|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра практической и прикладной информатики (ППИ)**

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ**

по дисциплине «Разработка баз данных»

**Практические работы № 1 - 4**

|  |  |  |  |
| --- | --- | --- | --- |
| Студент группы ИНБО-01-17 | ИКБО-20-21 Сидоров С.Д. | (подпись) | |
| Преподаватель | Благовещенский В.Г. | (подпись) | |
| Отчет представлен | «\_\_\_»\_\_\_\_\_\_\_\_2023 г. | |  | |

Москва 2023 г.

**Практическая работа 1**

**Цель**: создание базы данных и таблицы в ней по теме «Каршеринг», на основе разработанных моделей.

**Результат работы:**

Разработанная модель в нотации IDEF1X представлена на рисунке 1.

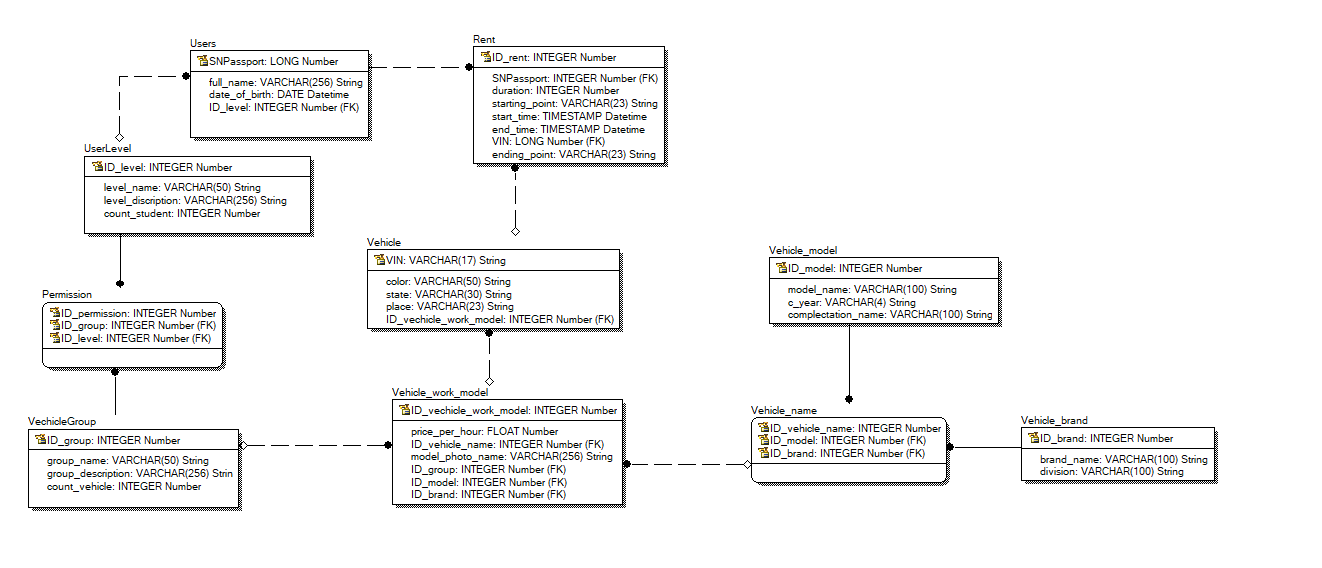


Рисунок 1 – Разработанная модель в нотации IDEF1X

Для реализации базы данных использовалась СУБД MySQL. Все последующие команды были выполнены в MySQL command line. С помощью команд была создана база данных “carsharing” и таблицы в ней, процесс создания представлен на рисунках 2-6.



Рисунок 2 – Создание базы данных и создание части таблиц



Рисунок 3 – Создание таблиц в базе данных “carsharing”

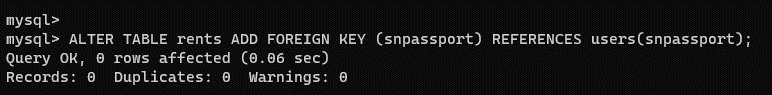


Рисунок 4 – Добавление внешнего ключа к таблицам базы данных “carsharing”

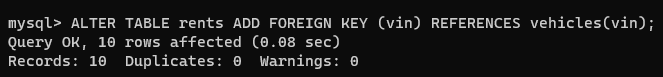


Рисунок 5 – Добавление внешнего ключа к таблицам базы данных “carsharing”

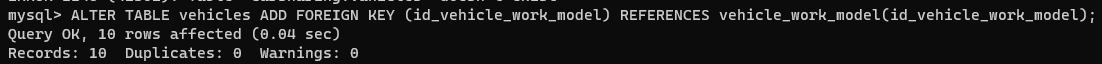


Рисунок 6 – Добавление внешнего ключа к таблицам базы данных “carsharing”

Результаты создания таблиц представлены на рисунках 7 – 9.

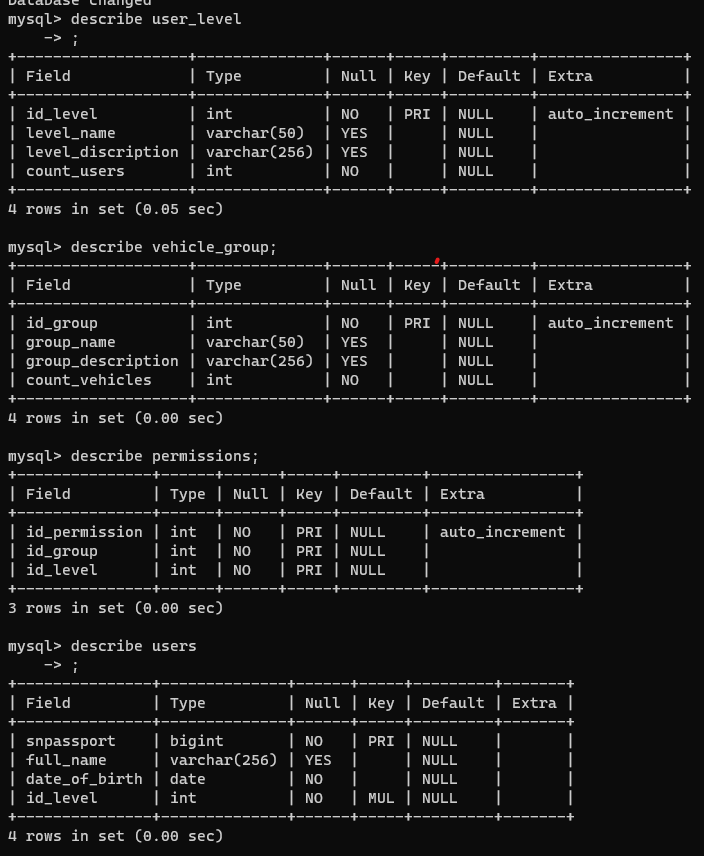


Рисунок 6 – Структура некоторых таблиц базы данных “carsharing”

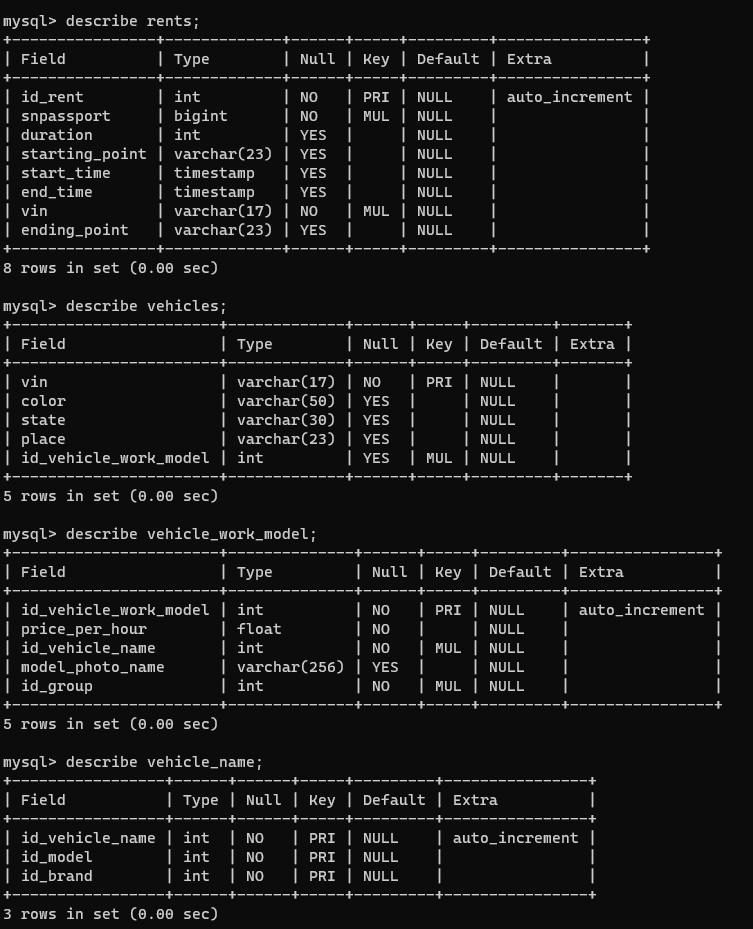


Рисунок 7 - Структура некоторых таблиц базы данных “carsharing”

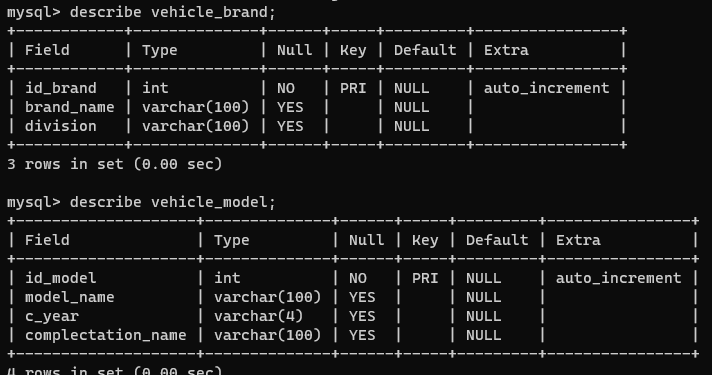


Рисунок 8 - Структура некоторых таблиц базы данных “carsharing”

Также с помощью MySQL Workbench была сгенерирована диаграмма по уже созданной базе данных (рис.10).

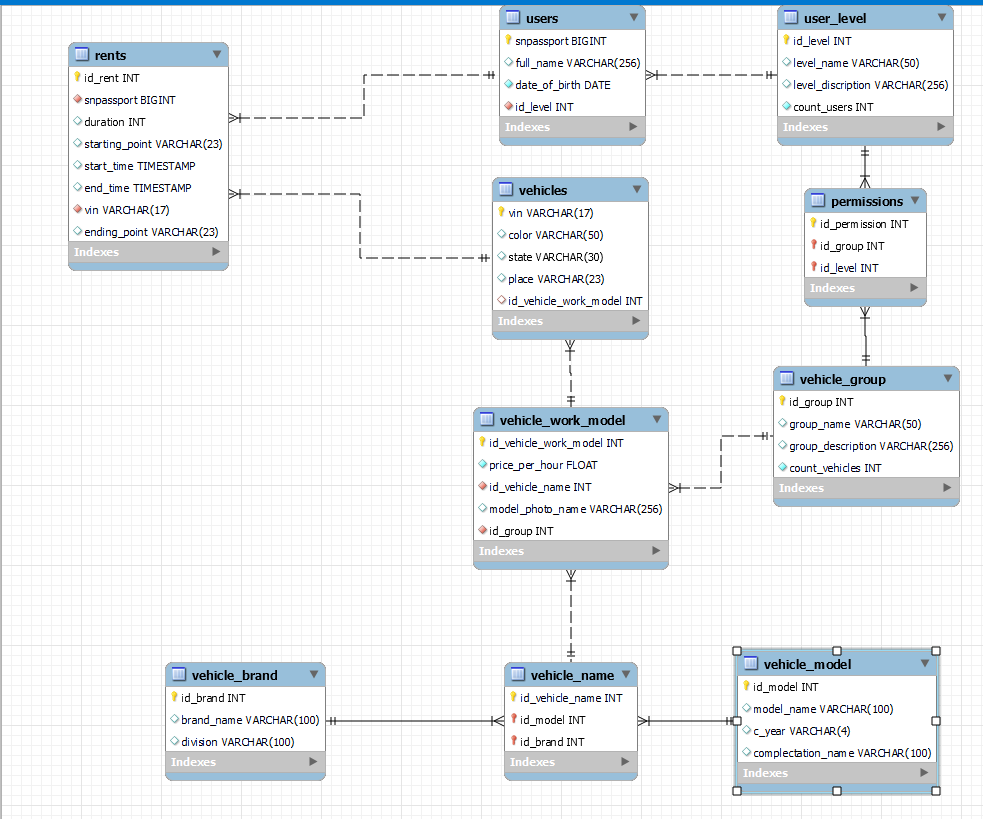


Рисунок 10 – Сгенерированная модель базы данных

Для создания базы данных использовался код представленный в листинге 1.

