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 системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху”

Студентки 2 курсу ІП-23 групи спеціальності 121 «Інженерія

програмного забезпечення» Федорєєва Д. А.

(прізвище та ініціали)

Керівник Марченко О. І.

(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 – 2023 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 ІП-23 Семестр 3

# З А В Д А Н Н Я

**НА КУРСОВУ РОБОТУ СТУДЕНТУ**

Федорєєвій Дарфї Анріївній

(прізвище, ім’я, по батькові)

1. Тема роботи “База даних для підтримки системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху” керівник роботи: Марченко Олена Іванівна

(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

1. Строк подання студентом роботи
2. Вихідні дані до роботи завдання на розробку бази даних підтримки системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху
3. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити
   1. Аналіз предметного середовища
   2. Побудова ER-моделі
   3. Побудова реляційної схеми з ER-моделі
   4. Створення бази даних, у форматі обраної СУБД
   5. Створення користувачів бази даних
   6. Імпорт даних з використанням засобів СУБД в створену базу даних
   7. Створення мовою SQL запитів
   8. Оптимізація роботи запитів
4. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
5. Дата видачі завдання

## КАЛЕНДАРНИЙ ПЛАН

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

**Студент**  Федорєєва Д. А.

(підпис ) (прізвище та ініціали)

**Керівник роботи**  Марченко О. І.

(підпис ) (прізвище та ініціали)

# Зміст

## ВСТУП

У сучасному світі, бази даних відіграють ключову роль у зберіганні, систематизації та управлінні обширними обсягами інформації. Вони є необхідною складовою сучасних обчислювальних систем, що застосовуються у різних галузях та контекстах.

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

Зокрема, бази даних є центральними сховищами даних, що гарантують їхню послідовність, точність та актуальність. Крім того, вони забезпечують важливі заходи безпеки, такі як автентифікація користувачів та шифрування даних, для захисту конфіденційної інформації від несанкціонованого доступу.

У рамках цієї курсової роботи розглядається база даних, спрямована на підтримку системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху.

## ОПИС ПРЕДМЕТНОГО СЕРЕДОВИЩА

Адміністративне правопорушення у сфері забезпечення безпеки дорожнього руху визначається як невідповідність чинних норм, правил та обмежень, які регулюють дорожній рух, та порушення встановлених правил, які мають адміністративний характер. Ці порушення можуть включати в себе різноманітні види невідповідностей водійській поведінці, стану транспортних засобів чи дорожньої інфраструктури, що створює загрозу безпеці дорожнього руху.

До адміністративних правопорушень у сфері безпеки дорожнього руху можуть відноситися:

1. Перевищення швидкості: порушення встановлених обмежень швидкості на дорогах.
2. Проїзд на червоне світло: проїзд перехрестя при червоному світлі світлофора без відповідного дозволу.
3. Порушення правил обгону: невідповідність вимогам при виконанні обгону інших транспортних засобів.
4. Водіння у стані алкогольного чи наркотичного сп'яніння: використання транспортного засобу під впливом алкоголю або наркотичних речовин.
5. Неправильне паркування: паркування у заборонених місцях або порушення інших правил паркування.
6. Недотримання дорожніх знаків та сигналізації: ігнорування дорожніх знаків, світлофорів та інших сигналізуючих засобів.

Адміністративні правопорушення у сфері безпеки дорожнього руху регулюються відповідним законодавством та можуть призводити до застосування адміністративних санкцій, таких як штрафи, позбавлення водійських прав чи інші заходи відповідно до закону. Їх фіксація та контроль важливі для забезпечення безпеки на дорозі та попередження небажаних ситуацій.

Предметне середовище для фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху містить різні об'єкти та сутності, які пов'язані з його функціонуванням. Основні об'єкти та сутності в цьому предметному середовищі можуть бути наступними:

* Правопорушення (опис правопорушення, дата та час фіксування, місце правопорушення)
* Водій
* Штраф
* Свідки

Ці об'єкти та сутності взаємодіють та утворюють цілісну систему для ефективного фіксування, аналізу та управління адміністративними правопорушеннями у сфері безпеки дорожнього руху

## ПОСТАНОВКА ЗАДАЧІ

Метою даної роботи є розробка бази даних для підтримки системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху. 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 багатокористувацького 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, реалізувавши розроблену багатокористувацьку 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 акт адміністративного правопорушення може бути заповнений лише одним офіцером.
2. Один акт правопорушення може мати лише один штраф.
3. Кожен акт правопорушення може підпадати лише під одну категорію правопорушення.
4. Правопорушення може мати місце лише на одній локації. Так само й акт цього правопорушення може мати в описі лише одну локацію.
5. Кожний акт правопорушення може мати від одного свідка.
6. Кожен акт правопорушення може стосуватися лише одного водія.
7. Один автомобіль може мати лише одного водія.
8. Власник може мати від одного автомобіля.
9. У базі необхідно зберігати інформацію про дату та час правопорушення, номер транспортного засобу, та дані про водія.

Вибір сутностей

Вибір сутностей для Бази Даних виглядатиме наступним чином:

* VehicleOwner
* Vehicle
* Driver
* Violation
* Witness
* ViolationCatagory
* ViolationAct
* ViolationLocation
* PoliceOfficer
* Fine

Набiр атрибутів сутностей

Таблиця 3.1 – Сутності та їх атрибути.

|  |  |  |
| --- | --- | --- |
| Сутність | Атрибути | Тип даних |
| VehicleOwner | owner\_id  name  last\_name  address  phone\_number  ipn\_code | serial  varchar(25)  varchar(25)  varchar(255)  varchar(12)  varchar(10) |
| Vehicle | vin  registration\_number  brand  model  manufacture\_year  owner\_id | varchar(17)  varchar(10)  varchar(25)  varchar(60)  integer  integer |
| Driver | drivers\_licence  name  last\_name  address  phone\_number  vin | varchar(20)  varchar(25)  varchar(25)  varchar(255)  varchar(12)  varchar(17) |

Продовження таблиці 3.1

|  |  |  |
| --- | --- | --- |
| Сутність | Атрибути | Тип даних |
| Violation | violation\_id  violation\_type  date\_time  status  violation\_catagory\_id  drivers\_licence | serial  varchar(255)  timestamp  varchar(25)  integer  varchar(20) |
| Witness | witness\_id  name  last\_name  phone\_number  violation\_act | serial  varchar(25)  varchar(25)  varchar(12)  integer |
| ViolationCatagory | violation\_catagory\_id  name  description | serial  varchar(70)  text |
| ViolationAct | violation\_act\_id  violation\_description  date\_time  witness\_testimony  status  location\_id  violation\_id  officer\_id  fine\_id | serial  text  timestamp  text  varchar(25)  integer  integer  integer  integer |
| ViolationLocation | location\_id  town  street  building\_number | serial  varchar(25)  varchar(255)  varchar(5) |

Продовження таблиці 3.1

|  |  |  |
| --- | --- | --- |
| Сутність | Атрибути | Тип даних |
| PoliceOfficer | officer\_id  name  last\_name  phone\_number | serial  varchar(25)  varchar(25)  varchar(12) |
| Fine | fine\_id  fine\_amount  payment\_term  status | serial  smalint  date  varchar(25) |

Сутність VehicleOwner буде пов’язана **один до багатьох** із сутністю Vehicle, адже кожен власник може мати кілька машин

Сутність Vehicle буде пов’язана **один до одного** із сутністю Driver, адже тільки одна людина може бути за кермом машини.

