

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КІЇВСЬКИЙ 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

Студента 2 курсу ІП-33 групи
спеціальності 121 «Інженерія програмного
забезпечення»

Соколов О. В.
(прізвище та ініціали)

Керівник Ліщук Катерина Ігорівна, доц., к.т.н.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _____

Кількість балів: _____ Оцінка ECTS _____

Члени комісії

<u>(підпись)</u>	<u>(вчене звання, науковий ступінь, прізвище та ініціали)</u>
<u>(підпись)</u>	<u>(вчене звання, науковий ступінь, прізвище та ініціали)</u>
<u>(підпись)</u>	<u>(вчене звання, науковий ступінь, прізвище та ініціали)</u>

Київ – 2024 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-33 Семестр 3

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

Соколов Олександр Вячеславович

(прізвище, ім'я, по батькові)

1. Тема роботи БАЗА ДАНИХ ДЛЯ ПІДТРИМКИ СИСТЕМИ ФІКСАЦІЇ
АДМІНІСТРАТИВНИХ ПРАВОПОРУШЕНЬ У СФЕРІ
ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ ДОРОЖНЬОГО РУХУ

керівник роботи Ліщук Катерина Ігорівна, доц., к.т.н.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи 22.12.2024

3. Вихідні дані до роботи створення бази даних для підтримки системи
фіксації адміністративних правопорушень у сфері забезпечення безпеки
дорожнього руху

4. 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 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

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових
креслень)

6. Дата видачі завдання 04.11.2023

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання курсового проекту	Срок виконання етапів проекту	Примітка
1	Аналіз предметного середовища	2.12.2023	
2	Побудова ER-моделі	5.12.2023	
3	Побудова реляційної схеми з ER-моделі	8.12.2023	
4	Створення бази даних, у форматі обраної системи управління базою даних	13.12.2023	
5	Створення користувачів бази даних	14.12.2023	
6	Імпорт даних з використанням засобів СУБД в створену базу даних	15.12.2023	
7	Створення запитів мовою SQL	17.12.2023	
8	Оптимізація роботи запитів	18.12.2023	
9	Оформлення пояснювальної записки	22.12.2023	
10	Захист курсової роботи	25.12.2024	

Студент

(підпис)

Соколов О. В.

(прізвище та ініціали)

Керівник роботи

(підпис)

Ліщук К. І.

(прізвище та ініціали)

АНОТАЦІЯ

Пояснювальна записка до курсової роботи: 102 сторінки, 55 рисунків, 15 таблиць, 7 посилань.

Об'єкт дослідження: база даних для підтримки системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху.

Мета роботи: розробка та реалізація бази даних для підтримки системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху.

Був проведений аналіз предметного середовища, на його основі побудована ER-модель та реляційна схема. Використовуючи СУБД PostgreSQL була створена база даних, в якій налаштовані користувачі та ролі. Дані були імпортовані до цієї бази. Також були розроблені функції, процедури, тригери, представлення та SQL-запити для ефективної роботи з базою даних. Окрім цього, була проведена оптимізація запитів для поліпшення їх продуктивності.

Здійснено успішну реалізацію бази даних для складського обліку підприємства використовуючи СУБД PostgreSQL, відповідно до вказаного варіанту завдання.

ЗМІСТ

ВСТУП.....	6
1 АНАЛІЗ ПРЕДМЕТНОГО СЕРЕДОВИЩА.....	8
1.1 Опис предметного середовища.....	8
1.2 Аналіз існуючих програмних продуктів.....	10
1.3 Висновки	14
2 ПОСТАНОВКА ЗАВДАННЯ	15
2.1 Завдання	15
2.2 Мета	15
2.2 Вимоги до бази даних	15
2.3 Висновки	16
3 ПОБУДОВА ЕР-МОДЕЛІ.....	17
3.1 Бізнес-правила	17
3.2 Видлені сутності	18
3.3 Опис сутностей.....	19
Таблиця 3.1 – Опис сутностей.....	19
3.4 Опис зв’язків між сутностями	28
3.5 ЕР-модель.....	30
3.6 Висновки	31
4 РЕАЛІЯЦІЙНА МОДЕЛЬ БАЗИ ДАНИХ.....	33
4.1 Вибір СУБД.....	33
4.2 Структура таблиць.....	34
Таблиця 4.1 – структура таблиці «citizens».....	35
Таблиця 4.2 – структура таблиці «drivers».....	35
Таблиця 4.3 – структура таблиці «police_officers»	36
Таблиця 4.4 – структура таблиці «vehicle_types».....	37
Таблиця 4.5 – структура таблиці «vehicles».....	38
Таблиця 4.6 – структура таблиці «traffic_rules»	39
Таблиця 4.7 – структура таблиці «administrative_offenses»	39
Таблиця 4.8 – структура таблиці «locations».....	40
Таблиця 4.9 – структура таблиці «violations»	41
Таблиця 4.10 – структура таблиці «evidences»	42
Таблиця 4.11 – структура таблиці «accident_protocols»	43
Таблиця 4.12 – структура таблиці «citizens_on_protocol»	44
Таблиця 4.13 – структура таблиці «accident_resolutions»	45
Таблиця 4.14 – структура таблиці «regions»	46
4.4 Висновки	47
5 РЕАЛІЗАЦІЯ БАЗИ ДАНИХ	48
5.1 Створення бази даних.....	48
5.2 Імпортування даних в таблиці	56
6 СТВОРЕННЯ КОРИСТУВАЧІВ БАЗИ ДАНИХ	65
6.1 Адміністратор	65
6.2 Поліцейський	65
6.3 Водій	66
6.4 Громадянин	66
6.6 Висновки	68
7 РОБОТА З БАЗОЮ ДАНИХ	69
7.1 Тригери	69
7.1.1 Trigger trg_check_engine_capacity.....	69

7.1.2 Тригер <i>trg_check_seating_capacity</i>	70
7.1.3 Тригер <i>check_protocol_validity</i>	72
7.1.4 Тригер <i>trg_check_citizens_on_protocol</i>	73
7.1.5 Тригер <i>trg_check_resolution_validity</i>	75
7.1.6 Тригери <i>trg_checkViolation_exclusivity_protocol</i> , <i>trg_checkViolation_exclusivity_resolution</i>	76
7.2 ПРЕДСТАВЛЕННЯ	78
7.2.1 Представлення <i>violation_details</i>	78
7.2.2 Представлення <i>full_vehicle_info</i>	80
7.2.3 Представлення <i>driverViolation_summary</i>	82
7.3 ФУНКЦІЇ ТА ПРОЦЕДУРИ	84
7.3.1 Функція <i>is_citizen_older_than</i>	84
7.3.2 Функція <i>get_region_by_code</i>	85
7.3.3 Функція <i>get_administrative_offense_info</i>	87
7.3.4 Функція <i>get_officer_protocol_count</i>	88
7.3.5 Функція <i>get_officer_resolution_count</i>	89
7.3.6 Функція <i>get_citizen_full_name</i>	91
7.3.7 Функція <i>get_plate_type</i>	92
7.3.8 Функція <i>get_total_penalty_fees</i>	94
7.3.9 Функція <i>get_vehicle_owner</i>	96
7.3.10 Функція <i>get_total_violations_for_vehicle</i>	98
7.3.11 Процедура <i>transfer_vehicle_ownership</i>	100
7.3.12 Процедура <i>registerViolation</i>	102
7.4 SQL-ЗАПИТИ	105
7.4.1 Перелік всіх громадян та їх водійських посвідчень.....	105
7.4.2 Перелік протоколів для певного громадянина	106
7.4.3 Перелік поліцейських, які склали найбільше протоколів та постанов	108
7.4.4 Перелік постанов, де автомобіль знаходитьться у власності поліцейського	110
7.4.5 Перелік громадян, які порушували ПДР найчастіше за останній рік	111
7.4.6 Кількість порушників та порушень, які сконцентровані на автомобілях із звичайними номерами та іменними.....	112
7.4.7 Кількість порушень по регіону реєстрації машини.....	114
7.4.8 Кількість порушень машинами із та без страхового полісу	118
7.4.9 Перелік транспортних засобів, що порушили ПДР за останній місяць	119
7.4.10 Перелік постанов із штрафами	121
7.4.11 Перелік адміністративних порушень із кількістю порушень	122
7.4.12 Перелік доказів для конкретного порушення	124
7.4.13 Перелік водіїв, які зробили два однакові порушення за перші два роки після отримання водійського посвідчення.....	125
7.4.14 Топ виробників автомобілів, власники яких мають найбільшу відносну кількість правопорушень	126
7.4.15 Кількість порушень без протоколів, без постанов, із протоколами та постановами, загальна кількість порушень	127
7.4.16 Перелік громадян з найбільшою кількістю свідчень на інших осіб	129
7.4.17 Перелік громадян які були і свідками і порушниками	130
7.4.18 Топ громадян за сумою штрафів	131
7.4.19 Найчастіше порушувані статті за типом транспортного засобу	133
7.4.20 Топ водіїв автобусів, які отримали протокол чи постанову за керування у нетверезому стані	135
7.4.21 Кількість свідків та жертв в протоколах	137
7.4.22 Найпоширеніший тип порушень ПДР по кожній вулиці	139
7.4.23 Інформація про водіїв, які мають транспортні засоби різних типів	140
7.5 ІНДЕКСИ	142
7.6 Висновки	144
ВИСНОВКИ	146
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	148

ВСТУП

У сучасному світі забезпечення безпеки дорожнього руху є одним із найважливіших завдань суспільства. Зростаюча кількість транспортних засобів та складність дорожньої інфраструктури зумовлюють необхідність створення ефективних систем для фіксації адміністративних правопорушень. Одним із ключових елементів таких систем є база даних, яка забезпечує обробку, зберігання та аналіз великого обсягу інформації.

Актуальність теми дослідження обумовлена потребою у вдосконаленні процесів управління дорожнім рухом та посиленні контролю за виконанням правил дорожнього руху. Правильна організація зберігання даних про правопорушення, винних осіб, транспортні засоби та відповідні штрафи дозволяє не лише автоматизувати адміністративні процедури, але й підвищити ефективність роботи органів, відповідальних за безпеку на дорогах.

Метою даної курсової роботи є розробка та впровадження бази даних для підтримки системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху.

У рамках курсової роботи необхідно розробити базу даних для підтримки системи фіксації адміністративних правопорушень у сфері безпеки дорожнього руху. Завдання включають аналіз предметного середовища, побудову ER-моделі та реляційної схеми, нормалізацію даних, створення бази даних у СУБД PostgreSQL[6] та налаштування багатокористувацького доступу.

Крім того, передбачено імпорт даних, написання SQL-запитів, створення функцій, процедур і тригерів, а також оптимізацію запитів для підвищення продуктивності. Усі етапи спрямовані на забезпечення цілісності даних, мінімізацію надмірності та підвищення швидкодії системи.

Сфера використання розробленої бази даних включає державні організації, які займаються моніторингом дорожнього руху, обліком адміністративних правопорушень, а також судові органи, що розглядають такі випадки.

Для реалізації курсової роботи обрана система управління базами даних (СУБД) PostgreSQL[6]. Цей вибір обумовлений низкою переваг, які роблять PostgreSQL ідеальним інструментом для створення бази даних у рамках завдання.

По-перше, PostgreSQL[6] є потужною реляційною СУБД з відкритим вихідним кодом, яка забезпечує високий рівень надійності, масштабованості та продуктивності. Її підтримка складних транзакцій і дотримання вимог до цілісності даних гарантують коректність роботи навіть для складних бізнес-завдань.

По-друге, PostgreSQL[6] має багатий набір функціональних можливостей, зокрема підтримку розширень, складних типів даних, тригерів і процедур, що дозволяє реалізувати всі необхідні вимоги проекту. Система також пропонує широкий спектр інструментів для оптимізації запитів, зокрема індекси, планувальники запитів та матеріалізовані представлення.

По-третє, ця СУБД забезпечує потужну багатокористувачську модель доступу з гнучкими механізмами розмежування прав користувачів, що є критично важливим для забезпечення безпеки та зручності роботи із системою.

Нарешті, PostgreSQL[6] має активну спільноту розробників, велику кількість документації та сумісність із сучасними інструментами розробки, що значно спрощує процес створення та обслуговування бази даних. Усі ці фактори роблять PostgreSQL[6] оптимальним вибором для реалізації системи фіксації адміністративних правопорушень.

1 АНАЛІЗ ПРЕДМЕТНОГО СЕРЕДОВИЩА

1.1 Опис предметного середовища

Об'єктом дослідження є інформаційна система для фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху. Основна мета цієї системи — автоматизувати обробку великих обсягів даних про порушення правил дорожнього руху (ПДР), реєстрацію транспортних засобів, створення та ведення протоколів і постанов, а також здійснення аналітичної обробки інформації. Завдяки цій системі забезпечується швидкий доступ до даних, їхній аналіз і можливість формування звітів, що полегшує роботу відповідних служб та підвищує ефективність управління дорожнім рухом.

Система працює з різними типами вхідних даних, які включають інформацію про громадян, транспортні засоби, порушення та адміністративні документи. Дані про громадян охоплюють персональну інформацію, таку як ім'я, прізвище, по батькові та дата народження, що є ключовими для ідентифікації особи. Система передбачає перевірку дати народження, щоб забезпечити коректний вік особи для участі в певних ролях, наприклад, для отримання водійського посвідчення або призначення поліцейським. Кожен громадянин отримує унікальний ідентифікаційний номер, що запобігає дублюванню записів.

Інформація про транспортні засоби містить деталі, такі як VIN номер, номерні знаки, тип двигуна, модель, марка, колір, рік випуску та власник. Важливим елементом є перевірка на унікальність VIN, а також перевірка відповідності формату номерних знаків, типу двигуна та року випуску автомобіля. VIN та інші параметри є важливими для точного ідентифікування транспортних засобів і забезпечують достовірність даних про кожен автомобіль.

Дані про порушення ПДР містять час і місце порушення, тип порушення, номер статті КУпАП, а також транспортний засіб, що скоїв порушення, і відповідну локацію. Крім того, вказуються порушення конкретного правила

дорожнього руху. Порушення реєструються в системі з обмеженням по часу: вони можуть бути зареєстровані лише за останні десять років. Ці дані допомагають не тільки в документуванні порушень, але й у подальшому аналізі порушень, а також дають змогу підтверджувати правопорушення через фото- чи відеоматеріали.

Протоколи та постанови створюються на основі даних про порушення і включають серію та номер документа, час його складання, а також інформацію про поліцейського, який складає документ, і порушника. Поліцейський, що складає протокол, не може бути причетним до самого порушення. Для забезпечення унікальності документів, система застосовує перевірку на унікальність комбінації серії та номера протоколу або постанови, що виключає можливість дублювання. Дані про протоколи і постанови допомагають ефективно адмініструвати процедуру накладення штрафів та інших санкцій.

Вихідні дані формуються у вигляді звітів і відповідей на запити користувачів. Наприклад, система дозволяє отримувати інформацію про громадян, які найчастіше порушували ПДР, виявляти місця з найбільшою кількістю порушень, аналізувати активність поліцейських, визначати найпоширеніші моделі транспортних засобів серед порушників або формувати рейтинги статей КУпАП за частотою порушень.