Листинг 1 – Создание базы данных.

 CREATE DATABASE IF NOT EXISTS carsharing;

 USE carsharing;

 CREATE TABLE IF NOT EXISTS users(

    snpassport BIGINT PRIMARY KEY,

    full\_name VARCHAR(256),

    date\_of\_birth DATE NOT NULL,

    id\_level INT NOT NULL

);

CREATE TABLE IF NOT EXISTS user\_level(

    id\_level INT AUTO\_INCREMENT PRIMARY KEY,

    level\_name VARCHAR(50),

    level\_discription VARCHAR(256),

    count\_users INTEGER NOT NULL

);

ALTER TABLE users ADD FOREIGN KEY (id\_level) REFERENCES user\_level(id\_level);

Продолжение листинга 1

CREATE TABLE IF NOT EXISTS vehicle\_group(

    id\_group INT AUTO\_INCREMENT PRIMARY KEY,

    group\_name VARCHAR(50),

    group\_description VARCHAR(256),

    count\_vehicles INT NOT NULL

);

CREATE TABLE IF NOT EXISTS permissions(

    id\_permission INT AUTO\_INCREMENT,

    id\_group INT,

    id\_level INT,

    primary key(id\_permission, id\_group, id\_level)

);

ALTER TABLE permissions ADD FOREIGN KEY (id\_group) REFERENCES user\_level(id\_level);

ALTER TABLE permissions ADD FOREIGN KEY (id\_level) REFERENCES vehicle\_group(id\_group);

CREATE TABLE IF NOT EXISTS vehicle\_model(

    id\_model INTEGER AUTO\_INCREMENT PRIMARY KEY,

    model\_name VARCHAR(100),

    c\_year VARCHAR(4),

    complectation\_name VARCHAR(100)

);

CREATE TABLE IF NOT EXISTS vehicle\_brand(

    id\_brand INTEGER AUTO\_INCREMENT PRIMARY KEY,

    brand\_name VARCHAR(100),

    division VARCHAR(100)

);

CREATE TABLE IF NOT EXISTS vehicle\_name(

    id\_vehicle\_name INTEGER AUTO\_INCREMENT,

    id\_model INT,

    id\_brand INT,

    PRIMARY KEY (id\_vehicle\_name, id\_model, id\_brand)

);

ALTER TABLE vehicle\_name ADD FOREIGN KEY (id\_model) REFERENCES vehicle\_model(id\_model);

ALTER TABLE vehicle\_name ADD FOREIGN KEY (id\_brand) REFERENCES vehicle\_brand(id\_brand);

CREATE TABLE IF NOT EXISTS vehicle\_work\_model(

    id\_vehicle\_work\_model INTEGER AUTO\_INCREMENT PRIMARY KEY,

    price\_per\_hour FLOAT NOT NULL,

    id\_vehicle\_name INT NOT NULL,

    model\_photo\_name VARCHAR(256),

Продолжение листинга 1

id\_group INT NOT NULL

);

ALTER TABLE vehicle\_work\_model ADD FOREIGN KEY (id\_vehicle\_name) REFERENCES vehicle\_name(id\_vehicle\_name);

ALTER TABLE vehicle\_work\_model ADD FOREIGN KEY (id\_group) REFERENCES vehicle\_group(id\_group);

CREATE TABLE IF NOT EXISTS vehicles(

    vin VARCHAR(17) NOT NULL PRIMARY KEY,

    color VARCHAR(50),

    state VARCHAR(30),

    place VARCHAR(23),

    id\_vehicle\_work\_model INT

);

CREATE TABLE IF NOT EXISTS rents(

    id\_rent INT AUTO\_INCREMENT PRIMARY KEY,

    snpassport BIGINT NOT NULL,

    duration INT,

    starting\_point VARCHAR(23),

    start\_time TIMESTAMP,

    end\_time TIMESTAMP,

    vin VARCHAR(17) NOT NULL,

    ending\_point VARCHAR(23)

);

ALTER TABLE rents ADD FOREIGN KEY (snpassport) REFERENCES users(snpassport);

ALTER TABLE rents ADD FOREIGN KEY (vin) REFERENCES vehicles(vin);

ALTER TABLE vahicles ADD FOREIGN KEY (id\_vehicle\_work\_model) REFERENCES vehicle\_work\_model(id\_vehicle\_work\_model);

После создания базы данных было произведено заполнение полей базы с помощью команды INSERT, процесс представлен на рисунках 11 – 20.

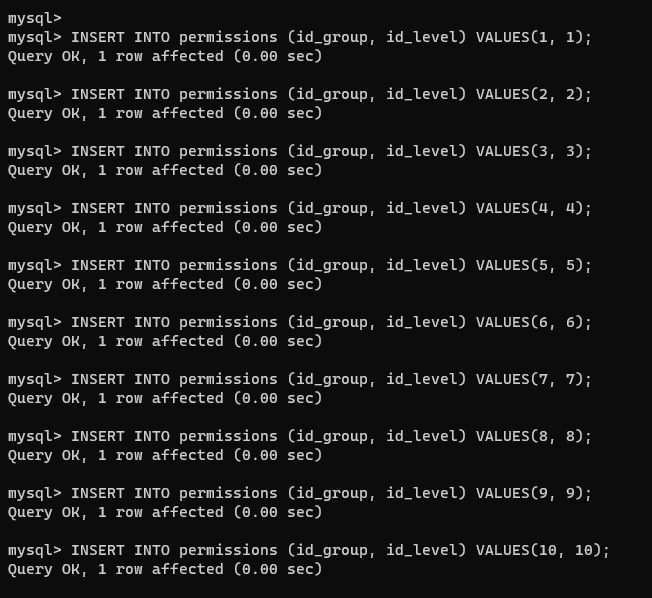


Рисунок 11 – Заполнение таблицы “permissions”

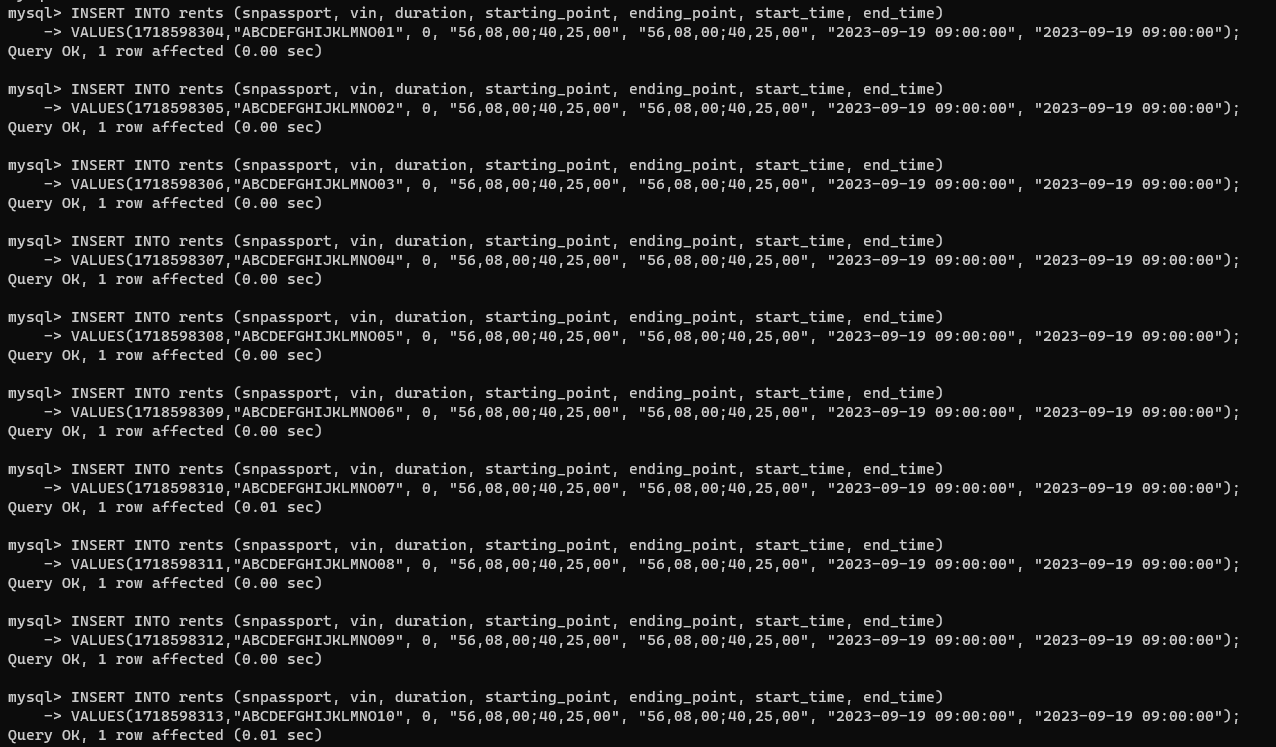


Рисунок 12 – Заполнение таблицы “rents”

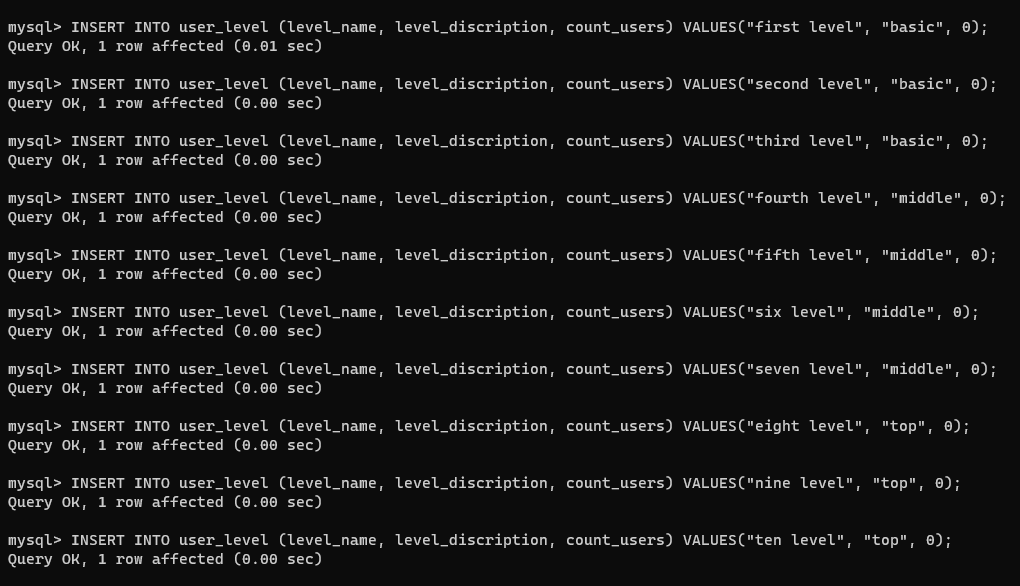


Рисунок 13 – Заполнение таблицы “user\_level”

