ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – СОФИЯ

ФАКУЛТЕТ КОМПЮТЪРНИ СИСТЕМИ И ТЕХНОЛОГИИ

Катедра „Компютърни системи“

КУРСОВ ПРОЕКТ

ПО БАЗИ ОТ ДАННИ

Студент : Кристиян Иванов

ФАК. № : 123221009 **Група** : 43

Тема №24
Напишете база данни, в която да се пази информация за автомобили (номер на рама, модел, марка…, регистрационен номер). В отделна таблица да се пази информация за собственика на даден автомобил (един или повече). Направете регистър – кой собственик кой автомобил е притежавал. Също така трябва да има възможност за съхранение на информация за това дали автомобилът е бил на ремонт - къде е обслужван и т.н. Допълнете таблиците с необходима информация по ваш избор.

1. Да се проектира база от данни и да се представи ER диаграма със съответни CREATE TABLE заявки за средата MySQL.
2. Напишете заявка, в която демонстрирате SELECT с ограничаващо условие по избор.
3. Напишете заявка, в която използвате агрегатна функция и GROUP BY по ваш избор.
4. Напишете заявка, в която демонстрирате INNER JOIN по ваш избор.
5. Напишете заявка, в която демонстрирате OUTER JOIN по ваш избор.
6. Напишете заявка, в която демонстрирате вложен SELECT по ваш избор.
7. Напишете заявка, в която демонстрирате едновременно JOIN и агрегатна функция.

8. Създайте тригер по ваш избор.
9. Създайте процедура, в която демонстрирате използване на курсор.

1. Да се проектира база от данни и да се представи ER диаграма със съответни CREATE TABLE заявки за средата MySQL.

Според условието на задачата, необходимо е да бъдат съхранени данни за основните обекти - автомобили, техните собсвеници, регистри, ремонти. Допълнително ще създадем още няколко таблици.

Започваме с таблицата 'owners', в която ще пазим информация за собствениците на автомобили като име, фамилия, адрес, имейл и телефонен номер.

Следваща таблица, която ще създадем пази информация за автомобилите, включително номер на рама, модел, марка, година на производство, пробег, цвят, мощност на двигателя и тип каросерия.

Тъй като един собсвеник може да има повече коли, а една кола повече собсвеници, ще създадем отделна таблица 'registers', в която, освен външните ключове към двете таблици, ще пазим и датата на регистрация, номера на регистрацията и дали това е първата регистрация на автомобила

Следващата таблица е свързана с сервизите, където се извързват като ремонти, така и технически прегледи. В 'services' ще пазим името на сервиза, адреса, телефонния номер и имейл адреса.

Следващата таблица 'repairs' пази информация за ремонтите на автомобили, включително кой сервиз е извършил ремонта, датата на ремонта, описание на проблема, разходите за ремонта и частите, които са били заменени.

В следващите таблици ще пазим допълнителна информация за базата 'car_service'.

Таблицата 'technical_inspections' пази информация за техническите прегледи на автомобили, включително кой сервиз е извършил прегледа, датата на прегледа, статуса на автомобила (проходен или непроходен) и датата на следващия преглед.

Таблицата 'taxes' пази информация за данъците за автомобили, включително датата на плащане, разходите за ремонт, разходите за частите и общата сума на плащането.

Таблицата 'repairs_has_taxes' е между таблиците 'repairs' и 'taxes' и свързва ремонтите и данъците, които са свързани с този ремонт.

Таблицата 'cars_has_repairs' е между таблиците 'cars' и 'repairs' и свързва автомобилите с техните ремонти.

Таблицата 'spare_parts' пази информация за наличните резервни части за автомобили, включително име на частта и цена.

Таблицата 'repairs_has_taxes' съдържа връзката между ремонти и такси, които са свързани с тези ремонти. Тя има два външни ключа, един към таблицата 'repairs' и един към таблицата 'taxes'.

Таблицата 'cars_has_repairs' съдържа връзката между автомобили и ремонти, които са свързани с тези автомобили. Тя има два външни ключа, един към таблицата 'cars' и един към таблицата 'repairs'.

Таблицата 'spare_parts' съдържа информация за наличните резервни части и техните цени.