Сутність Violation буде пов’язана **від нуля до одного** із сутністю Driver, адже водій може не мати, а може й мати штрафи.

Сутність Violation буде пов’язана **багато до одного** із сутністю ViolationCatagory, адже адміністративне правопорушення може підпадати лише під одну категорію.

Сутність Violation буде пов’язана **один до одного** із сутністю ViolationAct, адже на одне правопорушення можна скласти лише один акт.

Сутність ViolationAct буде пов’язана **один до багатьох** із сутністю Witness, адже одне правопорушення може мати багато свідків та їх свідчення мають бути записані в акт правопорушення.

Сутність ViolationAct буде пов’язана **багато до одного** із сутністю ViolationLocation, адже у одного правопорушення може бути лише одна локація, а на одній локації може статися багато правопорушень.

Сутність ViolationAct буде пов’язана **багато до одного** із сутністю PoliceOfficer, адже лише один офіцер може складати акт про правопорушення, а офіцер може складати багато правопорушень.

Сутність ViolationAct буде пов’язана **один до одного** із сутністю Fine, правопорушення може каратися лише одним штрафом.

Изображение выглядит как текст, диаграмма, План, Параллельный

Автоматически созданное описание

Рисунок 3.1 – ER-діаграма

Із системою мають змогу працювати наступні групи користувачів:

* Адміністратор – має повний доступ до всіх даних у базі даних. Може створювати, редагувати та видаляти записи.
* Оператор – має доступ до даних про правопорушення, транспортні засоби та водіїв. Може створювати та редагувати записи про правопорушення так акт правопорушення.
* Офіцер – має доступ до даних про акт правопорушення. Може створювати та редагувати дані акту правопорушення.

## РЕЛЯЦІЙНА МОДЕЛЬ БАЗИ ДАНИХ

Побудова необхідних відношень, визначення первинних та зовнішніх ключів.

Изображение выглядит как снимок экрана, текст

Автоматически созданное описание

Рисунок 4.1 – Реляційна схема бази даних

На цьому рисунку видно, що база даних знаходиться у третій нормальній формі, оскільки всі поля таблиць розкладені, а також всі атрибути таблиць функціонально повністю залежать від первинного ключа. Кожен неключовий атрибут не є транзитивно залежним від первинного ключа.

1. Обов'язкові атрибути таблиць мають обмеження NOT NULL, щоб уникнути помилок під час роботи з даними.
2. Забезпечено каскадні дії при видаленні зовнішніх ключів однієї з таблиць (ON DELETE CASCADE).

## РЕАЛІЗАЦІЯ БАЗИ ДАНИХ

Реалізація бази даних буде здійснюватися з використанням системи управління базами даних PostgreSQL. Це потужне рішення, що володіє широким спектром інструментів для ефективної роботи з базою даних. Однією з вагомих переваг PostgreSQL є його добре структурована та доступна документація. Вона є однією з найпопулярніших СУБД у світі, і її використовують для широкого спектру застосунків, включаючи веб-сайти, веб-додатки, бізнес-аналітику та наукові дослідження.

PostgreSQL має ряд переваг:

Розширена підтримка SQL: PostgreSQL дотримується стандартів SQL і включає ряд розширень, які роблять його більш потужним та функціональним у порівнянні зі стандартним SQL.

Гнучкість конфігурації: PostgreSQL надає розширені можливості конфігурації, що дозволяє налаштовувати різні параметри для оптимізації продуктивності та виконання завдань згідно з конкретними потребами.

Створення бази даних

CREATE TABLE IF NOT EXISTS Vehicle\_Owner(  
 owner\_id SERIAL PRIMARY KEY NOT NULL,  
 name VARCHAR(25) NOT NULL,  
 last\_name VARCHAR(25) NOT NULL,  
 address VARCHAR(255) NOT NULL,  
 phone\_number VARCHAR(12) NOT NULL,  
 ipn\_code VARCHAR(10) NOT NULL  
);  
  
CREATE TABLE IF NOT EXISTS Vehicle(  
 vin VARCHAR(17) PRIMARY KEY NOT NULL,  
 registration\_number VARCHAR(10) NOT NULL,  
 brand VARCHAR(25),  
 model VARCHAR(60),  
 manufacture\_year VARCHAR(4),  
 owner\_id INT REFERENCES Vehicle\_Owner(owner\_id) ON DELETE CASCADE NOT NULL  
);  
  
CREATE TABLE IF NOT EXISTS Driver(  
 drivers\_licence VARCHAR(22) PRIMARY KEY NOT NULL,  
 name VARCHAR(25) NOT NULL,  
 last\_name VARCHAR(25) NOT NULL,  
 address VARCHAR(255) NOT NULL,  
 phone\_number VARCHAR(12) NOT NULL,  
 vin VARCHAR(17) UNIQUE REFERENCES Vehicle(vin) ON DELETE CASCADE NOT NULL  
);  
  
CREATE TABLE IF NOT EXISTS Violation\_Category(  
 violation\_category\_id SERIAL PRIMARY KEY NOT NULL,  
 name VARCHAR(70) NOT NULL,  
 description TEXT  
);  
  
CREATE TYPE VIOLATION\_STATUS AS ENUM ('not registered', 'registered', 'closed');  
  
CREATE TABLE IF NOT EXISTS Violation(  
 violation\_id SERIAL PRIMARY KEY NOT NULL,  
 violation\_type VARCHAR(255) NOT NULL,  
 date\_time TIMESTAMP NOT NULL,  
 status VIOLATION\_STATUS NOT NULL,  
 violation\_category\_id INT REFERENCES Violation\_Category(violation\_category\_id) ON DELETE CASCADE NOT NULL,  
 drivers\_licence VARCHAR(22) REFERENCES Driver(drivers\_licence) ON DELETE CASCADE NOT NULL  
);  
  
CREATE TABLE IF NOT EXISTS Police\_Officer(  
 officer\_id SERIAL PRIMARY KEY NOT NULL,  
 name VARCHAR(25) NOT NULL,  
 last\_name VARCHAR(25) NOT NULL,  
 phone\_number VARCHAR(12) NOT NULL  
);  
  
CREATE TABLE IF NOT EXISTS Violation\_Location(  
 location\_id SERIAL PRIMARY KEY NOT NULL,  
 town VARCHAR(25) NOT NULL,  
 street VARCHAR(255) NOT NULL,  
 building\_number VARCHAR(5) NOT NULL  
);  
  
CREATE TYPE FINE\_STATUS AS ENUM ('not payed', 'payed');  
  
CREATE TABLE IF NOT EXISTS Fine(  
 fine\_id SERIAL PRIMARY KEY NOT NULL,  
 fine\_amount REAL NOT NULL,  
 payment\_term DATE NOT NULL,  
 status FINE\_STATUS NOT NULL  
);  
  
CREATE TYPE VIOLATION\_ACT\_STATUS AS ENUM ('active', 'closed');  
  