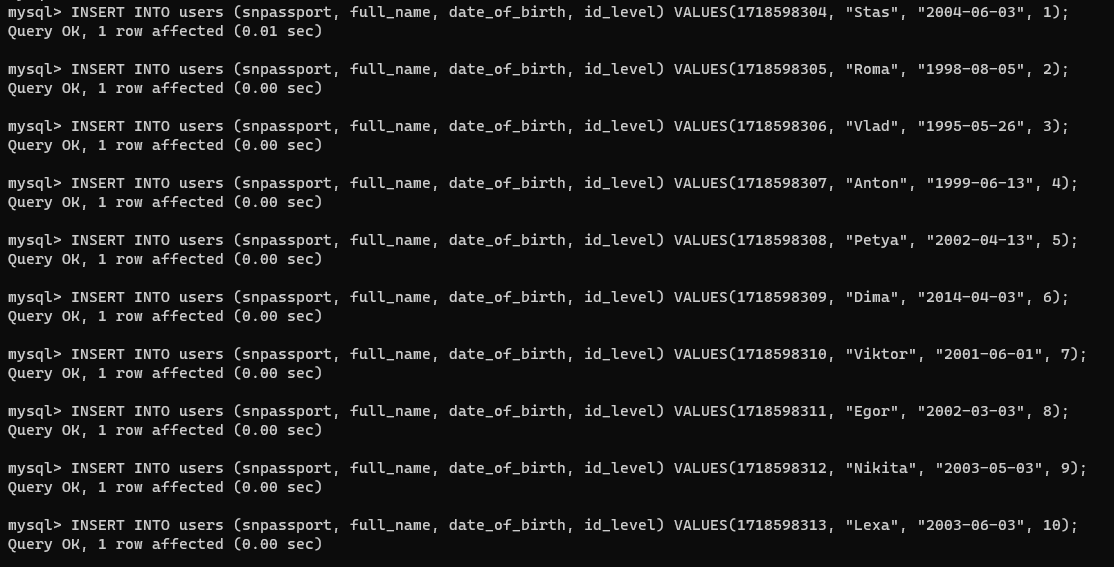


Рисунок 14 – Заполнение таблицы “users”



Рисунок 15 – Заполнение таблицы “vehicles”

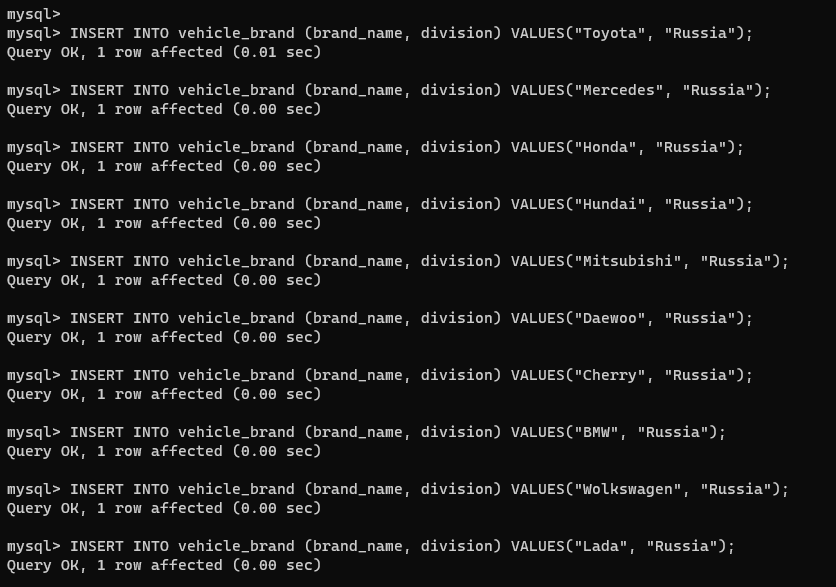


Рисунок 16 – Заполнение таблицы “vehicle\_brand”

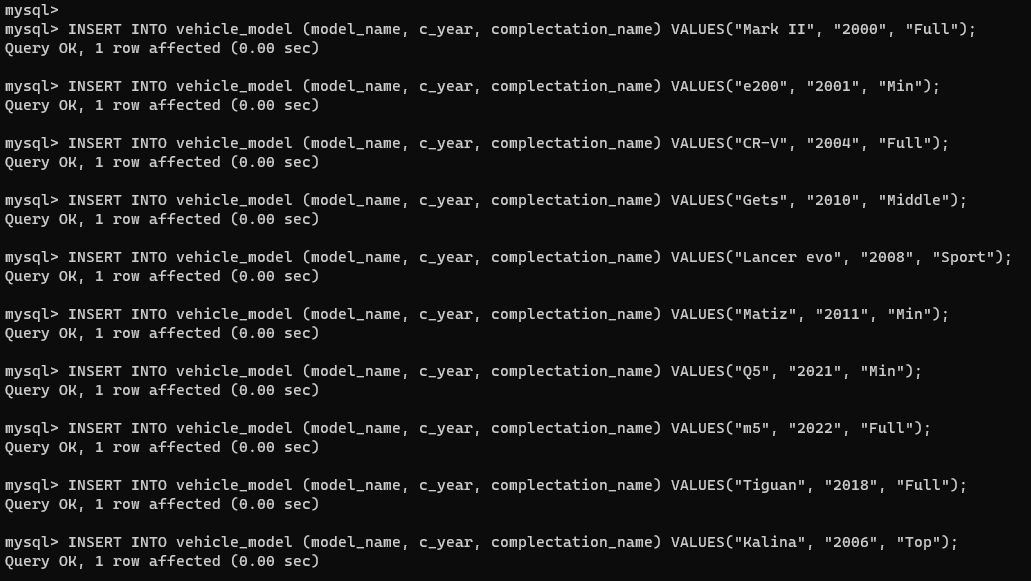


Рисунок 17 – Заполнение таблицы “vehicle\_model”

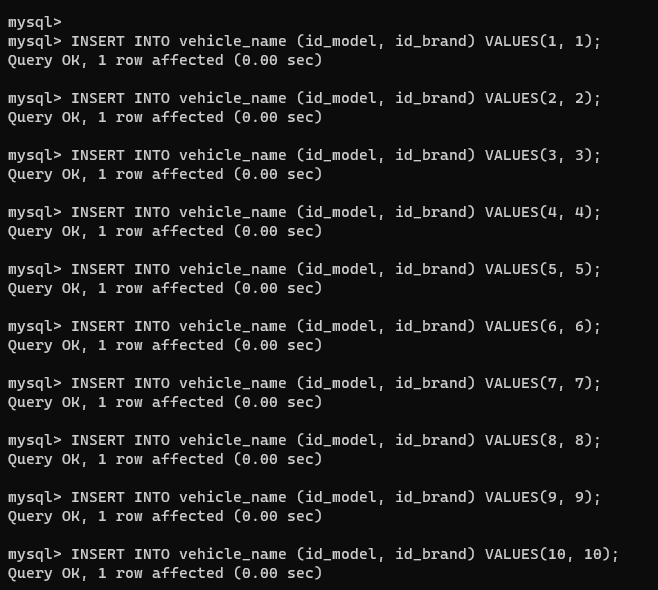


Рисунок 18 – Заполнение таблицы ‘vehicle\_name’



Рисунок 19 – Заполнение таблицы “vehicle\_work\_model”

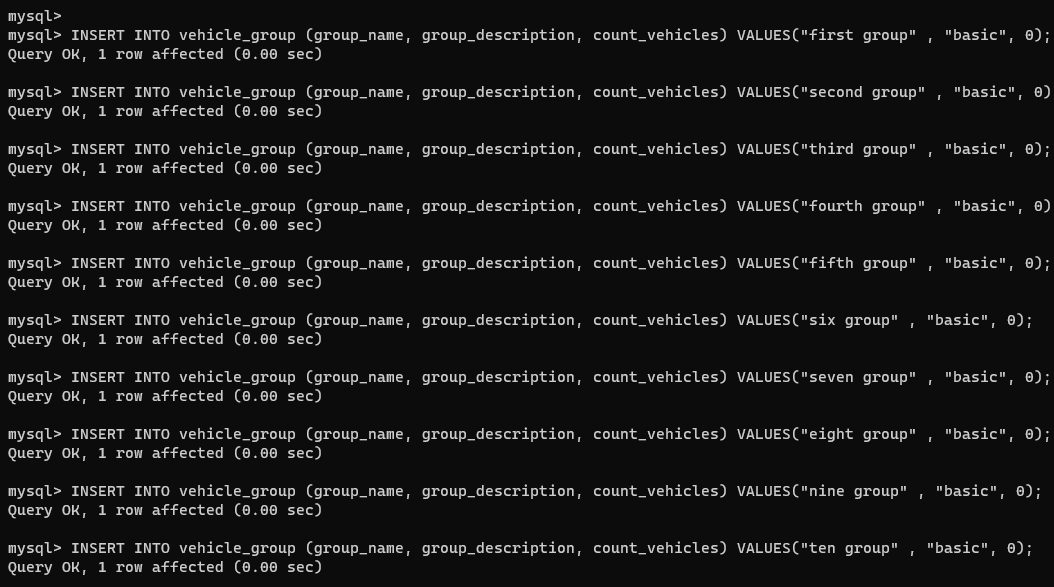


Рисунок 20 – Заполнение таблицы “vehicle\_group”

Результат заполнения представлен на рисунках 21 – 24.

