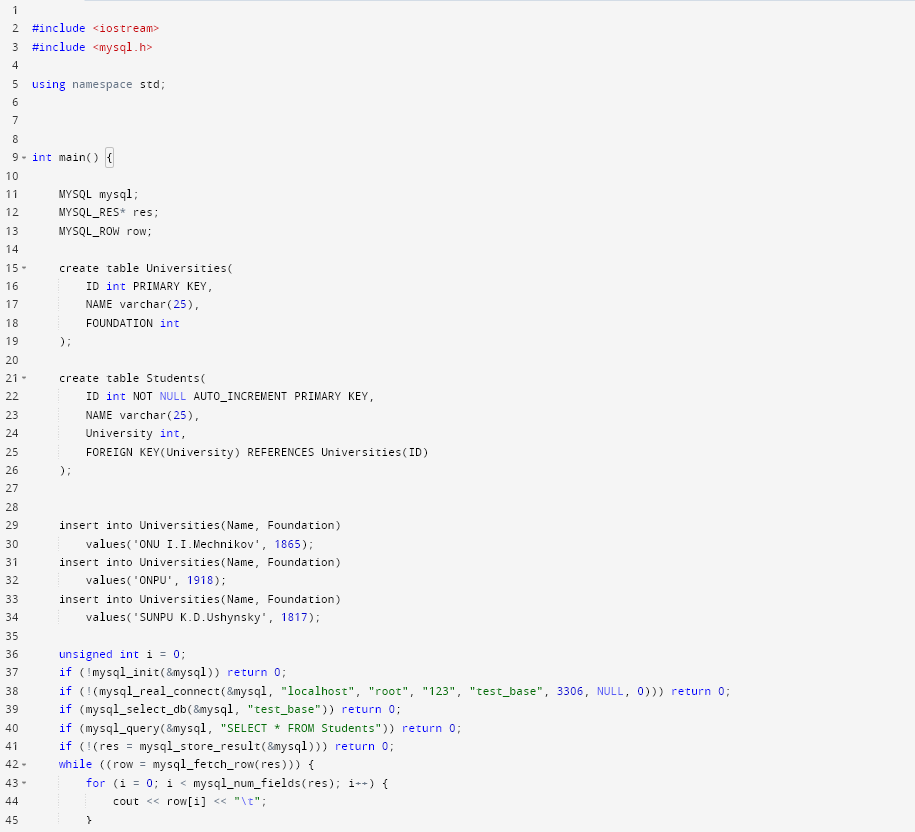
**Задание\_5**

Создал новую ветку dev, где создал свой проект по созданию и подключению в своей базе данных.

****

**Написал код по созданию базы данных и ее подключению**

**При написании использовал модель КИС, аккуратно раскладывал все файлы и исходники по папкам, структурировал.**





После залил ветку dev на гит, добавил также Read.me

**Контрольные вопросы**

1. **С консоли -sqlite или графическим приложением, например как**

**DB Browser (SQLite)**

1. **.databases - Список имен и файлов прикрепленных баз данных**
2. **Example:**

**SELECT \* FROM myTable**

**INTO OUTFILE '\tmp\myExportFile.csv'**

**FIELDS ENCLOSED BY '"'**

**TERMINATED BY ';'**

**ESCAPED BY '"'**

**LINES TERMINATED BY '\r\n';**

1. **zip ./dump.zip ./dump.sql**

**tar -pczf ./dump.tar.gz ./dump.sql**

1. **SELECT \* FROM ИМЯ\_ТАБЛИЦЫ**

**SELECT ИМЯ\_СТОЛБЦА FROM ИМЯ\_ТАБЛИЦЫ WHERE УСЛОВИЕ**

1. **Включение или выключение отображения заголовков**
2. **Первая — sqldiff, позволяет сравнивать базы (или отдельную таблицу) не только по структуре, но и по данным. Вторая — sqlite3\_analizer используется для вывода информации о том, как эффективно используется место таблицами и индексами в файле базы данных4**
3. **.tables**
4. **SELECT Customers.cname, Salespeople.sname, Salespeople.city**

**FROM Salespeople, Customers**

**WHERE Salespeople.city = Customers.city**

1. **UPDATE goods SET price = 150 WHERE num = 2**
2. **sqlite3\_open (const char \* имя файла, sqlite3 \*\* ppDb)**

**Эта подпрограмма открывает соединение с файлом базы данных SQLite и возвращает объект соединения с базой данных, который будет использоваться другими подпрограммами SQLite.**



**int sqlite3\_exec(**

**sqlite3\*,                                  /\* An open database \*/**

**const char \*sql,                           /\* SQL to be evaluated \*/**

**int (\*callback)(void\*,int,char\*\*,char\*\*),  /\* Callback function \*/**

**void \*,                                    /\* 1st argument to callback \*/**

**char \*\*errmsg                              /\* Error msg written here \*/**

**);**

**Интерфейс sqlite3\_exec() — это удобная оболочка для sqlite3\_prepare\_v2() , sqlite3\_step() и sqlite3\_finalize() , которая позволяет приложению запускать несколько операторов SQL без необходимости использовать много кода C.**



**sqlite3\_close (sqlite3 \*)**

**Эта процедура закрывает соединение с базой данных, ранее открытое вызовом sqlite3\_open (). Все подготовленные заявления, связанные с соединением, должны быть завершены до закрытия соединения.**

**Если остаются какие-либо запросы, которые не были завершены, sqlite3\_close () вернет SQLITE\_BUSY с сообщением об ошибке «Невозможно закрыть из-за незавершенных операторов».**



**int main(int argc, char\* argv[]) {**

**sqlite3 \*db;**

**char \*zErrMsg = 0;**

**int rc;**

**char \*sql;**

**/\* Open database \*/**

**rc = sqlite3\_open("test.db", &db);**

**if( rc ) {**

**fprintf(stderr, "Can't open database: %s\n", sqlite3\_errmsg(db));**

**return(0);**

**} else {**

**fprintf(stdout, "Opened database successfully\n");**

**}**

**/\* Create SQL statement \*/**

**sql = "CREATE TABLE COMPANY("  \**

**"ID INT PRIMARY KEY     NOT NULL," \**

**"NAME           TEXT    NOT NULL," \**

**"AGE            INT     NOT NULL," \**

**"ADDRESS        CHAR(50)," \**

**"SALARY         REAL );";**

**/\* Execute SQL statement \*/**

**rc = sqlite3\_exec(db, sql, callback, 0, &zErrMsg);**

**if( rc != SQLITE\_OK ){**

**fprintf(stderr, "SQL error: %s\n", zErrMsg);**

**sqlite3\_free(zErrMsg);**

**} else {**

**fprintf(stdout, "Table created successfully\n");**

**}**

**sqlite3\_close(db);**

**return 0;**

**}**



**connection.setAutoCommit(false); -**

**Транзакции дают нам возможность контролировать когда и где сохранять изменения в БД. Благодаря этому мы, например, можем объединить группу SQL – запросов в одну логическую группу и, если один из запросов не пройдёт – мы отменяем всю транзакцию.**

**кусочек кода:**

**try {**

**SQL = "INSERT INTO developer VALUES (6, 'John','C#', 2200)";**

**statement.executeUpdate(SQL);**

**SQL = "INSERT INTO developers VALUES (7, 'Ron', 'Ruby', 1900)";**

**statement.executeUpdate(SQL);**

**connection.commit();**

**} catch (SQLException e) {**

**System.out.println("SQLException. Executing rollback to savepoint...");**

**connection.rollback(savepointOne);**

**}**