CREATE TABLE IF NOT EXISTS Violation\_Act(  
 violation\_act\_id SERIAL PRIMARY KEY NOT NULL,  
 violation\_description TEXT NOT NULL,  
 date\_time TIMESTAMP NOT NULL,  
 witness\_testimony TEXT NOT NULL,  
 status VIOLATION\_ACT\_STATUS NOT NULL,  
 location\_id INT REFERENCES Violation\_Location(location\_id) ON DELETE CASCADE NOT NULL,  
 violation\_id INT UNIQUE REFERENCES Violation(violation\_id) ON DELETE CASCADE NOT NULL,  
 officer\_id INT REFERENCES Police\_Officer(officer\_id) ON DELETE CASCADE NOT NULL,  
 fine\_id INT UNIQUE REFERENCES Fine(fine\_id) ON DELETE CASCADE NOT NULL  
);  
  
CREATE TABLE IF NOT EXISTS Witness(

witness\_id SERIAL PRIMARY KEY NOT NULL,  
 name VARCHAR(25) NOT NULL,  
 last\_name VARCHAR(25) NOT NULL,  
 phone\_number VARCHAR(12) NOT NULL,  
 violation\_act\_id INT REFERENCES Violation\_Act(violation\_act\_id) ON DELETE CASCADE NOT NULL  
);

## ІМПОРТУВАННЯ ДАНИХ

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

Перелік .csv файлів зображено на рисунку:

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

Рисунок 5.1 – Створені файли для заповнення таблиць

Для завантаження даних були використані наступні команди:

\COPY Vehicle\_Owner FROM 'C:\Users\Admin\DataGripProjects\administrative\_violation\csv\_data\vehicle\_owner.csv' WITH (FORMAT csv, HEADER true, DELIMITER ',');  
  
\COPY Vehicle FROM 'C:\Users\Admin\DataGripProjects\administrative\_violation\csv\_data\vehicle.csv' WITH (FORMAT csv, HEADER true, DELIMITER ',');  
  
\COPY Driver FROM 'C:\Users\Admin\DataGripProjects\administrative\_violation\csv\_data\driver.csv' WITH (FORMAT csv, HEADER true, DELIMITER ',');  
  
\COPY Violation\_Category FROM 'C:\Users\Admin\DataGripProjects\administrative\_violation\csv\_data\violation\_category.csv' WITH (FORMAT csv, HEADER true, DELIMITER ',');  
  
\COPY Violation FROM 'C:\Users\Admin\DataGripProjects\administrative\_violation\csv\_data\violation.csv' WITH (FORMAT csv, HEADER true, DELIMITER ',');  
  
\COPY Police\_Officer FROM 'C:\Users\Admin\DataGripProjects\administrative\_violation\csv\_data\police\_officer.csv' WITH (FORMAT csv, HEADER true, DELIMITER ',');  
  
\COPY Violation\_Location FROM 'C:\Users\Admin\DataGripProjects\administrative\_violation\csv\_data\violation\_location.csv' WITH (FORMAT csv, HEADER true, DELIMITER ',');  
  
\COPY Fine FROM 'C:\Users\Admin\DataGripProjects\administrative\_violation\csv\_data\fine.csv' WITH (FORMAT csv, HEADER true, DELIMITER ',');  
  
\COPY Violation\_Act FROM 'C:\Users\Admin\DataGripProjects\administrative\_violation\csv\_data\violation\_act.csv' WITH (FORMAT csv, HEADER true, DELIMITER ',');  
  
\COPY Witness FROM 'C:\Users\Admin\DataGripProjects\administrative\_violation\csv\_data\witness.csv' WITH (FORMAT csv, HEADER true, DELIMITER ',');

СТВОРЕННЯ КОРИСТУВАЧІВ БАЗИ ДАНИХ

Адміністратор

CREATE ROLE administrator LOGIN PASSWORD 'administrator\_user\_password';

GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO administrator;  
GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA public TO administrator;

GRANT EXECUTE ON ALL FUNCTIONS IN SCHEMA public TO administrator;

Оператор

CREATE ROLE administrative\_operator LOGIN PASSWORD 'operator\_user\_password';

GRANT SELECT, INSERT, UPDATE ON TABLE Violation, Violation\_Act, Vehicle, Driver TO administrative\_operator;

GRANT USAGE, SELECT ON ALL SEQUENCES IN SCHEMA public TO administrative\_operator;

GRANT EXECUTE ON ALL FUNCTIONS IN SCHEMA public TO administrative\_operator;

Офіцер

CREATE ROLE officer LOGIN PASSWORD 'officer\_user\_password';

GRANT SELECT, INSERT, UPDATE ON TABLE Violation\_Act TO officer;

GRANT USAGE, SELECT ON ALL SEQUENCES IN SCHEMA public TO officer;

GRANT EXECUTE ON ALL FUNCTIONS IN SCHEMA public TO officer;

# SQL запити

**Створення тригерів на таблиці**

Тригер, що не дозволяє зменшувати ціну штрафу, що був виписаний за адміністративне правопорушення на дорозі.

CREATE OR REPLACE FUNCTION fine\_update\_check\_function()  
RETURNS TRIGGER AS $$  
BEGIN  
 IF NEW.fine\_amount < OLD.fine\_amount THEN  
 RAISE EXCEPTION 'New fine amount cannot be less, than old fine amount';  
 END IF;  
 RETURN NEW;  
END;  
$$ LANGUAGE plpgsql;

CREATE TRIGGER FineAmountCheck  
AFTER UPDATE ON fine  
FOR EACH ROW  
EXECUTE FUNCTION fine\_update\_check\_function();

Отримаємо штраф з id=11:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 6.1 – сума штрафу та інші дані, де id=11

Під час спроби змінити суму штрафу, на меншу, ніж було отримуємо помилку.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 6.2 – отримана помилка при спробі змінити суму штрафу на меншу

Тригер, який унеможливить додавання штрафів з сумою меншою 1.

CREATE OR REPLACE FUNCTION fine\_insert\_trigger\_function()  
RETURNS TRIGGER AS $$  
BEGIN  
 IF NEW.fine\_amount <= 0 THEN  
 RAISE EXCEPTION 'Fine amount cannot be less than 0';  
 END IF;  
 RETURN NEW;  
END;  
$$ LANGUAGE plpgsql;  
  
CREATE TRIGGER FinesInsertTrigger  
AFTER INSERT ON fine  
FOR EACH ROW  
EXECUTE FUNCTION fine\_insert\_trigger\_function();

Переконаємося, що цей тригер не дозволить вставити новий штраф, сума якого буде менше 1.

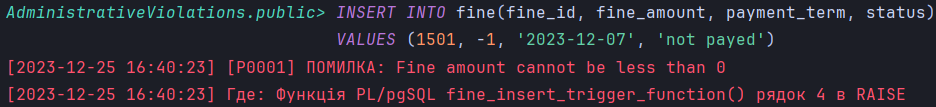


Рисунок 6.3 – отримана помилка при спробі додати штраф з від’ємною сумою

Тригер, який не дозволить видалити водія, в котрого досі активна справа по його адміністративному правопорушенню.