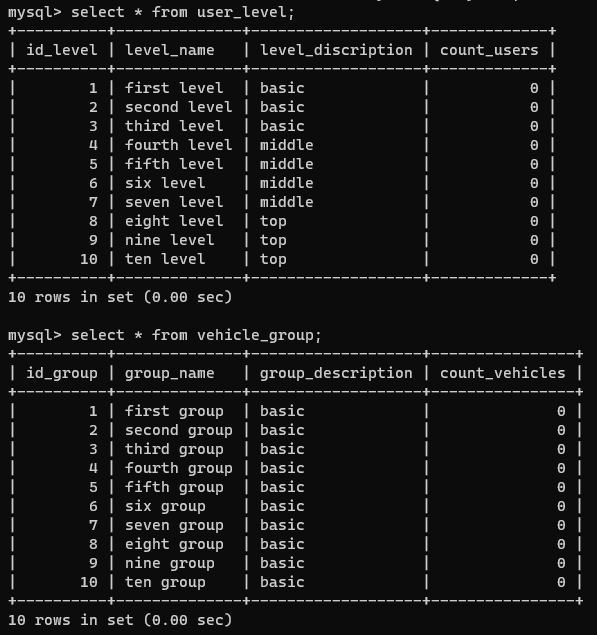


Рисунок 21 – Содержимое части таблиц

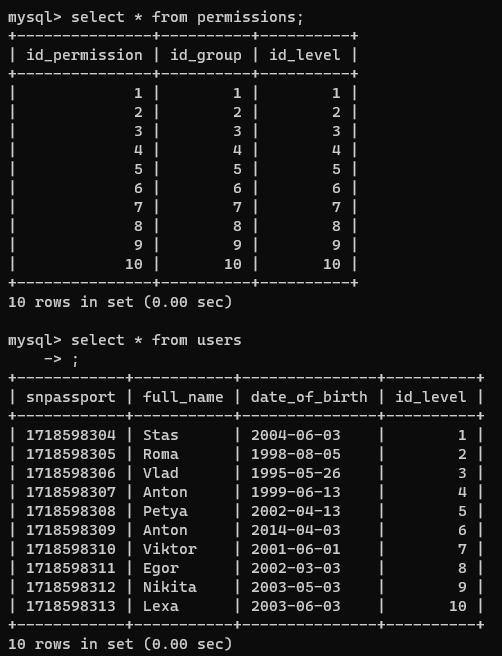


Рисунок 22 – Содержимое части таблиц

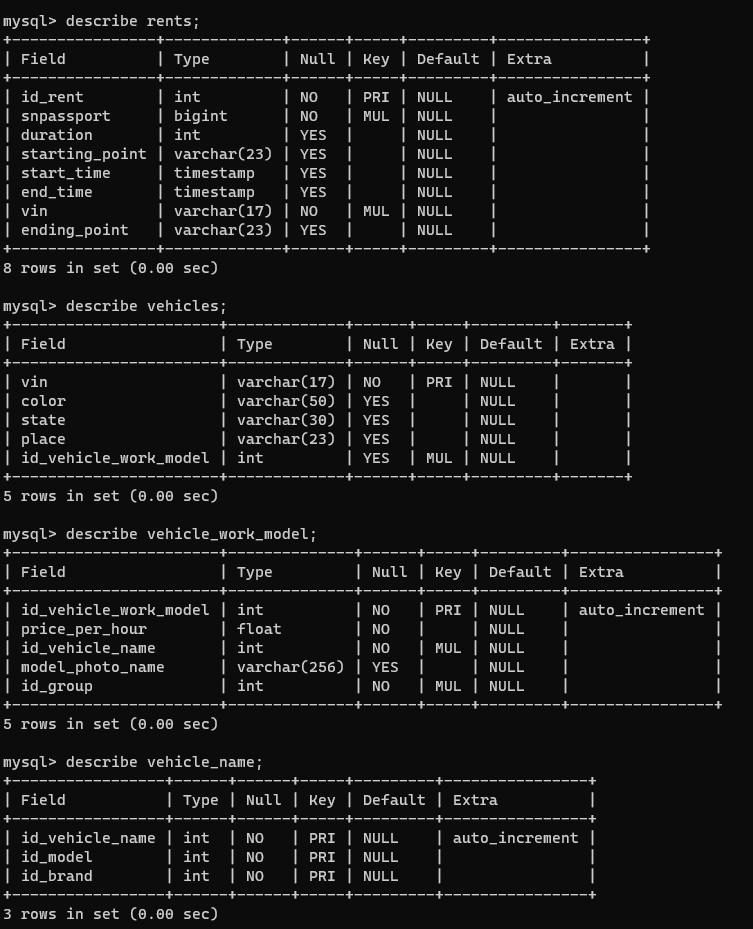


Рисунок 23 – Содержимое части таблиц

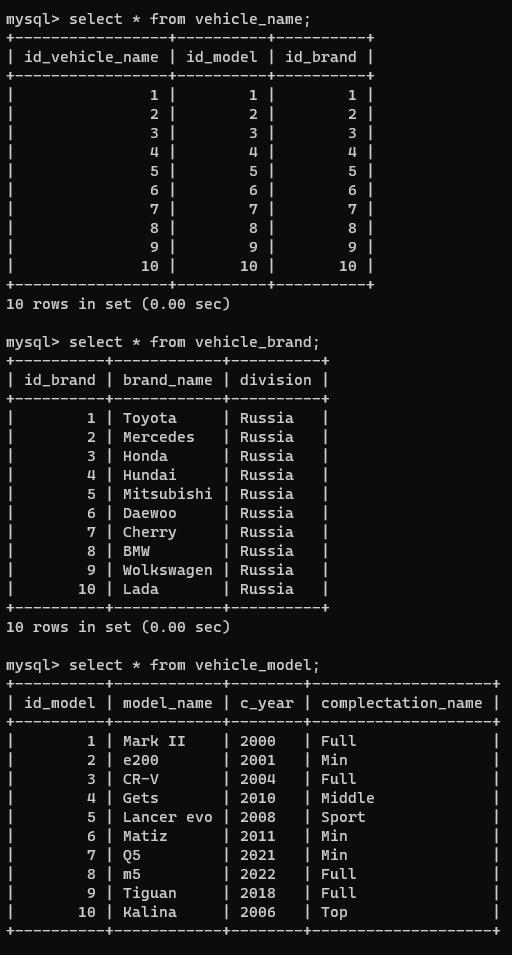


Рисунок 24 – Содержимое части таблиц

Для заполнения таблиц использовался код, представленный в листинге 2.

Листинг 2 – Код заполнения таблиц базы данных.

USE carsharing;

INSERT INTO user\_level (level\_name, level\_discription, count\_users) VALUES("first level", "basic", 0);

INSERT INTO user\_level (level\_name, level\_discription, count\_users) VALUES("second level", "basic", 0);

INSERT INTO user\_level (level\_name, level\_discription, count\_users) VALUES("third level", "basic", 0);

INSERT INTO user\_level (level\_name, level\_discription, count\_users) VALUES("fourth level", "middle", 0);

Продолжение листинга 2.

INSERT INTO user\_level (level\_name, level\_discription, count\_users) VALUES("fifth level", "middle", 0);

INSERT INTO user\_level (level\_name, level\_discription, count\_users) VALUES("six level", "middle", 0);

INSERT INTO user\_level (level\_name, level\_discription, count\_users) VALUES("seven level", "middle", 0);

INSERT INTO user\_level (level\_name, level\_discription, count\_users) VALUES("eight level", "top", 0);

INSERT INTO user\_level (level\_name, level\_discription, count\_users) VALUES("nine level", "top", 0);

INSERT INTO user\_level (level\_name, level\_discription, count\_users) VALUES("ten level", "top", 0);

INSERT INTO vehicle\_group (group\_name, group\_description, count\_vehicles) VALUES("first group" , "basic", 0);

INSERT INTO vehicle\_group (group\_name, group\_description, count\_vehicles) VALUES("second group" , "basic", 0);

INSERT INTO vehicle\_group (group\_name, group\_description, count\_vehicles) VALUES("third group" , "basic", 0);

INSERT INTO vehicle\_group (group\_name, group\_description, count\_vehicles) VALUES("fourth group" , "basic", 0);

INSERT INTO vehicle\_group (group\_name, group\_description, count\_vehicles) VALUES("fifth group" , "basic", 0);

INSERT INTO vehicle\_group (group\_name, group\_description, count\_vehicles) VALUES("six group" , "basic", 0);

INSERT INTO vehicle\_group (group\_name, group\_description, count\_vehicles) VALUES("seven group" , "basic", 0);

INSERT INTO vehicle\_group (group\_name, group\_description, count\_vehicles) VALUES("eight group" , "basic", 0);

INSERT INTO vehicle\_group (group\_name, group\_description, count\_vehicles) VALUES("nine group" , "basic", 0);

INSERT INTO vehicle\_group (group\_name, group\_description, count\_vehicles) VALUES("ten group" , "basic", 0);

INSERT INTO permissions (id\_group, id\_level) VALUES(1, 1);

INSERT INTO permissions (id\_group, id\_level) VALUES(2, 2);

INSERT INTO permissions (id\_group, id\_level) VALUES(3, 3);

INSERT INTO permissions (id\_group, id\_level) VALUES(4, 4);

INSERT INTO permissions (id\_group, id\_level) VALUES(5, 5);

INSERT INTO permissions (id\_group, id\_level) VALUES(6, 6);

INSERT INTO permissions (id\_group, id\_level) VALUES(7, 7);

INSERT INTO permissions (id\_group, id\_level) VALUES(8, 8);

INSERT INTO permissions (id\_group, id\_level) VALUES(9, 9);

INSERT INTO permissions (id\_group, id\_level) VALUES(10, 10);

INSERT INTO vehicle\_model (model\_name, c\_year, complectation\_name) VALUES("Mark II", "2000", "Full");

INSERT INTO vehicle\_model (model\_name, c\_year, complectation\_name) VALUES("e200", "2001", "Min");

Продолжение листинга 2.

INSERT INTO vehicle\_model (model\_name, c\_year, complectation\_name) VALUES("CR-V", "2004", "Full");

INSERT INTO vehicle\_model (model\_name, c\_year, complectation\_name) VALUES("Gets", "2010", "Middle");

INSERT INTO vehicle\_model (model\_name, c\_year, complectation\_name) VALUES("Lancer evo", "2008", "Sport");

INSERT INTO vehicle\_model (model\_name, c\_year, complectation\_name) VALUES("Matiz", "2011", "Min");

INSERT INTO vehicle\_model (model\_name, c\_year, complectation\_name) VALUES("Q5", "2021", "Min");

INSERT INTO vehicle\_model (model\_name, c\_year, complectation\_name) VALUES("m5", "2022", "Full");

INSERT INTO vehicle\_model (model\_name, c\_year, complectation\_name) VALUES("Tiguan", "2018", "Full");

INSERT INTO vehicle\_model (model\_name, c\_year, complectation\_name) VALUES("Kalina", "2006", "Top");

INSERT INTO vehicle\_brand (brand\_name, division) VALUES("Toyota", "Russia");

INSERT INTO vehicle\_brand (brand\_name, division) VALUES("Mercedes", "Russia");

INSERT INTO vehicle\_brand (brand\_name, division) VALUES("Honda", "Russia");

INSERT INTO vehicle\_brand (brand\_name, division) VALUES("Hundai", "Russia");

INSERT INTO vehicle\_brand (brand\_name, division) VALUES("Mitsubishi", "Russia");

INSERT INTO vehicle\_brand (brand\_name, division) VALUES("Daewoo", "Russia");

INSERT INTO vehicle\_brand (brand\_name, division) VALUES("Cherry", "Russia");

INSERT INTO vehicle\_brand (brand\_name, division) VALUES("BMW", "Russia");

INSERT INTO vehicle\_brand (brand\_name, division) VALUES("Wolkswagen", "Russia");

INSERT INTO vehicle\_brand (brand\_name, division) VALUES("Lada", "Russia");

INSERT INTO vehicle\_name (id\_model, id\_brand) VALUES(1, 1);

INSERT INTO vehicle\_name (id\_model, id\_brand) VALUES(2, 2);

INSERT INTO vehicle\_name (id\_model, id\_brand) VALUES(3, 3);

INSERT INTO vehicle\_name (id\_model, id\_brand) VALUES(4, 4);

INSERT INTO vehicle\_name (id\_model, id\_brand) VALUES(5, 5);

INSERT INTO vehicle\_name (id\_model, id\_brand) VALUES(6, 6);

INSERT INTO vehicle\_name (id\_model, id\_brand) VALUES(7, 7);

INSERT INTO vehicle\_name (id\_model, id\_brand) VALUES(8, 8);

INSERT INTO vehicle\_name (id\_model, id\_brand) VALUES(9, 9);

INSERT INTO vehicle\_name (id\_model, id\_brand) VALUES(10, 10);

INSERT INTO vehicle\_work\_model (price\_per\_hour, id\_vehicle\_name, model\_photo\_name, id\_group) VALUES(20.0, 1, "toymarkii2000", 1);

INSERT INTO vehicle\_work\_model (price\_per\_hour, id\_vehicle\_name, model\_photo\_name, id\_group) VALUES(25.5, 1, "mere2002001", 2);

INSERT INTO vehicle\_work\_model (price\_per\_hour, id\_vehicle\_name, model\_photo\_name, id\_group) VALUES(30.0, 1, "honcrv2004", 3);

INSERT INTO vehicle\_work\_model (price\_per\_hour, id\_vehicle\_name, model\_photo\_name, id\_group) VALUES(35.5, 1, "hungets2010", 4);

INSERT INTO vehicle\_work\_model (price\_per\_hour, id\_vehicle\_name, model\_photo\_name, id\_group) VALUES(40.0, 1, "mitlancerevo2008", 5);

Продолжение листинга 2.