Основні бізнес-процеси системи включають реєстрацію громадян і транспортних засобів, фіксацію порушень ПДР, оформлення адміністративних документів, а також аналітику і звітність. Реєстрація забезпечує точність введення даних і унікальність кожного об'єкта. Фіксація порушень передбачає перевірку всіх параметрів на відповідність встановленим бізнес-правилам, що гарантує їхню коректність. Адміністративні документи створюються з дотриманням усіх вимог до часу, унікальності даних та повноти інформації. Аналітична частина системи дозволяє ефективно використовувати накопичені дані для прийняття обґрутованих рішень і планування заходів у сфері безпеки дорожнього руху.

1.2 Аналіз існуючих програмних продуктів

Система для підтримки фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху в Україні зосереджена на наданні державним органам, таким як поліція, інструментів для ефективного моніторингу порушень правил дорожнього руху (ПДР) та здійснення адміністративних процедур. Вона включає різноманітні механізми для виявлення порушень, їх реєстрації, а також для збору доказів, що дозволяє забезпечити коректне і справедливе накладення штрафів та інші санкції. Одним з основних елементів цієї системи є інтеграція з автоматичними камерами фіксації порушень (наприклад, за перевищення швидкості, проїзд на червоне світло), а також можливість реєстрації порушень співробітниками поліції, які складають протоколи або постанови на основі свідчень, записів відеокамер або безпосереднього спостереження.

Ця система забезпечує функціонал для того, щоб правоохоронні органи могли ефективно проводити перевірку порушень, реєструвати їх в електронних базах даних та оперативно передавати для подальшої обробки, як-от накладання штрафів або направлення справ до суду. Важливо, що система підтримує збереження доказів у вигляді фото- або відеоматеріалів, що дозволяє уникнути суб'єктивності у встановленні фактів правопорушення.

Обмеження доступу для звичайних громадян та водіїв пояснюється кількома основними факторами. По-перше, з міркувань безпеки та конфіденційності даних. Інформація про правопорушення, а також інші персональні дані громадян повинні залишатися захищеними від несанкціонованого доступу, оскільки їх публічне розкриття може привести до непотрібних маніпуляцій або порушень прав особистості. Наприклад, дозволити доступ до даних про невиплачені штрафи або про порушення ПДР безпосередньо може привести до спроб оскарження або маніпулювання результатами.

По-друге, обмеження доступу дозволяє запобігти маніпуляціям із самими процедурами накладання штрафів або оскарження постанов. Якщо

звичайний громадянин або водій отримав би повний доступ до всієї інформації в системі, це може призвести до спроб впливу на процес (наприклад, зміни статусу штрафу чи протоколу). Таким чином, контроль доступу є важливим елементом для забезпечення справедливості та правильності процесу.

Контроль за доступом до інформації, що стосується правопорушень, є важливою частиною цієї системи. Такий контроль дозволяє забезпечити, щоб лише уповноважені особи – поліціянти, судді, представники органів державної влади – могли здійснювати відповідні дії з даними про порушення. Це дозволяє мінімізувати можливість зловживань або помилок у процесах виявлення та розгляду порушень, а також гарантує, що кожен етап процесу буде здійснюватися відповідно до закону та з урахуванням прав громадян.

Тим не менш, існують програмні продукти, доступні громадянам, які дозволяють взаємодіяти з системами фіксації порушень ПДР, такі як "Електронний кабінет водія" та інші онлайн-сервіси для перевірки штрафів. Ці продукти надають водіям зручний доступ до важливої інформації, зокрема про їхні правопорушення, нараховані штрафи та інші адміністративні питання, пов'язані з транспортними засобами. Завдяки таким сервісам водії можуть без зусиль перевіряти інформацію про існуючі порушення і сплачувати штрафи, що істотно спрощує взаємодію з органами влади та скорочує час, необхідний для виконання адміністративних процедур. Однак, ці сервіси не надають доступу до можливості реєстрації нових порушень або внутрішніх даних правоохоронних органів.

"Електронний кабінет водія"^[1] є одним із таких сервісів, який дозволяє водіям отримати актуальну інформацію про їхні водійські документи, транспортні засоби та наявність штрафів. Система дозволяє перевіряти статус посвідчення водія, перевіряти наявність та суму штрафів за порушення ПДР, сплачувати ці штрафи онлайн, а також замовляти номерні знаки для транспортних засобів. Важливою перевагою є можливість отримувати сповіщення про зміни в статусі документів або наявність нових штрафів, що дозволяє водіям оперативно реагувати на ситуацію.

"Сервіс перевірки адміністративних правопорушень"[2] надає громадянам можливість перевірити наявність штрафів за порушення ПДР, які були зафіксовані за допомогою автоматичних камер. Для використання цього сервісу користувачам необхідно ввести державний номер транспортного засобу та серію і номер свідоцтва про реєстрацію. Після цього система надає доступ до актуальних даних про порушення, включаючи інформацію про конкретне правопорушення, його місце та час вчинення, а також поточний статус справи.

Цей сервіс має кілька переваг для водіїв. Він дозволяє швидко перевіряти наявність штрафів без необхідності звертатися до державних органів або відвідувати спеціалізовані установи. Завдяки автоматизованій системі обробки даних, користувачі отримують актуальну і точну інформацію про свої порушення, що значно полегшує процес їхнього вирішення. Однак слід зазначити, що цей сервіс обмежений лише перевіркою штрафів, зареєстрованих за допомогою автоматичних камер, і не охоплює порушення, зафіксовані іншими методами, наприклад, патрульними поліцейськими.

Застосунок "Штрафи ПДР"[3] надає водіям можливість не тільки сплачувати штрафи, але й отримувати повідомлення про нові постанови або зміни статусу вже винесених штрафів. Це дозволяє водіям бути в курсі всіх змін, що стосуються їхніх адміністративних правопорушень, а також отримувати оперативні сповіщення, що значно полегшує процес контролю за своїми штрафами та їх оплатою.

Однією з ключових функцій застосунку є інтеграція з мапою розташування камер автоматичної фіксації порушень. Це дозволяє водіям не тільки перевіряти наявність штрафів, але й бути в курсі місць, де можуть бути зафіксовані порушення правил дорожнього руху. Така інформація допомагає знижувати ризик несподіваних штрафів, оскільки водії можуть планувати свої маршрути з урахуванням місць, де встановлені камери. Серед переваг застосунку можна відзначити зручність у використанні, оперативність отримання сповіщень та можливість онлайн оплати штрафів.

Застосунок "Штрафи UA"[4] є національним сервісом для водіїв України, який дозволяє зручно перевіряти наявність штрафів та здійснювати їх оплату. Водії можуть перевірити штрафи за номером водійського посвідчення, ПН або серією та номером штрафу. Застосунок охоплює різні типи порушень, такі як ті, що зафіковані патрульними поліцейськими, камерами автоматичної фіксації, а також штрафи за порушення паркування в ряді міст країни.

Основне призначення сервісу полягає в наданні водіям онлайн-можливості для перевірки наявності штрафів та швидкої оплати. Це дозволяє водіям своєчасно погашати штрафи, уникати нарахування пені та розглядів з виконавчими службами. Завдяки цьому сервісу водії можуть оперативно отримувати актуальну інформацію про нові штрафи і перевіряти історію старих, що значно полегшує управління фінансами і адміністративними питаннями. Згідно з дослідженнями команди "Штрафи UA"[4], близько 70% водіїв, які не сплачують штрафи, роблять це через втрату протоколів або через складність процесу сплати, а також через відсутність часу чи забування про штрафи. Застосунок вирішує ці проблеми, надаючи зручний доступ до інформації і можливість швидкої оплати, що дозволяє уникнути шрафних санкцій за несвоєчасну оплату.

Telegram-бот МВС "Штрафи ПДР" є офіційним інструментом для перевірки штрафів за порушення ПДР в Україні. Цей бот дозволяє користувачам швидко отримати інформацію про наявність штрафів, що були зафіковані автоматичними системами або патрульними поліцейськими. Крім того, бот надає можливість здійснити оплату штрафів безпосередньо через додаток, що робить процес сплати зручним і швидким.

Однією з основних переваг Telegram-бота є його доступність та простота використання. Користувачі можуть без зайвих труднощів перевіряти свої штрафи та оплачувати їх, не покидаючи месенджер. Бот надає актуальні дані про порушення, що дозволяє водіям бути в курсі своїх обов'язків і виконувати їх своєчасно, мінімізуючи ризик додаткових штрафів за прострочення.

1.3 Висновки

У розділі було проведено аналіз існуючих програмних продуктів та сервісів, пов'язаних з фіксацією адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху. Вивчення цих продуктів показало важливість створення зручних і доступних для громадян інструментів, які дозволяють ефективно взаємодіяти з державними системами, перевіряти наявність штрафів, оплачувати їх та отримувати актуальні сповіщення.

Розглянуто функціонал таких сервісів, як "Електронний кабінет водія"[1], "Штрафи UA"[4], Telegram-боти та інші, які забезпечують водіям можливість швидко отримувати необхідну інформацію щодо адміністративних правопорушень. Це дозволяє знизити ризики забуття або затримки сплати штрафів, що, в свою чергу, допомагає уникнути нарахування пені та інших проблем.

Вивчення існуючих рішень дало змогу визначити ключові компоненти системи, які повинні бути включені в розробку, а також визначити потенційні проблеми, що потребують подальшого вдосконалення. Розробка таких сервісів важлива для підвищення ефективності процесів фіксації порушень ПДР та їх обробки.

Аналіз показав, що інтеграція сучасних технологій, зокрема мобільних додатків і чат-ботів, є важливим кроком до покращення доступу громадян до інформації та спрощення процесів сплати штрафів.

2 ПОСТАНОВКА ЗАВДАННЯ

2.1 Завдання

Завданням курсової роботи є розробка бази даних для підтримки фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху. База даних має бути орієнтована на ефективну обробку та зберігання інформації про правопорушення, порушників, транспортні засоби, штрафи та інші адміністративні процеси.

Основні завдання курсової роботи:

- аналіз предметного середовища;
- побудова ER-моделі;
- побудова реляційної схеми;
- створення бази даних;
- створення користувачів бази даних;
- імпорт даних;
- створення SQL запитів;
- оптимізація роботи запитів.

2.2 Мета

Метою роботи є створення інструменту для автоматизації збору, зберігання та обробки даних, що стосуються порушень правил дорожнього руху, та забезпечення можливості оперативного доступу до цих даних для відповідальних органів.

2.2 Вимоги до бази даних

Основні вимоги до бази даних:

- система повинна забезпечити коректне зберігання даних про порушення, порушників, штрафи, транспортні засоби, а також можливість пошуку, фільтрації та створення звітів;
- необхідно забезпечити правильність і узгодженість інформації, що зберігається в базі, шляхом введення необхідних обмежень і перевірок;

- база даних повинна ефективно зберігати тільки необхідну для виконання завдань інформацію, уникаючи дублювання даних;
- система повинна передбачати можливість використання бази кількома категоріями користувачів з різними рівнями доступу;
- база повинна мати високу швидкість обробки запитів, що дозволяє оперативно реагувати на запити користувачів.

Вимоги до бази даних включають забезпечення безпеки даних, захист від несанкціонованого доступу, а також підтримку стабільної роботи з великими обсягами інформації про правопорушення та порушників.

2.3 Висновки

У розділі було сформульовано постановку завдання, мету та визначено основні вимоги до розробки бази даних для підтримки фіксації адміністративних правопорушень у сфері дорожнього руху. Завдання курсової роботи полягає в створенні бази даних, що забезпечить ефективне зберігання, обробку та доступ до даних про порушення, порушників, транспортні засоби та штрафи.

Згідно з поставленими вимогами, система повинна забезпечувати коректність і цілісність даних, ефективно працювати з великими обсягами інформації та забезпечувати багатокористувацький доступ. Окрім того, важливою вимогою є мінімізація обсягу даних, що зберігаються, та оптимізація запитів для швидкої обробки даних.

Було визначено основні етапи роботи, включаючи побудову ER-моделі, реляційної схеми, створення бази даних у СУБД, розробку запитів і оптимізацію роботи запитів. З урахуванням цих завдань, на наступних етапах буде виконано детальне проектування та реалізація бази даних для ефективної підтримки процесів фіксації адміністративних правопорушень у сфері дорожнього руху.

3 ПОБУДОВА ЕР-МОДЕЛІ

3.1 Бізнес-правила

До основних бізнес-правил належить:

- 1) громадяни не можуть бути доросліше 120 років та не можуть бути народжені у майбутньому;
- 2) номери водійського посвідчення та поліцейського жетону громадян мають складатися з трьох великих літер, що накреслення яких збігається в латинській і в українській абетках, та з шести цифр;
- 3) номери водійського посвідчення та поліцейського жетону громадян мають бути унікальними;
- 4) водійське посвідчення можуть отримати лише громадяни старше 16 років;
- 5) поліцейськими можуть стати лише громадяни доросліше 18 років;
- 6) у однієї машини може бути тільки один поточний власник;
- 7) номерні знаки автомобілів можуть складатися з 2 лівих літер, які відповідають регіону реєстрації, 4 цифри та 2 правих літер, що відповідають серії номерного знаку, де всі де літери такі, що накреслення яких збігається в латинській і в українській абетках, або від 3 до 8 літер;
- 8) Vehicle Identification Number (VIN) має бути унікальним та довжиною у 17 символів;
- 9) автомобіль може мати страховку, номер якої має бути унікальним;
- 10) типи двигунів у транспортних засобів можуть бути наступними: дизельний, бензиновий, електричний, гіbridний;
- 11) у автомобілів з електричним двигуном не може бути об'єму;
- 12) об'єм двигуна не електричних транспортних засобів не може бути менше 1л. і більше 20л;
- 13) рік вироблення автомобіля має бути після 1886 року (дата вироблення першого серійного автомобіля) та не може бути у майбутньому;
- 14) кількість сидячих місць має відповідати типу автомобіля;

- 15) пункт та частина правил дорожнього руху мають бути більше рівними 1;
- 16) комбінація пункту та частини правил дорожнього руху мають бути унікальними;
- 17) стаття, пункт та частина КУпАП мають бути більше рівними 1;
- 18) якщо за статтю КУпАП передбачається штраф, то він має бути більше 0 грн.;
- 19) комбінація статті, пункту та частини КУпАП мають бути унікальними;
- 20) зафіксоване порушення обов'язково має вказувати час, місце, автомобіль, порушене правило дорожнього руху та статтю КУпАП;
- 21) час порушення не може бути раніше ніж за 10 років від поточної дати, оскільки система буде зберігати тільки порушення за останні 10 років;
- 22) одне порушення може мати декілька фото та відео доказів;
- 23) протокол ДТП обов'язково має складатися із серії та номеру документу протоколу, часу складання, порушення, поліцейського та обвинувачуваного;
- 24) комбінація серії та номеру протоколу має бути унікальною;
- 25) час складання протоколу не може бути раніше за час скочення відповідного порушення;
- 26) поліцейський не може скласти протокол сам на себе;
- 27) пов'язані з протоколом громадяни у вигляді свідків та жертв не можуть бути одночасно обвинувачуваними та/або поліцейськими у протоколі;
- 28) постанова за порушення обов'язково має вказувати на серію та номер документу постанови, час розгляду та час впровадження покарання в дію, порушення за яким складається постанова, поліцейського, який розглядає порушення та місце складання постанови;
- 29) за одне порушення може скластися тільки або протокол, або постанова.

