

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ  
ІГОРЯ СІКОРСЬКОГО»  
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ  
(повна назва інституту/факультету)

КАФЕДРА інформатики та програмної інженерії  
(повна назва кафедри)

**КУРСОВА РОБОТА**

з дисципліни «Бази даних»  
(назва дисципліни)

на тему: “База даних для підтримки діяльності авторинку”

Студента 2 курсу ІІ-13 групи  
спеціальності 121 «Інженерія  
програмного забезпечення»  
Недельчева Є. О.  
(прізвище та ініціали)

Керівник Ліщук Олександр Васильович  
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала \_\_\_\_\_

Кількість балів: \_\_\_\_ Оцінка ECTS \_\_\_\_\_

Члени комісії

_____	_____
(підпис)	(вчене звання, науковий ступінь, прізвище та ініціали)
_____	_____
(підпис)	(вчене звання, науковий ступінь, прізвище та ініціали)
_____	_____
(підпис)	(вчене звання, науковий ступінь, прізвище та ініціали)

Київ – 2022 рік

**Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»**

Факультет Інформатики та обчислювальної техніки  
(повна назва)

Кафедра Інформатики та програмної інженерії  
(повна назва)

Дисципліна Бази даних

Курс 2 Група ІІІ-13 Семестр 3

**З А В Д А Н Н Я  
НА КУРСОВУ РОБОТУ СТУДЕНТУ**

Недельчеву Євгену Олександровичу  
(прізвище, ім'я, по батькові)

1. Тема роботи “ База даних для підтримки діяльності авторинку ”  
керівник роботи: Лішук Олександр Васильович  
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
2. Строк подання студентом роботи 10.01.2023
3. Вихідні дані до роботи завдання на розробку бази даних підтримки для підтримки діяльності авторинку
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
  - a) Аналіз предметного середовища
  - b) Побудова ER-моделі
  - c) Побудова реляційної схеми з ER-моделі
  - d) Створення бази даних, у форматі обраної СУБД
  - e) Створення користувачів бази даних
  - f) Імпорт даних з використанням засобів СУБД в створену базу даних
  - g) Створення мовою SQL запитів
  - h) Оптимізація роботи запитів
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)  

---
6. Дата видачі завдання 08.11.2022

## КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання курсового проекту	Строк виконання етапів проекту	Примітка
1	Аналіз предметного середовища	29.12.2022	
2	Побудова ER-моделі	29.12.2022	
3	Побудова реляційної схеми з ER-моделі	29.12.2022	
4	Створення бази даних, у форматі обраної системи управління базою даних	01.01.2023	
5	Створення користувачів бази даних	01.01.2023	
6	Імпорт даних з використанням засобів СУБД в створену базу даних	02.01.2023	
7	Створення мовою SQL запитів	04.01.2023	
8	Оптимізація роботи запитів	04.01.2023	
9	Оформлення пояснювальної записки	06.01.2023	
10	Захист курсової роботи	11.01.2023	

**Студент**

\_\_\_\_\_  
(підпис )

Недельчев Є.О.

(прізвище та ініціали)

**Керівник роботи**

\_\_\_\_\_  
(підпис )

Ліщук О.В.

(прізвище та ініціали)

## Зміст

<b>ВСТУП.....</b>	<b>5</b>
<b>ОПИС ПРЕДМЕТНОГО СЕРЕДОВИЩА.....</b>	<b>6</b>
<b>ПОСТАНОВКА ЗАДАЧІ .....</b>	<b>7</b>
<b>ER-діаграма .....</b>	<b>9</b>
Бізнес правила.....	9
Вибір сутностей .....	9
Набір атрибутів сутностей.....	10
<b>РЕЛЯЦІЙНА МОДЕЛЬ БАЗИ ДАНИХ .....</b>	<b>14</b>
Побудова необхідних відношень, визначення первинних та зовнішніх ключів. .....	14
<b>РЕАЛІЗАЦІЯ БАЗИ ДАНИХ .....</b>	<b>15</b>
Створення бази даних .....	15
Імпортування даних .....	19
<b>СТВОРЕННЯ КОРИСТУВАЧІВ БАЗИ ДАНИХ.....</b>	<b>20</b>
Розробник .....	20
Продавець.....	20
Покупець .....	20
<b>SQL запити .....</b>	<b>21</b>
Створення тригерів на таблиці.....	21
Створення процедур.....	25
Створення функцій.....	27
Створення представлень .....	29
Створення різних запитів .....	34
Створення індексів .....	42
<b>ВИСНОВОК .....</b>	<b>43</b>
<b>ПЕРЕЛІК ПОСИЛАНЬ.....</b>	<b>44</b>

## **ВСТУП**

У сучасному світі бази даних відіграють важливу роль у зберіганні, організації та управлінні великими обсягами даних. Вони є важливою частиною сучасних обчислювальних систем, які використовуються в широкому діапазоні галузей і застосувань.

Бази даних дозволяють організаціям ефективно зберігати та отримувати великі обсяги структурованих даних, наприклад інформацію про клієнтів, записи про продажі та дані про запаси. Вони також надають інструменти для організації, обробки й аналізу даних, що дозволяє отримувати цінну інформацію та приймати обґрунтовані рішення.

Крім того, бази даних є центральним сховищем даних, що забезпечує їх послідовність, точність і актуальність. Вони також підтримують такі заходи безпеки, як автентифікація користувачів і шифрування даних для захисту конфіденційної інформації від несанкціонованого доступу.

У цій курсовій роботі представлена база даних для підтримки діяльності авторинку.

## **ОПИС ПРЕДМЕТНОГО СЕРЕДОВИЩА**

Авторинок – це місце, де люди можуть продати свій автомобіль, або придбати інший. Зазвичай авторинки знаходяться на великих відкритих площадках, де володарі можуть зручно розташувати свої авто. Зазвичай угоди про купівлю/продаж укладаються просто на місці.

Перед тим, як виставити свій автомобіль на продаж, продавці надають детальну інформацію про свої автівки, щоб покупці могли зробити правильний вибір. Покупці ж перед придбанням авто детально ознайомлюються з хар-ками бажаних авто.

Предметне середовище для авторинку містить різні об'єкти та сутності, які пов'язані з його функціонуванням. Основні об'єкти, які можуть бути включені до такого середовища:

- Автомобілі
- Характеристики авто (двигун, ходова частина)
- Володарі авто
- Покупці

## ПОСТАНОВКА ЗАДАЧІ

Метою даної роботи є розробка бази даних для підтримки діяльності кіностудії. Результатом проектування повинна бути БД з визначеною структурою, заповнена даними та оптимізована для потреб користувача. Необхідно створити об'єкти, які покращать роботу розробника та користувача.

Завданням курсової роботи є розробка бази даних і її використання для вирішення практичних задач.

При розробці бази даних необхідно враховувати:

- вимоги до функціональності (наявність усіх функцій, які необхідні для реалізації поставленої задачі);
- вимоги до цілісності даних;
- вимоги до мінімізації об'єму даних, що зберігаються;
- наявність багатокористувальницького режиму;
- вимоги до швидкодії.

В процесі роботи над курсовою роботою повинні бути виконані наступні завдання:

- побудувати ER-модель, для чого необхідно:
  - детально проаналізувати предметне середовище;
  - сформулювати бізнес-правила, які будуть основою завдання обмежень при проектуванні та реалізації бази даних;
  - виявити необхідний набір сутностей;
  - визначити необхідний набір атрибутів для кожної сутності; - визначити зв'язки між об'єктами;
  - описати отриману ER-модель в одній з відомих нотацій;
  - розробити модель користувачів бази даних з описом їх прав;
- побудувати реляційну схему з ER-моделі, для чого необхідно:
  - побудувати набір необхідних відношень бази даних;
  - виділити первинні і зовнішні ключі у кожному з відношень;

- привести отримані відношення до третьої нормальної формі;
  - визначити обмеження цілісності для спроектованих відношень;
- створити базу даних, що була спроектована, у форматі обраної системи управління базою даних (СУБД);
- створити користувачів бази даних, реалізуючи розроблену багатокористувальницьку модель;
- імпортувати дані з використанням засобів СУБД в створену базу даних;
- мовою SQL написати запити для визначених на етапі аналізу предметного середовища потреб користувачів;
- оптимізувати роботу запитів (продемонструвати роботу до і після оптимізації).



## ER-діаграма

### Бізнес правила

Для правильної роботи БД необхідно ввести певні бізнес-правила:

- 1) Кожен автомобіль може бути одночасно виставлений на продаж лише один раз.
- 2) Кожен продавець може виставити на продаж будь-яку кількість автівок.
- 3) Кожен покупець може купити будь-яку кількість автівок.
- 4) Кожна автівка може мати декілька полісів страхування, в тому числі протерміновані, а може не мати жодного.