INSERT INTO vehicle\_work\_model (price\_per\_hour, id\_vehicle\_name, model\_photo\_name, id\_group) VALUES(45.5, 1, "daematiz2011", 6);

INSERT INTO vehicle\_work\_model (price\_per\_hour, id\_vehicle\_name, model\_photo\_name, id\_group) VALUES(50.0, 1, "cherq52021", 7);

INSERT INTO vehicle\_work\_model (price\_per\_hour, id\_vehicle\_name, model\_photo\_name, id\_group) VALUES(55.5, 1, "bmwm52022", 8);

INSERT INTO vehicle\_work\_model (price\_per\_hour, id\_vehicle\_name, model\_photo\_name, id\_group) VALUES(60.0, 1, "woltiguan2018", 9);

INSERT INTO vehicle\_work\_model (price\_per\_hour, id\_vehicle\_name, model\_photo\_name, id\_group) VALUES(65.5, 1, "ladakalina2006", 10);

INSERT INTO vehicles (vin, color, state, place, id\_vehicle\_work\_model) VALUES("ABCDEFGHIJKLMNO01", "red", "inactive", "56,08,00;40,25,00", 1);

INSERT INTO vehicles (vin, color, state, place, id\_vehicle\_work\_model) VALUES("ABCDEFGHIJKLMNO02", "green", "inactive", "56,08,00;40,25,00", 2);

INSERT INTO vehicles (vin, color, state, place, id\_vehicle\_work\_model) VALUES("ABCDEFGHIJKLMNO03", "black", "inactive", "56,08,00;40,25,00", 3);

INSERT INTO vehicles (vin, color, state, place, id\_vehicle\_work\_model) VALUES("ABCDEFGHIJKLMNO04", "blue", "inactive", "56,08,00;40,25,00", 4);

INSERT INTO vehicles (vin, color, state, place, id\_vehicle\_work\_model) VALUES("ABCDEFGHIJKLMNO05", "blue", "inactive", "56,08,00;40,25,00", 5);

INSERT INTO vehicles (vin, color, state, place, id\_vehicle\_work\_model) VALUES("ABCDEFGHIJKLMNO06", "black", "inactive", "56,08,00;40,25,00", 6);

INSERT INTO vehicles (vin, color, state, place, id\_vehicle\_work\_model) VALUES("ABCDEFGHIJKLMNO07", "green", "inactive", "56,08,00;40,25,00", 7);

INSERT INTO vehicles (vin, color, state, place, id\_vehicle\_work\_model) VALUES("ABCDEFGHIJKLMNO08", "red", "inactive", "56,08,00;40,25,00", 8);

INSERT INTO vehicles (vin, color, state, place, id\_vehicle\_work\_model) VALUES("ABCDEFGHIJKLMNO09", "green", "inactive", "56,08,00;40,25,00", 9);

INSERT INTO vehicles (vin, color, state, place, id\_vehicle\_work\_model) VALUES("ABCDEFGHIJKLMNO10", "blakc", "inactive", "56,08,00;40,25,00", 10);

INSERT INTO users (snpassport, full\_name, date\_of\_birth, id\_level) VALUES(1718598304, "Stas", "2004-06-03", 1);

INSERT INTO users (snpassport, full\_name, date\_of\_birth, id\_level) VALUES(1718598305, "Roma", "1998-08-05", 2);

INSERT INTO users (snpassport, full\_name, date\_of\_birth, id\_level) VALUES(1718598306, "Vlad", "1995-05-26", 3);

INSERT INTO users (snpassport, full\_name, date\_of\_birth, id\_level) VALUES(1718598307, "Anton", "1999-06-13", 4);

INSERT INTO users (snpassport, full\_name, date\_of\_birth, id\_level) VALUES(1718598308, "Petya", "2002-04-13", 5);

INSERT INTO users (snpassport, full\_name, date\_of\_birth, id\_level) VALUES(1718598309, "Dima", "2014-04-03", 6);

INSERT INTO users (snpassport, full\_name, date\_of\_birth, id\_level) VALUES(1718598310, "Viktor", "2001-06-01", 7);

INSERT INTO users (snpassport, full\_name, date\_of\_birth, id\_level) VALUES(1718598311, "Egor", "2002-03-03", 8);

Продолжение листинга 2.

INSERT INTO users (snpassport, full\_name, date\_of\_birth, id\_level) VALUES(1718598312, "Nikita", "2003-05-03", 9);

INSERT INTO users (snpassport, full\_name, date\_of\_birth, id\_level) VALUES(1718598313, "Lexa", "2003-06-03", 10);

INSERT INTO rents (snpassport, vin, duration, starting\_point, ending\_point, start\_time, end\_time)

VALUES(1718598304,"ABCDEFGHIJKLMNO01", 0, "56,08,00;40,25,00", "56,08,00;40,25,00", "2023-09-19 09:00:00", "2023-09-19 09:00:00");

INSERT INTO rents (snpassport, vin, duration, starting\_point, ending\_point, start\_time, end\_time)

VALUES(1718598305,"ABCDEFGHIJKLMNO02", 0, "56,08,00;40,25,00", "56,08,00;40,25,00", "2023-09-19 09:00:00", "2023-09-19 09:00:00");

INSERT INTO rents (snpassport, vin, duration, starting\_point, ending\_point, start\_time, end\_time)

VALUES(1718598306,"ABCDEFGHIJKLMNO03", 0, "56,08,00;40,25,00", "56,08,00;40,25,00", "2023-09-19 09:00:00", "2023-09-19 09:00:00");

INSERT INTO rents (snpassport, vin, duration, starting\_point, ending\_point, start\_time, end\_time)

VALUES(1718598307,"ABCDEFGHIJKLMNO04", 0, "56,08,00;40,25,00", "56,08,00;40,25,00", "2023-09-19 09:00:00", "2023-09-19 09:00:00");

INSERT INTO rents (snpassport, vin, duration, starting\_point, ending\_point, start\_time, end\_time)

VALUES(1718598308,"ABCDEFGHIJKLMNO05", 0, "56,08,00;40,25,00", "56,08,00;40,25,00", "2023-09-19 09:00:00", "2023-09-19 09:00:00");

INSERT INTO rents (snpassport, vin, duration, starting\_point, ending\_point, start\_time, end\_time)

VALUES(1718598309,"ABCDEFGHIJKLMNO06", 0, "56,08,00;40,25,00", "56,08,00;40,25,00", "2023-09-19 09:00:00", "2023-09-19 09:00:00");

INSERT INTO rents (snpassport, vin, duration, starting\_point, ending\_point, start\_time, end\_time)

VALUES(1718598310,"ABCDEFGHIJKLMNO07", 0, "56,08,00;40,25,00", "56,08,00;40,25,00", "2023-09-19 09:00:00", "2023-09-19 09:00:00");

INSERT INTO rents (snpassport, vin, duration, starting\_point, ending\_point, start\_time, end\_time)

VALUES(1718598311,"ABCDEFGHIJKLMNO08", 0, "56,08,00;40,25,00", "56,08,00;40,25,00", "2023-09-19 09:00:00", "2023-09-19 09:00:00");

INSERT INTO rents (snpassport, vin, duration, starting\_point, ending\_point, start\_time, end\_time)

VALUES(1718598312,"ABCDEFGHIJKLMNO09", 0, "56,08,00;40,25,00", "56,08,00;40,25,00", "2023-09-19 09:00:00", "2023-09-19 09:00:00");

INSERT INTO rents (snpassport, vin, duration, starting\_point, ending\_point, start\_time, end\_time)

VALUES(1718598313,"ABCDEFGHIJKLMNO10", 0, "56,08,00;40,25,00", "56,08,00;40,25,00", "2023-09-19 09:00:00", "2023-09-19 09:00:00");

Итоговый список таблиц представлен на рисунке 25.

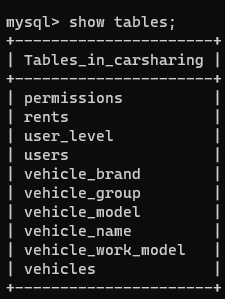


Рисунок 25 – Итоговый список таблиц

**Вывод:**

В результате данной практической работы была создана база данных, а также было произведено наполнение её тестовыми данными.

**Практическая работа 2**

**Цель**: изучение и создание выборки, сортировки данных. Изучение и применение операторов для изменения данных.

**Результат работы:**

Создание выборки всех колонок без фильтрации и сортировки таблицы “user\_level” с помощью команды SELECT представлено на рисунке 26.

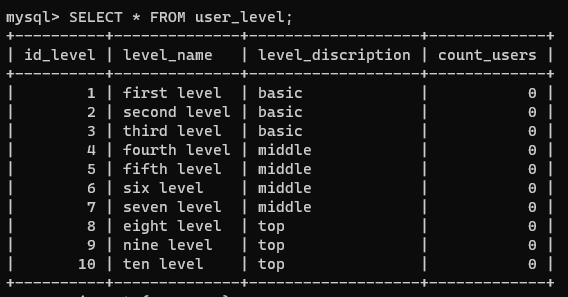


Рисунок 26 –Создание выборки

Создание выборки всех колонок с фильтром на колонку description таблицы “user\_level” с помощью команды SELECT и параметра WHERE представлено на рисунке 27.

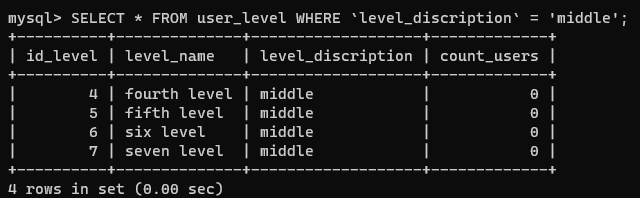


Рисунок 27 – Создание выборки с фильтрацией

Создание выборки всех колонок с фильтром на колонку description и на колонку id таблицы “user\_level” с помощью команды SELECT и параметра WHERE представлено на рисунке 28.

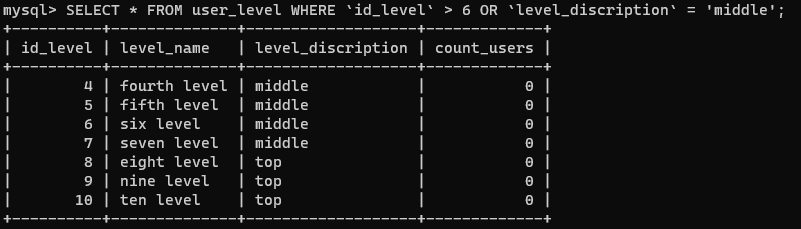


Рисунок 28 – Создание выборки с множественной фильтрацией

Создание выборки части колонок таблицы “user\_level” представлено на рисунке 29.

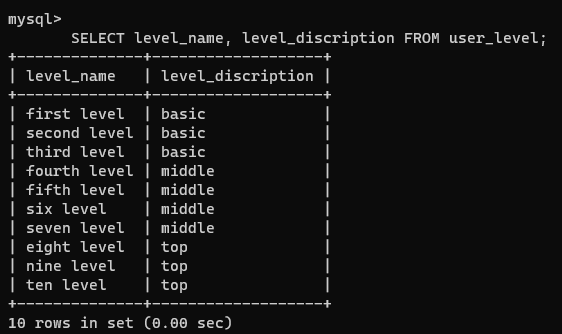


Рисунок 29 – Создание выборки с фильтрацией колонок

Создание выборки с объединением данных нескольких таблиц с помощью параметра JOIN представлено на рисунке 30.

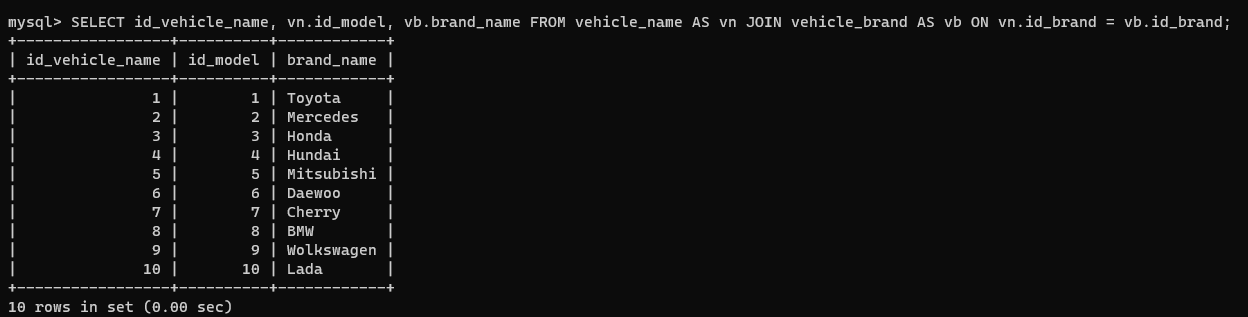


Рисунок 30 – Создание выборки с объединением

Создание выборки с множественным объединением данных нескольких таблиц с помощью параметра JOIN представлено на рисунке 31.