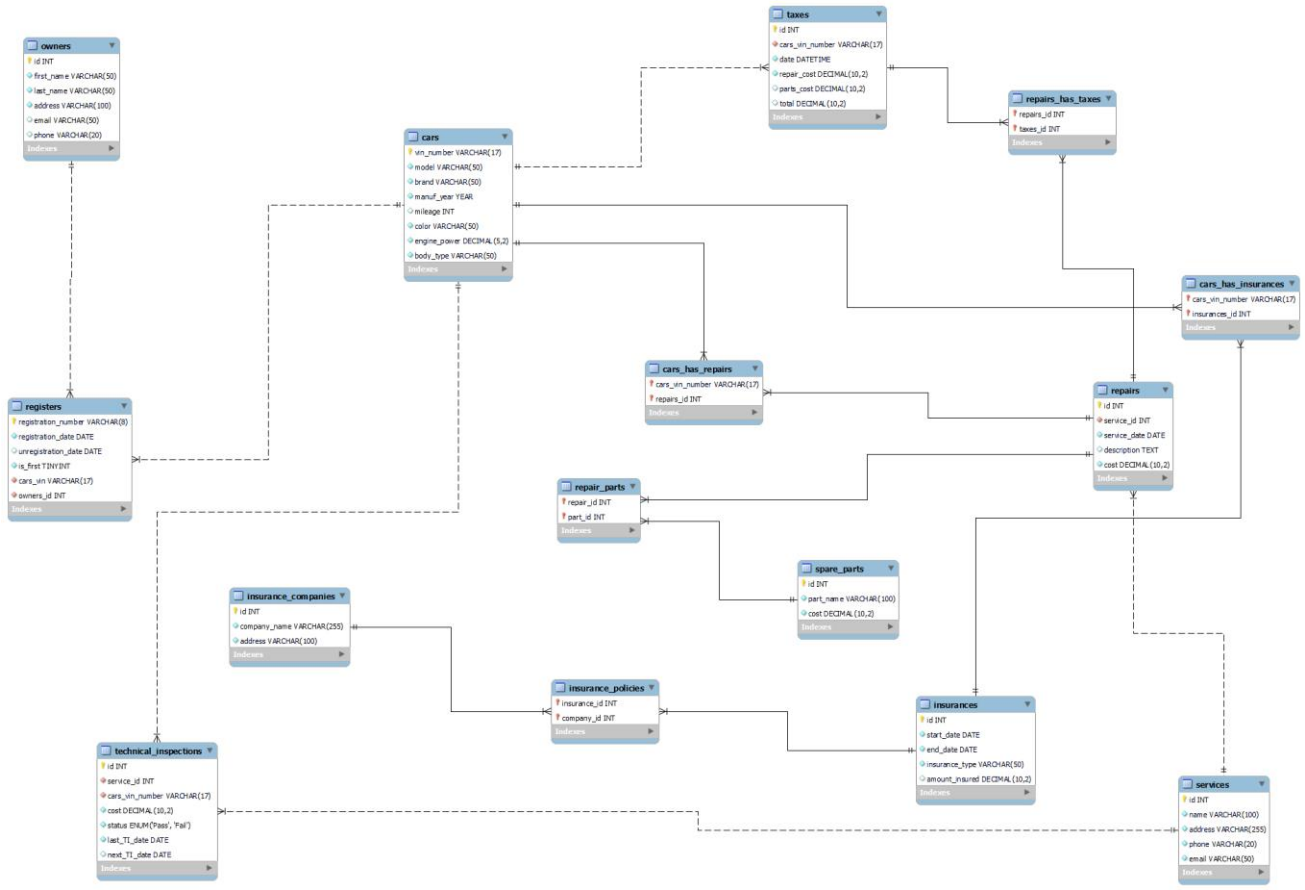
Таблицата 'repair_parts' съдържа връзката между ремонти и резервни части, които са използвани за тези ремонти. Тя има два външни ключа, един към таблицата 'repairs' и един към таблицата 'spare_parts'.

Таблицата 'insurance_companies' пази информацията за застрахователните компании, включително уникален идентификатор, име на компанията и адрес.

Таблицата 'insurances' пази информацията за застраховките, включително уникален идентификатор, дата на начало и край на застраховката, вид застраховка (Гражданска, Каско) и осигурена сума.

Таблицата 'insurance_policies' пази информацията за полиците, които свързват застраховките и застрахователните компании. Тя включва външни ключове към двете таблици.

Таблицата 'cars_has_insurances' пази информацията за връзката между застраховките и колите, външните ключове към двете таблици.