### Вибір сутностей

Вибір сутностей для Бази Даних виглядатиме наступним чином:

- Car\_accidents
- Engine
- Transmission
- Insurance
- Sellers
- Cars
- Listings
- Car\_types
- Transaction
- Buyers

## Набор атрибутів сутностей

Таблиця 3.1 – Сутності та їх атрибути.

Сутність	Атрибути
Car_accidents	<b>accident_id</b> car_id event date_of_event
Engine	<b>engine_id</b> engine_type horse_power engine_size
Transmission	<b>transmission_id</b> transmission_type type_of_drive
Insurance	<b>insurance_id</b> car_id provider policy_number start_date end_date

Продовження таблиці 3.1

Cars	<b>car_id</b> engine_id transmission_id type_id brand model color year mileage
Sellers	<b>seller_id</b> first_name last_name email phone
Listing	<b>listing_id</b> seller_id car_id price listing_date
Car_type	<b>type_id</b> typename amount_of_doors amount_of_seats

Продовження таблиці 3.1

Transaction	<b>transaction_id</b> listing_id buyer_id price transaction_date
Buyer	<b>buyer_id</b> first_name last_name email phone

Сутність Car\_accident буде пов'язана **багато до одного** із сутністю Car, адже кожна автівка може мати декілька ДТП.

Сутність Insurance буде пов'язана **багато до одного** із сутністю Car, адже кожна автівка може мати декілька страхових полісів.

Сутність Car буде пов'язана **багато до одного** із сутністю Engine, адже декілька автівок можуть мати однаковий тип двигуна.

Сутність Car буде пов'язана **багато до одного** із сутністю Transmission, адже декілька автівок можуть мати однаковий тип трансмісії.

Сутність Car буде пов'язана **багато до одного** із сутністю Car\_types, адже декілька автівок можуть мати однаковий тип кузова.

Сутність Car буде пов'язана **один до одного** із сутністю Listing, адже одна автівка може бути виставлена на продаж лише 1 раз.

Сутність Listing буде пов'язана **багато до одного** із сутністю Seller, адже кожен продавець може мати декілька об'яв про продаж авто.

Сутність Transaction буде пов'язана **один до одного** із сутністю Listing, адже кожна об'ява про продаж може мати лише одну угоду.

Сутність Transaction буде пов'язана **один до багатьох** із сутністю Buyer, адже кожен покупець може укласти декілька угод про купівлю авто.

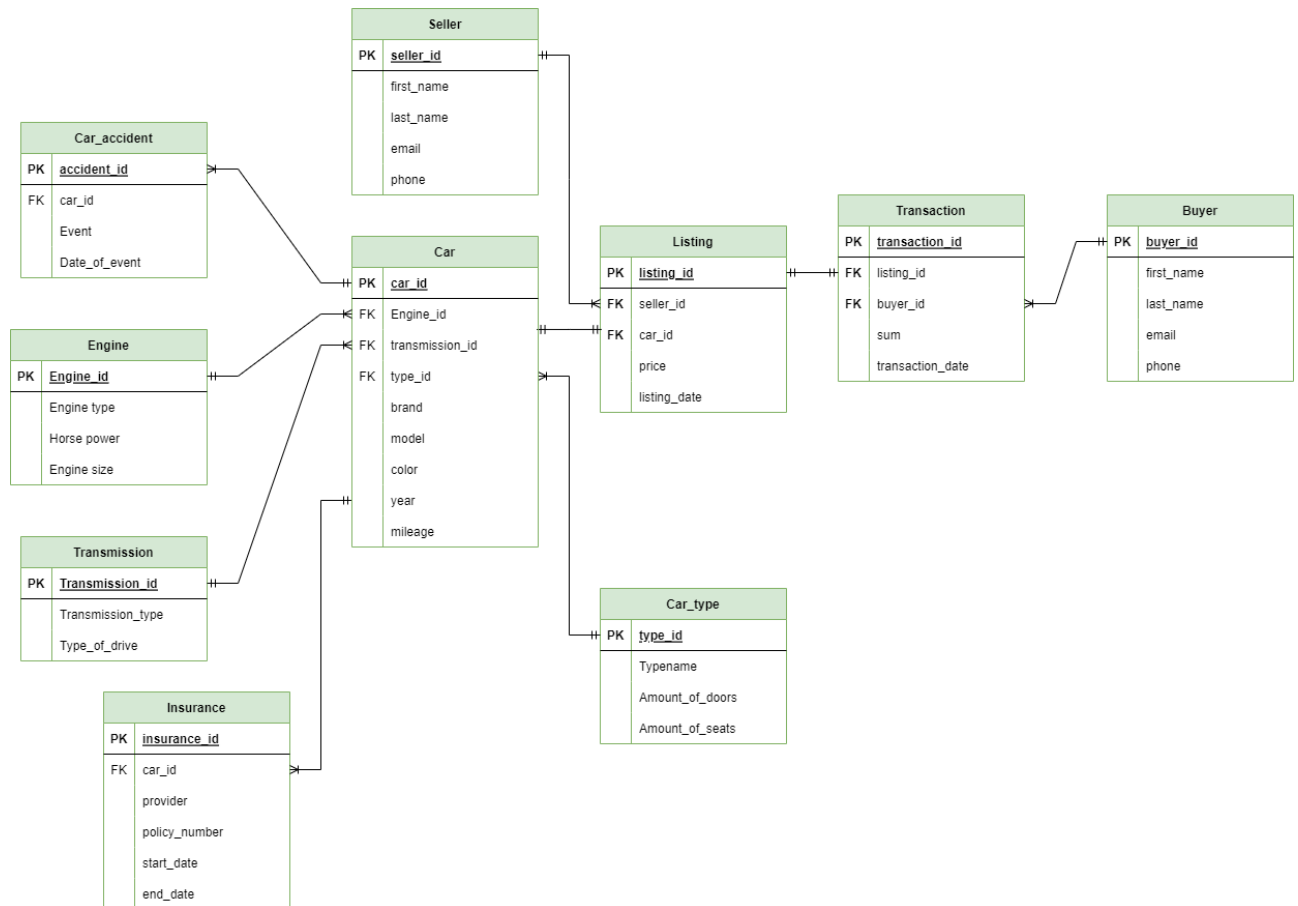


Рисунок 3.1 – ER-діаграма

## РЕЛЯЦІЙНА МОДЕЛЬ БАЗИ ДАНИХ

Побудова необхідних відношень, визначення первинних та зовнішніх ключів.