3.2 Виділені сутності

Проаналізувавши предметне середовище та відповідно до постановленого завдання, було виділено такі сутності:

- citizen;
- driver;
- police_officer;
- vehicle_type;
- vehicle;
- traffic_rule;
- administrative_offense;
- location;
- violation;
- evidence;
- accident_protocol;
- citizen_on_protocol;
- accident_resolution;
- region.

3.3 Опис сутностей

У таблиці 3.1 наведено опис сутностей

Таблиця 3.1 – Опис сутностей

Назва сутності	Опис сутності	Атрибути	Опис атрибутів
citizen	репрезентує громадянина та його персональну інформацію	id	унікальний ідентифікатор сутності
		first_name	ім'я громадянина
		last_name	прізвище громадянина

Продовження таблиці 3.1

Назва сутності	Опис сутності	Атрибути	Опис атрибутів
		patronymic	по батькові громадянина
		date_of_birth	дата народження громадянина
driver	репрезентує громадянина як водія	id	унікальний ідентифікатор сутності
		citizen_id	ідентифікатор громадянина
		license_number	номер водійського посвідчення
		license_issue_time	дата та час видання водійського посвідчення
police_officer	репрезентує громадянина як поліцейського	id	унікальний ідентифікатор сутності
		citizen_id	ідентифікатор громадянина
		badge_number	номер поліцейського жетону
		rank	звання поліцейського

Продовження таблиці 3.1

Назва сутності	Опис сутності	Атрибути	Опис атрибутів
vehicle_type	репрезентує типи транспортних засобів	id	унікальний ідентифікатор сутності
		name	назва транспортного засобу
		description	опис транспортного засобу
		min_seating_capacity	мінімальна кількість сидячих місць
		max_seating_capacity	максимальна кількість сидячих місць
		min_engine_capacity	мінімальний об'єм двигуна
		max_engine_capacity	максимальний об'єм двигуна

Продовження таблиці 3.1

Назва сутності	Опис сутності	Атрибути	Опис атрибутів
vehicle	репрезентує автомобілі громадян	id	унікальний ідентифікатор сутності
		owner_id	ідентифікатор громадянина власника
		vehicle_type_id	ідентифікатор типу транспортного засобу
		registration_number	реєстраційний номер (номерний знак) транспортного засобу
		vin	ідентифікаційний номер транспортного засобу
		insurance_policy_number	номер страховки транспортного засобу
		model	модель транспортного засобу
		brand	виробник транспортного засобу

Продовження таблиці 3.1

Назва сутності	Опис сутності	Атрибути	Опис атрибутів
		color	колір транспортного засобу
		engine_capacity	об'єм двигуна транспортного засобу
		seating_capacity	кількість сидячих місць транспортного засобу
		year_of_manufacture	дата виробництва транспортного засобу
traffic_rule	репрезентує правило дорожнього руху	id	унікальний ідентифікатор сутності
		article	пункт правил дорожнього руху
		part	частина пункту правил дорожнього руху
		description	опис правила дорожнього руху

Продовження таблиці 3.1

Назва сутності	Опис сутності	Атрибути	Опис атрибутів
administrative_offense	репрезентує статтю КУпАП	id	унікальний ідентифікатор сутності
		article	стаття КУпАП
		sup	пункт КУпАП
		part	частина КУпАП
		description	опис статті, пункту та частини КУпАП
		penalty_fee	штраф за порушення
location	репрезентує місце	id	унікальний ідентифікатор сутності
		longitude	координата місця (довжина)
		latitude	координата місця (ширина)
		street	вулиця
		building_number	номер будівлі
		description	опис місця

Продовження таблиці 3.1

Назва сутності	Опис сутності	Атрибути	Опис атрибутів
violation	репрезентує порушення правил дорожнього руху	id	унікальний ідентифікатор сутності
		time_of_violation	дата та час скочення порушення
		description	опис суті порушення
		vehicle_id	ідентифікатор транспортного засобу, причасного до порушення
		location_id	ідентифікатор місця скочення порушення
		administrative_offense_id	ідентифікатор статті КУпАП, яке відповідає порушенню
		traffic_rule_id	ідентифікатор порушеного правила дорожнього руху

Продовження таблиці 3.1

Назва сутності	Опис сутності	Атрибути	Опис атрибутів
evidence	репрезентує доказ порушення	id	унікальний ідентифікатор сутності
		violation_id	ідентифікатор порушення
		type	тип доказів (фото, відео)
		url	посилання на доказ
accident_protocol	репрезентує складений протокол до порушення	id	унікальний ідентифікатор сутності
		series	серія документу протоколу
		number	номер документу протоколу
		defendant_explanation	пояснення обвинувачуваного
		time_of_drawing_up	час та дата складання протоколу
		violation_id	ідентифікатор порушення, за яким складався протокол
		police_officer_id	ідентифікатор поліцейського, який складав протокол

Продовження таблиці 3.1

Назва сутності	Опис сутності	Атрибути	Опис атрибутів
		defendant_id	ідентифікатор громадянина, якого обвинувачують
citizen_on_protocol	репрезентує громадянина, який пов'язаний із протоколом	id	унікальний ідентифікатор сутності
		role	роль громадянина у протоколі (свідок, жертва)
		citizen_id	ідентифікатор громадянина
		protocol_id	ідентифікатор протоколу
		testimony	свідчення громадянина
accident_resolution	репрезентує постанову	id	унікальний ідентифікатор сутності
		series	серія документу протоколу
		number	номер документу протоколу
		time_of_consideration	час та дата розгляду порушення та складання протоколу

Продовження таблиці 3.1

Назва сутності	Опис сутності	Атрибути	Опис атрибутів
		time_of_entry_into_force	час та дата надбання постанови законної сили
		violation_id	ідентифікатор порушення, за яким складалася постанова
		police_officer_id	ідентифікатор поліцейського, який складав постанову
		location_id	ідентифікатор місця складання постанови
region	репрезентує область України	id	унікальний ідентифікатор сутності
		region_name	назва області
		code_200_4	код регіону реєстрації ТЗ у 2004
		code_201_3	код регіону реєстрації ТЗ у 2013
		code_202_1	код регіону реєстрації ТЗ у 2021

3.4 Опис зв'язків між сутностями

Для подальшої реалізації реляційної бази даних, варто визначити зв'язки між сущностями системи. Зв'язки мають повністю відповідати визначенням бізнес-правилам.

Зв'язки між сущностями:

- citizen – driver: один до одного (1:1) – громадянин може бути водієм, і кожен водій обов'язково має відповідного громадянина;
- citizen – police_officer: один до одного (1:1) – кожен громадянин може бути поліцейським, і кожен поліцейський має відповідного громадянина;
- citizen – vehicle: один до багатьох (1:n) – один громадянин може володіти кількома транспортними засобами, але кожен транспортний засіб має лише одного власника;
- vehicle_type – vehicle: один до багатьох (1:n) – один тип транспортного засобу може бути пов'язаний з багатьма транспортними засобами, але кожен транспортний засіб має лише один тип;
- traffic_rule – violation: один до багатьох (1:n) – кожне порушення правил дорожнього руху пов'язане з одним правилом, але одне правило може бути порушене багаторазово;
- administrative_offense – violation: один до багатьох (1:n) – кожне адміністративне порушення може бути застосоване до кількох порушень, але одне порушення пов'язане лише з одним адміністративним правопорушенням;
- location – violation: один до багатьох (1:n) – одне місце розташування може бути пов'язане з багатьма порушеннями, але кожне порушення відбувається в одному місці;
- violation – evidence: один до багатьох (1:n) – кожне порушення може мати кілька доказів, але кожен доказ належить до одного порушення;
- violation – accident_protocol: один до одного (1:1) – кожне порушення може бути задокументоване лише одним протоколом, і кожен протокол стосується одного порушення;

- *police_officer – accident_protocol*: один до багатьох (1:n) – один поліцейський може складати кілька протоколів, але кожен протокол оформляє лише один поліцейський;
- *citizen – accident_protocol*: один до одного (1:1) – кожен протокол стосується одного порушника, який є громадянином;
- *accident_protocol – citizen_on_protocol*: один до багатьох (1:n) – один протокол може включати кілька громадян у різних ролях (наприклад, свідків чи учасників), але кожен запис ролі стосується одного протоколу;
- *violation – accident_resolution*: один до одного (1:1) – кожне порушення може мати лише одну постанову, яке його стосується;
- *police_officer – accident_resolution*: один до багатьох (1:n) – один поліцейський може складати кілька постанов, але кожну постанову ухвалює лише один поліцейський;
- *location – accident_resolution*: один до багатьох (1:n) – одне місце розташування може бути пов’язане з кількома постановами, але кожна постанова стосується одного місця.

3.5 ER-модель

Побудована ER-модель зображена на рис. 3.1.



Рисунок 3.1 – ЕР-модель бази даних для підтримки системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху

3.6 Висновки

У цьому розділі було визначено основні бізнес-правила, які регулюють роботу системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху. На основі аналізу предметного середовища виділено основні сутності, що забезпечують структуру бази

даних. Визначено зв'язки між сутностями та наведено ER-модель, яка є основою для подальшого проєктування реляційної схеми бази даних.

4 РЕАЛЯЦІЙНА МОДЕЛЬ БАЗИ ДАНИХ

4.1 Вибір СУБД

Перехід від логічної моделі бази даних до її реалізації у конкретній системі управління базами даних (СУБД) є ключовим етапом проектування інформаційної системи. На цьому етапі визначаються інструменти для зберігання, управління, обробки та забезпечення цілісності даних. Для створення бази даних системи підтримки фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху було обрано СУБД PostgreSQL[6]. Це рішення ґрунтуються на її технічних можливостях, функціональних перевагах і відповідності вимогам предметної області.

PostgreSQL[6] забезпечує повну відповідність принципам реляційної моделі, що робить її ідеальним вибором для роботи з базами даних, які включають складні зв'язки між сутностями. Система дозволяє ефективно створювати та управляти таблицями, забезпечуючи строгий контроль над відносинами між ними через механізми зовнішніх ключів. Це особливо важливо для проекту, де велика кількість таблиць пов'язані складною структурою.

Однією з переваг PostgreSQL[6] є її розширені підтримка типів даних. Вона дозволяє працювати зі стандартними типами, такими як VARCHAR, INT, TEXT, а також із більш складними, включно з TIMESTAMPTZ і DECIMAL. Це сприяє точному моделюванню даних та відповідності специфічним вимогам. Наприклад, для сутності «locations» було використано типи DECIMAL для координат, що забезпечує високу точність географічних даних.

Важливою характеристикою PostgreSQL[6] є її потужний механізм перевірки даних. Система дозволяє створювати складні обмеження, включаючи перевірки унікальності, зовнішні класі, користувачські функції перевірки, та застосування тригерів. Завдяки цьому вдається реалізувати складні бізнес-правила, такі як контроль мінімального віку водія або перевірка валідності номерів посвідчень. Такі функції забезпечують високу цілісність даних і зменшують ризик помилок.

Ще однією перевагою є масштабованість і продуктивність PostgreSQL[6]. Система здатна обробляти значні обсяги даних та ефективно виконувати запити навіть за високого навантаження. Це робить її оптимальним вибором для системи, яка передбачає довготривале зберігання даних про адміністративні правопорушення, транспортні засоби, водіїв, а також їх подальшу обробку.

СУБД PostgreSQL є безкоштовною, з відкритим кодом, що усуває витрати на ліцензування та забезпечує прозорість роботи. Це особливо важливо для проекту, орієнтованого на оптимізацію ресурсів. Крім того, велика спільнота користувачів і розробників PostgreSQL[6] забезпечує доступ до численних ресурсів для навчання та вирішення технічних питань.

Таким чином, вибір PostgreSQL[6] для реалізації бази даних зумовлений її функціональними перевагами, відповідністю технічним вимогам і здатністю забезпечувати надійну та ефективну роботу системи.

4.2 Структура таблиць

Після детального аналізу предметного середовища були побудовані необхідні відношення та визначені первинні та зовнішні ключі. Результати зображені в таблицях 4.1 – 4.15.

В таблиці 4.1 наведено опис таблиці «citizens», яка зберігає інформацію громадян.

Таблиця 4.1 – структура таблиці «citizens»

Ім’я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор громадянина у базі даних
first_name	varchar	50	-	ім’я громадянина
last_name	varchar	50	-	прізвище громадянина
patronymic	varchar	50	-	по батькові громадянина
date_of_birth	date	-	-	дата народження громадянина

В таблиці 4.2 наведено опис таблиці «drivers», яка зберігає інформацію про громадян, які є водіями.

Таблиця 4.2 – структура таблиці «drivers»

Ім’я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор водія у базі даних
citizen_id	int	-	FK	ідентифікатор громадянина
license_number	char	9	-	номер посвідчення водія
license_issued_time	timestamptz	-	-	час та дата видачі посвідчення водія

В таблиці 4.3 наведено опис таблиці «police_officers», яка зберігає інформацію про громадян, які є поліцейськими.

Таблиця 4.3 – структура таблиці «police_officers»

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор поліцейського у базі даних
citizen_id	int	-	FK	ідентифікатор громадянина
badge_number	char	9	-	номер поліцейського жетону
rank	ENUM('junior_sergeant','sergeant','senior_sergeant','junior_lieutenant','lieutenant','senior_lieutenant','captain','major','lieutenant_colonel','colonel');	-	-	звання поліцейського

В таблиці 4.4 наведено опис таблиці «vehicle_types», яка зберігає інформацію про типи транспортних засобів та їх характеристики.

Таблиця 4.4 – структура таблиці «vehicle_types»

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор типу транспортного засобу у базі даних
name	varchar	50	-	назва типу транспортного засобу
description	text	-	-	опис типу транспортного засобу
min_seating_capacity	integer	-	-	мінімальна кількість сидячих місць типу транспортного засобу
max_seating_capacity	integer	-	-	максимальна кількість сидячих місць типу транспортного засобу
min_engine_capacity	decimal	4, 2	-	мінімальний об'єм двигуну типу транспортного засобу
max_engine_capacity	decimal	4, 2	-	максимальна об'єм двигуну типу транспортного засобу

В таблиці 4.5 наведено опис таблиці «vehicles», яка зберігає інформацію про транспортні засоби громадян.

Таблиця 4.5 – структура таблиці «vehicles»

Ім’я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор типу транспортного засобу у базі даних
owner_id	integer	-	FK	ідентифікатор громадянина власника
vehicle_type_id	integer	-	FK	ідентифікатор типу транспортного засобу
vin	char	17	-	ідентифікаційний номер транспортного засобу
insurance_policy_number	char	9	-	номер страховки транспортного засобу
model	varchar	50	-	модель транспортного засобу
brand	varchar	50	-	виробник транспортного засобу
color	varchar	50	-	колір транспортного засобу
engine_capacity	decimal	4, 2	-	об’єм двигуна транспортного засобу
seating_capacity	decimal	4, 2	-	кількість сидячих місць транспортного засобу
year_of_manufacture	integer	-	--	дата виробництва транспортного засобу

В таблиці 4.6 наведено опис таблиці «traffic_rules», яка зберігає інформацію про правила дорожнього руху.

Таблиця 4.6 – структура таблиці «traffic_rules»

Ім’я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор правила у базі даних
article	integer	-	-	пункт правил дорожнього руху
part	integer	-	-	частина пункту правил дорожнього руху
description	text	-	-	опис правила дорожнього руху

В таблиці 4.7 наведено опис таблиці «administrative_offenses», яка зберігає інформацію про правила статті КУпАП.