Заявките, с които създаваме базата данни и таблиците, са:

```
DROP DATABASE IF EXISTS car_service;
CREATE DATABASE car_service;
USE car_service;


CREATE TABLE IF NOT EXISTS car_service.`owners` (
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    address VARCHAR(100) NOT NULL,
    email VARCHAR(50) DEFAULT NULL,
    phone VARCHAR(20) DEFAULT NULL
);
```

```
CREATE TABLE IF NOT EXISTS car_service.`cars` (
    vin_number VARCHAR(17) PRIMARY KEY,
    model VARCHAR(50) NOT NULL,
    brand VARCHAR(50) NOT NULL,
    manuf_year YEAR NOT NULL,
    mileage INT DEFAULT 0,
    color VARCHAR(50) NOT NULL,
    engine_power DECIMAL(5, 2) NOT NULL,
    body_type VARCHAR(50) NOT NULL
);

CREATE TABLE IF NOT EXISTS car_service.`registers` (
    registration_number VARCHAR(10) PRIMARY KEY,
    registration_date DATE NOT NULL,
    unregistration_date DATE DEFAULT NULL,
    is_first TINYINT NOT NULL DEFAULT 1,
    cars_vin VARCHAR(17) NOT NULL,
    owners_id INT NOT NULL,
    CONSTRAINT FOREIGN KEY (cars_vin) REFERENCES car_service.`cars` (vin_number),
    CONSTRAINT FOREIGN KEY (owners_id) REFERENCES car_service.`owners` (id)
);

CREATE TABLE IF NOT EXISTS car_service.`services` (
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    address VARCHAR(255) NOT NULL,
    phone VARCHAR(20) NOT NULL,
    email VARCHAR(50) NOT NULL UNIQUE
);
```

```
CREATE TABLE IF NOT EXISTS car_service.`repairs` (
   id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
   service_id INT NOT NULL,
   service_date DATE NOT NULL,
   description TEXT DEFAULT NULL,
   cost DECIMAL(10, 2) NOT NULL,
   CONSTRAINT FOREIGN KEY (service_id) REFERENCES car_service.`services` (id)
);

CREATE TABLE IF NOT EXISTS car_service.`technical_inspections` (
   id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
   service_id INT NOT NULL,
   cars_vin_number VARCHAR(17) NOT NULL,
   cost DECIMAL(10, 2) NOT NULL,
   status ENUM("Pass", "Fail") NOT NULL,
   last_TI_date DATE NOT NULL,
   next_TI_date DATE DEFAULT NULL,
   CONSTRAINT FOREIGN KEY (service_id) REFERENCES car_service.`services` (id),
   CONSTRAINT FOREIGN KEY (cars_vin_number) REFERENCES car_service.`cars`
(vin_number)
);

CREATE TABLE IF NOT EXISTS car_service.`taxes` (
   id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
   cars_vin_number VARCHAR(17) NOT NULL,
   date DATETIME NOT NULL,
   repair_cost DECIMAL(10, 2) NOT NULL,
   parts_cost DECIMAL(10, 2) NULL,
   total DECIMAL(10, 2) NULL,
   CONSTRAINT FOREIGN KEY (cars_vin_number) REFERENCES car_service.`cars`
(vin_number)
);

CREATE TABLE IF NOT EXISTS car_service.`repairs_has_taxes` (
   repairs_id INT NOT NULL,
   taxes_id INT NOT NULL,
   PRIMARY KEY (repairs_id, taxes_id),
   CONSTRAINT FOREIGN KEY (repairs_id) REFERENCES car_service.`repairs` (id),
   CONSTRAINT FOREIGN KEY (taxes_id) REFERENCES car_service.`taxes` (id)
);

CREATE TABLE IF NOT EXISTS car_service.`cars_has_repairs` (
   cars_vin_number VARCHAR(17) NOT NULL,
   repairs_id INT NOT NULL,
   PRIMARY KEY (cars_vin_number, repairs_id),
   CONSTRAINT FOREIGN KEY (cars_vin_number) REFERENCES car_service.`cars`
(vin_number),
    CONSTRAINT FOREIGN KEY (repairs_id) REFERENCES car_service.`repairs` (id)
);
```

```
CREATE TABLE IF NOT EXISTS car_service.`spare_parts` (
   id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
   part_name VARCHAR(100) NOT NULL,
   cost DECIMAL(10,2) NOT NULL
);

CREATE TABLE IF NOT EXISTS car_service.`repair_parts` (
   repair_id INT NOT NULL,
   part_id INT NOT NULL,
   PRIMARY KEY (repair_id, part_id),
   CONSTRAINT FOREIGN KEY (repair_id) REFERENCES car_service.`repairs` (id),
   CONSTRAINT FOREIGN KEY (part_id) REFERENCES car_service.`spare_parts` (id)
);

CREATE TABLE IF NOT EXISTS car_service.`insurance_companies` (
   id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
   company_name VARCHAR(255) NOT NULL,
   address VARCHAR(100) NOT NULL
);

CREATE TABLE IF NOT EXISTS car_service.`insurances` (
   id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
   start_date DATE NOT NULL,
   end_date DATE NOT NULL,
   insurance_type VARCHAR(50) NOT NULL,
   amount_insured DECIMAL(10, 2) DEFAULT NULL
);

CREATE TABLE IF NOT EXISTS car_service.`insurance_policies` (
   insurance_id INT NOT NULL,
   company_id INT NOT NULL,
   PRIMARY KEY (insurance_id, company_id),
   CONSTRAINT FOREIGN KEY (insurance_id) REFERENCES car_service.`insurances` (id),
   CONSTRAINT FOREIGN KEY (company_id) REFERENCES
car_service.`insurance_companies` (id)
);

CREATE TABLE IF NOT EXISTS car_service.`cars_has_insurances` (
   cars_vin_number VARCHAR(17) NOT NULL,
   insurances_id INT NOT NULL,
   PRIMARY KEY (cars_vin_number, insurances_id),
   CONSTRAINT FOREIGN KEY (cars_vin_number) REFERENCES car_service.`cars`
(vin_number),
   CONSTRAINT FOREIGN KEY (insurances_id) REFERENCES car_service.`insurances` (id)
);
```