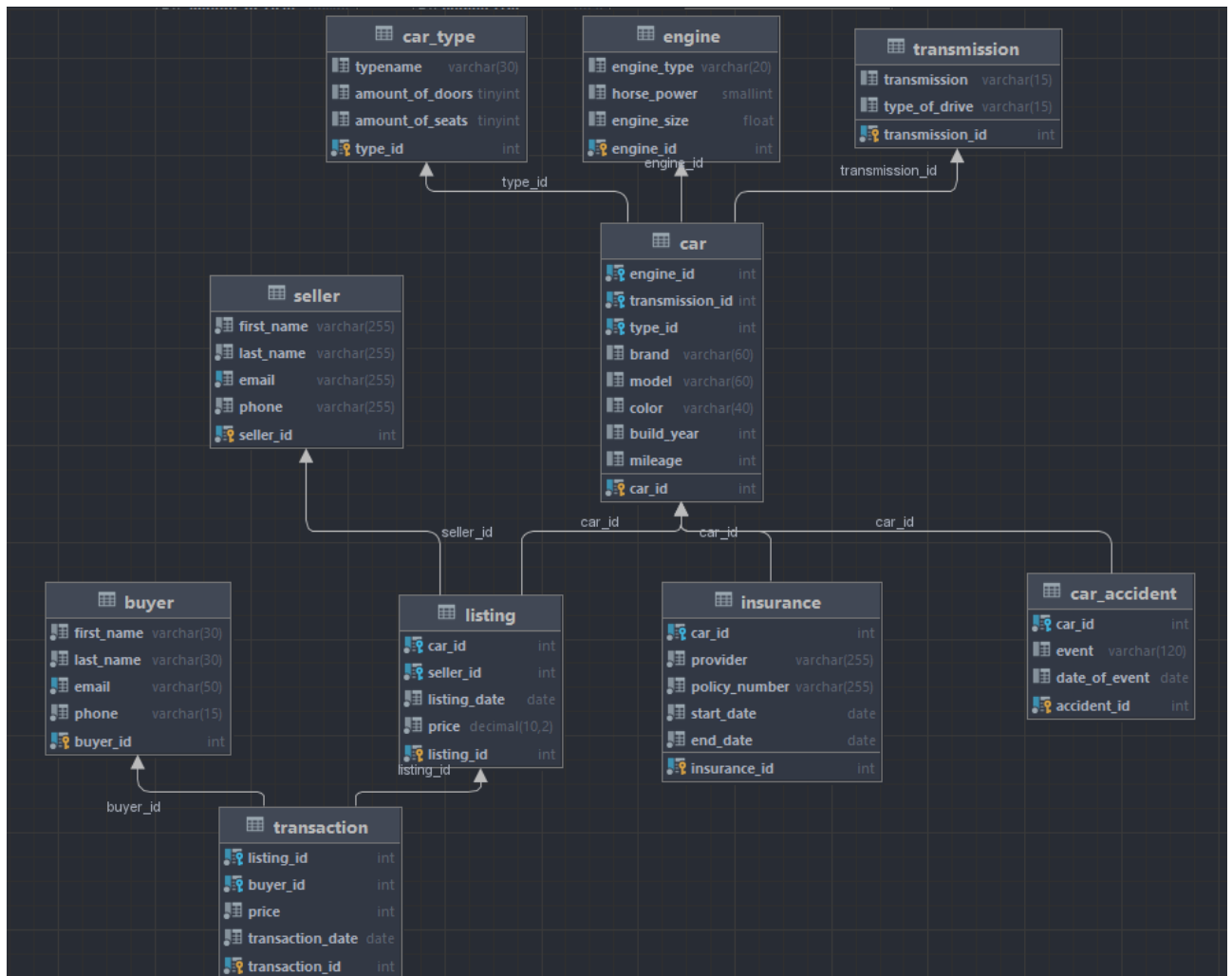


Рисунок 4.1 – Реляційна схема бази даних

На даній схемі видно, що база даних знаходиться у 3 нормальній формі, адже всі поля таблиць декомпозовані, також всі атрибути таблиць функціонально повно залежать від первинного ключа, кожен неключовий атрибут не є транзитивно залежним від первинного ключа.

1. Обов'язкові атрибути таблиць мають обмеження NOT NULL, для запобігання помилок при роботі з даними.
2. Забезпечуються каскадні дії при видаленні зовнішніх ключів однієї з таблиць (ON DELETE CASCADE).

## РЕАЛІЗАЦІЯ БАЗИ ДАНИХ

Створення бази даних

```
DROP DATABASE IF EXISTS car_market;
```

```
CREATE DATABASE car_market;
```

```
use car_market;
```

```
SET foreign_key_checks = 0;
```

```
DROP TABLE IF EXISTS buyer;
```

```
DROP TABLE IF EXISTS seller;
```

```
DROP TABLE IF EXISTS `engine`;
```

```
DROP TABLE IF EXISTS transmission;
```

```
DROP TABLE IF EXISTS car_type;
```

```
DROP TABLE IF EXISTS car_accident;
```

```
DROP TABLE IF EXISTS insurance;
```

```
DROP TABLE IF EXISTS car;
```

```
DROP TABLE IF EXISTS listing;
```

```
DROP TABLE IF EXISTS `transaction`;
```

```
SET foreign_key_checks = 1;
```

```
CREATE TABLE buyer
```

```
(  
    buyer_id INT NOT NULL AUTO_INCREMENT,  
    first_name VARCHAR(30) NOT NULL,  
    last_name VARCHAR(30) NOT NULL,  
    email VARCHAR(50) NOT NULL,  
    phone VARCHAR(15) NOT NULL,  
    PRIMARY KEY (buyer_id)  
);
```

```
ALTER TABLE buyer ADD CONSTRAINT email_unique UNIQUE  
(email);
```

```
CREATE TABLE seller
```

```
(  
    seller_id INT NOT NULL AUTO_INCREMENT,  
    first_name VARCHAR(255) NOT NULL,  
    last_name VARCHAR(255) NOT NULL,  
    email VARCHAR(255) NOT NULL,  
    phone VARCHAR(255) NOT NULL,
```

```
PRIMARY KEY (seller_id)
);
```

```
ALTER TABLE seller ADD CONSTRAINT email_unique UNIQUE (email);
```

```
CREATE TABLE `engine`
(
    engine_id INT NOT NULL AUTO_INCREMENT,
    engine_type VARCHAR(20),
    horse_power SMALLINT,
    engine_size FLOAT,
    PRIMARY KEY (engine_id)
);
```

```
CREATE TABLE transmission
(
    transmission_id INT NOT NULL AUTO_INCREMENT,
    transmission VARCHAR(15),
    type_of_drive VARCHAR(15),
    PRIMARY KEY (transmission_id)
);
```

```
CREATE TABLE car_type
(
    type_id INT NOT NULL AUTO_INCREMENT,
    typename VARCHAR(30),
    amount_of_doors TINYINT,
    amount_of_seats TINYINT,
    PRIMARY KEY (type_id)
);
```

```
CREATE TABLE car
(
    car_id INT NOT NULL AUTO_INCREMENT,
    engine_id INT NOT NULL,
    transmission_id INT NOT NULL,
    type_id INT NOT NULL,
    brand VARCHAR(60),
    model VARCHAR(60),
    color VARCHAR(40),
```



```
build_year    INT,
mileage       INT,
PRIMARY KEY (car_id),
FOREIGN KEY (engine_id) REFERENCES `engine` (engine_id) ON
DELETE CASCADE,
FOREIGN KEY (transmission_id) REFERENCES transmission
(transmission_id) ON DELETE CASCADE,
FOREIGN KEY (type_id) REFERENCES car_type (type_id) ON
DELETE CASCADE
);
```

```
CREATE TABLE insurance
(
insurance_id INT      NOT NULL AUTO_INCREMENT,
car_id      INT      NOT NULL,
provider    VARCHAR(255) NOT NULL,
policy_number VARCHAR(255) NOT NULL,
start_date  DATE      NOT NULL,
end_date    DATE      NOT NULL,
PRIMARY KEY (insurance_id),
FOREIGN KEY (car_id) REFERENCES car (car_id) ON DELETE
CASCADE
);
```

```
CREATE TABLE car_accident
(
accident_id INT NOT NULL AUTO_INCREMENT,
car_id      INT NOT NULL,
`event`     VARCHAR(120),
date_of_event DATE,
PRIMARY KEY (accident_id),
FOREIGN KEY (car_id) REFERENCES car (car_id) ON DELETE
CASCADE
);
```

```
CREATE TABLE listing
(
listing_id INT      NOT NULL AUTO_INCREMENT,
car_id     INT      NOT NULL,
```

```

seller_id INT NOT NULL,
listing_date DATE NOT NULL,
price DECIMAL(10, 2) NOT NULL,
PRIMARY KEY (listing_id),
FOREIGN KEY (car_id) REFERENCES car (car_id) ON DELETE
CASCADE,
FOREIGN KEY (seller_id) REFERENCES seller (seller_id) ON DELETE
CASCADE
);

```

```

CREATE TABLE `transaction`
(
transaction_id INT NOT NULL AUTO_INCREMENT,
listing_id INT NOT NULL,
buyer_id INT NOT NULL,
price INT NOT NULL,
transaction_date DATE NOT NULL,
PRIMARY KEY (transaction_id),
FOREIGN KEY (listing_id) REFERENCES listing (listing_id) ON
DELETE CASCADE,
FOREIGN KEY (buyer_id) REFERENCES buyer (buyer_id) ON DELETE
CASCADE
);

```

## Імпортування даних

Для імпортування даних у була використана можливість завантаження даних за .csv файлу. Було створено .csv файли для кожної з таблиць.

Перелік основних файлів зображено на рисунку:

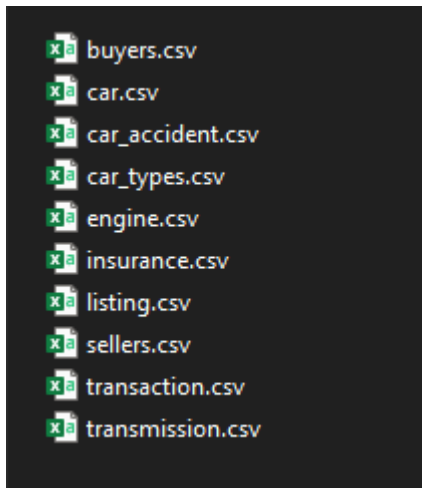


Рисунок 5.1 – Створені файли для заповнення

Для завантаження даних були використані запити виду:

```
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\buyers.csv'
INTO TABLE buyer
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

## СТВОРЕННЯ КОРИСТУВАЧІВ БАЗИ ДАНИХ

### Розробник

```
create role if not exists developer;  
grant all on car_market.* to developer;  
create user if not exists dev@localhost identified by 'admin';  
grant developer to dev@localhost;
```

### Продавець

```
create role if not exists seller;  
grant select on car_market.* to seller;  
grant create, update, delete on car_market.listing to seller;  
create user if not exists seller@localhost identified by 'seller';  
grant seller to seller@localhost;
```

### Покупець