CREATE OR REPLACE FUNCTION prevent\_delete\_if\_active\_violation()  
RETURNS TRIGGER AS $$  
BEGIN  
 IF EXISTS (  
 SELECT 1  
 FROM Violation\_Act va  
 JOIN Violation v ON va.violation\_id = v.violation\_id  
 WHERE v.drivers\_licence = OLD.drivers\_licence AND va.status = 'active'  
 ) THEN  
 RAISE EXCEPTION 'Cannot delete driver with active violations';  
 END IF;  
 RETURN OLD;  
END;  
$$ LANGUAGE plpgsql;  
  
CREATE TRIGGER CheckToDeleteDriver  
BEFORE DELETE ON driver  
FOR EACH ROW  
EXECUTE FUNCTION prevent\_delete\_if\_active\_violation();

Переконаємося, що проти водія з посвідченням 'BK-549847-63-37' є як мінімум одна активна справа по його адміністративному правопорушенню.

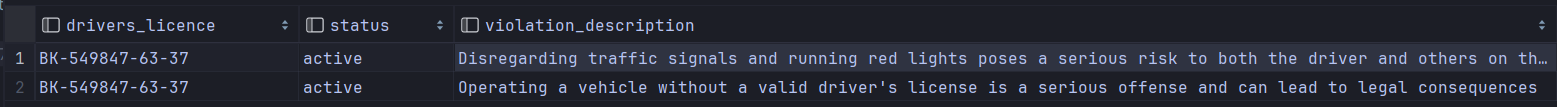


Рисунок 6.4 – активні справи за посвідченням 'BK-549847-63-37'

Під час спроби видалити водія з цим посвідченням ми отримуємо повідомлення про помилку.

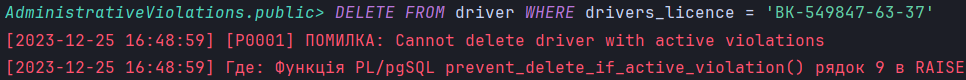


Рисунок 6.5 – посилка про видалення водія за посвідченням 'BK-549847-63-37'

Тригер, що не дозволить відредагувати статус справи про адміністративне правопорушення на дорозі, якщо не був сплачений штраф.

Переконаємося, що водій, який занесений до справи де id=11 не сплатив штраф за правопорушення.

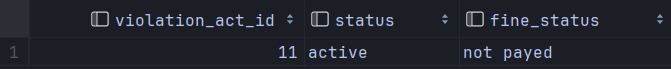


Рисунок 6.5 – справа, по якій не був сплачений штраф

Спробуємо змінити статус справи на «закрита».

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 6.6 – помилка під час спроби змінити статус справи

Тригер, що самостійно встановить статус нового правопорушення за замовчуванням на «не зареєстровано», якщо статус не буде прописаний власноруч.

CREATE OR REPLACE FUNCTION set\_default\_violation\_status()  
RETURNS TRIGGER AS $$  
BEGIN  
 NEW.status := 'not registered';  
 RETURN NEW;  
END;  
$$ LANGUAGE plpgsql;  
  
CREATE TRIGGER DefaultViolationStatusTrigger  
BEFORE INSERT ON violation  
FOR EACH ROW  
WHEN (NEW.status IS NULL)  
EXECUTE FUNCTION set\_default\_violation\_status();

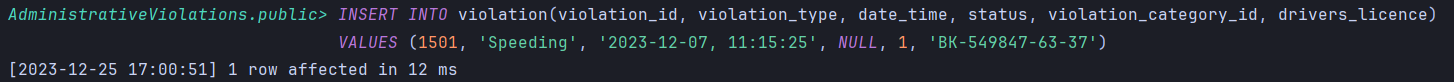


Рисунок 6.6 – вставлення нового запису без явно прописаного статусу

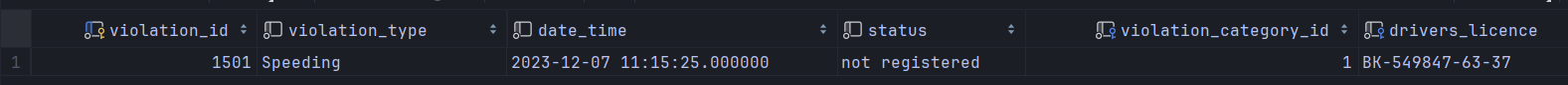


Рисунок 6.7 – перевірка записаних даних

Як видно з рисунку 6.7 тригер виконав поставлену задачу та встановив статус за замовчуванням «не зареєстровано».

Тригер, що за замовчуванням встановить статус «не сплачено» для штрафів, що щойно були занесені до таблиць.

CREATE OR REPLACE FUNCTION set\_default\_fine\_status()  
RETURNS TRIGGER AS $$  
BEGIN  
 NEW.status := 'not payed';  
 RETURN NEW;  
END;  
$$ LANGUAGE plpgsql;  
  
CREATE TRIGGER DefaultFineStatusTrigger  
BEFORE INSERT ON fine  
FOR EACH ROW  
WHEN (NEW.status IS NULL)  
EXECUTE FUNCTION set\_default\_fine\_status();

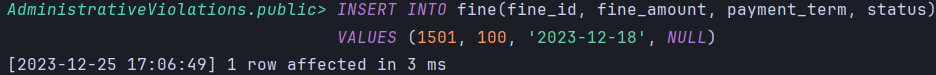


Рисунок 6.8 – вставлення нового запису без явно прописаного статусу

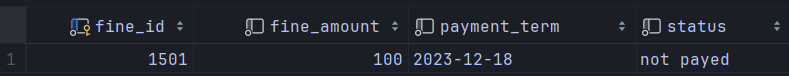


Рисунок 6.9 – перевірка записаних даних

Передивляючись рисунок 6.9 можна переконатися в тому, що тригер виконав поставлену задачу та встановив статус за замовченням «не сплачено».

**Створення функцій**

Функція, що рахує середню суму штрафу за специфіковане адміністративне правопорушення.

CREATE OR REPLACE FUNCTION average\_fine\_by\_violation(IN violation VARCHAR(70))  
RETURNS NUMERIC  
AS $$  
DECLARE  
 avg\_fine\_amount NUMERIC;  
BEGIN  
 SELECT AVG(f.fine\_amount) INTO avg\_fine\_amount  
 FROM fine f  
 INNER JOIN violation\_act va ON f.fine\_id = va.fine\_id  
 INNER JOIN violation v ON va.violation\_id = v.violation\_id  
 WHERE v.violation\_type = violation;  
 RETURN avg\_fine\_amount;  
END;  
$$ LANGUAGE plpgsql;

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

Автоматически созданное описание

Рисунок 7.1 – результат функції average\_fine\_by\_violation

Функція, яка виводить усіх офіцерів у базі даних та підраховує на яку суму кожен з них зареєстрував адміністративних правопорушень.

CREATE OR REPLACE FUNCTION officer\_total\_penalty\_sum()  
RETURNS TABLE (police\_officer\_name TEXT, total\_fine\_sum REAL)  
AS $$  
BEGIN  
 RETURN QUERY  
 SELECT CONCAT(p.name, ' ', p.last\_name) AS police\_officer\_name,  
 SUM(f.fine\_amount) AS total\_fine\_sum  
 FROM police\_officer p  
 JOIN violation\_act va ON p.officer\_id = va.officer\_id  
 JOIN fine f ON va.fine\_id = f.fine\_id  
 GROUP BY police\_officer\_name  
 ORDER BY total\_fine\_sum DESC;  
END;  
$$ LANGUAGE plpgsql;

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

Автоматически созданное описание

Рисунок 7.2 – результат функції officer\_total\_penalty\_sum()

Функція, що буде виводити усю інформацію про автомобіль та ім’я його власника, що використовувався під час правопорушення, за допомогою вхідного параметру, що вказує на номер справи про правопорушення.