Добавяме тестови данни в таблиците:

```
INSERT INTO car_service.owners (first_name, last_name, address, email, phone)
VALUES  ('Иван', 'Иванов', 'ул. Георги Бенковски 10', 'ivanov123@abv.bg', '0888123456'),
        ('Петър', 'Петров', 'ул. Братя Миладинови 5', 'peterpetrov@gmail.com',
'0899123456'),
        ('Мария', 'Георгиева', 'жк. Люлин 9, бл. 912, вх. А', 'maria.georgieva@mail.bg',
'0877123456'),
        ('Анна', 'Иванова', 'жк. Младост 1А, бл. 509, вх. Г', 'anna_ivanova@gmail.com',
    '0882730947'),
        ('Иванка', 'Иванова', 'жк. Лозенец, ул. Якубица 5', 'ivanka.ivanova@mail.bg',
'0898123456'),
        ('Георги', 'Петков', 'ул. Бузлуджа 3', 'georgi_petkov@gmail.com', '0886123456'),
        ('Диана', 'Маринова', 'ул. Балканджийска 7', 'diana.marinova@yahoo.com',
'0895123456'),
        ('Александър', 'Илиев', 'жк. Младост 1, бл. 1', 'alex_iliev@mail.com', '0889123456'),
        ('Мирослав', 'Павлов', 'ул. Цар Симеон 50', 'miroslav.pavlov@gmail.com',
'0878123456'),
        ('Десислава', 'Иванова', 'жк. Дружба 2, бл. 231, вх. Б', 'desiivanova@abv.bg',
'0887123456'),
        ('Иван', 'Петров', 'бул. Сливница 123', 'ivan.petrov@gmail.com', '0873498281'),
        ('Ана', 'Андреева', 'бул. Цар Освободител 12', 'ana_andreeva@yahoo.com',
'0897421090'),
        ('Кристиян', 'Иванов', 'ул. Бачо Киро 15', 'ivanovkristiyan8@yahoo.com',
'0889123456'),
        ('Кирил', 'Георгиев', 'ул. 9-а 3', 'kiril.georgiev@abv.bg', '0897123456'),
        ('Иван', 'Петров', 'бул. Сливница 123', 'ivan.petrov@gmail.com', NULL),
        ('Петър', 'Георгиев', 'жк. Изток, ул. Андрей Сахаров 8', NULL, '0876123456'),
        ('Георги', 'Георгиев', 'ул. Бачо Киро 20', NULL, '0898123456'),
        ('Димитър', 'Петров', 'ул. Цар Самуил 29', NULL, NULL);
```

```
INSERT INTO car_service.cars (vin_number, model, brand, manuf_year, mileage, color,
engine_power, body_type)
VALUES  ('YS3FD45E7Y3021311', '9-5', 'Saab', 2000, 150000, 'Blue', 150, 'Sedan'),
        ('WBAAZ720X0ND79291', '3-Series', 'BMW', 2000, 200000, 'Silver', 170, 'Coupe'),
        ('JT2BG22K3V0085804', 'Camry', 'Toyota', 2020, 30000, 'Beige', 125, 'Sedan'),
        ('JTDKN3DU1C5467407', 'Prius', 'Toyota', 2012, 120000, 'Green', 98, 'Hatchback'),
        ('5N1DR2MN3HC665412', 'Pathfinder', 'Nissan', 2017, 80000, 'Black', 284, 'SUV'),
        ('1GCHK29U93E303886', 'Silverado', 'Chevrolet', 2003, 250000, 'Red', 300, 'Pickup'),
        ('JM1BK32FX41886212', 'Protege', 'Mazda', 2004, 150000, 'Silver', 130, 'Sedan'),
        ('JTHKD5BH4F2238764', 'CT', 'Lexus', 2015, 50000, 'White', 200, 'Hatchback'),
        ('1HGCR2F33EA312245', 'Accord', 'Honda', 2014, 70000, 'Gray', 185, 'Sedan'),
        ('WBA3B1C54EPV79899', '328i', 'BMW', 2014, 60000, 'Black', 240, 'Sedan'),
        ('1G1BE5SM4H7141438', 'Cruze', 'Chevrolet', 2017, 40000, 'Blue', 153, 'Sedan'),
        ('JN1CV6FE1HM301231', 'Q50', 'Infiniti', 2022, 35000, 'Red', 300, 'Sports car'),
        ('1N4AL3AP3FN381498', 'Altima', 'Nissan', 2015, 80000, 'Black', 182, 'Sedan'),
        ('1G1ZD5ST5JF173856', 'Malibu', 'Chevrolet', 2018, 50000, 'Silver', 160, 'Sedan'),
        ('5NPEB4AC3DH622462', 'Sonata', 'Hyundai', 2013, 90000, 'White', 190, 'Sedan'),
        ('1FTFX1EF3BKD79935', 'Focus', 'Ford', 2011, 150000, 'Red', 365, 'Wagon'),
        ('JF2GPADC9EH235352', 'XV Crosstrek', 'Subaru', 2014, 70000, 'Orange', 148, 'SUV'),
        ('5XXGM4A79EG314789', 'Optima', 'Kia', 2014, 90000, 'Black', 200, 'Sedan'),
        ('WBA7F0C50HG422488', '740i', 'BMW', 2017, 45000, 'Gray', 320, 'Sedan'),
        ('1N6AA1F47JN519184', 'Titan', 'Nissan', 2018, 40000, 'Red', 390, 'Pickup');

INSERT INTO car_service.registers (registration_number, registration_date, is_first, cars_vin,
owners_id)
VALUES  ('CA1234AB', '2020-01-01', 1, 'YS3FD45E7Y3021311', 1),
        ('CB5678CM', '2021-02-15', 1, 'WBAAZ720X0ND79291', 2),
        ('CT2468AB', '2019-06-10', 0, 'JT2BG22K3V0085804', 3),
        ('PB9012BC', '2022-04-01', 1, 'JTDKN3DU1C5467407', 4),
        ('E1234CD', '2021-05-10', 0, '5N1DR2MN3HC665412', 5),
        ('CA1111AB', '2020-12-31', 1, '1GCHK29U93E303886', 6),
        ('PB2222BC', '2021-06-15', 0, 'JM1BK32FX41886212', 7),
        ('PB3333BC', '2019-11-25', 0, 'JTHKD5BH4F2238764', 8),
        ('KH4444CD', '2020-07-15', 0, '1HGCR2F33EA312245', 9),