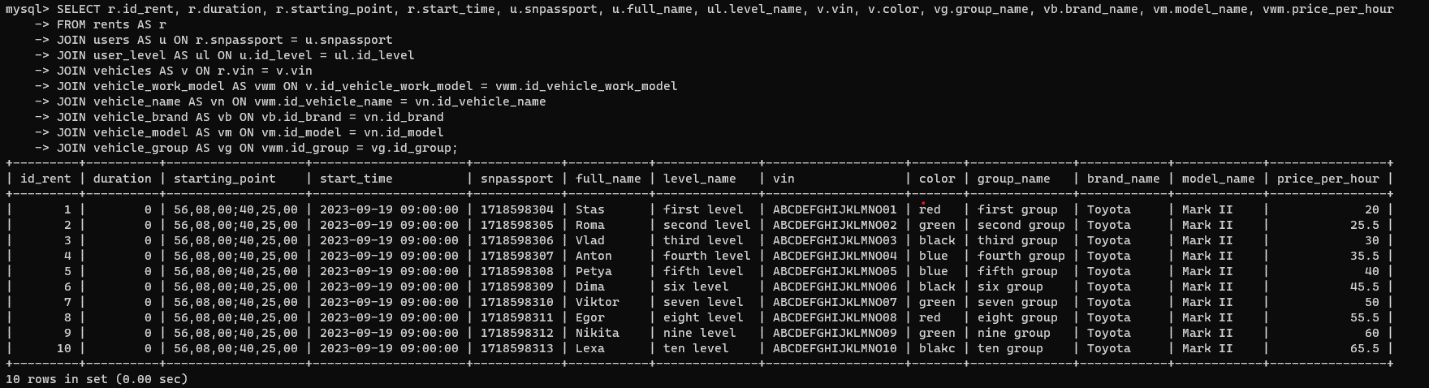


Рисунок 31 – Создание выборки с множественным объединением

Создание выборки из таблицы с использованием сортировки с помощью ORDER BY представлены на рисунке 32.



Рисунок 32 – Создание выборки с множественной сортировкой

Для обновления базы данных использовалась команда UPDATE результат выполнения представлен на рисунке 33.

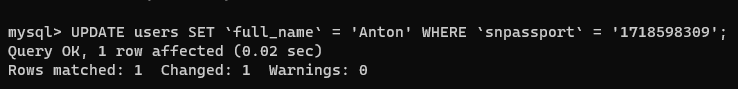


Рисунок 33 – Обновление данных в базе данных

Для удаления строки из таблицы “permissions” использовалась команда

DELETE, результат выполнения представлен на рисунке 34.

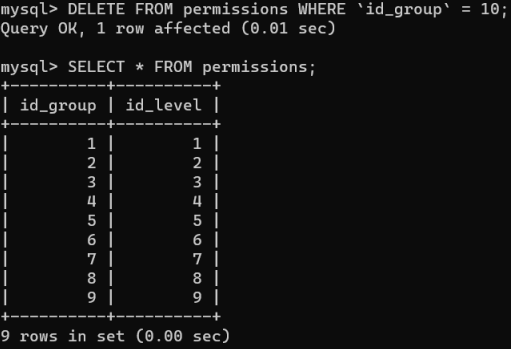


Рисунок 34 - Удаление данных из базы

Выборка внутри таблиц по определённому паттерну выполнялась с помощью параметра LIKE. (Рис. 35)

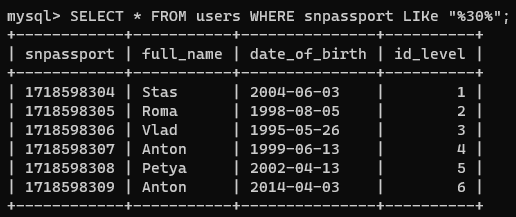


Рисунок 35 - Выборка с параметром LIKE

Содержимое команд выполнение которых было представлено выше представлено в листинге 1.

Листинг 3 – Содержимое выполненных команд.

SELECT \* FROM user\_level;

SELECT level\_name, level\_discription FROM user\_level;

SELECT \* FROM user\_level WHERE `level\_discription` = 'middle';

SELECT \* FROM user\_level WHERE `id` > 6 OR `level\_discription` = 'middle';

SELECT id\_vehicle\_name, vn.id\_model, vb.brand\_name FROM vehicle\_name AS vn JOIN vehicle\_brand AS vb ON vn.id\_brand = vb.id\_brand;

SELECT r.id\_rent, r.duration, r.starting\_point, r.start\_time, u.snpassport, u.full\_name, ul.level\_name, v.vin, v.color, vg.group\_name, vb.brand\_name, vm.model\_name, vwm.price\_per\_hour

FROM rents AS r

JOIN users AS u ON r.snpassport = u.snpassport

JOIN user\_level AS ul ON u.id\_level = ul.id\_level

JOIN vehicles AS v ON r.vin = v.vin

JOIN vehicle\_work\_model AS vwm ON v.id\_vehicle\_work\_model = vwm.id\_vehicle\_work\_model

JOIN vehicle\_name AS vn ON vwm.id\_vehicle\_name = vn.id\_vehicle\_name

JOIN vehicle\_brand AS vb ON vb.id\_brand = vn.id\_brand

JOIN vehicle\_model AS vm ON vm.id\_model = vn.id\_model

JOIN vehicle\_group AS vg ON vwm.id\_group = vg.id\_group;

SELECT r.id\_rent, r.duration, r.starting\_point, r.start\_time, u.snpassport, u.full\_name, ul.level\_name, v.vin, v.color, vg.group\_name, vb.brand\_name, vm.model\_name, vwm.price\_per\_hour

FROM rents AS r

JOIN users AS u ON r.snpassport = u.snpassport

JOIN user\_level AS ul ON u.id\_level = ul.id\_level

JOIN vehicles AS v ON r.vin = v.vin

JOIN vehicle\_work\_model AS vwm ON v.id\_vehicle\_work\_model = vwm.id\_vehicle\_work\_model

JOIN vehicle\_name AS vn ON vwm.id\_vehicle\_name = vn.id\_vehicle\_name

JOIN vehicle\_brand AS vb ON vb.id\_brand = vn.id\_brand

JOIN vehicle\_model AS vm ON vm.id\_model = vn.id\_model

JOIN vehicle\_group AS vg ON vwm.id\_group = vg.id\_group

ORDER BY u.snpassport ASC;

SELECT \* FROM users ORDER BY full\_name, id\_level DESC;

SELECT \* FROM users WHERE snpassport LIKE "%30%";

UPDATE users SET `full\_name` = 'Anton' WHERE `snpassport` = '1718598309';

**Вывод:**

В результате данной практической работы была создана база данных, а также было произведено наполнение её тестовыми данными.

**Практическая работа 3**

**Цель занятия:**

Написание триггеров, функций и процедур различной степени сложности.

**Постановка задачи:**

Изучить и создать несколько (по 3) триггеров, функций и процедур.

Результат работы в виде отчета должен содержать:

- снимок экрана (скриншоты) триггеров, функций и процедур

- снимок экрана (скриншоты) результата работы триггеров, функций и процедур;

- снимки экрана (скриншоты)

**Результат работы:**

**Написание триггеров**

Триггер, который обновляет статус машины если появилась аренда с ней.

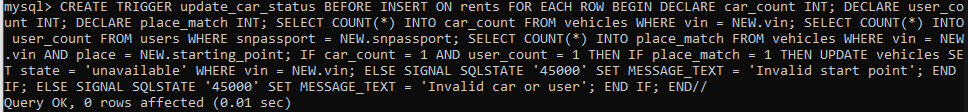


Рисунок 36 – Триггер update\_car\_status

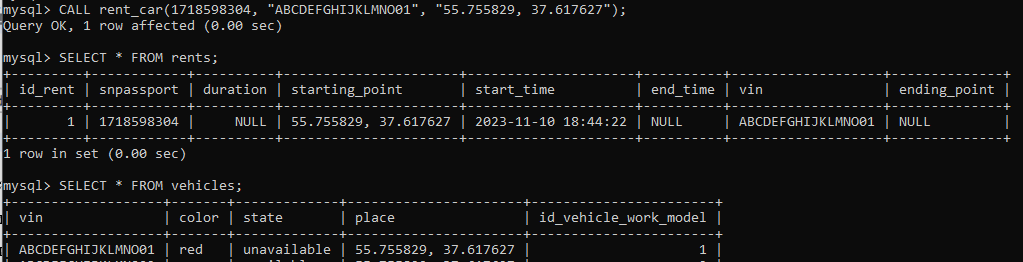


Рисунок 37 – Результат работы триггера

Триггер, который проверяет правильность данных и устанавливает базовые значения для place и state при создании новой машины представлен на рисунке 38.

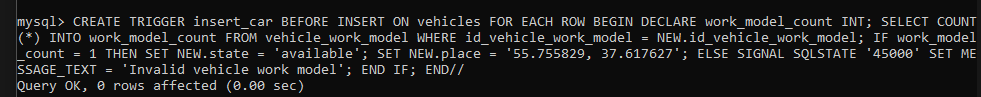


Рисунок 38 – Триггер insert\_car

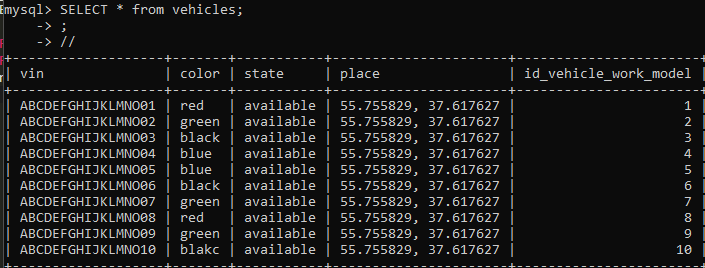


Рисунок 39 – Результат работы триггера при добавлении машины

Триггер, который после закрытия аренды устанавливает статус и место для машины, которая была задействована в аренде представлен на рисунке 40.



Рисунок 41 – Триггер close\_the\_rent

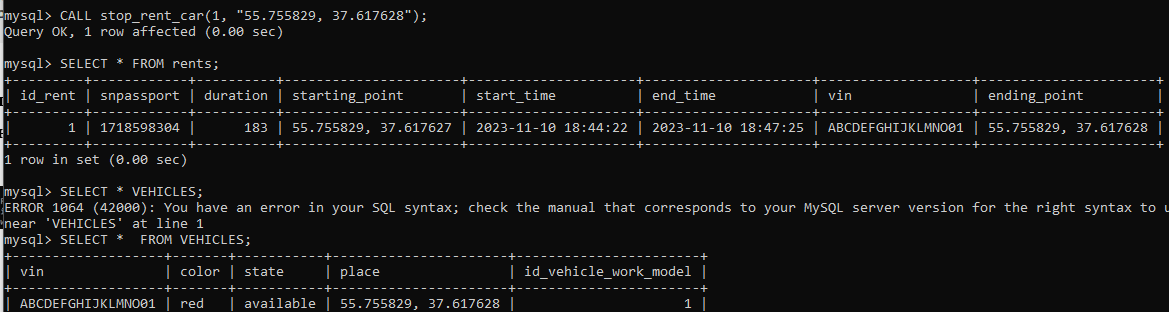


Рисунок 42 – Результат работы триггера

**Написание функций**

Функция, которая рассчитывает длительность аренды в зависимости от времени начала и конца.