CREATE OR REPLACE FUNCTION get\_vehicle\_data\_by\_violation\_id(IN v\_violation\_id INT)  
RETURNS TABLE (  
 vin VARCHAR(17) ,  
 registration\_number VARCHAR(10),  
 brand VARCHAR(25),  
 model VARCHAR(60),  
 manufacture\_year VARCHAR(4),  
 owner\_id INT,  
 owner\_name TEXT  
) AS $$  
BEGIN  
 RETURN QUERY  
 SELECT v.vin,  
 v.registration\_number,  
 v.brand,  
 v.model,  
 v.manufacture\_year,  
 v.owner\_id,  
 CONCAT(o.name, ' ', o.last\_name) AS owner\_name  
 FROM vehicle v  
 JOIN driver d ON v.vin = d.vin  
 JOIN violation vio ON d.drivers\_licence = vio.drivers\_licence  
 JOIN violation\_act va ON vio.violation\_id = va.violation\_id  
 JOIN vehicle\_owner o ON v.owner\_id = o.owner\_id  
 WHERE va.violation\_act\_id = v\_violation\_id;  
END;  
$$ LANGUAGE plpgsql;

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 7.3 – результат функції get\_vehicle\_data\_by\_violation\_id

Функція, що виведе на екран тип правопорушення та їх кількість, що виникла за останній місяць.

CREATE OR REPLACE FUNCTION violation\_count\_for\_the\_last\_month()  
RETURNS TABLE (violation\_description TEXT, violation\_count INT)  
AS $$  
BEGIN  
 RETURN QUERY  
 SELECT va.violation\_description,  
 COUNT(\*)::INT AS violation\_count  
 FROM violation\_act va  
 WHERE va.date\_time >= NOW() - INTERVAL '1 month'  
 GROUP BY va.violation\_description  
 ORDER BY violation\_count DESC;  
END;  
$$ LANGUAGE plpgsql;

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

Автоматически созданное описание

Рисунок 7.4 – результат функції violation\_count\_for\_the\_last\_month()

Функція, що відбере офіцерів поліції, що за останні три роки не зареєстрували жодного правопорушення.

CREATE OR REPLACE FUNCTION officers\_with\_no\_violation\_acts\_recently()  
RETURNS TABLE (police\_officer\_name TEXT)  
AS $$  
BEGIN  
 RETURN QUERY  
 SELECT CONCAT(p.name, ' ', p.last\_name) AS police\_officer\_name  
 FROM police\_officer p  
 LEFT JOIN (  
 SELECT DISTINCT va.officer\_id  
 FROM violation\_act va  
 WHERE va.date\_time >= NOW() - INTERVAL '3 years'  
 ) AS recent\_violations ON p.officer\_id = recent\_violations.officer\_id  
 WHERE recent\_violations.officer\_id IS NULL;  
END;  
$$ LANGUAGE plpgsql;

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

Автоматически созданное описание

Рисунок 7.5 – результат функції officers\_with\_no\_violation\_acts\_recently

Функція, що підраховує середню кількість правопорушень спираючись на марку автівки.

CREATE OR REPLACE FUNCTION avg\_violations\_by\_vehicle\_make()  
RETURNS TABLE (vehicle\_make VARCHAR(25), avg\_num\_of\_violations REAL)  
AS $$  
BEGIN  
 RETURN QUERY  
 SELECT v.brand AS vehicle\_make,  
 AVG(va.violation\_act\_id)::REAL AS avg\_num\_of\_violations  
 FROM violation\_act va  
 JOIN violation vio ON va.violation\_id = vio.violation\_id  
 JOIN driver d ON vio.drivers\_licence = d.drivers\_licence  
 JOIN vehicle v ON d.vin = v.vin  
 WHERE v.brand IS NOT NULL  
 GROUP BY v.brand  
 ORDER BY avg\_num\_of\_violations DESC;  
END;  
$$ LANGUAGE plpgsql;

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

Автоматически созданное описание

Рисунок 7.6 – результат функції avg\_violations\_by\_vehicle\_make()

Функція, що знайде свідків, які давали показання на переданий параметром тип адміністративного правопорушення на дорозі.

CREATE OR REPLACE FUNCTION get\_witnesses\_by\_violation\_description(violation\_description\_param TEXT)  
RETURNS TABLE (witness\_name TEXT)  
AS $$  
BEGIN  
 RETURN QUERY  
 SELECT CONCAT(w.name, ' ', w.last\_name) AS witness\_name  
 FROM witness w  
 JOIN violation\_act va ON w.violation\_act\_id = va.violation\_act\_id  
 JOIN violation v ON va.violation\_id = v.violation\_id  
 WHERE v.violation\_type = violation\_description\_param;  
END;  
$$ LANGUAGE plpgsql;

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

Автоматически созданное описание

Рисунок 7.7 – результат функції get\_witnesses\_by\_violation\_description

Функція, що виведе автівки, котрі були зафіксовані у адміністративному правопорушенні у певному місті переданому функції через параметр.

CREATE OR REPLACE FUNCTION get\_vehicles\_by\_violation\_location(town\_param VARCHAR(25))  
RETURNS TABLE (vin VARCHAR(17), registration\_number VARCHAR(10), brand VARCHAR(25), model VARCHAR(60))  
AS $$  
BEGIN  
 RETURN QUERY  
 SELECT DISTINCT v.vin, v.registration\_number, v.brand, v.model  
 FROM vehicle v  
 JOIN driver d ON v.vin = d.vin  
 JOIN violation vio ON d.drivers\_licence = vio.drivers\_licence  
 JOIN violation\_act va ON vio.violation\_id = va.violation\_id  
 JOIN violation\_location vl ON va.location\_id = vl.location\_id  
 WHERE vl.town = town\_param;  
END;  
$$ LANGUAGE plpgsql;

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

Автоматически созданное описание

Рисунок 7.8 – результат функції get\_vehicles\_by\_violation\_location

Функція, що виводить на екран тип правопорушення, час, статус справи та місто, в якому воно було зафіксовано за водійським посвідченням, яке було передано у функцію.

CREATE OR REPLACE FUNCTION get\_violations\_by\_license(drivers\_licence\_param VARCHAR(22))  
RETURNS TABLE (violation\_type VARCHAR(255), date\_time TIMESTAMP, status VIOLATION\_ACT\_STATUS, town VARCHAR(25))  
AS $$  
BEGIN  
 RETURN QUERY  
 SELECT v.violation\_type, v.date\_time, va.status, l.town  
 FROM violation v  
 JOIN violation\_act va ON v.violation\_id = va.violation\_id  
 JOIN violation\_location l ON va.location\_id = l.location\_id  
 WHERE v.drivers\_licence = drivers\_licence\_param;  
END;  
$$ LANGUAGE plpgsql;

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

Автоматически созданное описание

Рисунок 7.9 – результат функції get\_violations\_by\_license

Функція, що виведе на екран ім’я та прізвище власника автівки, що була зафіксована у адміністративному правопорушенні у певному місті, переданому функції як параметр.