```
create role if not exists seller;  
grant select on car_market.* to seller;  
grant create, update, delete on car_market.listing to seller;  
create user if not exists seller@localhost identified by 'seller';  
grant seller to seller@localhost;
```

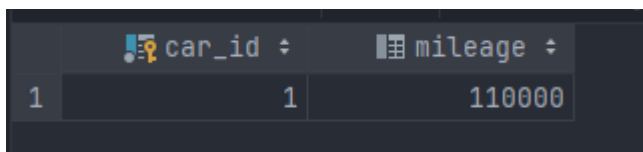
## SQL запити

Створення тригерів на таблиці

Тригер, який не дозволяє зменшувати пробіг для занесених в базу даних автовок.

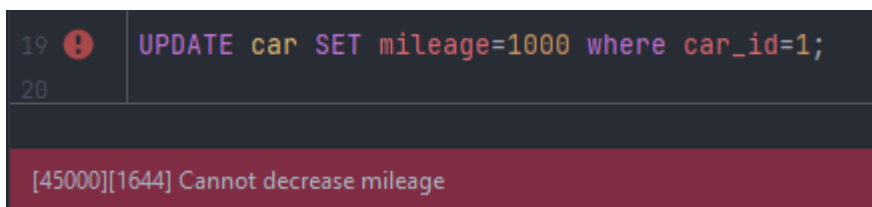
```
DROP TRIGGER IF EXISTS restrict_mileage_decrease;
DELIMITER $$
CREATE TRIGGER restrict_mileage_decrease
BEFORE UPDATE ON car
FOR EACH ROW
BEGIN
    IF NEW.mileage < OLD.mileage THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Cannot decrease
mileage';
    END IF;
END$$
DELIMITER ;
```

Отримаємо пробіг авто з id=1:



	car_id	mileage
1	1	110000

При спробі зменшити його пробіг отримаємо помилку:



Тригер, який унеможливить видалення продавців, в яких є хоча б одна активна об'ява.

```

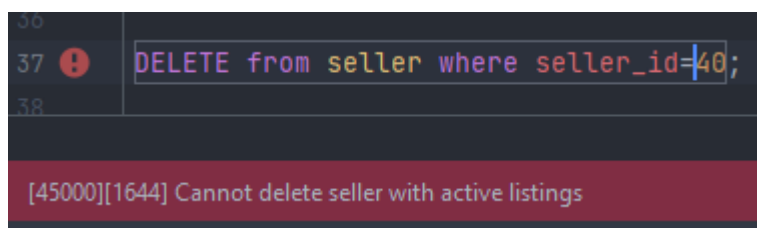
DELIMITER $$
DROP TRIGGER IF EXISTS restrict_seller_delete;
CREATE TRIGGER restrict_seller_delete
BEFORE DELETE ON seller
FOR EACH ROW
BEGIN
    IF (SELECT COUNT(*) FROM listing WHERE seller_id = OLD.seller_id) > 0
    THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Cannot delete seller
with active listings';
    END IF;
END$$
DELIMITER ;

```

Переконаємося, що в продавця з id=1 є хоча б одна активна об'ява:

	listing_id	car_id	seller_id	listing_date	price
1	1	1	40	2022-01-13	22837.00

При спробі його видалити маємо помилку:



Тригер, який унеможливить створення записів про угоди, дата яких більш рання за дату подачі об'яви про продаж.

```

DELIMITER $$
DROP TRIGGER IF EXISTS transaction_date_trigger;
CREATE TRIGGER transaction_date_trigger

```

```
BEFORE INSERT ON `transaction`
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    IF NEW.transaction_date < (SELECT l.listing_date FROM listing l WHERE  
l.listing_id = NEW.listing_id) THEN
```

```
        SIGNAL SQLSTATE '45000'
```

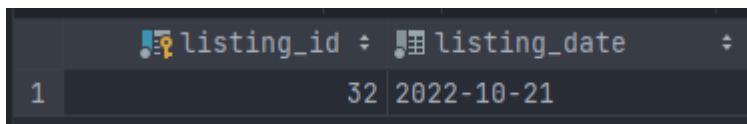
```
        SET MESSAGE_TEXT = 'Transaction date cannot be earlier than listing date';
```

```
    END IF;
```

```
END$$
```

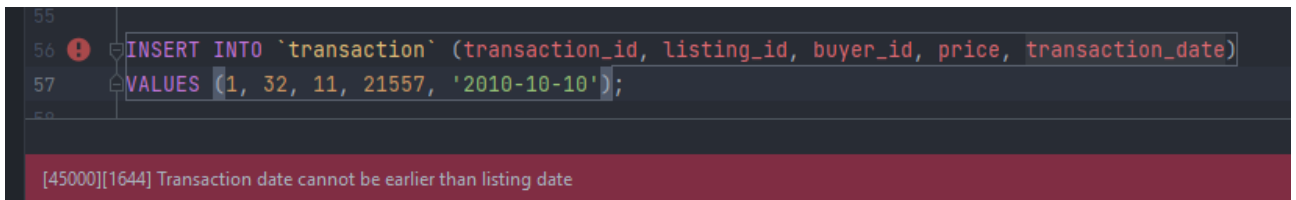
```
DELIMITER ;
```

Виконаємо запит та побачимо дату об'яви з id=1:



	listing_id	listing_date
1	32	2022-10-21

Спробуємо створити угоду, дата якої задовго до дати зверху:



```
55  
56 ! INSERT INTO `transaction` (transaction_id, listing_id, buyer_id, price, transaction_date)  
57 VALUES (1, 32, 11, 21557, '2010-10-10');  
58  
[45000][1644] Transaction date cannot be earlier than listing date
```

Тригер, який не дозволить подавати декілька об'яв на одну й ту саму машину.

```
DELIMITER $$
```

```
DROP TRIGGER IF EXISTS prevent_duplicate_listings;
```

```
CREATE TRIGGER prevent_duplicate_listings
```

```
    BEFORE INSERT ON listing
```

```
    FOR EACH ROW
```

```
BEGIN
```

```
    DECLARE duplicate INT;
```

```
    SELECT COUNT(*) INTO duplicate FROM listing WHERE car_id =
```

```

NEW.car_id;
IF duplicate > 0 THEN
    SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Cannot create listing for car that is already listed';
END IF;
END$$
DELIMITER ;

```

Переконаємося, що запит про авто з id=1 існує:

	car_id	brand	model
1	1	Ford	Grand Cherokee

Спробуємо створити ще одну об'яву на цю автівку:

```

77
78 INSERT INTO listing(listing_id,car_id, seller_id, listing_date, price)
79 VALUES (54,1,2,'2023-01-01',30000);

```

[45000][1644] Cannot create listing for car that is already listed



## Створення процедур

Процедура, яка виводить середню ціну по ринку на автівки заданої марки

```
DELIMITER $$
```

```
CREATE PROCEDURE average_price_by_brand(IN p_brand VARCHAR(60))
```

```
BEGIN
```

```
    SELECT AVG(`transaction`.price) AS avg_price
```

```
    FROM car
```

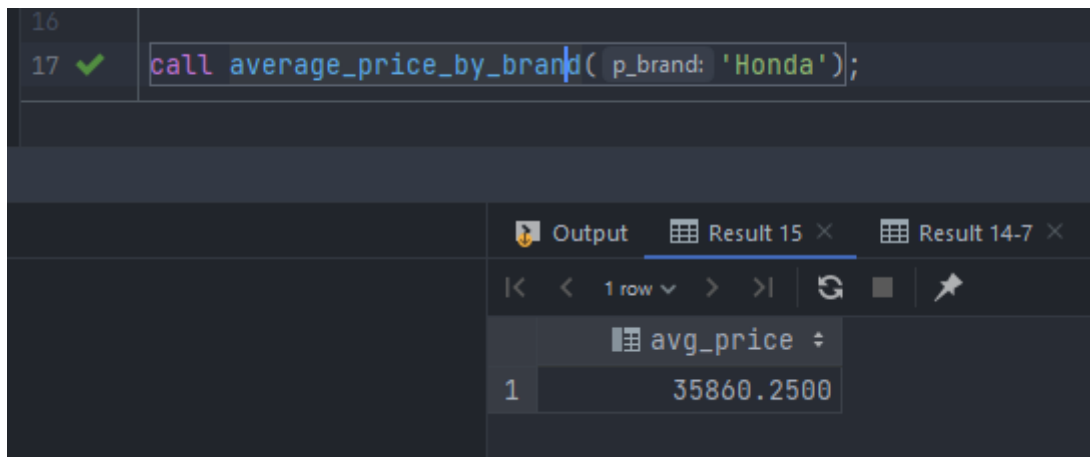
```
    INNER JOIN listing ON car.car_id = listing.car_id
```

```
    INNER JOIN `transaction` ON listing.listing_id = `transaction`.listing_id
```

```
    WHERE car.brand = p_brand;
```

```
END$$
```

```
DELIMITER ;
```



Процедура, яка перевіряє, чи дійсний страховий поліс, айді якого було передано в процедуру як параметр.