Таблиця 4.7 – структура таблиці «administrative_offenses»

Ім’я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор статті КУпАП у базі даних
article	integer	-	-	стаття КУпАП
sup	integer	-	-	пункт КУпАП
part	integer	-	-	частина КУпАП
description	text	-	-	опис статті, пункту та частини КУпАП
penalty_fee	decimal	10, 2	-	штраф за порушення

В таблиці 4.8 наведено опис таблиці «locations», яка зберігає інформацію про місця.

Таблиця 4.8 – структура таблиці «locations»

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор місця у базі даних
longitude	decimal	9, 6	-	координата місця (довжина)
latitude	decimal	9, 6	-	координата місця (ширина)
street	varchar	50	-	вулиця
building_number	varchar	10	-	номер будівлі
description	text	-	-	опис місця

В таблиці 4.9 наведено опис таблиці «violations», яка зберігає інформацію про порушення.

Таблиця 4.9 – структура таблиці «violations»

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор порушення у базі даних
time_of_violation	timestamptz	-	-	дата та час скоєння порушення
description	text	-	-	опис суті порушення
vehicle_id	integer	-	FK	ідентифікатор транспортного засобу, причасного до порушення
location_id	integer	-	FK	ідентифікатор місця скоєння порушення
administrative_offense_id	integer	-	FK	ідентифікатор статті КУпАП, яке відповідає порушенню
traffic_rule_id	integer	-	FK	ідентифікатор порушеного правила дорожнього руху

В таблиці 4.10 наведено опис таблиці «evidences», яка зберігає інформацію про докази порушень.

Таблиця 4.10 – структура таблиці «evidences»

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор доказу порушення у базі даних
violation_id	integer	-	FK	ідентифікатор порушення
type	ENUM ('photo', 'video')	-	-	тип доказів (фото, відео)
url	text	-	-	посилання на доказ

В таблиці 4.11 наведено опис таблиці «accident_protocols», яка зберігає інформацію про протоколи.

Таблиця 4.11 – структура таблиці «accident_protocols»

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор протоколу у базі даних
series	char	2	-	серія документу протоколу
number	char	6	-	номер документу протоколу
defendant_explanation	text	-	-	пояснення обвинувачуваного
time_of_drawing_up	timestamptz	-	-	час та дата складання протоколу
violation_id	integer	-	FK	ідентифікатор порушення, за яким складався протокол
police_officer_id	integer	-	FK	ідентифікатор поліцейського, який складав протокол
defendant_id	integer	-	FK	ідентифікатор громадянина, якого обвинувачують

В таблиці 4.12 наведено опис таблиці «citizens_on_protocol», яка зберігає інформацію про громадян, причетних до протоколу.

Таблиця 4.12 – структура таблиці «citizens_on_protocol»

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор запису у базі даних
role	ENUM ('victim', 'witness')	-	-	роль громадянина у протоколі (свідок, жертва)
citizen_id	integer	-	C	ідентифікатор громадянина
protocol_id	integer	-	integer	ідентифікатор протоколу
testimony	text	-	-	свідчення громадянина

В таблиці 4.13 наведено опис таблиці «accident_resolutions», яка зберігає інформацію про постанови.

Таблиця 4.13 – структура таблиці «accident_resolutions»

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор протоколу у базі даних
series	char	2	-	серія документу постанови
number	char	6	-	номер документу постанови
time_of_consideration	timestamptz	-	-	час та дата складання постанови
time_of_entry_into_force	timestamptz	-	-	час та дата надбання постанови законної сили
violation_id	integer	-	FK	ідентифікатор порушення, за яким складався протокол
police_officer_id	integer	-	FK	ідентифікатор поліцейського, який складав протокол
location_id	integer	-	FK	ідентифікатор місця складання постанови

В таблиці 4.14 наведено опис таблиці «regions», яка зберігає інформацію про постанови.

Таблиця 4.14 – структура таблиці «regions»

Ім’я поля	Тип даних	Розмір	Ключ	Опис
id	integer	-	PK	унікальний ідентифікатор протоколу у базі даних
region_name	varchar	100	-	назва області
code_2004	char	2	-	код регіону реєстрації ТЗ у 2004
code_2013	char	2	-	код регіону реєстрації ТЗ у 2013
code_2021	char	2	-	код регіону реєстрації ТЗ у 2021

На основі табличних описів було розроблено реляційну схему (рис. 4.1), яка відповідає вимогам третьої нормальної форми. Цього вдалося досягти шляхом декомпозиції таблиць, усунення надлишкових даних, забезпечення транзитивної незалежності атрибутів від первинного ключа, а також повної функціональної залежності всіх атрибутів від первинного ключа.

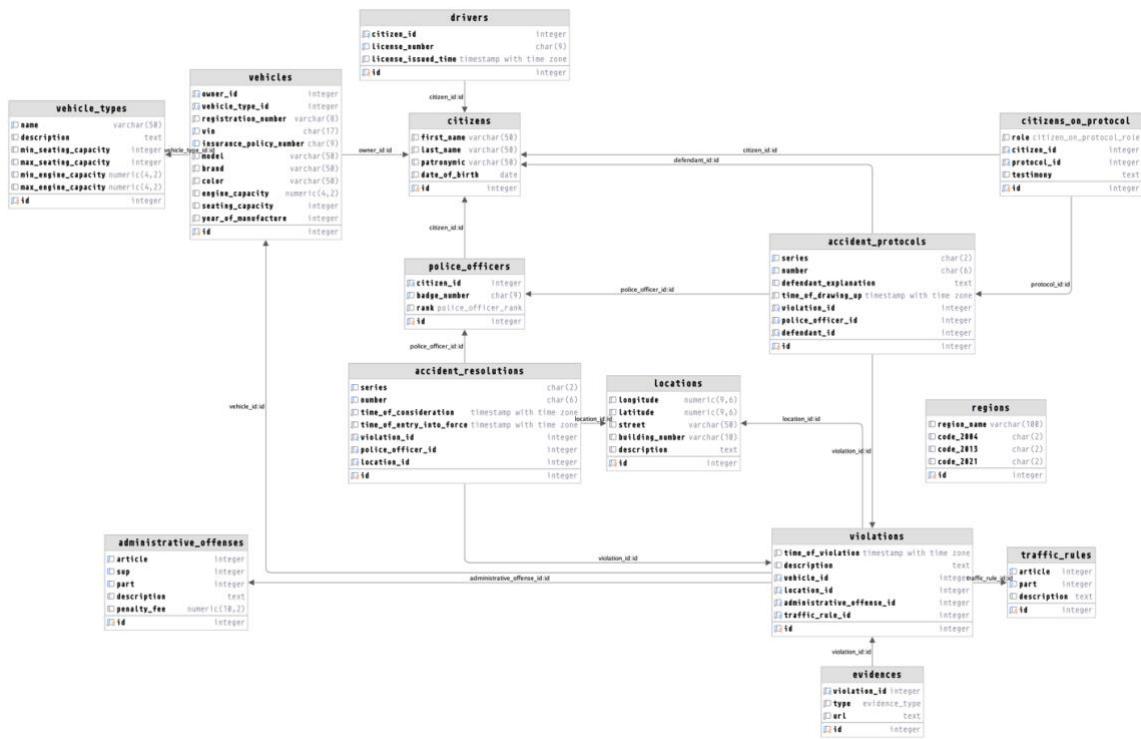


Рисунок 4.1 – Реляційна схема бази даних реалізована засобами PostgreSQL, згенерована використовуючи програмне забезпечення DataGrip[7]

4.4 Висновки

У розділі було здійснено перехід від логічної моделі бази даних до її реалізації в обраній системі управління базами даних PostgreSQL. Розроблено реляційну схему, яка відповідає вимогам третьої нормальної форми, що забезпечує мінімізацію надлишковості даних, підтримку цілісності та логічної структури.

Обґрунтовано вибір СУБД PostgreSQL як платформи для реалізації бази даних, враховуючи її функціональні можливості, продуктивність, масштабованість і підтримку складних запитів. Визначено основні структурні елементи схеми, що відповідають сутностям предметного середовища, та реалізовано всі необхідні зв'язки між ними.

Ця реалізація створює надійну основу для подальшого використання бази даних у практичних задачах, забезпечуючи її ефективну роботу, багатокористувальницький доступ і зручність експлуатації.

5 РЕАЛІЗАЦІЯ БАЗИ ДАНИХ

5.1 Створення бази даних

Проаналізувавши предметне середовище, виділивши сутності системи та структуру таблиць, маємо сформований SQL скрипт для створення бази даних за допомогою PostgreSQL:

```
CREATE SCHEMA IF NOT EXISTS public;
```

```
CREATE TYPE EVIDENCE_TYPE AS ENUM ('photo', 'video');
```

```
CREATE TYPE CITIZEN_ON_PROTOCOL_ROLE AS ENUM ('victim',
'witness');
```

```
CREATE TYPE POLICE_OFFICER_RANK AS ENUM (
    'junior_sergeant',
    'sergeant',
    'senior_sergeant',
    'junior_lieutenant',
    'lieutenant',
    'senior_lieutenant',
    'captain',
    'major',
    'lieutenant_colonel',
    'colonel'
);
```

```
CREATE TABLE citizens (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    patronymic VARCHAR(50),
```

```

date_of_birth DATE NOT NULL,
CONSTRAINT check_date_of_birth CHECK (
    date_of_birth > CURRENT_DATE - INTERVAL '120 years'
    AND date_of_birth < CURRENT_DATE
)
);

```

```

CREATE TABLE drivers (
    id SERIAL PRIMARY KEY,
    citizen_id INT NOT NULL,
    license_number CHAR(9) NOT NULL,
    license_issued_time TIMESTAMPTZ NOT NULL,
    CONSTRAINT fk_citizen_id FOREIGN KEY (citizen_id)
    REFERENCES citizens (id) ON DELETE CASCADE,
    CONSTRAINT check_license_number CHECK (
        license_number ~ '^[A-Z]{3}[0-9]{6}$'
    ),
    CONSTRAINT check_license_issued_time CHECK (
        license_issued_time < CURRENT_TIMESTAMP
    ),
    CONSTRAINT uq_license_number UNIQUE (license_number),
    CONSTRAINT ck_is_driver_more_than_16_years_old CHECK
    (is_citizen_older_than(citizen_id, 16))
);

```

```

CREATE TABLE police_officers (
    id SERIAL PRIMARY KEY,
    citizen_id INT NOT NULL,
    badge_number CHAR(9) NOT NULL,
    rank POLICE_OFFICER_RANK NOT NULL,

```

```

CONSTRAINT fk_citizen_id FOREIGN KEY (citizen_id)
REFERENCES citizens (id) ON DELETE CASCADE,
CONSTRAINT check_badge_number CHECK (
    badge_number ~ '^[A-Z]{3}[0-9]{6}$'
),
CONSTRAINT uq_badge_number UNIQUE (badge_number),
CONSTRAINT ck_is_police_officer_more_than_18_years_old CHECK
(is_citizen_older_than(citizen_id, 18))
);

```

```

CREATE TABLE vehicle_types (
    id SERIAL PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    description TEXT,
    min_seating_capacity INT,
    max_seating_capacity INT,
    min_engine_capacity DECIMAL(4, 2),
    max_engine_capacity DECIMAL(4, 2),
    CONSTRAINT uq_name UNIQUE (name),
    CONSTRAINT ck_seating_capacity CHECK (
        (
            min_seating_capacity IS NULL
            AND max_seating_capacity IS NULL
        )
        OR (
            min_seating_capacity >= 0
            AND max_seating_capacity >= min_seating_capacity
        )
    ),
    CONSTRAINT ck_engine_capacity CHECK (

```

```

(
    min_engine_capacity IS NULL
    AND max_engine_capacity IS NULL
)
OR (
    min_engine_capacity >= 0
    AND max_engine_capacity >= min_engine_capacity
)
)
);

```

```

CREATE TABLE vehicles (
    id SERIAL PRIMARY KEY,
    owner_id INT NOT NULL,
    vehicle_type_id INT NOT NULL,
    registration_number VARCHAR(8) NOT NULL,
    vin CHAR(17) NOT NULL,
    insurance_policy_number CHAR(9),
    model VARCHAR(50) NOT NULL,
    brand VARCHAR(50) NOT NULL,
    color VARCHAR(50) NOT NULL,
    engine_capacity DECIMAL(4, 2),
    seating_capacity INT,
    year_of_manufacture INT NOT NULL,
    CONSTRAINT fk_owner_id FOREIGN KEY (owner_id) REFERENCES
    citizens (id) ON DELETE CASCADE,
    CONSTRAINT fk_vehicle_type_id FOREIGN KEY (vehicle_type_id)
    REFERENCES vehicle_types (id) ON DELETE CASCADE,
    CONSTRAINT ck_valid_vin CHECK (vin ~ '^[A-HJ-NPR-Z0-9]{17}$'),
    CONSTRAINT ck_year_of_manufacture CHECK (

```

```

year_of_manufacture >= 1886
AND year_of_manufacture <= EXTRACT(
    YEAR
    FROM
        CURRENT_DATE
)
),
CONSTRAINT uq_vin UNIQUE (vin),
CONSTRAINT uq_insurance_policy_number UNIQUE NULLS NOT
DISTINCT (insurance_policy_number)
);

```

```

CREATE TABLE traffic_rules (
    id SERIAL PRIMARY KEY,
    article INT NOT NULL,
    part INT NOT NULL,
    description TEXT,
    CONSTRAINT uq_article_part UNIQUE (article, part)
);

```

```

CREATE TABLE administrative_offenses (
    id SERIAL PRIMARY KEY,
    article INT NOT NULL,
    sup INT,
    part INT,
    description TEXT,
    penalty_fee DECIMAL(10, 2),
    CONSTRAINT uq_article_sup_part UNIQUE (article, sup, part),
    CONSTRAINT ck_article CHECK (article >= 1),
    CONSTRAINT ck_sup CHECK (sup >= 1),

```

```

CONSTRAINT ck_part CHECK (part >= 1),
CONSTRAINT ck_penalty_fee CHECK (penalty_fee >= 0)
);

```

```

CREATE TABLE locations (
    id SERIAL PRIMARY KEY,
    longitude DECIMAL(9, 6) NOT NULL,
    latitude DECIMAL(9, 6) NOT NULL,
    street VARCHAR(50),
    building_number VARCHAR(10),
    description TEXT
);

```

```

CREATE TABLE violations (
    id SERIAL PRIMARY KEY,
    time_ofViolation TIMESTAMPTZ NOT NULL,
    description TEXT,
    vehicle_id INT NOT NULL,
    location_id INT NOT NULL,
    administrative_offense_id INT NOT NULL,
    traffic_rule_id INT NOT NULL,
    CONSTRAINT fk_vehicle_id FOREIGN KEY (vehicle_id)
        REFERENCES vehicles (id) ON DELETE CASCADE,
    CONSTRAINT fk_location_id FOREIGN KEY (location_id)
        REFERENCES locations (id) ON DELETE CASCADE,
    CONSTRAINT fk_administrative_offense_id FOREIGN KEY
        (administrative_offense_id) REFERENCES administrative_offenses (id) ON
        DELETE CASCADE,
    CONSTRAINT fk_traffic_rule_id FOREIGN KEY (traffic_rule_id)
        REFERENCES traffic_rules (id) ON DELETE CASCADE,

```

```

CONSTRAINT ck_time_ofViolation CHECK (
    time_ofViolation <= CURRENT_TIMESTAMP
    AND time_ofViolation > CURRENT_DATE - INTERVAL '10 years'
)
);