CREATE OR REPLACE FUNCTION get\_vehicle\_owners\_by\_location(town\_param VARCHAR(25))  
RETURNS TABLE (owner\_name VARCHAR(25), owner\_last\_name VARCHAR(25))  
AS $$  
BEGIN  
 RETURN QUERY  
 SELECT DISTINCT o.name AS owner\_name, o.last\_name AS owner\_last\_name  
 FROM vehicle\_owner o  
 JOIN vehicle v ON o.owner\_id = v.owner\_id  
 JOIN driver d ON v.vin = d.vin  
 JOIN violation vio ON d.drivers\_licence = vio.drivers\_licence  
 JOIN violation\_act va ON vio.violation\_id = va.violation\_id  
 JOIN violation\_location vl ON va.location\_id = vl.location\_id  
 WHERE vl.town = town\_param;  
END;  
$$ LANGUAGE plpgsql;

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

Автоматически созданное описание

Рисунок 7.10 – результат функції get\_vehicle\_owners\_by\_location

**Створення представлень**

Представлення, яке демонструє кількість автівок, що має власник та інформацію про власника.

CREATE OR REPLACE VIEW owner\_data AS  
SELECT CONCAT(name, ' ', last\_name) AS full\_name,  
 phone\_number,  
 address,  
 ipn\_code,  
 COUNT(v.vin) AS total\_vehicles  
FROM vehicle\_owner  
INNER JOIN vehicle v ON vehicle\_owner.owner\_id = v.owner\_id  
GROUP BY full\_name, phone\_number, address, ipn\_code;

Изображение выглядит как текст, снимок экрана, меню, число

Автоматически созданное описание

Рисунок 8.1 – результат представлення owner\_data

Представлення, що покаже інформацію про водіїв та їх адміністративне правопорушення, а також інформацію про зафіксовану автівку.

CREATE OR REPLACE VIEW request\_driver\_data AS  
SELECT CONCAT(name, ' ', last\_name) AS full\_name,  
 vi.drivers\_licence,  
 phone\_number,  
 address,  
 v.vin,  
 v.registration\_number,  
 v.brand,  
 v.model,  
 v.manufacture\_year,  
 vi.violation\_type  
FROM driver  
INNER JOIN vehicle v ON v.vin = driver.vin  
INNER JOIN violation vi ON driver.drivers\_licence = vi.drivers\_licence;

Изображение выглядит как снимок экрана, текст, черный

Автоматически созданное описание

Рисунок 8.2 – результат представлення request\_driver\_data

Пердставлення, що продемонструє опис адміністративного правопорушення, інформацію, про водія, суму та статус штрафу.

CREATE OR REPLACE VIEW violation\_act\_information AS  
SELECT va.violation\_description,  
 CONCAT(d.name, ' ', d.last\_name) AS driver\_name,  
 d.drivers\_licence,  
 f.fine\_amount,  
 f.status AS fine\_status  
FROM violation\_act va  
INNER JOIN fine f ON f.fine\_id = va.fine\_id  
INNER JOIN violation v ON v.violation\_id = va.violation\_id  
INNER JOIN driver d ON v.drivers\_licence = d.drivers\_licence;

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

Автоматически созданное описание

Рисунок 8.3 – результат представлення violation\_act\_information

Представлення, що визначить водіїв, що їхали на високій швидкості та зафіксовану автівку.

CREATE OR REPLACE VIEW speeding\_drivers\_and\_vehicles AS  
SELECT CONCAT(d.name, ' ', d.last\_name) AS driver\_name,  
 d.drivers\_licence,  
 v.vin,  
 v.registration\_number,  
 vio.violation\_type  
FROM driver d  
INNER JOIN vehicle v ON v.vin = d.vin  
INNER JOIN violation vio ON vio.drivers\_licence = d.drivers\_licence  
WHERE vio.violation\_type = 'Speeding';

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

Автоматически созданное описание

Рисунок 8.4 – результат представлення speeding\_drivers\_and\_vehicles

**Створення запитів**

Запит, що продемонструє власників автівок та загальну суму штрафів, що були зафіксовані у справи про адміністративне правопорушення на дорозі принаймні двічі.

SELECT vo.name,  
 vo.last\_name,  
 SUM(f.fine\_amount) AS total\_fine\_amount  
FROM vehicle\_owner vo  
INNER JOIN vehicle ch ON vo.owner\_id = ch.owner\_id  
INNER JOIN driver d ON ch.vin = d.vin  
INNER JOIN violation v ON d.drivers\_licence = v.drivers\_licence  
INNER JOIN violation\_act va ON v.violation\_id = va.violation\_id  
INNER JOIN fine f ON va.fine\_id = f.fine\_id  
WHERE v.status = 'registered'  
GROUP BY vo.owner\_id, vo.name, vo.last\_name  
HAVING COUNT(\*) >= 2;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 9.01 – результат запиту про автівки, що були зафіксовані більше, ніж двічі

Запит, що покаже інформацію про свідка та власника автівки, що була зафіксована у правопорушенні з специфікованим id справи.

SELECT CONCAT(w.name, ' ', w.last\_name) AS witness\_name,  
 CONCAT(vo.name, ' ', vo.last\_name) AS owner\_name  
FROM witness w  
INNER JOIN violation\_act va ON w.violation\_act\_id = va.violation\_act\_id  
INNER JOIN violation v ON va.violation\_id = v.violation\_id  
INNER JOIN driver d ON v.drivers\_licence = d.drivers\_licence  
INNER JOIN vehicle vh ON d.vin = vh.vin  
INNER JOIN vehicle\_owner vo ON vh.owner\_id = vo.owner\_id  
WHERE va.violation\_act\_id = ?;

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 9.02 – результат запиту про свідків та власника автівки у зареєстрованій справі

Запит, що продемонструє інформацію про автівки та ПІБ їх власників.

SELECT v.vin,  
 v.registration\_number,  
 CONCAT(vo.name, ' ', vo.last\_name) AS owner\_name,  
 vio.violation\_type,  
 vio.date\_time  
FROM vehicle v  
JOIN vehicle\_owner vo ON v.owner\_id = vo.owner\_id  
JOIN driver d ON v.vin = d.vin  
LEFT JOIN violation vio ON d.drivers\_licence = vio.drivers\_licence;

Изображение выглядит как текст, снимок экрана, меню

Автоматически созданное описание

Рисунок 9.03 – результат запиту про автівки та їх власників

Запит, що продемонструє інформацію про автівки, їх власників та відповідну інформацію про здійснене правопорушення.

SELECT v.vin,  
 v.registration\_number,  
 CONCAT(vo.name, ' ', vo.last\_name) AS owner\_name,  
 vio.violation\_type,  
 vio.date\_time  
FROM vehicle v  
JOIN vehicle\_owner vo ON v.owner\_id = vo.owner\_id  
JOIN driver d ON v.vin = d.vin  
LEFT JOIN violation vio ON d.drivers\_licence = vio.drivers\_licence;

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 9.04 – результат запиту про автівки, їх власників та правопорушення

Запит, що продемонструє водіїв, що не сплатили їхній штраф.