        ('BP5555AB', '2022-01-02', 1, 'WBA3B1C54EPV79899', 10),
        ('CB1357DF', '2022-04-22', 1, '1G1BE5SM4H7141438', 11),
        ('M2468CF', '2020-07-05', 0, '5NPEB4AC3DH622462', 11),
        ('CB1234GH', '2021-08-12', 0, '1GCHK29U93E303886', 12),
        ('CB5678IJ', '2022-02-10', 1, '5NPEB4AC3DH622462', 13),
        ('A2468KL', '2018-12-01', 0, '1FTFX1EF3BKD79935', 14),
        ('C1234MN', '2022-01-02', 0, '5N1DR2MN3HC665412', 15),
        ('CB5678OP', '2020-11-20', 0, '1FTFX1EF3BKD79935', 16),
        ('PK2468QR', '2021-05-15', 1, '5XXGM4A79EG314789', 17),
        ('B1234ST', '2019-09-30', 0, 'WBA7F0C50HG422488', 18);

INSERT INTO car_service.services (name, address, phone, email)
VALUES  ('Sofia Truck Center', 'ул. Христо Белчев 20', '0888123456', 'office@service-
truck.com'),
        ('Service Plovdiv', 'ул. Иван Вазов 35', '0899123456', 'info@service-plovdiv.bg'),
        ('BAT Service', 'ул. Тодор Каблешков 2', '0877123456', 'office@bat-service.com'),
        ('Varna Auto Service', 'ул. Сливница 15', '0888234567', 'office@varna-auto.bg'),
        ('Auto Moto Center', 'ул. Балканска 25', '0899123456', 'office@auto-moto-center.com');
```

```
INSERT INTO car_service.repairs (service_id, service_date, description, cost)
VALUES  (1, '2020-02-01', 'Смяна на масло и филтри', 120),
        (2, '2021-03-15', 'Ремонт на двигател', 800),
        (3, '2021-05-10', 'Ремонт на окачване', 550),
        (3, '2022-01-02', 'Смяна на акумулатор', 600),
        (4, '2022-03-20', 'Ремонт на скоростна кутия', 1200),
        (5, '2021-03-15', 'Ремонт на двигател', 750);

INSERT INTO car_service.technical_inspections (service_id, cars_vin_number, cost, status,
last_TI_date, next_TI_date)
VALUES  (1, "JTHKD5BH4F2238764", 50, "Pass", "2021-01-01", "2022-01-01"),
        (2, "5XXGM4A79EG314789", 50, "Pass", "2021-02-15", "2022-02-15"),
        (2, "JT2BG22K3V0085804", 75, "Fail", "2021-03-20", NULL),
        (3, "WBA7F0C50HG422488", 60, "Pass", "2021-04-10", "2022-04-10"),
        (3, "1GCHK29U93E303886", 75, "Fail", "2021-05-05", NULL);

INSERT INTO car_service.taxes (cars_vin_number, date, repair_cost, parts_cost, total)
VALUES  ('JT2BG22K3V0085804', '2022-01-01', 75, 25, 100),
        ('JM1BK32FX41886212', '2022-02-15', 150, 0, 150),
        ('WBA3B1C54EPV79899', '2022-03-10', 200, 50, 250),
        ('JF2GPADC9EH235352', '2022-04-20', 50, 20, 70),
        ('1N6AA1F47JN519184', '2022-05-15', 300, 100, 400);

INSERT INTO car_service.repairs_has_taxes (repairs_id, taxes_id)
VALUES  (1, 1),
        (2, 2),
        (2, 1),
        (3, 2),
        (3, 3);

INSERT INTO car_service.cars_has_repairs (cars_vin_number, repairs_id)
VALUES  ('YS3FD45E7Y3021311', 1),
        ('YS3FD45E7Y3021311', 4),
        ('WBAAZ720X0ND79291', 2),
        ('JTHKD5BH4F2238764', 3),
        ('1N4AL3AP3FN381498', 4),
        ('5XXGM4A79EG314789', 5);

INSERT INTO car_service.spare_parts (part_name, cost)
VALUES  ('Oil filter', 10.50),
        ('Air filter', 15.20),
        ('Brake pads', 50.00),
        ('Timing belt', 98.60),
        ('Spark plugs', 30.00);

INSERT INTO car_service.repair_parts (repair_id, part_id)
VALUES  (1, 1),
        (1, 2),
        (2, 3),
        (3, 2),
        (4, 1);
```

```
INSERT INTO car_service.insurance_companies (company_name, address)
VALUES  ('Бул Инс АД', 'ул. „Околовръстен път“ 238, 1330 София'),
        ('ДЗИ - Генерали АД', 'ул. „Гурко“ 25, 1000 София'),
        ('Евроинс ООД', 'бул. „Цар Освободител“ 87, 1000 София');

INSERT INTO car_service.insurances (start_date, end_date, insurance_type, amount_insured)
VALUES  ('2022-01-01', '2023-01-01', 'Liability', 100000.00),
        ('2022-02-15', '2023-02-15', 'Comprehensive', 50000.00),
        ('2022-07-15', '2023-07-15', 'Liability', 100000.00),
        ('2022-08-25', '2023-08-25', 'Comprehensive', 50000.00),
        ('2022-09-05', '2023-09-05', 'Liability', 75000.00);

INSERT INTO car_service.insurance_policies (insurance_id, company_id)
VALUES  (1, 1),
        (2, 2),
        (3, 1),
        (4, 2),
        (4, 3);

INSERT INTO car_service.cars_has_insurances (cars_vin_number, insurances_id)
VALUES  ('JM1BK32FX41886212', 1),
        ('1N6AA1F47JN519184', 2),
        ('5N1DR2MN3HC665412', 3),
        ('WBA7F0C50HG422488', 4),
        ('5NPEB4AC3DH622462', 5);
```