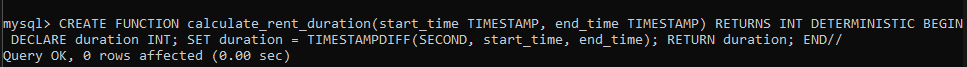


Рисунок 43 – Функция calculate\_rent\_duration

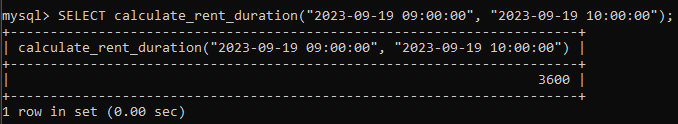


Рисунок 44 – Вызов функции

Функция, которая возвращает количество свободных машин;



Рисунок 45 – Функция get\_availeable\_vehicle\_count

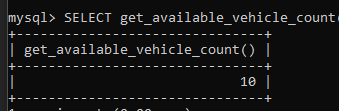


Рисунок 46 – Вызов функции

Функция, которая рассчитывает стоимость всех поездок пользователя.

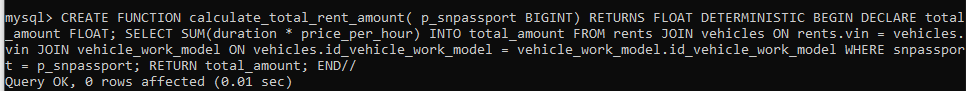


Рисунок 47 – Функция calculate\_total\_rent\_amount

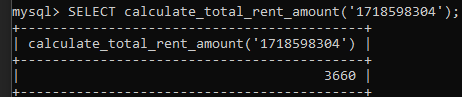


Рисунок 48 – Вызов функции

**Написание процедур**

Процедура, которая завершает аренду.

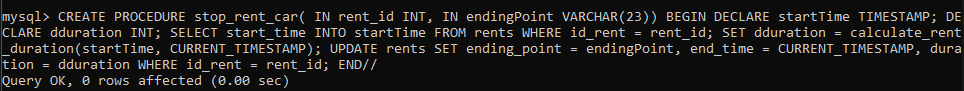


Рисунок 49 – Процедура stop\_rent\_car

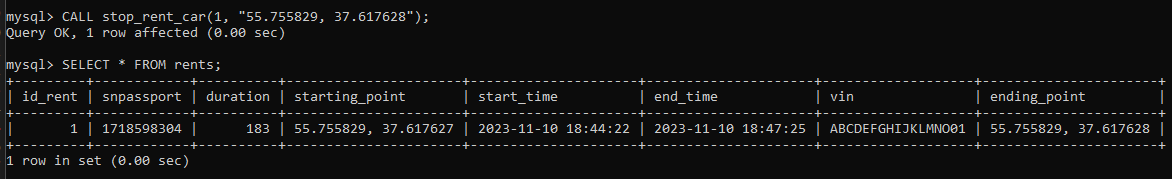


Рисунок 50 – Результат работы процедуры

Процедура, которая начинает аренду.

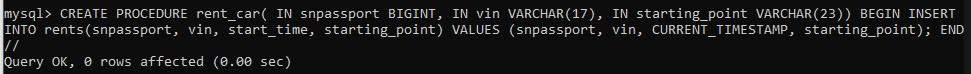


Рисунок 51 – Процедура rent\_car

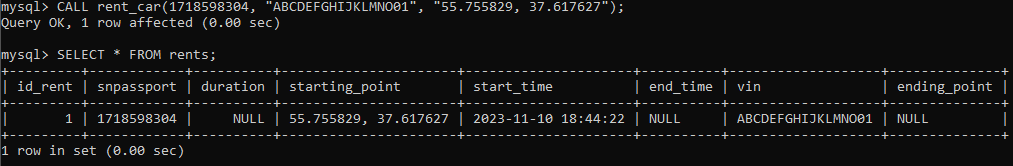


Рисунок 52 – результат вызова процедуры

Процедура, которая удаляет пользователя со всеми связанными данными.



Рисунок 53 – Процедура delete\_user

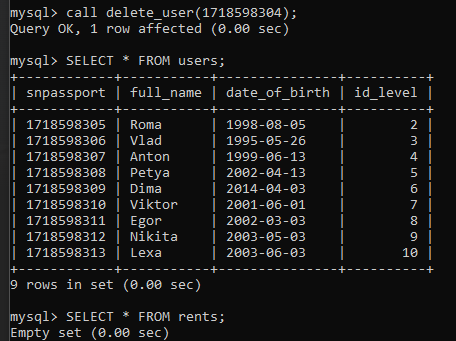


Рисунок 54 – Результат вызова процедуры

Для переноса базы данных использовался функционал MySQL Workbench – функция Export data. Результат представлен на рисунке 55.

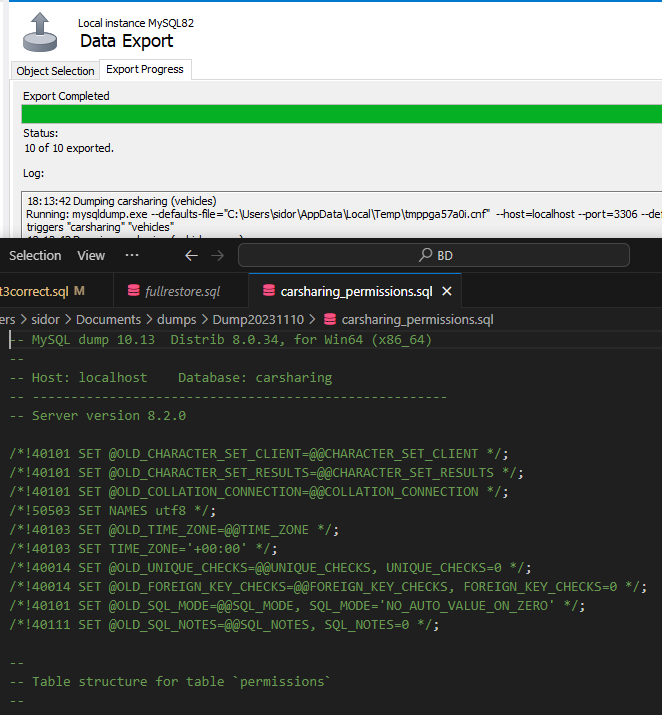


Рисунок 55 – Результат экспорта БД

Листинг 4 – Триггер 1

|  |
| --- |
| DELIMITER //  CREATE TRIGGER update\_car\_status  BEFORE INSERT ON rents  FOR EACH ROW  BEGIN  DECLARE car\_count INT;  DECLARE user\_count INT;  DECLARE place\_match INT;    SELECT COUNT(\*) INTO car\_count FROM vehicles WHERE vin = NEW.vin;  SELECT COUNT(\*) INTO user\_count FROM users WHERE snpassport = NEW.snpassport;  SELECT COUNT(\*) INTO place\_match FROM vehicles WHERE vin = NEW.vin AND place = NEW.starting\_point;    IF car\_count = 1 AND user\_count = 1 THEN  IF place\_match = 1 THEN  UPDATE vehicles SET state = 'unavailable' WHERE vin = NEW.vin;  ELSE  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Invalid start point';  END IF;  ELSE  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Invalid car or user';  END IF;  END//  DELIMITER ; |

Листинг 5 – Триггер 2

|  |
| --- |
| DELIMITER //  CREATE TRIGGER insert\_car  BEFORE INSERT ON vehicles  FOR EACH ROW  BEGIN  DECLARE work\_model\_count INT;  SELECT COUNT(\*) INTO work\_model\_count FROM vehicle\_work\_model WHERE id\_vehicle\_work\_model = NEW.id\_vehicle\_work\_model;    IF work\_model\_count = 1 THEN  SET NEW.state = 'available';  SET NEW.place = '55.755829, 37.617627';  ELSE  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Invalid vehicle work model';  END IF;  END//  DELIMITER ; |

Листинг 6 – Триггер 3

|  |
| --- |
| DELIMITER //  CREATE TRIGGER update\_car\_status  BEFORE INSERT ON rents  FOR EACH ROW  BEGIN  DECLARE car\_count INT;  DECLARE user\_count INT;  DECLARE place\_match INT;    SELECT COUNT(\*) INTO car\_count FROM vehicles WHERE vin = NEW.vin;  SELECT COUNT(\*) INTO user\_count FROM users WHERE snpassport = NEW.snpassport;  SELECT COUNT(\*) INTO place\_match FROM vehicles WHERE vin = NEW.vin AND place = NEW.starting\_point;    IF car\_count = 1 AND user\_count = 1 THEN  IF place\_match = 1 THEN  UPDATE vehicles SET state = 'unavailable' WHERE vin = NEW.vin;  ELSE  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Invalid start point';  END IF;  ELSE  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Invalid car or user';  END IF;  END//  DELIMITER ; |

Листинг 7 – Процедура 1

|  |
| --- |
| DELIMITER //  CREATE PROCEDURE delete\_user (IN p\_snpassport BIGINT)  BEGIN  DELETE FROM rents WHERE snpassport = p\_snpassport;  DELETE FROM users  WHERE snpassport = p\_snpassport;  END//  DELIMITER ; |

Листинг 8 – Процедура 2

|  |
| --- |
| DELIMITER //  CREATE PROCEDURE rent\_car(  IN snpassport BIGINT,  IN vin VARCHAR(17),  IN starting\_point VARCHAR(23)  )  BEGIN  INSERT INTO rents (snpassport, vin, start\_time, starting\_point)  VALUES (snpassport, vin, CURRENT\_TIMESTAMP, starting\_point);  END//  DELIMITER ; |

Листинг 9 – Процедура 3

|  |
| --- |
| DELIMITER //  CREATE PROCEDURE stop\_rent\_car(  IN rent\_id INT,  IN endingPoint VARCHAR(23)  )  BEGIN  DECLARE startTime TIMESTAMP;  DECLARE dduration INT;  SELECT start\_time INTO startTime  FROM rents  WHERE id\_rent = rent\_id;  SET dduration = calculate\_rent\_duration(startTime, CURRENT\_TIMESTAMP);  UPDATE rents  SET ending\_point = endingPoint,  end\_time = CURRENT\_TIMESTAMP,  duration = dduration  WHERE id\_rent = rent\_id;  END//  DELIMITER ; |

Листинг 10 – Функция 1

|  |
| --- |
| DELIMITER //  CREATE FUNCTION calculate\_total\_rent\_amount (  p\_snpassport BIGINT  )  RETURNS FLOAT  DETERMINISTIC  BEGIN  DECLARE total\_amount FLOAT;  SELECT SUM(duration \* price\_per\_hour) INTO total\_amount  FROM rents  JOIN vehicles ON rents.vin = vehicles.vin  JOIN vehicle\_work\_model ON vehicles.id\_vehicle\_work\_model = vehicle\_work\_model.id\_vehicle\_work\_model  WHERE snpassport = p\_snpassport;  RETURN total\_amount;  END//  DELIMITER ; |

Листинг 11 – Функция 2

|  |
| --- |
| DELIMITER //  CREATE FUNCTION get\_available\_vehicle\_count()  RETURNS INT  DETERMINISTIC  BEGIN  DECLARE count INT;  SELECT COUNT(\*) INTO count  FROM vehicles  WHERE state = 'available';  RETURN count;  END//  DELIMITER ; |

Листинг 12 – Функций 3

|  |
| --- |
| DELIMITER //  CREATE FUNCTION calculate\_rent\_duration(start\_time TIMESTAMP, end\_time TIMESTAMP) RETURNS INT  DETERMINISTIC  BEGIN  DECLARE duration INT;  SET duration = TIMESTAMPDIFF(SECOND, start\_time, end\_time);  RETURN duration;  END//  DELIMITER ; |

**Вывод:**

В результате данной практической работы были созданы три процедуры, три функции и три триггера. А также создан backup для базы данных.

**Практическая работа 4**

**Цель:** cсоздание оконных функций различных видов.

**Постановка задачи:** создать пять различных оконных функций используя каждый вид как минимум по одному разу.

**Результат работы:**

В ходе выполнения данной практической работы были созданы 5 различных функций.   
 Первая функция рассчитывает сумму длительностей всех аренд представленных в базе сгруппированных по пользователю. Результат работы функции представлен на рисунке 56. Код функции представлен в листинге 13.