SELECT CONCAT(d.name, ' ', d.last\_name) AS driver\_name,  
 d.drivers\_licence,  
 f.fine\_amount  
FROM driver d  
INNER JOIN violation v ON d.drivers\_licence = v.drivers\_licence  
INNER JOIN violation\_act va ON v.violation\_id = va.violation\_id  
INNER JOIN fine f ON va.fine\_id = f.fine\_id  
WHERE f.status = 'not payed';

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 9.05 – результат запиту про водіїв, що не сплатили штраф

Запит, що продемонструє усі завершені справи по адміністративним правопорушенням, їх інформацію та ім’я водія.

SELECT CONCAT(d.name, ' ', d.last\_name) AS driver\_name,  
 d.drivers\_licence,  
 va.violation\_description,  
 va.status  
FROM driver d  
INNER JOIN violation v ON d.drivers\_licence = v.drivers\_licence  
INNER JOIN violation\_act va ON v.violation\_id = va.violation\_id  
WHERE va.status = 'closed';

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 9.06 – результат запиту про закриті справи по адміністративним правопорушенням

Запит, що виведе id активних справ про адміністративне правопорушення, ПІБ свідка та його свідчень, а також інформацію про водія.

SELECT va.violation\_act\_id,  
 CONCAT(w.name, ' ', w.last\_name) AS witness\_name,  
 va.witness\_testimony,  
 va.status,  
 CONCAT(d.name, ' ', d.last\_name) AS driver\_name,  
 d.drivers\_licence  
FROM violation\_act va  
INNER JOIN witness w ON va.violation\_act\_id = w.violation\_act\_id  
INNER JOIN violation v ON v.violation\_id = va.violation\_id  
INNER JOIN driver d ON v.drivers\_licence = d.drivers\_licence  
WHERE va.status = 'active';

Изображение выглядит как снимок экрана, текст, Шрифт

Автоматически созданное описание

Рисунок 9.07 – результат запиту про активні справи, свідків та водіїв

Запит, що підрахує загальну кількість правопорушень за їх типом та підрахує середню суму штрафу на кожен тип.

SELECT v.violation\_type,  
 COUNT(\*) AS num\_violations,  
 AVG(f.fine\_amount) AS avg\_fine\_amount  
FROM violation v  
INNER JOIN violation\_act va on v.violation\_id = va.violation\_id  
INNER JOIN fine f ON va.fine\_id = f.fine\_id  
GROUP BY violation\_type  
ORDER BY num\_violations DESC;

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

Автоматически созданное описание

Рисунок 9.08 – результат запиту про типи правопорушень, їх загальну кількість та середню суму штрафу

Запит, що продемонструє усіх офіцерів та загальну кількість активних, ними зареєстрованих справ відповідно до кожного типу адміністративного правопорушення.

SELECT p.officer\_id,  
 CONCAT(p.name, ' ', p.last\_name) AS police\_officer\_name,  
 v.violation\_type,  
 COUNT(\*) AS num\_violations  
FROM police\_officer p  
INNER JOIN violation\_act va ON va.officer\_id = p.officer\_id  
INNER JOIN violation v ON va.violation\_id = v.violation\_id  
WHERE va.status = 'active'  
GROUP BY p.officer\_id, v.violation\_type  
ORDER BY police\_officer\_name;

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

Автоматически созданное описание

Рисунок 9.09 – результат запиту про типи правопорушень та їх кількість зафіксованим окремим офіцером

Запит, що підрахує загальну кількість правопорушень на кожній із зафіксованих локацій.

SELECT vc.name AS violation\_category,  
 CONCAT(vl.town, ' ', vl.street, ' ', vl.building\_number) AS violation\_location,  
 COUNT(v.violation\_id) AS num\_violations  
FROM violation\_location vl  
INNER JOIN violation\_act va ON vl.location\_id = va.location\_id  
INNER JOIN violation v ON va.violation\_id = v.violation\_id  
INNER JOIN violation\_category vc ON v.violation\_category\_id = vc.violation\_category\_id  
GROUP BY violation\_category, violation\_location  
ORDER BY violation\_location;

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

Автоматически созданное описание

Рисунок 9.10 – результат запиту про типи правопорушень та їх кількість на відповідних локаціях

Запит, що демонструє інформацію про водіїв та термін проплати штрафу, що мали проплатити штрафи протягом попереднього року проте не зробили цього.

SELECT CONCAT(d.name, ' ', d.last\_name) AS driver\_name,  
 d.drivers\_licence,  
 f.payment\_term  
FROM driver d  
INNER JOIN violation v ON v.drivers\_licence = d.drivers\_licence  
INNER JOIN violation\_act va ON v.violation\_id = va.violation\_id  
INNER JOIN fine f ON f.fine\_id = va.fine\_id  
WHERE f.status = 'not payed'  
AND f.payment\_term BETWEEN (NOW() - INTERVAL '1 year') AND NOW();

Изображение выглядит как текст, снимок экрана, меню, Шрифт

Автоматически созданное описание

Рисунок 9.11 – результат запиту про водіїв, що не сплатили штраф протягом року

Запит, що підрахує загальну суму штрафів, що має бути сплачена відповідного до кожного типу адміністративних правопорушень на дорозі.

SELECT v.violation\_type,  
 SUM(f.fine\_amount) AS sum\_fine\_amount  
FROM violation v  
INNER JOIN violation\_act va on v.violation\_id = va.violation\_id  
INNER JOIN fine f ON va.fine\_id = f.fine\_id  
WHERE f.status = 'not payed'  
GROUP BY violation\_type;

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 9.12 – результат запиту про загальну сушу штрафу до сплати

Запит, що демонструє офіцерів та підраховує кількість свідків, що давали їм показання.

SELECT p.officer\_id,  
 CONCAT(p.name, ' ', p.last\_name) AS police\_officer\_name,  
 COUNT(DISTINCT w.name) AS num\_witness\_occurrences  
FROM police\_officer p  
JOIN violation\_act va ON p.officer\_id = va.officer\_id  
JOIN witness w ON va.violation\_act\_id = w.violation\_act\_id  
GROUP BY p.officer\_id, police\_officer\_name  
ORDER BY num\_witness\_occurrences DESC;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 9.13 – результат запиту про офіцерів та кількість свідків, що давали показання

Запит, що продемонструє кожну марку автівки та скільки разів вона фігурувала у правопорушенні.

SELECT v.brand,  
 COUNT(DISTINCT vl.location\_id) AS num\_locations\_seen  
FROM Vehicle v  
LEFT JOIN Driver d ON v.vin = d.vin  
LEFT JOIN Violation viol ON d.drivers\_licence = viol.drivers\_licence  
LEFT JOIN Violation\_Act va ON viol.violation\_id = va.violation\_id  
LEFT JOIN Violation\_Location vl ON va.location\_id = vl.location\_id  
WHERE v.brand IS NOT NULL  
GROUP BY v.brand  
ORDER BY num\_locations\_seen DESC;

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

Автоматически созданное описание

Рисунок 9.14 – результат запиту про марки автомобілів та їх фіксування у правопорушеннях

Запит, що підрахує загальну кількість правопорушень у кожному місті.