2. Напишете заявка, в която демонстрирате SELECT с ограничаващо условие по избор.

Ще изведем номера на рама, модела и регистрационния номер на всички коли с марка BMW.

```
SELECT cars.vin_number, cars.model, registers.registration_number
FROM cars
JOIN registers
ON cars_vin = cars.vin_number
WHERE brand = 'BMW';
```

vin_number	model	registration_number
WBA3B1C54EPV79899	328i	BP5555AB
WBA7F0C50HG422488	740i	B1234ST
WBAAZ720X0ND79291	3-Series	CB5678CM

3. Напишете заявка, в която използвате агрегатна функция и GROUP BY по ваш избор.

Ще изведем номера на рама, модела, марката и броя на извършените ремонти за колите с повече от 1 ремонт.

```
SELECT c.vin_number, c.brand, c.model, COUNT(cr.repairs_id) AS num_repairs
FROM cars c
JOIN cars_has_repairs cr
ON c.vin_number = cr.cars_vin_number
JOIN repairs r ON cr.repairs_id = r.id
GROUP BY c.vin_number
HAVING num_repairs > 1;
```

	vin_number	brand	model	num_repairs
▶	YS3FD45E7Y3021311	Saab	9-5	2

4. Напишете заявка, в която демонстрирате INNER JOIN по ваш избор.

Ще напишем заявка, с която ще изведем името и имейла на собствениците, както и модела, марката и годината на производство на колите, произведени между 2015 и 2020 година, подредени по пробег в низходящ ред.

```
SELECT concat(owners.first_name, owners.last_name) AS owner, owners.email, cars.brand,
cars.model, cars.manuf_year
FROM car_service.owners
INNER JOIN car_service.registers
ON owners.id = registers.owners_id
INNER JOIN car_service.cars
ON cars_vin = cars.vin_number
WHERE cars.manuf_year BETWEEN 2015 AND 2020
ORDER BY cars.mileage DESC;
```

	owner	email	brand	model	manuf_year
▶	Иван Петров	ivan.petrov@gmail.com	Nissan	Pathfinder	2017
	Иванка Иванова	ivanka.ivanova@mail.bg	Nissan	Pathfinder	2017
	Александър Илиев	alex_iliev@mail.com	Lexus	CT	2015
	Димитър Петров	NULL	BMW	740i	2017
	Иван Петров	ivan.petrov@gmail.com	Chevrolet	Cruze	2017
	Мария Георгиева	maria.georgieva@mail.bg	Toyota	Camry	2020

5. Напишете заявка, в която демонстрирате OUTER JOIN по ваш избор.

Ще изведем модела, марката и номера на рама на всички коли, независимо дали за тях са плащани такси. За целта ще използваме LEFT JOIN.

```
SELECT c.brand, c.model, c.vin_number, t.total
FROM cars c
LEFT OUTER JOIN taxes t
ON c.vin_number = t.cars_vin_number;
```

	brand	model	vin_number	total
▶	Ford	Focus	1FTFX1EF3BKD79935	NULL
	Chevrolet	Cruze	1G1BE5SM4H7141438	NULL
	Chevrolet	Malibu	1G1ZD5ST5JF173856	NULL
	Chevrolet	Silverado	1GCHK29U93E303886	NULL
	Honda	Accord	1HGCR2F33EA312245	NULL
	Nissan	Altima	1N4AL3AP3FN381498	NULL
	Nissan	Titan	1N6AA1F47JN519184	400.00
	Nissan	Pathfinder	5N1DR2MN3HC665412	NULL
	Hyundai	Sonata	5NPEB4AC3DH622462	NULL
	Kia	Optima	5XXGM4A79EG314789	NULL
	Subaru	XV Crosst...	JF2GPADC9EH235352	70.00
	Mazda	Protege	JM1BK32FX41886212	150.00
	Infiniti	Q50	JN1CV6FE1HM301231	NULL
	Toyota	Camry	JT2BG22K3V0085804	100.00
	Toyota	Prius	JTDKN3DU1C5467407	NULL
	Lexus	CT	JTHKD5BH4F2238764	NULL
	BMW	328i	WBA3B1C54EPV79899	250.00
	BMW	740i	WBA7F0C50HG422488	NULL
	BMW	3-Series	WBAAZ720X0ND79291	NULL
	Saab	9-5	YS3FD45E7Y3021311	NULL

6. Напишете заявка, в която демонстрирате вложен SELECT по ваш избор.