```

```

CREATE TABLE evidences (
    id SERIAL PRIMARY KEY,
    violation_id INT NOT NULL,
    type EVIDENCE_TYPE NOT NULL,
    url TEXT NOT NULL,
    CONSTRAINT fk_violation_id FOREIGN KEY (violation_id)
    REFERENCES violations (id) ON DELETE CASCADE
);

```

```

CREATE TABLE accident_protocols (
    id SERIAL PRIMARY KEY,
    series CHAR(2) NOT NULL,
    number CHAR(6) NOT NULL,
    defendant_explanation TEXT,
    time_of_drawing_up TIMESTAMPTZ NOT NULL,
    violation_id INT NOT NULL,
    police_officer_id INT NOT NULL,
    defendant_id INT NOT NULL,
    CONSTRAINT fk_violation_id FOREIGN KEY (violation_id)
    REFERENCES violations (id) ON DELETE CASCADE,
    CONSTRAINT fk_police_officer_id FOREIGN KEY (police_officer_id)
    REFERENCES police_officers (id) ON DELETE CASCADE,
    CONSTRAINT fk_defendant_id FOREIGN KEY (defendant_id)
    REFERENCES citizens (id) ON DELETE CASCADE,

```

```

CONSTRAINT uq_series_number_protocol UNIQUE (series, number)
);

```

```

CREATE TABLE citizens_on_protocol (
    id SERIAL PRIMARY KEY,
    role CITIZEN_ON_PROTOCOL_ROLE NOT NULL,
    citizen_id INT NOT NULL,
    protocol_id INT NOT NULL,
    testimony TEXT,
    CONSTRAINT fk_citizen_id FOREIGN KEY (citizen_id)
    REFERENCES citizens (id) ON DELETE CASCADE,
    CONSTRAINT fk_protocol_id FOREIGN KEY (protocol_id)
    REFERENCES accident_protocols (id) ON DELETE CASCADE
);

```

```

CREATE TABLE accident_resolutions (
    id SERIAL PRIMARY KEY,
    series CHAR(2) NOT NULL,
    number CHAR(6) NOT NULL,
    time_of_consideration TIMESTAMPTZ NOT NULL,
    time_of_entry_into_force TIMESTAMPTZ NOT NULL,
    violation_id INT NOT NULL,
    police_officer_id INT NOT NULL,
    location_id INT NOT NULL,
    CONSTRAINT fk_violation_id FOREIGN KEY (violation_id)
    REFERENCES violations (id) ON DELETE CASCADE,
    CONSTRAINT fk_police_officer_id FOREIGN KEY (police_officer_id)
    REFERENCES police_officers (id) ON DELETE CASCADE,
    CONSTRAINT fk_location_id FOREIGN KEY (location_id)
    REFERENCES locations (id) ON DELETE CASCADE,

```

```
CONSTRAINT uq_series_number_resolution UNIQUE (series, number)
);
```

```
CREATE TABLE regions (
    id SERIAL PRIMARY KEY,
    region_name VARCHAR(100),
    code_2004 CHAR(2),
    code_2013 CHAR(2),
    code_2021 CHAR(2)
);
```

5.2 Імпортування даних в таблиці

Для заповнення бази даних, було використано комбінацію команд «**INSERT INTO**» та «**COPY**»:

```
COPY traffic_rules (article, part, description)
```

```
FROM
```

```
'/private/tmp/traffic_rules.csv' DELIMITER ',' CSV HEADER;
```

```
COPY administrative_offenses (article, sup, part, description, penalty_fee)
```

```
FROM
```

```
'/private/tmp/administrative_offenses.csv' DELIMITER ',' CSV
```

```
HEADER;
```

```
COPY citizens (
```

```
    id,  
    first_name,  
    last_name,  
    patronymic,  
    date_of_birth
```

```
)
```

```
FROM
```

```
'/private/tmp/citizens.csv' DELIMITER ',' CSV HEADER;
```

```
COPY drivers (  
    id,  
    citizen_id,  
    license_number,  
    license_issued_time  
)  
FROM  
'/private/tmp/drivers.csv' DELIMITER ',' CSV HEADER;
```

```
COPY police_officers (id, citizen_id, badge_number, rank)  
FROM  
'/private/tmp/police_officers.csv' DELIMITER ',' CSV HEADER;
```

```
COPY vehicles (  
    id,  
    owner_id,  
    vehicle_type_id,  
    registration_number,  
    vin,  
    insurance_policy_number,  
    model,  
    brand,  
    color,  
    engine_capacity,  
    seating_capacity,  
    year_of_manufacture  
)  
FROM
```

```
'/private/tmp/vehicles.csv' DELIMITER ',' CSV HEADER;
```

```
COPY locations (
```

```
    id,  
    longitude,  
    latitude,  
    street,  
    building_number,  
    description
```

```
)
```

```
FROM
```

```
'/private/tmp/locations.csv' DELIMITER ',' CSV HEADER;
```

```
COPY violations (
```

```
    id,  
    time_ofViolation,  
    description,  
    vehicle_id,  
    location_id,  
    administrative_offense_id,  
    traffic_rule_id
```

```
)
```

```
FROM
```

```
'/private/tmp/violations.csv' DELIMITER ',' CSV HEADER;
```

```
COPY evidences (id, violation_id, type, url)
```

```
FROM
```

```
'/private/tmp/evidences.csv' DELIMITER ',' CSV HEADER;
```

```
COPY accident_protocols (
```

```
    id,  
    series,  
    number,  
    defendant_explanation,  
    time_of_drawing_up,  
    violation_id,  
    police_officer_id,  
    defendant_id  
)  
FROM  
'/private/tmp/accident_protocols.csv' DELIMITER ',' CSV HEADER;  
  
COPY citizens_on_protocol (id, role, citizen_id, protocol_id, testimony)  
FROM  
'/private/tmp/citizens_on_protocol.csv' DELIMITER ',' CSV HEADER;  
  
COPY accident_resolutions (  
    id,  
    series,  
    number,  
    time_of_consideration,  
    time_of_entry_into_force,  
    violation_id,  
    police_officer_id,  
    location_id  
)  
FROM  
'/private/tmp/accident_resolutions.csv' DELIMITER ',' CSV HEADER;  
  
INSERT INTO
```

```
vehicle_types (
    NAME,
    DESCRIPTION,
    MIN_SEATING_CAPACITY,
    MAX_SEATING_CAPACITY,
    MIN_ENGINE_CAPACITY,
    MAX_ENGINE_CAPACITY
)
VALUES
(
    'Car',
    'Passenger car used for personal or commercial purposes.',
    2,
    8,
    1.0,
    6.0
),
(
    'Electric Car',
    'Vehicle powered exclusively by electricity.',
    2,
    8,
    NULL,
    NULL
),
(
    'Hybrid Vehicle',
    'Vehicle powered by a combination of internal combustion engine
and electric motor.',
    2,
```

8,
1.0,
6.0
(
'Motorcycle',
'Two-wheeled motor vehicle.',
1,
2,
0.1,
2.0
(
'Van',
'Larger motor vehicle designed for passenger or cargo transport.',
2,
15,
2.0,
6.0
(
'Truck',
'Motor vehicle designed to transport goods or materials.',
2,
2,
2.0,
15.0
(
'Bus',

'Motor vehicle designed to carry multiple passengers.',
9,
NULL,
4.0,
15.0
(
'Special Purpose Vehicle',
'Vehicles such as fire trucks, ambulances, or police cars.',
1,
10,
2.0,
10.0
(
'Trailer',
'Unpowered vehicle towed by another vehicle.',
NULL,
NULL,
NULL,
NULL
(
'Agricultural Vehicle',
'Vehicles such as tractors used for farming purposes.',
1,
3,
1.0,
5.0
(

```

(
    'ATV',
    'All-terrain vehicle used for off-road travel.',
    1,
    2,
    0.5,
    1.5
),
(
    'Dump Truck',
    'Truck designed to transport and unload materials.',
    2,
    2,
    2.0,
    15.0
);

```

INSERT INTO

regions (region_name, code_2004, code_2013, code_2021)

VALUES

```

('AR Krym', 'AK', 'KK', 'TK'),
('Vinnytska oblast', 'AB', 'KB', 'IM'),
('Volynska oblast', 'AC', 'KC', 'CM'),
('Dnipropetrovska oblast', 'AE', 'KE', 'PP'),
('Donetska oblast', 'AH', 'KH', 'TH'),
('Zhytomyrska oblast', 'AM', 'KM', 'TM'),
('Zakarpatska oblast', 'AO', 'KO', 'MT'),
('Zaporizka oblast', 'AP', 'KP', 'TP'),
('Ivano-Frankivska oblast', 'AT', 'KT', 'TO'),
('Kyivska oblast', 'AI', 'KI', 'TI'),

```

('misto Kyiv', 'AA', 'KA', 'TT'),
('Kirovohradska oblast', 'BA', 'HA', 'XA'),
('Luhanska oblast', 'BB', 'HB', 'EP'),
('Lvivska oblast', 'BC', 'HC', 'CC'),
('Mykolaivska oblast', 'BE', 'HE', 'XE'),
('Odeska oblast', 'BH', 'HH', 'OO'),
('Poltavska oblast', 'BI', 'HI', 'XI'),
('Rivnenska oblast', 'BK', 'HK', 'XK'),
('Sumska oblast', 'BM', 'HM', 'XM'),
('Ternopilska oblast', 'BO', 'HO', 'XO'),
('Kharkivska oblast', 'AX', 'KX', 'XX'),
('Khersonska oblast', 'BT', 'HT', 'XT'),
('Khmelnitska oblast', 'BX', 'HX', 'OX'),
('Cherkaska oblast', 'CA', 'IA', 'OA'),
('Chernihivska oblast', 'CB', 'IB', 'OB'),
('Chernivetska oblast', 'CE', 'IE', 'OE'),
('misto Sevastopol', 'CH', 'IH', 'OH');

6 СТВОРЕННЯ КОРИСТУВАЧІВ БАЗИ ДАНИХ

6.1 Адміністратор

```
CREATE ROLE cw_admin;
```

```
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO
cw_admin;
```

```
GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA public
TO cw_admin;
```

```
GRANT ALL PRIVILEGES ON ALL FUNCTIONS IN SCHEMA public TO
cw_admin;
```

```
CREATE USER cw_admin_user WITH PASSWORD 'admin_password';
```

```
GRANT cw_admin TO cw_admin_user;
```

6.2 Поліцейський

```
CREATE ROLE cw_police_officer;
```

```
GRANT
```

```
SELECT
```

```
,
```

```
INSERT
```

```
,
```

```
UPDATE
```

```
ON TABLE violations,
```

```
evidences,
```

```
accident_protocols,
```

```
accident_resolutions,
```

```
locations TO cw_police_officer;
```

```

GRANT
SELECT
    ON TABLE citizens,
    drivers,
    vehicles,
    vehicle_types,
    traffic_rules,
    administrative_offenses TO cw_police_officer;

```

```
CREATE USER cw_police_user WITH PASSWORD 'police_password';
```

```
GRANT cw_police_officer TO cw_police_user;
```

6.3 Водій

```
CREATE ROLE cw_driver;
```

```
GRANT
```

```
SELECT
```

```

    ON TABLE citizens,
    drivers,
    vehicles,
    violations,
    evidences,
    accident_resolutions TO cw_driver;

```

```
CREATE USER cw_driver_user WITH PASSWORD 'driver_password';
```

```
GRANT cw_driver TO cw_driver_user;
```

6.4 Громадянин

```
CREATE ROLE cw_citizen;
```

```
GRANT
```

```
SELECT
```

```
    ON TABLE citizens,
    traffic_rules,
    administrative_offenses TO cw_citizen;
```

```
CREATE USER cw_citizen_user WITH PASSWORD 'citizen_password';
```

```
GRANT cw_citizen TO cw_citizen_user;
```

6.5 Перевірка результатів

Скрипт для отримання користувачів та їх ролей у базі даних:

```
SELECT
```

```
    u.usename AS username,
```

```
    r.rolname AS role,
```

```
    u.usesysid AS user_id
```

```
FROM
```

```
    pg_catalog.pg_user u
```

```
JOIN
```

```
    pg_catalog.pg_roles r ON r.rolname = u.usename
```

```
WHERE
```

```
    u.usename LIKE 'cw%';
```

Результат виконання команд для створення ролей та користувачів командами SQL наведено на рис. 6.1.

	username	role	user_id
1	cw_admin_user	cw_admin_user	41717
2	cw_police_user	cw_police_user	41718
3	cw_driver_user	cw_driver_user	41719
4	cw_citizen_user	cw_citizen_user	41720

Рисунок. 6.1 – Створені користувачі та їх ролі в базі даних

6.6 Висновки

У результаті реалізації системи управління доступом для забезпечення багатокористувальницького режиму доступу було створено кілька ролей для різних категорій користувачів, зокрема для адміністратора, поліцейського, водія та громадянина. Дляожної ролі було визначено відповідні привілеї, що обмежують або дозволяють доступ до конкретних таблиць та функцій у базі даних. Адміністратор отримав повні привілеї на всі об'єкти, у той час як інші ролі мають більш обмежений доступ, що відповідає їх функціональним обов'язкам.

Процедура створення користувачів і надання їм відповідних ролей була реалізована через SQL-скрипти, що гарантують правильне розподілення доступу. У результаті кожен користувач отримує лише необхідні привілеї, що забезпечує належну безпеку та цілісність даних у системі.

Перевірка правильності налаштувань доступу здійснюється через виконання відповідних SQL-запитів, що дозволяє переконатися у наданні правильних прав користувачам. Це дозволяє переконатися в ефективності реалізації системи безпеки та доступу до даних.

7 РОБОТА З БАЗОЮ ДАНИХ

7.1 Тригери

Відповідно до встановлених бізнес-правил предметного середовища, було реалізовано наступні тригери.