SELECT vl.town,  
 COUNT(v.violation\_id) AS num\_violations  
FROM violation\_location vl  
INNER JOIN violation\_act va ON vl.location\_id = va.location\_id  
INNER JOIN violation v ON va.violation\_id = v.violation\_id  
GROUP BY town  
ORDER BY num\_violations DESC;

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Рисунок 9.15 – результат запиту про міста та кількість у них правопорушень

Запит, що підраховує загальну кількість штрафів за рік та загальну суму усіх штрафів.

SELECT EXTRACT(YEAR FROM va.date\_time) AS year,  
 COUNT(\*) AS num\_fines,  
 SUM(f.fine\_amount) AS total\_fine\_amount  
FROM Violation\_Act va  
JOIN Fine f ON va.fine\_id = f.fine\_id  
GROUP BY year  
ORDER BY year;

Изображение выглядит как текст, снимок экрана, число

Автоматически созданное описание

Рисунок 9.16 – результат запиту про загальні кількість штрафів та їх суму за кожен рік

Запит, що демонструє інформацію про водіїв, що були зафіксовані під час комендантської години.

SELECT CONCAT(d.name, ' ', d.last\_name) AS driver\_name,  
 d.drivers\_licence  
FROM driver d  
WHERE d.drivers\_licence IN (  
 SELECT DISTINCT v.drivers\_licence  
 FROM violation v  
 JOIN violation\_act va ON v.violation\_id = va.violation\_id  
 WHERE EXTRACT(HOUR FROM v.date\_time) >= 0 AND EXTRACT(HOUR FROM v.date\_time) < 5  
 AND EXTRACT(YEAR FROM v.date\_time) >= 2022  
);

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 9.17 – результат запиту про водіїв, що були зафіксовані під час комендантської години

Запит, що демонструє інформацію про водіїв, чиї справи про адміністративне правопорушення активні та сума штрафу менша від 1200.

SELECT CONCAT(d.name, ' ', d.last\_name) AS driver\_name,  
 d.drivers\_licence  
FROM driver d  
WHERE d.drivers\_licence IN (  
 SELECT v.drivers\_licence  
 FROM violation v,  
 violation\_Act va,  
 fine f  
 WHERE va.violation\_id = v.violation\_id  
 AND va.fine\_id = f.fine\_id  
 AND f.fine\_amount < 1200  
 AND va.status = 'active'  
);

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 9.18 – результат запиту про водіїв чиї справи активні та сума штрафу менша, ніж 1200

Запит, що демонструє офіцерів поліції, що зафіксували найбільшу кількість правопорушень відповідно до типу правопорушення.

SELECT officer\_id,  
 police\_officer\_name,  
 violation\_type,  
 num\_violation\_acts  
FROM (  
 SELECT p.officer\_id,  
 CONCAT(p.name, ' ', p.last\_name) AS police\_officer\_name,  
 v.violation\_type,  
 COUNT(va.violation\_act\_id) AS num\_violation\_acts,  
 ROW\_NUMBER() OVER (PARTITION BY v.violation\_type ORDER BY COUNT(va.violation\_act\_id) DESC) AS rank  
 FROM police\_Officer p  
 INNER JOIN violation\_Act va ON p.officer\_id = va.officer\_id  
 INNER JOIN violation v ON va.violation\_id = v.violation\_id  
 GROUP BY p.officer\_id, CONCAT(p.name, ' ', p.last\_name), v.violation\_type  
) AS RankedOfficers  
WHERE rank = 1  
ORDER BY num\_violation\_acts DESC;

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 9.19 – результат запиту про офіцерів, що склали найбільшу кількість справ

Запит, що демонструє інформацію про водіїв, що скоїли адміністративне правопорушення минулого року, сплатили штраф, через що їх справу було закрито, та не здійснювали правопорушень цього року.

SELECT CONCAT(d.name, ' ', d.last\_name) AS driver\_name,  
 d.drivers\_licence  
FROM driver d  
WHERE d.drivers\_licence IN (  
 SELECT DISTINCT v.drivers\_licence  
 FROM violation v  
 JOIN violation\_act va ON v.violation\_id = va.violation\_id  
 WHERE va.status = 'closed'  
 AND v.date\_time BETWEEN CURRENT\_DATE - INTERVAL '1 year' AND CURRENT\_DATE  
 AND v.drivers\_licence NOT IN (  
 SELECT DISTINCT v2.drivers\_licence  
 FROM violation v2  
 WHERE EXTRACT(YEAR FROM v2.date\_time) = EXTRACT(YEAR FROM CURRENT\_DATE)  
 )  
);

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 9.20 – результат запиту про водіїв, чию справу було закрито минулого року та не було жодної нової протягом цього року

**Створення індексів**

Запустимо перший запит із попереднього пункту та перевіримо скільки часу відводиться на його виконання.

EXPLAIN ANALYZE SELECT vo.name,  
 vo.last\_name,  
 SUM(f.fine\_amount) AS total\_fine\_amount  
FROM vehicle\_owner vo  
INNER JOIN vehicle ch ON vo.owner\_id = ch.owner\_id  
INNER JOIN driver d ON ch.vin = d.vin  
INNER JOIN violation v ON d.drivers\_licence = v.drivers\_licence  
INNER JOIN violation\_act va ON v.violation\_id = va.violation\_id  
INNER JOIN fine f ON va.fine\_id = f.fine\_id  
WHERE v.status = 'registered'  
GROUP BY vo.owner\_id, vo.name, vo.last\_name  
HAVING COUNT(\*) >= 2;

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 10.1 – час виконання запиту без оптимізації індексами

Тепер створимо індекси та запустимо цей самий запит ще один раз.

CREATE INDEX vehicle\_owner\_idx ON vehicle\_owner(name, last\_name, address, phone\_number, ipn\_code);  
  
CREATE INDEX vehicle\_idx ON vehicle(vin, registration\_number, brand, model, manufacture\_year);  
  
CREATE INDEX driver\_idx ON driver(name,last\_name,address,phone\_number);  
  
CREATE INDEX violation\_category\_idx ON violation\_category(name,description);  
  
CREATE INDEX violation\_idx ON violation(violation\_type,date\_time,status);  
  
CREATE INDEX police\_officer\_idx ON police\_officer(name,last\_name,phone\_number);  
  
CREATE INDEX violation\_location\_idx ON violation\_location(town,street,building\_number);  
  
CREATE INDEX fine\_idx ON fine(fine\_amount,payment\_term,status);  
  
CREATE INDEX violation\_act\_idx ON violation\_act(violation\_description,date\_time,witness\_testimony,status);  
  
CREATE INDEX witness\_idx ON witness(name,last\_name,phone\_number);

Изображение выглядит как текст, снимок экрана, Шрифт, меню

Автоматически созданное описание

Рисунок 10.21 – час виконання запиту з оптимізацією

Як можна побачити, перший раз запит виконувався 12.5ms, а запит з використанням індексів було виконано за 3,5ms, що є значно кращим результатом. У великій базі даних ця різниця може бути критичною. Проте слід зазначити, що на збереження послідовності індексів додатково виділяється пам’ять.

## ВИСНОВОК

Перед виконанням завдання по створенню бази для підтримки системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху, 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 системи фіксації адміністративних правопорушень. Було створено 10 різних таблиць та визначені зв’язки між ними за допомогою зовнішніх ключів і каскадних дій. Були закріплені навички створювання та використовування різних об’єктів бази даних, такі як тригери та функції, 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/](http://www.w3schools.com/mysql/)