Ще напишем заявка, с която ще изведем името на собственика на колата и имейла му за тези коли, които са регистрирани след 01.01.2021..

```
SELECT first_name, last_name, email
FROM owners
WHERE id IN (
        SELECT owners_id
        FROM registers
        WHERE registration_date > '2022-01-01'
);
```

first_name	last_name	email
Десислава	Иванова	desiivanova@abv.bg
Иван	Петров	ivan.petrov@gmail.com
Иван	Петров	ivan.petrov@gmail.com
Кристиян	Иванов	ivanovkristiyna8@yahoo.com
Анна	Иванова	anna_ivanova@gmail.com

7. Напишете заявка, в която демонстрирате едновременно JOIN и агрегатна функция.

Ще изведем броя на всички сервизни записи за всяка кола, сортирани по възходящ ред на броя на записите, за всички коли, произведени след 2010 година.

```
SELECT cars.vin_number, COUNT(registers.registration_number) as num_services
FROM cars
JOIN registers ON cars.vin_number = registers.cars_vin
WHERE cars.manuf_year > 2010
GROUP BY cars.vin_number
ORDER BY num_services ASC;
```

	vin_number	num_services
▶	1G1BE5SM4H7141438	1
	1HGCR2F33EA312245	1
	5XXGM4A79EG314789	1
	JT2BG22K3V0085804	1
	JTDKN3DU1C5467407	1
	JTHKD5BH4F2238764	1
	WBA3B1C54EPV79899	1
	WBA7F0C50HG422488	1
	1FTFX1EF3BKD79935	2
	5N1DR2MN3HC665412	2
	5NPEB4AC3DH622462	2

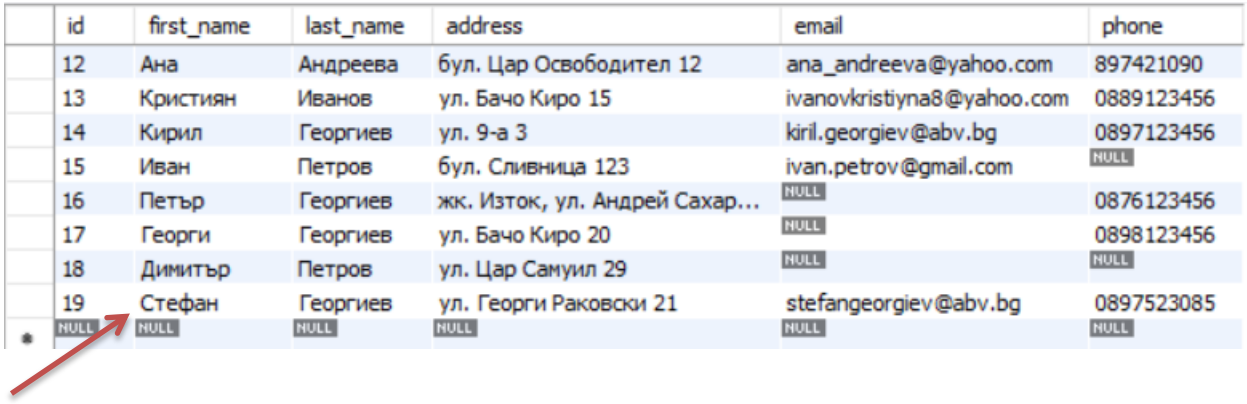
8. Създайте тригер по ваш избор.

Ще създадем тригер, който да проверява телефона на собственика и го добавя в таблицата 'owners', освен в следните случаи: ако той няма телефонен номер, или сме го грешно въвели. Тригерът ще изписва различни съобщения в зависимост от въведения номер.

```
DELIMITER &&
DROP TRIGGER IF EXISTS insert_owners_trigger;
CREATE TRIGGER insert_owners_trigger
BEFORE INSERT ON owners
FOR EACH ROW
BEGIN

	IF (NEW.phone IS NULL) THEN
		SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Phone number is missing!';

	ELSEIF (NEW.phone NOT REGEXP '^[0-9]+$') THEN
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Letters are not allowed in phone number!';

	ELSEIF LENGTH(NEW.phone) < 10 THEN
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Phone number must be at least 10 digits!';

	END IF;

END
&&
DELIMITER ;
```

При правилно вмъкнати данни:

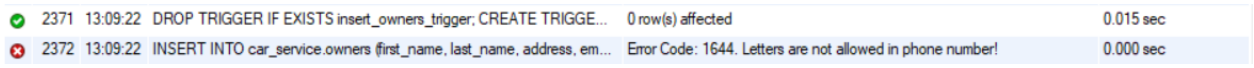
```
INSERT INTO car_service.owners (first_name, last_name, address, email, phone)
VALUE  ('Стефан', 'Георгиев', 'ул. Георги Раковски 21', 'stefangeorgiev@abv.bg', '0897523085');
```



id	first_name	last_name	address	email	phone
12	Ана	Андреева	бул. Цар Освободител 12	ana_andreeva@yahoo.com	897421090
13	Кристиян	Иванов	ул. Бачо Киро 15	ivanovkristiyna8@yahoo.com	0889123456
14	Кирил	Георгиев	ул. 9-а 3	kiril.georgiev@abv.bg	0897123456
15	Иван	Петров	бул. Сливница 123	ivan.petrov@gmail.com	NULL
16	Петър	Георгиев	жк. Изток, ул. Андрей Сахар...	NULL	0876123456
17	Георги	Георгиев	ул. Бачо Киро 20	NULL	0898123456
18	Димитър	Петров	ул. Цар Самуил 29	NULL	NULL
19	Стефан	Георгиев	ул. Георги Раковски 21	stefangeorgiev@abv.bg	0897523085
NULL	NULL	NULL	NULL	NULL	NULL

При грешно въведени данни:

```
INSERT INTO car_service.owners (first_name, last_name, address, email, phone)
VALUE  ('Стефан', 'Георгиев', 'ул. Георги Раковски 21', 'stefangeorgiev@abv.bg', '087749I216');
```



✓	2371	13:09:22	DROP TRIGGER IF EXISTS insert_owners_trigger; CREATE TRIGGE...	0 row(s) affected	0.015 sec
✗	2372	13:09:22	INSERT INTO car_service.owners (first_name, last_name, address, em...	Error Code: 1644. Letters are not allowed in phone number!	0.000 sec

9. Създайте процедура, в която демонстрирате използване на курсор.

Ще създадем процедура, която като входен параметър приема годината на производство на автомобила и в резултантната таблица извежда само тези коли, които са произведени преди 2010 година, информация за собственика на автомобила и дали е минал технически преглед. Курсорът ще обходи таблиците, чиито колони участват в резултантното решение.

```
DELIMITER |

DROP PROCEDURE IF EXISTS getCarInfo;
CREATE PROCEDURE getCarInfo(IN carYear INT)
BEGIN
	DECLARE cursorFinished INT DEFAULT 0;
	DECLARE tempCarBrand VARCHAR(255);
	DECLARE tempCarModel VARCHAR(255);
	DECLARE tempCarVINNumber VARCHAR(100);
	DECLARE tempOwnerName VARCHAR(255);
	DECLARE tempOwnerAddress VARCHAR(255);
	DECLARE tempOwnerPhone VARCHAR(255);
	DECLARE tempOwnerEmail VARCHAR(255);
	DECLARE ispassedTI VARCHAR(50);

	DECLARE tempCursor CURSOR FOR
		SELECT DISTINCT cars.brand, cars.model, cars.vin_number,
		owners.first_name, owners.address, owners.phone, owners.email,
		ti.status
		FROM owners
		JOIN registers ON registers.owners_id = owners.id
		JOIN cars ON cars.vin_number = registers.cars_vin
		JOIN technical_inspections ti ON cars.vin_number = ti.cars_vin_number
		WHERE cars.manuf_year < 2010
		ORDER BY cars.vin_number, cars.brand, cars.model;

	DECLARE CONTINUE HANDLER FOR NOT FOUND SET cursorFinished = 1;

	DROP TABLE IF EXISTS tempCarInfo;
	CREATE TEMPORARY TABLE tempCarInfo (
		brand VARCHAR(255) NOT NULL,
		model VARCHAR(255) NOT NULL,
		vin_number VARCHAR(17) NOT NULL,
		owner_name VARCHAR(255) NOT NULL,
		owner_address VARCHAR(255) NOT NULL,
		owner_phone VARCHAR(255) NOT NULL,
		owner_email VARCHAR(255) NOT NULL,
		isPassedTI VARCHAR(50) NOT NULL
	) Engine = Memory;
```

```
        OPEN tempCursor;

                cursorLoop: LOOP
                        FETCH tempCursor INTO tempCarBrand, tempCarModel,
        tempCarVINNumber, tempOwnerName, tempOwnerAddress,
                                tempOwnerPhone, tempOwnerEmail, isPassedTI;

                        IF cursorFinished
                    THEN
                                LEAVE cursorLoop;
                        END IF;

                        INSERT INTO tempCarInfo(brand, model, vin_number,
                                owner_name, owner_address, owner_phone, owner_email, isPassedTI)
                        VALUES(tempCarBrand, tempCarModel, tempCarVINNumber,
        tempOwnerName, tempOwnerAddress, tempOwnerPhone,
                                tempOwnerEmail, isPassedTI);
                END LOOP;

                CLOSE tempCursor;

                SELECT * FROM tempCarInfo ORDER BY brand, model, vin_number;

                DROP TABLE tempCarInfo;

END;
|

DELIMITER ;

CALL getCarInfo(2009);
```

	brand	model	vin_number	owner_name	owner_address	owner_phone	owner_email	isPassedTI
▶	Chevrolet	Silverado	1GCHK29U93E303886	Георги	ул. Бузлуджа 3	0886123456	georgi_petkov@gmail.com	Fail