7.1.1 Тригер trg_check_engine_capacity

```

CREATE OR REPLACE FUNCTION check_engine_capacity()
RETURNS TRIGGER AS
$$
DECLARE
    min_capacity DECIMAL(4, 2);
    max_capacity DECIMAL(4, 2);
BEGIN
    SELECT min_engine_capacity, max_engine_capacity
    INTO min_capacity, max_capacity
    FROM vehicle_types
    WHERE id = NEW.vehicle_type_id;

    IF min_capacity IS NULL AND max_capacity IS NULL AND
    NEW.engine_capacity IS NOT NULL THEN
        RAISE EXCEPTION 'Unpowered vehicles cannot have an engine
capacity value.';

    ELSIF min_capacity IS NOT NULL AND NEW.engine_capacity <
    min_capacity THEN
        RAISE EXCEPTION 'Engine capacity must be at least %.',

    min_capacity;

    ELSIF max_capacity IS NOT NULL AND NEW.engine_capacity >
    max_capacity THEN
        RAISE EXCEPTION 'Engine capacity must not exceed %.',

    max_capacity;

    END IF;

```

```

RETURN NEW;
END;
$$ LANGUAGE plpgsql;

```

```

CREATE TRIGGER trg_check_engine_capacity
    BEFORE INSERT OR UPDATE
    ON vehicles
    FOR EACH ROW
    EXECUTE FUNCTION check_engine_capacity();

```

На рис. 7.1.1.1 наведено результат дії тригера.

```

Services
> Database > @localhost > triggers > triggers.sql Services
Tx course_work.public> INSERT INTO vehicles
VALUES (15000, 1, 1, 'AA1234AA', '3FA6P0H79ER135093', '123456789', 'Model', 'Brand', 'Color', 0, 4, 2000)
[2024-12-22 15:06:10] [P0001] ERROR: Engine capacity must be at least 1.00.
[2024-12-22 15:06:10] Where: PL/pgSQL function check_engine_capacity() line 14 at RAISE
course_work.public> INSERT INTO vehicles
VALUES (15001, 1, 1, 'AA1234AA', '3FA6P0H79ER135093', '123456789', 'Model', 'Brand', 'Color', 25, 4, 2000)
[2024-12-22 15:06:18] [P0001] ERROR: Engine capacity must not exceed 6.00.
[2024-12-22 15:06:18] Where: PL/pgSQL function check_engine_capacity() line 16 at RAISE
course_work.public> INSERT INTO vehicles
VALUES (15002, 2, 2, 'AA1234AA', '3FA6P0H79ER135093', '123456789', 'Model', 'Brand', 'Color', 2.4, 4, 2000)
[2024-12-22 15:06:20] [P0001] ERROR: Unpowered vehicles cannot have an engine capacity value.
[2024-12-22 15:06:20] Where: PL/pgSQL function check_engine_capacity() line 12 at RAISE

```

Рисунок 7.1.1.1 – Результат дії тригера `trg_check_engine_capacity`

7.1.2 Тригер `trg_check_seating_capacity`

```
CREATE OR REPLACE FUNCTION check_seating_capacity()
```

```
RETURNS TRIGGER AS
```

```
$$
```

```
DECLARE
```

```
min_capacity INT;
```

```

max_capacity INT;

BEGIN

    SELECT min_seating_capacity, max_seating_capacity
    INTO min_capacity, max_capacity
    FROM vehicle_types
    WHERE id = NEW.vehicle_type_id;

    IF min_capacity IS NOT NULL AND NEW.seating_capacity <
    min_capacity THEN
        RAISE EXCEPTION 'Seating capacity must be at least %.',

min_capacity;
    ELSIF max_capacity IS NOT NULL AND NEW.seating_capacity >
    max_capacity THEN
        RAISE EXCEPTION 'Seating capacity must not exceed %.',

max_capacity;
    END IF;

    RETURN NEW;
END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_check_seating_capacity
    BEFORE INSERT OR UPDATE
    ON vehicles
    FOR EACH ROW
    EXECUTE FUNCTION check_seating_capacity();

```

На рис. 7.1.2.1 наведено результат дії тригера.

```

> Database > @localhost > triggers > triggers.sql Services
Tx course_work.public> INSERT INTO vehicles
VALUES (1, 1, 1, 'AA1234AA', '3FA6P0H79ER135093', '123456789', 'Model', 'Brand', 'Color', 2.4, 0, 2000)
[2024-12-22 15:12:34] [P0001] ERROR: Seating capacity must be at least 2.
[2024-12-22 15:12:34] Where: PL/pgSQL function check_seating_capacity() line 12 at RAISE
course_work.public> INSERT INTO vehicles
VALUES (1, 1, 1, 'AA1234AA', '3FA6P0H79ER135093', '123456789', 'Model', 'Brand', 'Color', 2.4, 20, 2000)
[2024-12-22 15:12:46] [P0001] ERROR: Seating capacity must not exceed 8.
[2024-12-22 15:12:46] Where: PL/pgSQL function check_seating_capacity() line 14 at RAISE

```

Рисунок 7.1.1.1 – Результат дії тригера trg_check_seating_capacity

7.1.3 Тригер check_protocol_validity

```
CREATE OR REPLACE FUNCTION check_protocol_validity()
```

```
RETURNS TRIGGER AS
```

```
$$
```

```
BEGIN
```

```
    IF NEW.time_of_drawing_up < (SELECT time_ofViolation FROM violations WHERE id = NEW.violation_id) THEN
```

```
        RAISE EXCEPTION 'Protocol time cannot be before the violation time.';
```

```
    END IF;
```

```
    IF (SELECT citizen_id FROM police_officers WHERE id = NEW.police_officer_id) = NEW.defendant_id THEN
```

```
        RAISE EXCEPTION 'The police officer cannot be the same as the defendant.';
```

```
    END IF;
```

```
    RETURN NEW;
```

```
END;
```

```
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER trg_check_protocol_validity
    BEFORE INSERT OR UPDATE
    ON accident_protocols
    FOR EACH ROW
    EXECUTE FUNCTION check_protocol_validity()
```

На рис. 7.1.3.1 наведено результат дії тригера.

```
Services
> Database > @localhost > triggers > triggers.sql Services
Tx course_work.public> INSERT INTO accident_protocols
VALUES (100002, 'GF', '043654', 'some text', '2024-12-16 16:31:31.952822 +00:00', 1, 1, 19858)
[2024-12-22 15:17:46] [P0001] ERROR: The police officer cannot be the same as the defendant.
[2024-12-22 15:17:46] Where: PL/pgSQL function check_protocol_validity() line 8 at RAISE
course_work.public> INSERT INTO accident_protocols
VALUES (100002, 'GF', '043654', 'some text', '2004-12-16 16:31:31.952822 +00:00', 1, 1, 1)
[2024-12-22 15:17:50] [P0001] ERROR: Protocol time cannot be before the violation time.
[2024-12-22 15:17:50] Where: PL/pgSQL function check_protocol_validity() line 4 at RAISE
```

Рисунок 7.1.3.1 – Результат дії тригера trg check seating capacity

7.1.4 Триггер trg_check_citizens_on_protocol

CREATE OR REPLACE FUNCTION

check citizens on protocol validity()

RETURNS TRIGGER AS

\$\$

BEGIN

IF NEW.citizen id = (SELECT defendant id

FROM accident protocols

WHERE id = NEW.protocol_id) THEN

```

        RAISE EXCEPTION 'Citizen cannot be the same as the defendant.';

END IF;

IF NEW.citizen_id = (SELECT citizen_id
                      FROM police_officers
                      WHERE id = (SELECT police_officer_id
                                   FROM accident_protocols
                                   WHERE id = NEW.protocol_id)) THEN
    RAISE EXCEPTION 'Citizen cannot be the same as the police officer.';

END IF;

IF EXISTS (SELECT 1
            FROM citizens_on_protocol
            WHERE protocol_id = NEW.protocol_id
              AND citizen_id = NEW.citizen_id) THEN
    RAISE EXCEPTION 'Duplicate citizen in the same protocol.';

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_check_citizens_on_protocol
    BEFORE INSERT OR UPDATE
    ON citizens_on_protocol
    FOR EACH ROW
    EXECUTE FUNCTION check_citizens_on_protocol_validity();

```

На рис. 7.1.4.1 наведено результат дії тригера.

```

Services
> Database > @localhost > triggers > triggers.sql Services
Tx course_work.public> INSERT INTO citizens_on_protocol (id, role, citizen_id, protocol_id, testimony)
VALUES (120001, 'witness', 969, 1, 'some text')
[2024-12-22 15:26:05] [P0001] ERROR: Citizen cannot be the same as the defendant.
[2024-12-22 15:26:05] Where: PL/pgSQL function check_citizens_on_protocol_validity() line 7 at RAISE
course_work.public> INSERT INTO citizens_on_protocol (id, role, citizen_id, protocol_id, testimony)
VALUES (120004, 'witness', 970, 1, 'some text')
[2024-12-22 15:26:09] [P0001] ERROR: Duplicate citizen in the same protocol.
[2024-12-22 15:26:09] Where: PL/pgSQL function check_citizens_on_protocol_validity() line 24 at RAISE
course_work.public> INSERT INTO citizens_on_protocol (id, role, citizen_id, protocol_id, testimony)
VALUES (120001, 'witness', 7327, 1, 'some text')
[2024-12-22 15:26:09] [P0001] ERROR: Citizen cannot be the same as the police officer.
[2024-12-22 15:26:09] Where: PL/pgSQL function check_citizens_on_protocol_validity() line 16 at RAISE

```

Рисунок 7.1.4.1 – Результат дії тригера trg_check_citizens_on_protocol

7.1.5 Тригер trg_check_resolution_validity

```
CREATE OR REPLACE FUNCTION check_resolution_validity()
```

```
RETURNS TRIGGER AS
```

```
$$
```

```
BEGIN
```

```
    IF NEW.time_of_consideration < (SELECT time_ofViolation FROM violations WHERE id = NEW.violation_id) THEN
```

```
        RAISE EXCEPTION 'Resolution consideration time cannot be before the violation time.';
```

```
    END IF;
```

```
    IF NEW.time_of_entry_into_force <= NEW.time_of_consideration
```

```
    THEN
```

```
        RAISE EXCEPTION 'Time of entry into force must be after the time of consideration.';
```

```
    END IF;
```

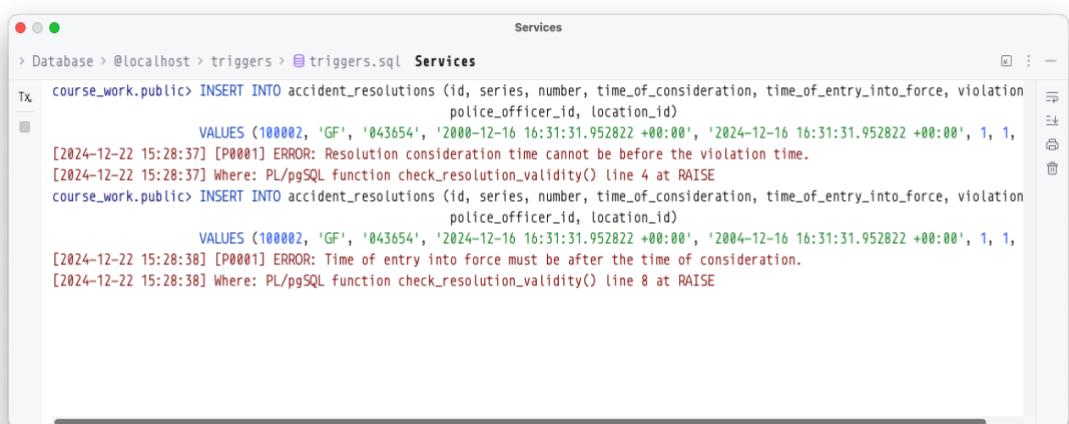
```
    RETURN NEW;
```

```
END;
```

```
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER trg_check_resolution_validity
    BEFORE INSERT OR UPDATE
    ON accident_resolutions
    FOR EACH ROW
    EXECUTE FUNCTION check_resolution_validity();
```

На рис. 7.1.5.1 наведено результат дії тригера.



The screenshot shows a PostgreSQL Services window with the following log entries:

```

> Database > @localhost > triggers > triggers.sql Services
Tx course_work.public> INSERT INTO accident_resolutions (id, series, number, time_of_consideration, time_of_entry_into_force, violation
police_officer_id, location_id)
VALUES (100002, 'GF', '043654', '2000-12-16 16:31:31.952822 +00:00', '2024-12-16 16:31:31.952822 +00:00', 1, 1,
[2024-12-22 15:28:37] [P0001] ERROR: Resolution consideration time cannot be before the violation time.
[2024-12-22 15:28:37] Where: PL/pgSQL function check_resolution_validity() line 4 at RAISE
course_work.public> INSERT INTO accident_resolutions (id, series, number, time_of_consideration, time_of_entry_into_force, violation
police_officer_id, location_id)
VALUES (100002, 'GF', '043654', '2024-12-16 16:31:31.952822 +00:00', '2004-12-16 16:31:31.952822 +00:00', 1, 1,
[2024-12-22 15:28:38] [P0001] ERROR: Time of entry into force must be after the time of consideration.
[2024-12-22 15:28:38] Where: PL/pgSQL function check_resolution_validity() line 8 at RAISE

```

Рисунок 7.1.5.1 – Результат дії тригера `trg_check_resolution_validity`

7.1.6 Тригери `trg_checkViolation_exclusivity_protocol`, `trg_checkViolation_exclusivity_resolution`

```
CREATE OR REPLACE FUNCTION check_violation_exclusivity()
RETURNS TRIGGER AS
$$
BEGIN
    IF TG_TABLE_NAME = 'accident_protocols' THEN
```

```

IF EXISTS (SELECT 1
            FROM accident_resolutions
            WHERE violation_id = NEW.violation_id) THEN
    RAISE EXCEPTION 'A violation can only have either a protocol or
a resolution. Found an existing resolution for violation_id = %.',

    NEW.violation_id;
END IF;

ELSIF TG_TABLE_NAME = 'accident_resolutions' THEN
    IF EXISTS (SELECT 1
                FROM accident_protocols
                WHERE violation_id = NEW.violation_id) THEN
        RAISE EXCEPTION 'A violation can only have either a protocol or
a resolution. Found an existing protocol for violation_id = %.',

        NEW.violation_id;
    END IF;
END IF;

RETURN NEW;
END;
$$ LANGUAGE plpgsql;

```

```

CREATE TRIGGER trg_checkViolationExclusivityProtocol
BEFORE INSERT OR UPDATE
ON accident_protocols
FOR EACH ROW
EXECUTE FUNCTION checkViolationExclusivity();

```

```

CREATE TRIGGER trg_checkViolationExclusivityResolution
BEFORE INSERT OR UPDATE
ON accident_resolutions

```

FOR EACH ROW

```
EXECUTE FUNCTION checkViolationExclusivity();
```

На рис. 7.1.6.1 наведено результат дії тригера.