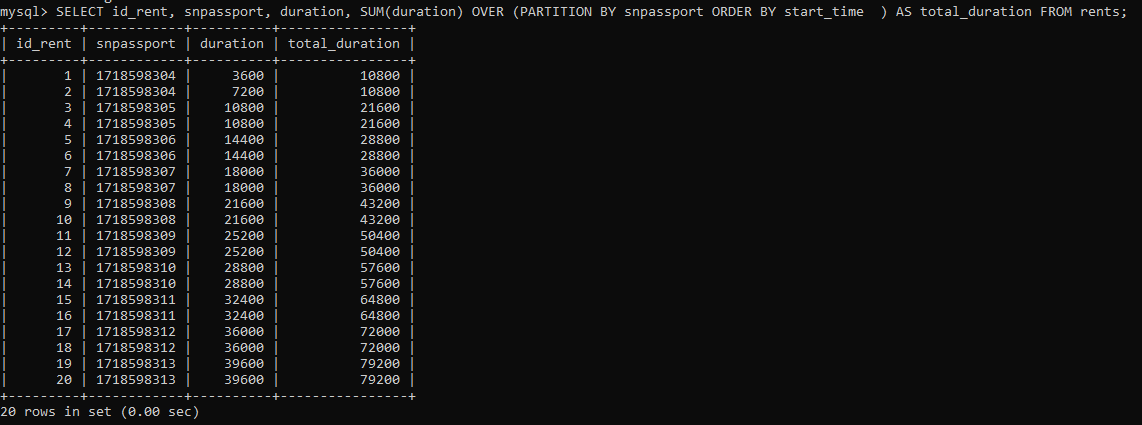


Рисунок 56 – Результат работы функции 1

Листинг 13 – Код функции 1

SELECT

    id\_rent,

    snpassport,

    duration,

    starting\_point,

    start\_time,

    end\_time,

    vin,

    ending\_point,

    SUM(duration) OVER (ORDER BY start\_time) AS total\_duration

FROM rents;

Вторая функция рассчитывает среднюю длительность всех аренд представленных в базе сгруппированных по рабочей модели т.с. . Результат работы функции представлен на рисунке 57. Код функции представлен в листинге 14.

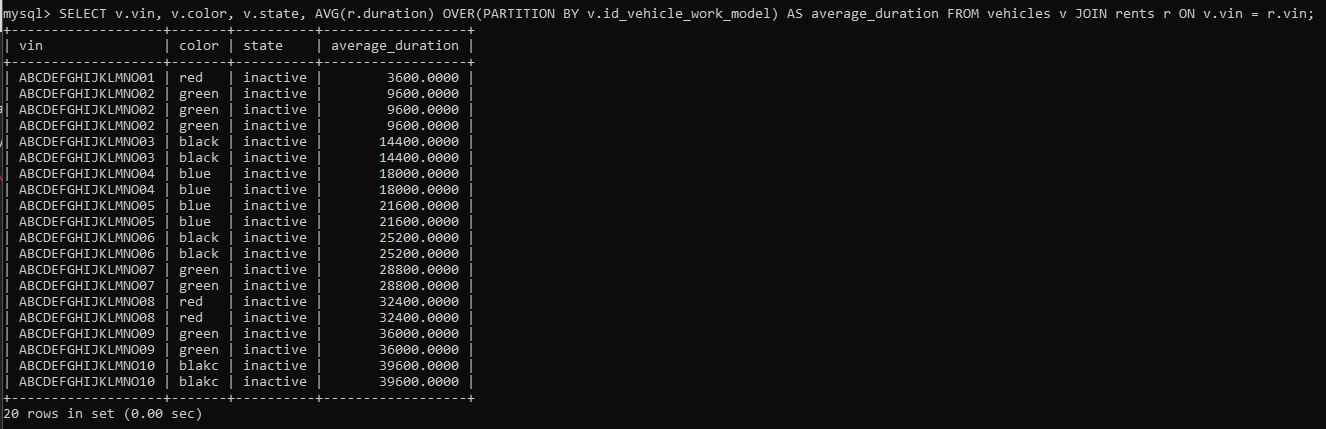


Рисунок 57 – Результат работы функции 2

Листинг 14 – Код функции 2

SELECT

v.vin,

v.color,

v.state,

AVG(r.duration) OVER (PARTITION BY v.id\_vehicle\_work\_model) AS average\_duration

FROM

vehicles v

JOIN

rents r ON v.vin = r.vin;

Третья функция проставляет ранги машинам в зависимости от средней длительности аренды, используя функцию DENSE\_RANK(). Результат работы функции представлен на рисунке 58. Код функции представлен в листинге 15.

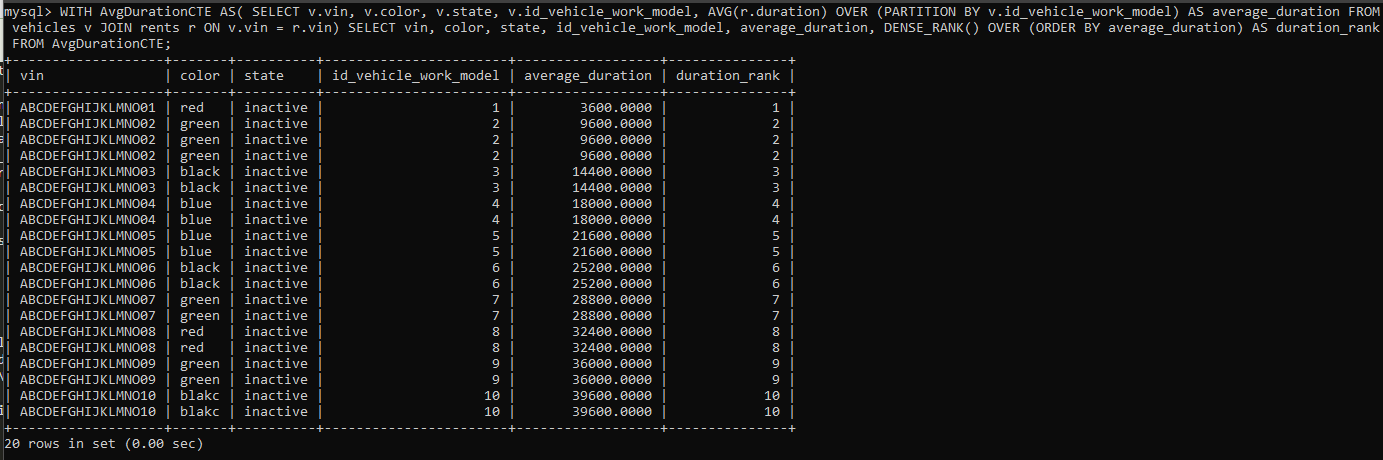


Рисунок 57 – Результат работы функции 3

Листинг 15 – Код функции 3

WITH AvgDurationCTE AS (

SELECT

v.vin,

v.color,

v.state,

v.id\_vehicle\_work\_model,

AVG(r.duration) OVER (PARTITION BY v.id\_vehicle\_work\_model) AS average\_duration

FROM

vehicles v

JOIN

rents r ON v.vin = r.vin

)

SELECT

vin,

color,

state,

id\_vehicle\_work\_model,

average\_duration,

DENSE\_RANK() OVER (ORDER BY average\_duration) AS duration\_rank

FROM

AvgDurationCTE;

Четвертая функция отображает предыдущую аренду пользователя в списке аренд, используя LAG(). Результат работы функции представлен на рисунке 58. Код функции представлен в листинге 16.

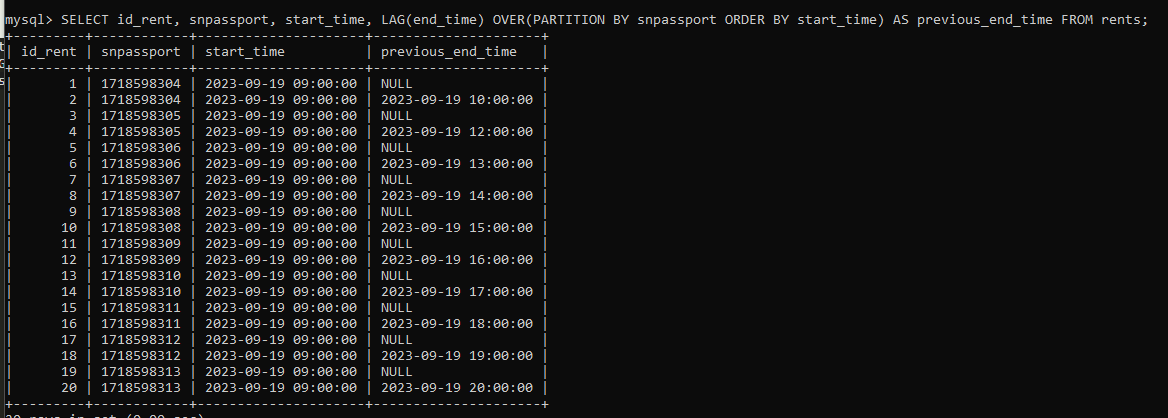


Рисунок 58 – Результат работы функции 4

Листинг 16 – Код функции 4

SELECT id\_rent, snpassport, start\_time,

LAG(end\_time) OVER (PARTITION BY snpassport ORDER BY start\_time) AS previous\_end\_time

FROM rents;

Пятая функция отображает конечное время первой поездки пользователя используя функцию FIRST\_VALUE(). Результат работы функции представлен на рисунке 59. Код функции представлен в листинге 17.

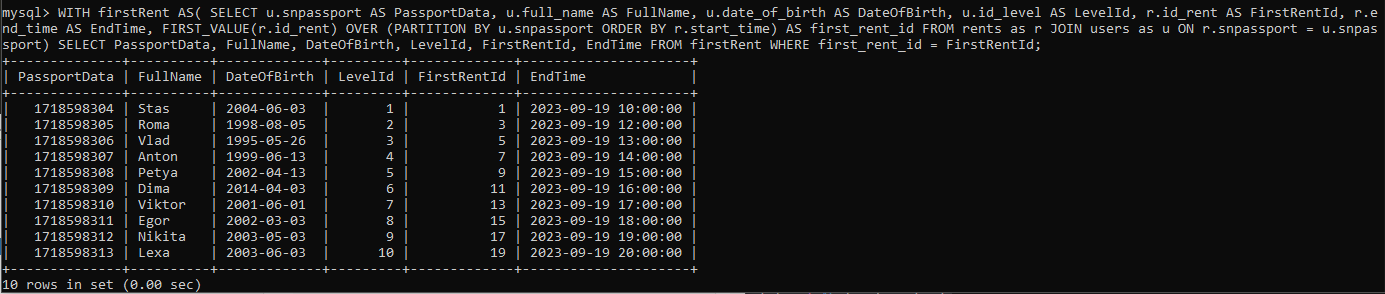


Рисунок 59 – Результат работы функции 5

Листинг 17 – Код функции 5

WITH firstRent AS (

SELECT

u.snpassport AS PassportData,

u.full\_name AS FullName,

u.date\_of\_birth AS DateOfBirth,

u.id\_level AS LevelId,

r.id\_rent AS FirstRentId,

r.end\_time AS EndTime,

FIRST\_VALUE(r.id\_rent) OVER (PARTITION BY u.snpassport ORDER BY r.start\_time) AS first\_rent\_id

FROM rents as r

JOIN users as u ON r.snpassport = u.snpassport

)

SELECT

PassportData,

FullName,

DateOfBirth,

LevelId,

FirstRentId,

EndTime

FROM firstRent

WHERE first\_rent\_id = FirstRentId;

**Вывод:**

В ходе выполнения данной практической работы были получены навыки работы с различными видами оконных функций.

**Список использованных источников и литературы:**

1. Методические указания к практической работе по дисциплине “Разработка баз данных” [Электронный ресурс]: Режим доступа: <https://online-edu.mirea.ru/mod/folder/view.php?id=263588> [Дата обращения : 13.09.2023]