```
DELIMITER $$
```

```
CREATE PROCEDURE insurance_in_effect(IN p_insurance_id INT)
```

```
BEGIN
```

```
    SELECT policy_number,
```

```
    CASE
```

```
        WHEN NOW() BETWEEN start_date AND end_date THEN 'In effect'
```

```
        ELSE 'Expired'
```

END AS status

FROM insurance

WHERE insurance\_id = p\_insurance\_id;

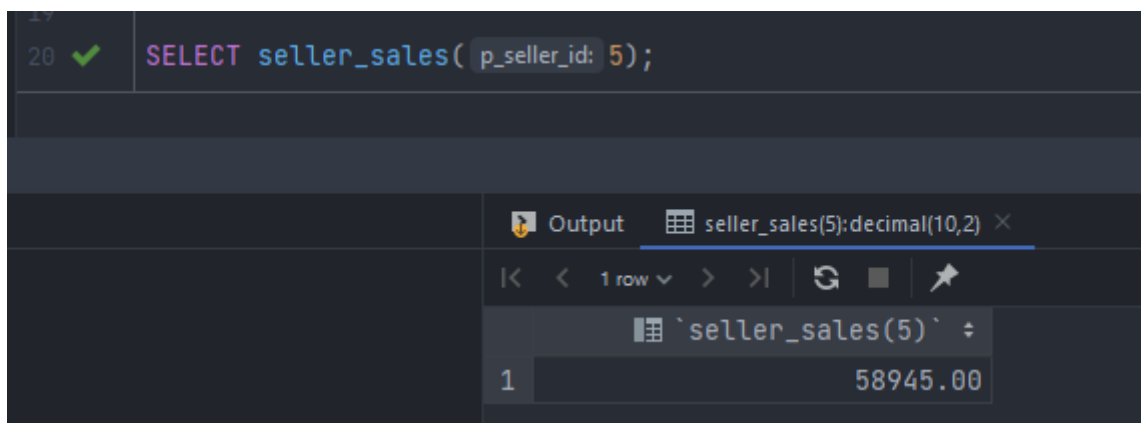
END\$\$

DELIMITER ;

## Створення функцій

Функція, яка повертає суму, на яку конкретний продавець здійснив угоди.

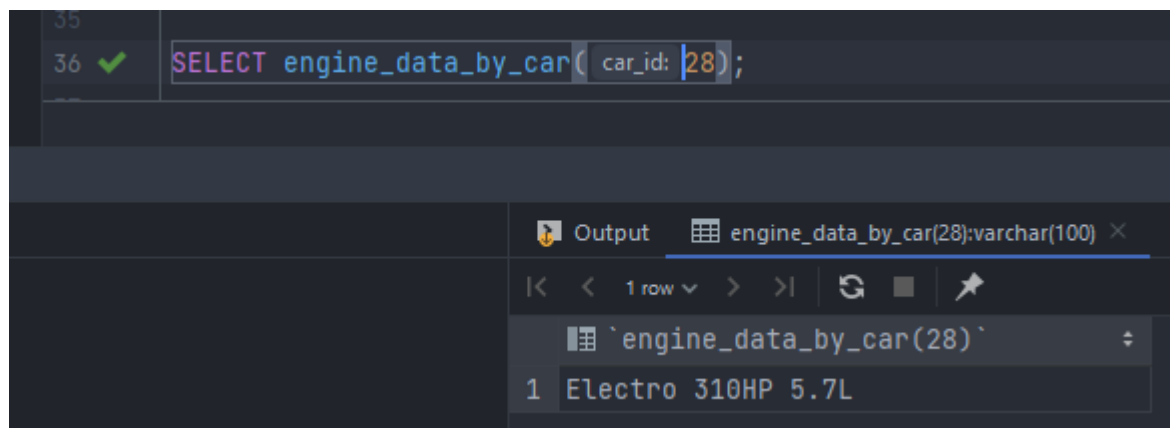
```
DELIMITER $$  
DROP FUNCTION IF EXISTS seller_sales;  
CREATE FUNCTION seller_sales(p_seller_id INT)  
    RETURNS DECIMAL(10, 2)  
    DETERMINISTIC  
BEGIN  
    DECLARE v_sales DECIMAL(10, 2);  
    SELECT COALESCE(SUM(t.price), 0.0)  
    INTO v_sales  
    FROM `transaction` t  
        INNER JOIN listing l ON t.listing_id = l.listing_id  
    WHERE l.seller_id = p_seller_id;  
    RETURN v_sales;  
END$$  
DELIMITER ;
```



Функція, яка повертає інформацію про двигун конкретної автівки

```
DELIMITER $$  
DROP FUNCTION IF EXISTS engine_data_by_car;  
CREATE FUNCTION engine_data_by_car(car_id INT) RETURNS  
VARCHAR(100) DETERMINISTIC
```

```
BEGIN  
    DECLARE engine_data VARCHAR(100);  
    SELECT CONCAT(e.engine_type, ' ', e.horse_power, 'HP ', e.engine_size, 'L')  
    INTO engine_data  
    FROM `engine` e  
    INNER JOIN car c ON e.engine_id = c.engine_id  
    WHERE c.car_id = car_id;  
    RETURN engine_data;  
END$$  
DELIMITER ;
```



The screenshot shows a SQL IDE interface. The top panel displays a query being executed: `SELECT engine_data_by_car( car_id: 28);`. A green checkmark indicates successful execution. The bottom panel shows the output window with the title `engine_data_by_car(28):varchar(100)`. The output is a single row: `1 Electro 310HP 5.7L`.

`engine_data_by_car(28)`		
1	Electro	310HP 5.7L

## Створення представлень

Представлення, яке демонструє середню ціну автівок по їхнім маркам:

```
DROP VIEW IF EXISTS avg_price_by_brand;  
CREATE VIEW avg_price_by_brand AS  
SELECT car.brand, AVG(`transaction`.price) AS avg_price  
FROM car  
INNER JOIN listing ON car.car_id = listing.car_id  
INNER JOIN `transaction` ON listing.listing_id = `transaction`.listing_id  
GROUP BY car.brand;
```

	brand	avg_price
1	Mazda	41811.7500
2	Hyundai	30132.6667
3	Toyota	34632.6667
4	Dodge	41048.0000
5	Porsche	39791.7143
6	Ford	28969.2500
7	Honda	35860.2500
8	Chevrolet	27289.5000
9	Kia	29132.5000
10	Jeep	44285.0000
11	Nissan	49107.3333

Представлення, яке демонструє «пригоди» кожної автівки:

```
DROP VIEW IF EXISTS accidents_by_car;  
CREATE VIEW accidents_by_car AS  
SELECT car.car_id, car.brand, car.model, car_accident.event,  
car_accident.date_of_event  
FROM car  
INNER JOIN car_accident ON car.car_id = car_accident.car_id;
```

	car_id	brand	model	event	date_of_event
1	5	Porsche	Quattro	Collision	2020-03-05
2	18	Chevrolet	Outback	Flood	2020-01-20
3	29	Kia	Fiesta	Theft	2020-03-19
4	35	Hyundai	Cruze	Fire	2020-08-20
5	26	Honda	Grand Cherokee	Rollover	2020-02-07
6	26	Honda	Grand Cherokee	Collision	2020-02-27
7	16	Hyundai	Quattro	Fire	2020-12-13
8	34	Hyundai	Quattro	Flood	2020-10-05
9	1	Ford	Grand Cherokee	Rollover	2020-09-27
10	11	Chevrolet	Charger	Fire	2020-07-13
11	11	Chevrolet	Charger	Flood	2020-05-09
12	19	Dodge	Charger	Collision	2020-05-07
13	21	Kia	Optima	Theft	2020-04-11
14	11	Chevrolet	Charger	Theft	2020-03-05
15	29	Kia	Fiesta	Collision	2020-03-13
16	40	Honda	Elantra	Collision	2020-08-04
17	42	Chevrolet	Accord	Theft	2020-11-14
18	25	Porsche	Grand Cherokee	Flood	2020-11-25
19	43	Porsche	Civic	Collision	2020-08-18
20	24	Porsche	Outback	Flood	2020-10-14

Представлення, яке демонструє топ-продавців виходячи з кількості проданих ними автівок:

```

DROP VIEW IF EXISTS top_sellers;
CREATE VIEW top_sellers AS
SELECT seller.seller_id, seller.first_name, seller.last_name,
COUNT(`transaction`.transaction_id) AS num_cars_sold
FROM seller
INNER JOIN listing ON seller.seller_id = listing.seller_id
INNER JOIN `transaction` ON listing.listing_id = `transaction`.listing_id
GROUP BY seller.seller_id
ORDER BY num_cars_sold DESC;

```

	seller_id	first_name	last_name	num_cars_sold
1	11	Ortensia	MacGinley	4
2	45	Mauricio	Savery	3
3	29	Beau	Dozdill	3
4	35	Shelton	Gluyus	2
5	25	Sheff	d' Eye	2
6	10	Janeva	Reinger	2
7	36	Guntar	Cuniam	2
8	15	Lilly	Bubear	2
9	22	Grete	Szymonwicz	2
10	5	Marris	Dwerryhouse	2
11	42	Bibi	Rossander	2
12	40	Evvy	Glavias	1
13	26	Donaugh	Vinnick	1
14	43	Andy	Struther	1
15	50	Guinna	Twinning	1
16	21	Trevor	Brand-Hardy	1
17	23	Roy	Shimon	1
18	9	Jorie	MacLoughlin	1
19	49	Merissa	Paskerful	1
20	4	Eileen	Turbitt	1
21	30	Becka	Pulley	1
22	33	Bill	Ferfulle	1
23	27	Chloe	Thoms	1
24	13	Lani	Hunnicutt	1

Представлення, яке демонструє інформацію про кожен страховий поліс кожної автівки:

```

DROP VIEW IF EXISTS cars_with_insurance;
CREATE VIEW cars_with_insurance AS
SELECT car.car_id, car.brand, car.model, insurance.provider,
insurance.policy_number
FROM car
INNER JOIN insurance ON car.car_id = insurance.car_id;

```

	car_id	brand	model	provider	policy_number
1	30	Toyota	Charger	KPI INSURANCE	9S03B
2	11	Chevrolet	Charger	God save you	WVDTX
3	30	Toyota	Charger	KPI INSURANCE	MFL02
4	19	Dodge	Charger	Kiliya Industries	0JKI2
5	12	Jeep	Charger	Best Agency	J0BDO
6	20	Chevrolet	Outback	God save you	DSHNY
7	14	Porsche	Grand Cherokee	Best Agency	3E69V
8	2	Ford	Accord	God save you	5S87X
9	44	Chevrolet	Quattro	KPI INSURANCE	K8I0J
10	12	Jeep	Charger	God save you	WVDTX
11	36	Ford	Optima	Best Agency	MQUHL
12	28	Kia	Cruiser	God save you	3E69V
13	12	Jeep	Charger	KPI INSURANCE	R6UJC
14	31	Porsche	Outback	Best Agency	E3HDK
15	39	Toyota	Altima	Best Agency	5NMQT
16	12	Jeep	Charger	Best Agency	MULFQ
17	49	Mazda	Cruiser	God save you	25IEN
18	47	Kia	Grand Cherokee	God save you	FGNV6
19	3	Nissan	Quattro	GoodLuckTM	YALFC
20	26	Honda	Grand Cherokee	Best Agency	WMT0B
21	39	Toyota	Altima	KPI INSURANCE	3E69V
22	19	Dodge	Charger	KPI INSURANCE	J0BDO
23	36	Ford	Optima	KPI INSURANCE	0JKI2
24	17	Honda	Grand Cherokee	Best Agency	DSHNY
25	7	Mazda	Charger	Best Agency	EB3CC
26	46	Honda	Tress	KPI INSURANCE	JJU2S
27	22	Ford	Tress	Kiliya Industries	2SF60
28	36	Ford	Optima	Kiliya Industries	YZ80I
29	30	Toyota	Charger	KPI INSURANCE	9S03B
30	8	Mazda	Civic	God save you	R6UJC
31	20	Chevrolet	Outback	Kiliya Industries	60RFA
32	25	Porsche	Grand Cherokee	KPI INSURANCE	IPCQH
33	28	Kia	Cruiser	God save you	FGNV6
34	35	Hyundai	Cruze	Best Agency	EB3CC
35	38	Honda	Elantra	KPI INSURANCE	KKITP

Представлення для відображення об'яв від кожного з продавців:

```

DROP VIEW IF EXISTS car_listings_by_seller;
CREATE VIEW car_listings_by_seller AS
SELECT seller.seller_id, seller.first_name, seller.last_name, car.car_id, car.brand,
car.model, listing.price, listing.listing_date
FROM seller

```



INNER JOIN listing ON seller.seller\_id = listing.seller\_id

INNER JOIN car ON listing.car\_id = car.car\_id;

	🔍 seller_id 🔍	🔍 first_name 🔍	🔍 last_name 🔍	🔍 car_id 🔍	🔍 brand 🔍	🔍 model 🔍	🔍 price 🔍	🔍 listing_date 🔍
1	40	Evvy	Glavias	1	Ford	Grand Cherokee	22837.00	2022-01-13
2	35	Shelton	Gluyus	2	Ford	Accord	42944.00	2021-06-21
3	25	Sheff	d' Eye	3	Nissan	Quattro	57423.00	2021-04-04
4	11	Ortensia	MacGinley	4	Porsche	Grand Cherokee	40103.00	2021-04-16
5	45	Mauricio	Savery	5	Porsche	Quattro	31390.00	2021-04-19
6	10	Janeva	Reinger	6	Chevrolet	Cruiser	22688.00	2021-05-14
7	35	Shelton	Gluyus	7	Mazda	Charger	49505.00	2022-12-14
8	36	Guntar	Cuniam	8	Mazda	Civic	39786.00	2022-05-13
9	29	Beau	Dozdill	9	Honda	Cruiser	55961.00	2021-12-17
10	26	Donaugh	Vinnick	10	Nissan	Altima	52511.00	2021-08-10
11	45	Mauricio	Savery	11	Chevrolet	Charger	46578.00	2022-07-17
12	43	Andy	Struther	12	Jeep	Charger	41163.00	2022-05-21
13	50	Guinna	Twinning	13	Toyota	Grand Cherokee	20019.00	2022-09-20
14	45	Mauricio	Savery	14	Porsche	Grand Cherokee	57485.00	2021-03-13
15	11	Ortensia	MacGinley	15	Kia	Cruiser	34464.00	2022-07-16
16	15	Lilly	Bubear	16	Hyundai	Quattro	49581.00	2021-05-15
17	21	Trevor	Brand-Hardy	17	Honda	Grand Cherokee	12012.00	2022-09-20
18	23	Roy	Shimon	18	Chevrolet	Outback	14446.00	2021-03-24
19	22	Grete	Szymonwicz	19	Dodge	Charger	41048.00	2021-03-26
20	15	Lilly	Bubear	20	Chevrolet	Outback	25446.00	2021-01-06
21	9	Jorie	MacLoughlin	21	Kia	Optima	16752.00	2022-08-07
22	49	Merissa	Paskerful	22	Ford	Tress	34744.00	2022-11-11
23	5	Marris	Dwerryhouse	23	Nissan	Fiesta	37388.00	2021-05-19
24	4	Eileen	Turbitt	24	Porsche	Outback	37731.00	2022-10-10
25	42	Bibi	Rossander	25	Porsche	Grand Cherokee	28909.00	2022-08-17
26	29	Beau	Dozdill	26	Honda	Grand Cherokee	38321.00	2022-08-07
27	11	Ortensia	MacGinley	27	Porsche	Accord	26583.00	2021-08-26
28	36	Guntar	Cuniam	28	Kia	Cruiser	52668.00	2022-09-24

## Створення різних запитів

Цей запит демонструє список автівок, з якими відбулися «неприємності»

```
SELECT car.car_id, car.brand, car.model, COUNT(car_accident.accident_id) AS  
num_accidents  
FROM car  
INNER JOIN car_accident ON car.car_id = car_accident.car_id  
GROUP BY car.car_id;
```

	car_id	brand	model	num_accidents
1	1	Ford	Grand Cherokee	1
2	5	Porsche	Quattro	1
3	11	Chevrolet	Charger	3
4	16	Hyundai	Quattro	1
5	18	Chevrolet	Outback	1
6	19	Dodge	Charger	1
7	21	Kia	Optima	1
8	24	Porsche	Outback	1
9	25	Porsche	Grand Cherokee	1
10	26	Honda	Grand Cherokee	2
11	29	Kia	Fiesta	2
12	34	Hyundai	Quattro	1
13	35	Hyundai	Cruze	1
14	40	Honda	Elantra	1
15	42	Chevrolet	Accord	1
16	43	Porsche	Civic	1

Цей запит демонструє усі автівки, страховий поліс яких закінчиться протягом наступного місяця:

```
SELECT car.car_id, insurance_id, insurance.policy_number, insurance.end_date  
FROM car  
INNER JOIN insurance ON car.car_id = insurance.car_id  
WHERE end_date BETWEEN NOW() AND DATE_ADD(NOW(), INTERVAL 1  
MONTH);
```

	car_id	insurance_id	policy_number	end_date
1	36	11	MQUHL	2023-02-01
2	46	26	JJU2S	2023-02-01
3	27	40	BA200	2023-02-01
4	38	53	S2T85	2023-02-01
5	29	62	IZ0Y1	2023-02-01

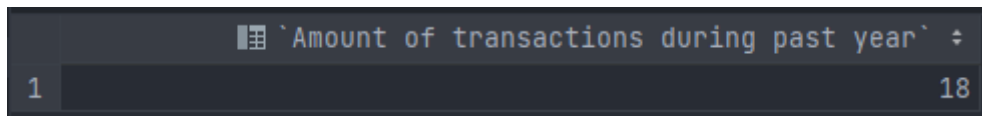
Цей запит демонструє кількість об'яв від кожного з продавців

```
SELECT seller.seller_id,
       concat(seller.first_name, ' ', seller.last_name) as seller,
       COUNT(listing.listing_id)                        AS num_listings
FROM seller
      INNER JOIN listing ON seller.seller_id = listing.seller_id
GROUP BY seller.seller_id;
```

	seller_id	seller	num_listings
1	2	Ranice Hewins	2
2	4	Eileen Turbitt	1
3	5	Marris Dwerryhouse	2
4	9	Jorie MacLoughlin	1
5	10	Janeva Reinger	2
6	11	Ortensia MacGinley	4
7	13	Lani Hunnicutt	1
8	14	Celisse McCarthy	1
9	15	Lilly Bubear	2
10	21	Trevor Brand-Hardy	1
11	22	Grete Szymonwicz	3
12	23	Roy Shimon	1
13	25	Sheff d' Eye	3
14	26	Donaugh Vinnick	1
15	27	Chloe Thoms	2
16	29	Beau Dozdill	3
17	30	Becka Pulley	1
18	32	Zena Issit	1
19	33	Bill Ferfulle	2
20	35	Shelton Gluyus	2
21	36	Guntar Cuniam	2
22	40	Evvy Glavias	1
23	42	Bibi Rossander	3
24	43	Andy Struther	2
25	45	Mauricio Savery	3

Цей запит демонструє кількість угод, укладених протягом минулого року:

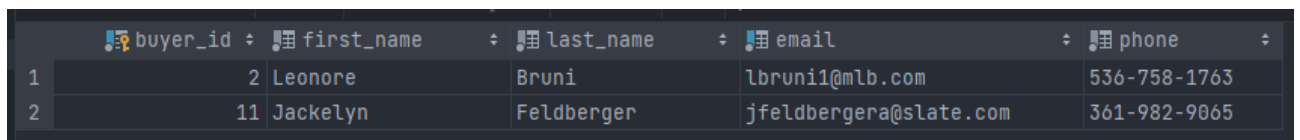
```
SELECT COUNT(*) as `Amount of transactions during past year`  
FROM `transaction`  
WHERE transaction_date BETWEEN DATE_SUB(NOW(), INTERVAL 1 YEAR)  
AND NOW();
```



	`Amount of transactions during past year`
1	18

Цей запит показує інформацію про покупців, які укладали угоди с продавцем під id=5:

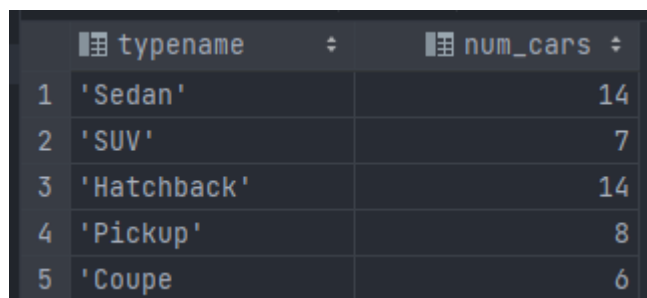
```
SELECT buyer.*  
FROM buyer  
    INNER JOIN `transaction` ON buyer.buyer_id = `transaction`.buyer_id  
    INNER JOIN listing ON `transaction`.listing_id = listing.listing_id  
WHERE listing.seller_id = 5;
```



	buyer_id	first_name	last_name	email	phone
1	2	Leonore	Bruni	lbruni1@mlb.com	536-758-1763
2	11	Jackelyn	Feldberger	jfeldbergera@slate.com	361-982-9065

Цей запит демонструє кількість автівок кожного типу:

```
SELECT car_type.typename, COUNT(car.car_id) AS num_cars  
FROM car_type  
    LEFT JOIN car ON car_type.type_id = car.type_id  
GROUP BY car_type.typename;
```



	typename	num_cars
1	'Sedan'	14
2	'SUV'	7
3	'Hatchback'	14
4	'Pickup'	8
5	'Coupe'	6

Цей запит виводить інформацію про 10 найдорожчих автомобілів

```
SELECT car.car_id, brand, model, price
FROM car
      INNER JOIN listing ON car.car_id = listing.car_id
ORDER BY price DESC
LIMIT 10;
```

	car_id	brand	model	price
1	39	Toyota	Altima	57714.00
2	14	Porsche	Grand Cherokee	57485.00
3	3	Nissan	Quattro	57423.00
4	37	Mazda	Cruiser	56399.00
5	31	Porsche	Outback	56341.00
6	9	Honda	Cruiser	55961.00
7	46	Honda	Tress	55173.00
8	28	Kia	Cruiser	52668.00
9	10	Nissan	Altima	52511.00
10	16	Hyundai	Quattro	49581.00

Цей запит демонструє кількість двигунів різних типів в автівках, які виставлені на продаж:

```
SELECT engine.engine_type, COUNT(car.car_id) AS num_cars
FROM engine
      LEFT JOIN car ON engine.engine_id = car.engine_id
GROUP BY engine.engine_type;
```

	engine_type	num_cars
1	Diesel	9
2	Electro	11
3	Petrol	21
4	Hybrid	8

Цей запит виводить середню ціну усіх автівок по марках:

```
SELECT car.brand, AVG(price) AS average_price
FROM car
      INNER JOIN listing ON car.car_id = listing.car_id
GROUP BY car.brand
ORDER BY (average_price) DESC;
```

	brand	average_price
1	Nissan	49107.333333
2	Jeep	44285.000000
3	Dodge	41048.000000
4	Porsche	36510.250000
5	Mazda	35621.166667
6	Honda	35220.833333
7	Toyota	34632.666667
8	Kia	34451.666667
9	Hyundai	30132.666667
10	Ford	29001.800000
11	Chevrolet	25746.500000

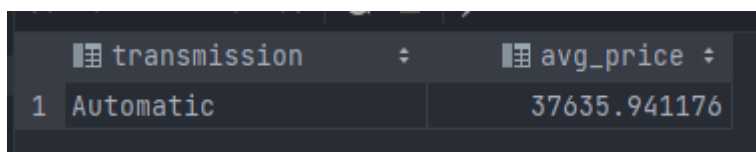
Цей запит демонструє кількість типів КПП, які встановлені на автівки, що виставлені на продаж:

```
SELECT transmission.transmission, COUNT(transmission.transmission_id) AS
Amount_of_listings
FROM transmission
      INNER JOIN car ON transmission.transmission_id = car.transmission_id
      INNER JOIN listing ON car.car_id = listing.car_id
GROUP BY transmission.transmission;
```

	transmission	Amount_of_listings
1	Automatic	17
2	Manual	15
3	CVT	17

Цей запит виводить середню ціну автовок з автоматичною коробкою передач:

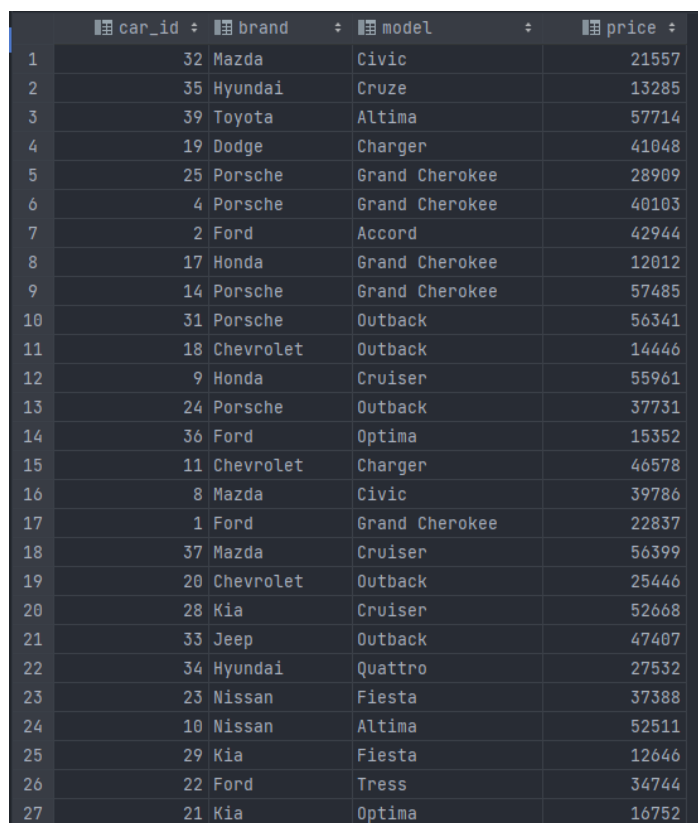
```
SELECT transmission, AVG(price) AS avg_price
FROM car
    INNER JOIN transmission ON car.transmission_id =
transmission.transmission_id
    INNER JOIN listing ON car.car_id = listing.car_id
WHERE transmission.transmission = 'Automatic';
```



	transmission	avg_price
1	Automatic	37635.941176

Цей запит демонструє продані автовки та їхні ціни:

```
SELECT car.car_id, car.brand, car.model, `transaction`.price
FROM car
    INNER JOIN listing ON car.car_id = listing.car_id
    INNER JOIN `transaction` ON listing.listing_id = `transaction`.listing_id;
```



	car_id	brand	model	price
1	32	Mazda	Civic	21557
2	35	Hyundai	Cruze	13285
3	39	Toyota	Altima	57714
4	19	Dodge	Charger	41048
5	25	Porsche	Grand Cherokee	28909
6	4	Porsche	Grand Cherokee	40103
7	2	Ford	Accord	42944
8	17	Honda	Grand Cherokee	12012
9	14	Porsche	Grand Cherokee	57485
10	31	Porsche	Outback	56341
11	18	Chevrolet	Outback	14446
12	9	Honda	Cruiser	55961
13	24	Porsche	Outback	37731
14	36	Ford	Optima	15352
15	11	Chevrolet	Charger	46578
16	8	Mazda	Civic	39786
17	1	Ford	Grand Cherokee	22837
18	37	Mazda	Cruiser	56399
19	20	Chevrolet	Outback	25446
20	28	Kia	Cruiser	52668
21	33	Jeep	Outback	47407
22	34	Hyundai	Quattro	27532
23	23	Nissan	Fiesta	37388
24	10	Nissan	Altima	52511
25	29	Kia	Fiesta	12646
26	22	Ford	Tress	34744
27	21	Kia	Optima	16752

Цей запит виводить продавців, які продали хоча б одну автівку:

```
SELECT seller.seller_id,  
       seller.first_name,  
       seller.last_name,  
       COUNT(`transaction`.transaction_id) AS num_cars_sold,  
       AVG(`transaction`.price)           AS avg_price  
FROM seller  
      INNER JOIN listing ON seller.seller_id = listing.seller_id  
      INNER JOIN `transaction` ON listing.listing_id = `transaction`.listing_id  
GROUP BY seller.seller_id;
```

	seller_id	first_name	last_name	num_cars_sold	avg_price
1	5	Marris	Dwerryhouse	2	29472.5000
2	11	Ortensia	MacGinley	4	28608.7500
3	22	Grete	Szymonwicz	2	49381.0000
4	42	Bibi	Rossander	2	28220.5000
5	35	Shelton	Gluyus	2	46224.5000
6	21	Trevor	Brand-Hardy	1	12012.0000
7	45	Mauricio	Savery	3	45151.0000
8	27	Chloe	Thoms	1	56341.0000
9	23	Roy	Shimon	1	14446.0000
10	29	Beau	Dozdill	3	47229.6667
11	4	Eileen	Turbitt	1	37731.0000
12	13	Lani	Hunnicutt	1	15352.0000
13	36	Guntar	Cuniam	2	46227.0000
14	40	Evvy	Glavias	1	22837.0000
15	25	Sheff	d' Eye	2	56911.0000
16	15	Lilly	Bubear	2	37513.5000
17	26	Donagh	Vinnick	1	52511.0000
18	30	Becka	Pulley	1	12646.0000
19	49	Merissa	Paskerful	1	34744.0000
20	9	Jorie	MacLoughlin	1	16752.0000
21	50	Guinna	Twinning	1	20019.0000
22	10	Janeva	Reinger	2	29917.5000
23	43	Andy	Struther	1	41163.0000
24	33	Bill	Ferfulle	1	26165.0000

Цей запит виводить список автівок, страховий поліс на поточний момент є чинним:

```
SELECT car.car_id, car.brand, car.model, insurance.provider,  
       insurance.policy_number, insurance.end_date  
FROM car
```



INNER JOIN insurance ON car.car\_id = insurance.car\_id

WHERE CURDATE() BETWEEN insurance.start\_date AND insurance.end\_date;

	car_id	brand	model	provider	policy_number	end_date
1	20	Chevrolet	Outback	God save you	DSHNY	2023-04-01
2	14	Porsche	Grand Cherokee	Best Agency	3E69V	2023-04-01
3	2	Ford	Accord	God save you	5S87X	2023-03-01
4	36	Ford	Optima	Best Agency	MQUHL	2023-02-01
5	12	Jeep	Charger	Best Agency	MULFQ	2023-04-01
6	49	Mazda	Cruiser	God save you	25IEN	2023-04-01
7	46	Honda	Tress	KPI INSURANCE	JJU2S	2023-02-01
8	36	Ford	Optima	Kiliya Industries	Y280I	2023-05-01
9	35	Hyundai	Cruze	Best Agency	EB3CC	2023-05-01
10	19	Dodge	Charger	Best Agency	VLK3L	2023-03-01
11	42	Chevrolet	Accord	God save you	Q8YSJ	2023-03-01
12	27	Porsche	Accord	GoodLuckTM	BA200	2023-02-01
13	9	Honda	Cruiser	KPI INSURANCE	MLZBJ	2023-04-01
14	48	Mazda	Tress	GoodLuckTM	EGKXR	2023-03-01
15	38	Honda	Elantra	God save you	S2T85	2023-02-01
16	36	Ford	Optima	GoodLuckTM	OV9KY	2023-03-01
17	36	Ford	Optima	God save you	DSHNY	2023-05-01
18	4	Porsche	Grand Cherokee	God save you	TXSCY	2023-03-01
19	29	Kia	Fiesta	Kiliya Industries	IZ0Y1	2023-02-01
20	3	Nissan	Quattro	GoodLuckTM	DSHNY	2023-03-01
21	27	Porsche	Accord	GoodLuckTM	YALFC	2023-05-01

Цей запит виводить найбільш популярний тип автовок:

SELECT car\_type.typename, COUNT(car\_type.typename) AS num\_cars

FROM car\_type

INNER JOIN car ON car\_type.type\_id = car.type\_id

GROUP BY car\_type.typename

ORDER BY num\_cars DESC

LIMIT 1;

	typename	num_cars
1	'Sedan'	14

## Створення індексів

Зробимо запит по полю з електронною адресою:

```
explain select * from buyer where email='gslowann@time.com';
```

```
[2023-01-10 06:29:47] [HY000][1003] /* select#1 */ select '24' AS `buyer_id`,`Glenn` AS `first_name`,`Slowan` AS `last_name`,`gslowann@time.com` AS `ema`  
[2023-01-10 06:29:47] 1 row retrieved starting from 1 in 38 ms (execution: 9 ms, fetching: 29 ms)
```

Тепер створимо індекс і зробимо запит ще раз:

```
create index buyer_email on buyer(email);
```

```
explain select * from buyer where email='gslowann@time.com';
```

```
[2023-01-10 06:31:05] [HY000][1003] /* select#1 */ select '24' AS `buyer_id`,`Glenn` AS `first_name`,`Slowan` AS `last_name`,`gslowann@time.com` AS `email`  
[2023-01-10 06:31:05] 1 row retrieved starting from 1 in 27 ms (execution: 3 ms, fetching: 24 ms)
```

Як можна побачити, перший запит виконувався 38ms, а запит з використанням індексів було виконано за 27ms, що є значно кращим результатом. У великій базі даних ця різниця може бути критичною.

## **ВИСНОВОК**

Перед виконанням завдання по створенню бази даних по підтримці діяльності автомаркету було спроектовану майбутню базу даних, виокремленні основні сутності та зв'язки між ними. Були сформульовані бізнес-правила та вимоги, які потрібно було реалізувати у роботі.

Отже, у даній роботі було успішно розроблено та впроваджено базу даних для підтримки діяльності автомаркету. Було створено 10 різних таблиць та визначені зв'язки між ними за допомогою зовнішніх ключів і каскадних дій. Були закріплені навички створювання та використання різних об'єктів бази даних, такі як тригери, функції та процедури, створення складних запитів, яке несуть практичне значення для потенційних користувачів сховища даних

## **ПЕРЕЛІК ПОСИЛАНЬ**

1. <https://dev.mysql.com/doc/>
2. <https://www.jetbrains.com/help/datagrip/meet-the-product.html>
3. <https://www.w3schools.com/mysql/>