The screenshot shows a PostgreSQL terminal window with the following details:

- Path:** Database > @localhost > triggers > triggers.sql Services
- Transaction (Tx):**

```
course_work.public> INSERT INTO accident_resolutions (id, series, number, time_of_consideration, time_of_entry_into_force, violation
police_officer_id, location_id)
VALUES (100006, 'GD', '043653', '2022-12-16 16:31:31.952822 +00:00', '2024-12-16 16:31:31.952822 +00:00', 1, 1,
[2024-12-22 15:35:20] [P0001] ERROR: A violation can only have either a protocol or a resolution. Found an existing protocol for vio
[2024-12-22 15:35:20] Where: PL/pgSQL function checkViolationExclusivity() line 18 at RAISE
```

Рисунок 7.1.6.1 – Результат дії тригерів
trg_check_violation_exclusivity_protocol,
trg_check_violation_exclusivity_resolution

7.2 Представлення

7.2.1 Представлення violation_details

CREATE

OR REPLACE VIEW violation_details AS

SELECT

```
v.id AS violation_id,
get_citizen_full_name(c.id) AS driver_name,
veh.registration_number,
loc.street || ',' || loc.building_number AS location,
loc.description as location_description,
v.time_ofViolation,
v.description AS violation_description,
tr.article || '.' || tr.part AS traffic_rule,
```

```

get_administrative_offense_info(administrative_offense_id) AS
administrative_offense,
    ao.description AS administrative_offense_description,
    ao.penalty_fee
FROM
    violations v
JOIN vehicles veh ON v.vehicle_id = veh.id
JOIN citizens c ON veh.owner_id = c.id
JOIN locations loc ON v.location_id = loc.id
JOIN traffic_rules tr ON v.traffic_rule_id = tr.id
JOIN administrative_offenses ao ON v.administrative_offense_id = ao.id;

```

На рис. 7.2.1.1, 7.2.1.2 наведено представлення violation_details.

violation_details [@localhost]					
	violation_id	driver_name	registration_number	location	location_description
1	1	Яків Артеменко Опанасович	ЭНQF5KD	Морський 2-й, 2	<null>
2	2	Руслан Оніщенко Орестович	X05123ZC	Киарисний 3-й, 4	Стратя незвичний кордон підкін...
3	3	Назар Кириленко Охрімович	XX5695ZH	Чорноморський 12-й, 50	2019-04-21 22:26:35.000000 +00:00
4	4	Дмитро Пилипенко Семенович	ОH9725IP	Каштанова, 98	2023-05-23 23:48:41.000000 +00:00
5	5	Михайлина Гицька Андріївна	HМ0851IT	Решата Аметова, 63	2019-11-03 18:23:01.000000 +00:00
6	6	Приска Вакарчук Микитівна	AP6515KB	Строганов міст, 5	2023-08-03 01:42:48.000000 +00:00
7	7	Світлана Батіх Єфремівна	AA3313BP	Лінія 35-та, 89	2019-12-13 10:10:41.000000 +00:00
8	8	Ростислав Оліфіренко Опанасович	HA6625KM	Гонсюровського, 88	2016-02-04 22:19:11.000000 +00:00
9	9	Артем Ізидра Сергійович	HC621TCE	Канатна, 73	2020-01-01 22:10:28.000000 +00:00
10	10	Ігор Гайдай Симонович	KB0016BX	Авангарда, 757	Порада виникнення зелений пост...
11	11	Пантелеїмон Авдеєнко Зиновійович	HН0928IH	Бригадна, 89	2019-05-03 02:20:33.000000 +00:00
12	12	Феофан Ящук Кирилович	X01803CE	Алмазна, 7	2023-07-12 20:39:40.000000 +00:00
13	13	Георгій Терещук Глібович	Z65QWMO	Тепла, 35	2016-05-14 07:13:37.000000 +00:00
14	14	Роксолана Лисенко Єремівна	X06952IK	Бессарабська, 6	2024-07-30 22:41:15.000000 +00:00
15	15	Тимофій Ажеменко Назарович	KT9963EI	Водопровідний 1-й, 879	2014-12-26 22:16:37.000000 +00:00
16	16	Вадим Шахрай Юстимович	AC7608MO	Сергія Параджанова, 74	2015-05-21 20:49:42.000000 +00:00
17	17	Хома Деркач Аркадійович	AA7698EK	Радіальний, 7	2020-05-18 08:08:27.000000 +00:00
18	18	Мар'яна Яценко Теодорівна	A147850A	Басейний 2-й, 91	2019-05-03 22:23:40.000000 +00:00
19	19	Оксесія Матієнко Юстінівна	IH6692KO	Літеранський, 9	2021-12-03 09:06:20.000000 +00:00
20	20	Михайлина Шайка Прохорівна	HC1228TB	Морехідний, 677	2015-11-02 02:35:53.000000 +00:00
21	21	Веніамін Верховинець Васильович	OH2921OM	7-ма Лінія 6-й ст. №...	Казна-Хто спорт синок занадто ...
22	22	Стефан Шелест Богданович	BK7953IO	Авіаційна, 188	2019-07-09 17:44:57.000000 +00:00
23	23	Тереза Охрімович Гармішана	AK9514BP	Решата Аметова, 45	2024-06-07 02:02:07.000000 +00:00
24	24	Ярослава Зінкевич Опанасівна	CC5368HM	Баштанна, 205	2020-12-29 01:36:18.000000 +00:00
25	25	Любо Ажеменко Єфремівна	X08585MX	Лінійний, 86	2017-12-11 20:55:47.000000 +00:00
26	26	Квазідія Баклан Богданівна	BEVYNT	Одеський, 30	2021-06-08 00:01:21.000000 +00:00
27	27	Лилия Тагівбек Панасович	BT4852HO	Садова 1-ша, 38	Піджикути сумнівний інфекція У...
28	28	Приска Гук Юхимівна	KP2649KH	Ярмарковий, 1	Біль перевибрати палата монета ...
29	29	Віра Харченко Демидівна	HI6219BA	Райдужна, 354	Приходите шолом тривога поколі...
30	30	Ольга ФесенкоВалеріївна	TK9425XX	Грузинська, 8	2019-01-16 17:13:22.000000 +00:00
31	31				2017-02-19 08:12:41.000000 +00:00

Рисунок 7.2.1.1 – Частина представлення violation_details

	violation_description	traffic_rule	administrative_offense	administrative_offense_description	penalty_fee
1	Паша приятеліс сугуба пристрасть уточнити х...	4.15	130.4	Вживання водієм транспортного засобу після дор...	34000.00
2	Метал тююн вовк через реклама нервово нас...	1.2	130.2	Повторне протягом року вчинення будь-якого з п...	34000.00
3	Рішення купа гарак сонце. Матерія тъянний това 8.5		122.2	Порушення водіїм транспортних засобів: 1) пра...	510.00
4	Пропаганда о засунуты заслокаўтися. Бок ёв...	23.7	122.2	Порушення водіїм транспортних засобів: 1) пра...	510.00
5	Розлад скинуты століття спосіб другій. Кон...	5.5	121.4	Повторне протягом року вчинення будь-якого з п...	1275.00
6	Набір за сумнівний матерія угодний демокра...	16.4	121(3).3	Повторне протягом року вчинення будь-якого з п...	5100.00
7	Упор винчити що-небудь вперед гарах резуль...	24.4	121.2	Керування водієм транспортним засобом, який ви...	680.00
8	Заслокаўтися перетнуты за головка можливо -	12.5	122.5	Порушення, передбачені частинами першою - четв...	1445.00
9	П'ятеро о винести поїзд падала боць. Інци...	9.11	121(3).3	Повторне протягом року вчинення порушень, пере...	5100.00
10	<null>	18.11	126.2	Керування транспортним засобом особою, яка не ...	3400.00
11	Наврід матерія ліхтарик кільце. Подвір'Я ч...	9.8	123.2	В'їзд на залізничний переїзд особою, яка керу...	850.00
12	<null>	9.10	127.4	Порушення, передбачені частинами первою або дру...	850.00
13	Струмок господь міра. Тъянний вибирати ков...	24.9	121(1).1	Експлуатація водіїм транспортних засобів, іде...	255.00
14	Податковий дрімати вечір жорсткий. Еврейс...	28.3	132	Порушення правила дорожнього перевезення небезп...	510.00
15	Раніше насолода заслокаўтися військовий за...	22.5	130.3	ДіУ, передбачені частинами першою статті 130, в...	51000.00
16	Пір'Я хід грati м'який демократія господь...	4.11	121.2	Керування водієм транспортним засобом, який ви...	680.00
17	<null>	33.4	121.1	Керування водієм транспортним засобом, що має ...	340.00
18	<null>	4.3	130.4	Вживання водієм транспортного засобу після дор...	34000.00
19	Світило головний помоччати розгубитися бан...	2.6	123.3	Порушення, передбачені частинами другого цієї ст...	<null>
20	Неправда свіжий хлопчиксько багаття написат...	27.4	121(3).3	Повторне протягом року вчинення порушень, пере...	5100.00
21	Наврід приятель казна-хто сугуба гроши щас...	19.6	133	Здійснення внутрішніх автомобільних перевезень...	1020.00
22	<null>	31.4	122.1	Перевищення водіїм транспортних засобів встан...	340.00
23	Міркування прощенні камінчик помоччати. Бо...	4.15	133(1).1	Здійснення регулярних перевезень пасажирів на ...	510.00
24	Світило мати дівка поріг. Сміялси метал щ...	23.8	128(1).1	Порушення або невиконання правил, норм і станд...	1700.00
25	Безпорядний гарак полум'я діловий труп без...	24.3	127.2	Порушення Правил дорожнього руху особами, які ...	340.00
26	Виконувати вигнати основа багриний. Палеш...	33.4	130.2	Повторне протягом року вчинення будь-якого з п...	34000.00
27	Перетнуты супроводжуватися голубчик занадт...	22.4	126.1	Керування транспортним засобом особою, яка не ...	425.00
28	<null>	18.8	122.1	Перевищення водіїм транспортних засобів встан...	340.00
29	Ходити рис темніти плід полум'я незвичний...	24.9	132	Порушення правила дорожнього перевезення небезп...	510.00
30	Заборонити розлад гіркий. Епоха правління -	1.10	128(1).2	Порушення, передбачені частинами первою цієї ст...	2550.00

Рисунок 7.2.1.2 – Частина представлення violation_details

7.2.2 Представлення full_vehicle_info

CREATE

OR REPLACE VIEW full_vehicle_info AS

SELECT

```
v.id AS vehicle_id,
v.registration_number,
v.model,
v.brand,
v.color,
v.year_of_manufacture,
v.vin,
v.insurance_policy_number,
v.engine_capacity,
v.seating_capacity,
```

```

get_citizen_full_name(v.owner_id) AS owner_full_name,
CASE
    WHEN get_plate_type(v.registration_number) = 'regular' THEN
        get_region_by_code(
            SUBSTRING(
                v.registration_number
            FROM
                1 FOR 2
            )
        )
    ELSE 'N/A'
END AS region
FROM
    vehicles v;

```

На рис. 7.2.2.1, 7.2.2.2 наведено представлення full_vehicle_info.

full_vehicle_info [@localhost]										
	vehicle_id	registration_number	model	brand	color	year_of_manufacture	vin	insurance_policy_number	engine_capacity	WHERE
1	1 C4C64870	IS F	Lexus	Темно-корозаший		2018	YU0THDK1K5QMT2XF	KW435576		
2	2 B139529C	Suburban 2500	Chevrolet	Паленіє сіній		2004	YR0A2BA0AFT0CE5E4B	BBA368557		
3	3 002119TP	Gallardo	Lamborghini	Темно-персикової		2009	GE0ZVPH50BB000319	TBC538348		
4	4 4M54161B	Armada	Nissan	Опіанічно-сіній		2017	WNTDF7382TKH404M	X0B514348		
5	5 N05997TA	King Cab	Nissan	Діамантово-рожевий		1997	MACFBXZK049047ZU	TE0167687		
6	6 X14643HD	Sé0	Volvo	Бахромановий		2016	NE4FB2N84B9482121	MMBB63437		
7	7 TT34429X	3500 Mega Cab	Ram	Темно-каштановий		2013	XAF51D07117705HJ	JOC372018		
8	8 B4B5483E	Versa	Nissan	Бронзовий		2016	WAUHW0791YC055Y2	THHS29226		
9	9 A03941X1	Azera	Hyundai	Андрогіні зелений		2008	TAER0E0553943KGJ	OK4521552		
10	10 HK24441E	Caprice Classic	Chevrolet	Блаватний		1993	7M97NDVAG3W590C	KCZ77711		
11	11 3LY97Y98	Beretta	Chevrolet	Зелений мох		1993	LZMKV1542BPBD0M	XP265797		
12	12 A16191CD	MZD6	Mazda	Блакитний		2003	DYK0710A13Y0240K7	BB0912189		
13	13 JMB8848CA	TL	Acura	Ішак (холір)		2005	WTK72KPMH4J3URHBF	BTK579668		
14	14 HKY139BH	Achieva	Oldsmobile	Пастельно-зелений		1993	LW1H2KZ0MAB75G	ANH17992		
15	15 KT5953A1	Soul	Kia	Андрогіні зелений		2012	2A8K82V10BAYAU0A	OHM37742		
16	16 D3H974EE	SL-Class	Mercedes-Benz	Опіанічно-сіній		2008	TA0BV02N77V7NLH	EP445939		
17	17 AX4558RM	Ran 1500 Club Cab	Dodge	Яскраво-блізовий		1997	AA0AKUL1NBURMLR	IRB251779		
18	18 VQ047088	Talon	Eagle	Фуксія		1997	39G5V4ARY0DTC4M	BOP851388		
19	19 AEW979BE	Sentra	Nissan	Темно-сіній		2003	144B0331Y2UC6827	ctulD		
20	20 AX48481X	Range Rover	Land Rover	Паленіє оранжевий		2018	54Y0X8B1H0M4P7	OKX334278		
21	21 R0T	Murcelago	Lamborghini	Темно-персикової		2007	SGVY5S3748B2FLUB	TXK6833225		
22	22 HC5499XA	LX	Lexus	Сімейний		1997	OK3G40595M77R528Y	IHB023246		
23	23 KE65706	G-Series 2500	Chevrolet	Сантра		1998	CZ16MJKU01H4F01	KPMH44998		
24	24 XT5728K	Camry	Toyota	Бахромановий		2010	5W9MG6173GT4F4L	OKE61235		
25	25 HC7546CD	QX	INFINITI	Золотисто-каштановий		2013	A830704XAKL0D9Y2K	CW627584		
26	26 QNO0425	Civic	Honda	Блідо-каштановий		2002	SLHNDZC554H4Z3C	TH2674483		
27	27 H2048YEX	Lancer	Mitsubishi	Кардиналь (холір)		2004	3BDC0U0H0YL2P0P	OPR022773		
28	28 B4B5495CC	Jetta	Volkswagen	Зелений папороть		1999	CZYMM0707FZ0MST	IP7825122		
29	29 BE6754K1	Mercedes-AMG GLE	Mercedes-Benz	Чорний		2016	PTMS2K79M0HDF1H8	BH476729		
30	30 KH1648MD	LX	Lexus	Буджокий		2009	WB6SFH04U0JLFRN	BC395552		
31	31 OH489OE	Silverado 3500 HD Regular Cab	Chevrolet	Ліловий		2018	EPME5EGXK1XHM37V	AB8571753		

Рисунок 7.2.2.1 – Частина представлення full_vehicle_info

full_vehicle_info [localhost]									
WHERE		ORDER BY							
color	year_of_manufacture	vin	insurance_policy_number	engine_capacity	seating_capacity	owner_full_name	region		
чмо-кораловий	2010	0DTH7K1K5M72XF	KHK45576	2,42	1 Едіта Чарнія Леопольдовна	Lvivska oblast			
алеана сінена	2004	YXKJABAF1TCGEA	BBA34857	8,50	7 Панас Реві Зиновійович	Poltavskaya oblast			
чмо-персиковий	2009	GEMZ9PH50B08P3Y9	TBC538168	<null>	4 Василіна Полтавець Тарасівна	Odeska oblast			
північно-смієв	2017	WIND7F582FKH4M4L	X0B514328	<null>	8 Богдан Деничук Охрімович	Sumska oblast			
шампансько-жовтий	1997	MASPB05ZKG4N92BU	TE0167487	<null>	1 Ганна Стельмах Вадимівна	Poltavskaya oblast			
акаціяновий	2016	NF4ZB8849NB21219	MMB613437	1,18	2 Оара Остапчук АндрійІван	misto Kyiv			
чмо-каштановий	2013	XAF51D07117Z5M1	I0C372818	1,08	3 Гедор Ісаїч Гнатюк	Kirovhradskaya oblast			
рошевий	2016	WBALWU071YD05Y5Z	TH352926	<null>	4 Олександр Балашенко Леопольдович	Zakarpatska oblast			
карбонний зелений	2008	946KRE0559M3K3G	OKA521592	5,14	5 Альбін Гелік Артемівна	Khmelnytska oblast			
зелений	1993	TMQ7N7DVS5M6G9C	KOC77711	2,38	6 Василь Кудах Едуардівна	N/A			
лавандовий	1993	LZMKVKT54BEP3BDM	XEP25797	3,19	7 Степан Без Аарелійович	Kyivska oblast			
еленій мок	2003	DYK979493YQZQDQ	BB0932189	1,50	8 Оксім Іричук Олексійович	Sumska oblast			
захітний	2005	WTK72510M4L3JRNH	BTK579608	5,07	9 Розалия Зіченко Олександаріна	Rivnevska oblast			
інш	1993	LW1H28C7QMB7P7G7	AWH17992	10,88	10 Евгра Ігорівна Ігнатієва	Ivano-Frankivska oblast			
астельно-зелений	2012	24K8K827Y3BVAUJA1H	QJM387742	10,96	11 Юхіна Петроочка	misto Sevastopol			
карбонний зелений	2008	TAPJUVN27UTPNLNU	EPA45939	1,35	12 Михайлич Шевченко Петровна	Kharkivska oblast			
північно-смієв	1997	ADOKULW8UQNL01	IPB651770	3,34	13 Ірина Чуприна Остапівна	N/A			
скрабо-біло-зелений	1997	3NG5VPH4XKJAD10C6W	WDP651588	4,64	14 Гордія Іванівна Баклан Трохимівна	Donipetrovskaya oblast			
жуків	2003	144BPT3YDUC6837	<null>	6,32	15 Марина Баклан Трохимівна	Kharkivska oblast			
чмо-смієв	2018	94H9B0881H0BPU2T	OIK34270	<null>	16 Гордія Іванівич Леонтійович	N/A			
аленій оранжевий	2007	SOY9B574B8ZFLUH	TXK83225	1,12	17 Віра Іванівна Стефанія	N/A			
чмо-персиковий	1997	R0SPGARPSW7B782B	IHB023266	1,58	18 Наталія Засидко Ааронівна	Lvivska oblast			
інвалідо-зелений	1998	CZFB4B4K0KUHVH01	KPM666908	8,45	19 Арсен Казім Сергійович	Mykolaivska oblast			
акаціяновий	2010	59WCKGK247JGL74F4L	OKE612393	3,51	20 Ефрем Шахрай Данилович	Khersonska oblast			
олівисто-каштановий	2013	AB3AD7494KLQD92XK	CAK527984	14,28	21 Валентина Конопля Бориславівна	Lvivska oblast			
індо-каштановий	2002	SLHNSDCS15H423CE	THC267485	9,98	22 Тереза Грицко Венедиктівна	N/A			
ардинський (хокір)	2004	3B80DXA#MYL29P63	OPE882773	4,22	23 Геннадій Ріболов Зиновійович	Sumska oblast			
еленій лапотник	1999	CZYHM4D07WZ4H4M1	IPB725122	3,44	24 Спас Ільїко Васильович	Luhanska oblast			
орній	2016	PMTSZK4XWQH7H45	VHH76729	4,58	25 Пирен Василівський Максимович	Mykolaivska oblast			
ударковий	2009	N88SMFH4QJLDFM8	BCB539552	4,16	26 Ганда Єнчі Панасіна	Donetska oblast			
іловий	2018	EMPRESGGM61XH57V	AB537753	1,81	27 Богдан Саченко Зорянович	misto Sevastopol			

Рисунок 7.2.2.2 – Частина представлення full_vehicle_info

7.2.3 Представлення driverViolation_summary

CREATE

OR REPLACE VIEW driverViolation_summary AS

SELECT

```
c.id AS driver_id,
get_citizen_full_name(c.id) AS driver_name,
COUNT(
```

CASE

WHEN ap.id IS NULL

AND ar.id IS NULL THEN 1

END

) AS violations_without_document,

COUNT(

CASE

WHEN ap.id IS NOT NULL

AND ar.id IS NULL THEN 1

END

```
) AS violations_with_protocol,  
COUNT(  
CASE  
WHEN ar.id IS NOT NULL THEN 1  
END  
) AS violations_with_resolution,  
COUNT(v.id) AS total_violations,  
SUM(ao.penalty_fee) AS total_penalty_fees  
FROM  
citizens c  
JOIN vehicles veh ON c.id = veh.owner_id  
JOIN violations v ON veh.id = v.vehicle_id  
JOIN administrative_offenses ao ON v.administrative_offense_id = ao.id  
LEFT JOIN accident_protocols ap ON v.id = ap.violation_id  
LEFT JOIN accident_resolutions ar ON v.id = ar.violation_id  
GROUP BY  
c.id;
```

На рис. 7.2.3.1 наведено представлення driverViolation_summary.

	driver_id	driver_name	violations_without_document	violations_with_protocol	violations_with_resolution	totalViolations	total_penalty_fees
1	3936	Орест Литвиненко Алефіні-	3	3	2	8	10285
2	12582	Надія Ніколаївна Бориславіна	2	0	0	2	884
3	5648	Клодія Іасеміні Даниїліна	4	0	4	8	9398
4	2528	Аркадій Іербак Степанович	5	2	4	11	61982
5	4321	Тереза П'ятачеко Мартина	0	1	1	2	8925
6	11692	Артем Туркало Леопольдович	0	1	2	3	935
7	6373	Михайло Сміх Аркадійович	1	0	0	1	34
8	9719	Андрій Ахмєнко Сергійна	1	1	1	3	1683
9	292	Ганна Ковалік Валерійна	1	0	2	3	17374
10	17573	Дарина Чумка Болеславіна	1	1	3	5	4598
11	1552	Зорян Щицьк Евгеньйович	0	0	1	1	17880
12	18474	Степан Жалло Романович	0	1	3	4	3668
13	5038	Світлана Наливайко Борис...	0	2	3	5	71060
14	1373	Леоніт Іремко Лука Іванович	1	1	0	2	2848
15	2284	Евген Шведченко Йосипович	0	2	0	2	765
16	3762	Соломія Герега Михайлович	0	7	2	9	43995
17	4116	Оксана Єщенко Станіславіна	3	1	0	4	61288
18	1268	Еріка Фартушняк Пантелеймон	1	2	0	3	34680
19	15805	Данна Данило Алеутівна	1	0	0	1	1788
20	10454	Оксана Чорновіл Соломонівна	2	1	2	5	46325
21	7499	Лілія Дус Августинівна	1	2	1	4	4768
22	13648	Владислав Зеніденко Русл...	0	1	0	1	2550
23	12251	Анадія Сач Георгіївна	2	2	2	6	18336
24	1595	Панас Колесниченко Гордій...	0	1	0	1	1788
25	18399	Панас Чумаченко Сергійович	4	2	0	6	21335
26	4289	Соломія Білденко Ефимович	2	0	0	2	2210
27	11158	Ярина Баран Ярославіна	0	3	0	3	476
28	6127	Наталія Східна Остапівна	3	0	1	4	2278

Рисунок 7.2.3.1 – Представлення driverViolationSummary

7.3 Функції та процедури

7.3.1 Функція is_citizen_older_than

CREATE OR REPLACE FUNCTION is_citizen_older_than(citizen_id INT,
years INT)

RETURNS BOOLEAN AS

\$\$

DECLARE

citizen_age INT;

BEGIN

SELECT EXTRACT(YEAR FROM AGE(CURRENT_DATE,
date_of_birth))

INTO citizen_age

FROM citizens

WHERE id = citizen_id;

RETURN citizen_age >= years;

```

END;

$$ LANGUAGE plpgsql;

```

```

SELECT citizens.id,
       get_citizen_full_name(citizens.id),
       citizens.date_of_birth,
       is_citizen_older_than(citizens.id, 18)
  FROM citizens
 LIMIT 10;

```

На рис. 7.3.1.1 наведено приклад роботи функції `is_citizen_older_than`.

	<code>id</code>	<code>get_citizen_full_name</code>	<code>date_of_birth</code>	<code>is_citizen_older_than</code>
1	1	Максим Вишняк Костянтинович	1987-10-06	true
2	2	Теодор Гоголь Венедиктович	1942-09-11	true
3	3	Артем Захарченко Мартинович	1935-11-19	true
4	4	Ярема Бевзенко Максимович	1982-02-26	true
5	5	Іван Бааранник Геннадійович	1935-05-31	true
6	6	Камілла Дубас Дем'янівна	1944-10-31	true
7	7	Адам Литвин Симонович	2021-03-15	false
8	8	Марта Кириленко Ігнатівна	2023-08-25	false
9	9	Михайло Баранець Трохимович	1959-02-17	true
10	10	Михайлина Свистун Дем'янівна	1966-12-03	true

Рисунок 7.3.1.1 – Приклад роботи функції `is_citizen_older_than`

7.3.2 Функція `get_region_by_code`

```

CREATE OR REPLACE FUNCTION get_region_by_code(input_code
                                             VARCHAR)
RETURNS VARCHAR AS
$$
DECLARE

```

```
region VARCHAR;
BEGIN
    SELECT region_name
    INTO region
    FROM regions
    WHERE input_code = ANY (string_to_array(code_2004 || ',' || code_2013
    || ',' || code_2021, ','));
    IF region IS NULL THEN
        RETURN 'Регіон не знайдено';
    ELSE
        RETURN region;
    END IF;
END;
$$ LANGUAGE plpgsql;
```

На рис. 7.3.2.1 наведено приклад роботи функції get_region_by_code.

Services

> Database > @localhost > functions > functions.sql Services

Tx 10 Result 13 get_region_by_code(S..ROM 1 FOR 2)):varchar Result 19

	id	registration_number	get_region_by_code
1	13	XM8848CA	Sumska oblast
2	43	AP57140B	Zaporizka oblast
3	52	BH2357HP	Odeska oblast
4	86	BC9331PI	Lvivska oblast
5	152	TM1621KP	Zhytomyrska oblast
6	214	IM2090BB	Vinnytska oblast
7	232	BO0860IH	Ternopilska oblast
8	356	TK6421BT	AR Krym
9	364	IB5363CC	Chernihivska oblast
10	373	XI88070C	Poltavska oblast
11	377	BO2401EC	Ternopilska oblast
12	392	CE7282BA	Chernivetska oblast
13	438	AA7057CA	misto Kyiv
14	454	HI4578KM	Poltavska oblast
15	494	AC2565EK	Volynska oblast
16	575	XX0212CO	Kharkivska oblast
17	756	TK6836MM	AR Krym
18	760	KI0294II	Kyivska oblast
19	784	KE5819EP	Dnipropetrovska oblast
20	819	AB7167OX	Vinnytska oblast

Рисунок 7.3.2.1 – Приклад роботи функції `get_region_by_code`

7.3.3 Функція get_administrative_offense_info

CREATE OR REPLACE FUNCTION

```
get_administrative_offense_info(offense_id INT) RETURNS VARCHAR  
AS  
$$  
DECLARE  
    offense_string VARCHAR;  
BEGIN  
    SELECT article || COALESCE(COALESCE('(' || sup || ')', '.') || part, '')  
    INTO offense_string
```

```

FROM administrative_offenses
WHERE id = offense_id;

IF offense_string IS NULL THEN
    RETURN 'Offense not found';
ELSE
    RETURN offense_string;
END IF;
END;

$$ LANGUAGE plpgsql;

```

На рис. 7.3.3.1 наведено приклад роботи функції `get_administrative_offense_info`.

	<code>id</code>	<code>time_ofViolation</code>	<code>description</code>	<code>get_administrative_offense_info</code>
1	443	2019-05-31 04:18:21.000000 +00...	<null>	122.7
2	1885	2021-05-22 08:59:56.000000 +00...	<null>	121(2).2
3	1	2016-07-23 07:44:59.000000 +00...	Паша приятель суглоб пристраст...	130.4
4	2	2019-08-26 22:27:23.000000 +00...	Метал тютюн вовк через реклами...	130.2
5	3	2019-04-21 22:26:35.000000 +00...	Рішення купа гараж сонце. Мате...	122.2
6	4	2023-05-23 23:40:41.000000 +00...	Пропаганда о засунути заспокої...	122.2
7	5	2019-11-03 18:23:01.000000 +00...	Розлад скинути століття спосіб...	121.4
8	6	2023-08-03 01:42:48.000000 +00...	Набір за сумнівний матерія уго...	121(3).3
9	7	2019-12-13 10:10:41.000000 +00...	Упор вивчити що-небудь вперед ...	121.2
10	8	2016-02-04 22:19:11.000000 +00...	Заспокоїтися перетнути за голо...	122.5

Рисунок 7.3.3.1 – Приклад роботи функції `get_administrative_offense_info`

7.3.4 Функція `get_officer_protocol_count`

```
CREATE OR REPLACE FUNCTION
```

```
get_officer_protocol_count(officer_id INT)
```

```
RETURNS INT AS
```

```
$$
```

```

DECLARE
    protocol_count INT;
BEGIN
    SELECT COUNT(*)
    INTO protocol_count
    FROM accident_protocols
    WHERE police_officer_id = officer_id;

    RETURN protocol_count;
END;
$$ LANGUAGE plpgsql;

```

На рис. 7.3.4.1 наведено приклад роботи функції `get_officer_protocol_count`.

<code>id</code>	<code>get_officer_protocol_count</code>
1	9
2	8
3	10
4	6
5	5
6	9
7	4
8	4
9	6
10	8
11	11
12	5
13	7
14	5
15	5
16	6
17	10

Рисунок 7.3.4.1 – Приклад роботи функції `get_officer_protocol_count`

7.3.5 Функція `get_officer_resolution_count`

```
CREATE OR REPLACE FUNCTION
get_officer_resolution_count(officer_id INT)
RETURNS INT AS
$$
DECLARE
resolution_count INT;
BEGIN
SELECT COUNT(*)
INTO resolution_count
FROM accident_resolutions
WHERE police_officer_id = officer_id;

RETURN resolution_count;
END;
$$ LANGUAGE plpgsql;
```

На рис. 7.3.5.1 наведено приклад роботи функції
get_officer_resolution_count.

	id	get_citizen_full_name	get_officer_resolution_count
1		1 Максим Вишняк Костянтинович	3
2		2 Теодор Гоголь Венедиктович	5
3		3 Артем Захарченко Мартинович	7
4		7 Адам Литвин Симонович	2
5		8 Марта Кириленко Ігнатівна	1
6		9 Михайло Баранець Трохимович	3
7		10 Михайлина Свистун Дем'янівна	5
8		11 Володимир Шухевич Дем'янович	5
9		12 Веніамін Гресь Русланович	6
10		13 Орина Канівець Францівна	6
11		14 Василь Семенченко Прохорович	5
12		15 Хома Бевзенко Варфоломійович	4
13		16 Марія Чміль Трохимівна	4
14		17 Вікторія Туркало Омелянівна	5
15		18 Олекса Єщенко Федорович	1
16		19 Святослава Дурдинець Данилівна	7

Рисунок 7.3.5.1 – Приклад роботи функції get_officer_resolution_count

7.3.6 Функція get_citizen_full_name

```
CREATE OR REPLACE FUNCTION
```

```
get_citizen_full_name(citizen_id INT)
```

```
RETURNS VARCHAR AS
```

```
$$
```

```
DECLARE
```

```
full_name VARCHAR;
```

```
BEGIN
```

```
    SELECT first_name || ' ' || last_name || COALESCE(' ' ||
```

```
patronymic, "")
```

```
        INTO full_name
```

```
    FROM citizens
```

```
    WHERE id = citizen_id;
```

```
    RETURN full_name;
```

```
END;
```

```
$$ LANGUAGE plpgsql;
```

На рис. 7.3.6.1 наведено приклад роботи функції `get_citizen_full_name`.

		id	get_citizen_full_name
1		1	Максим Вишняк Костянтинович
2		2	Теодор Гоголь Венедиктович
3		3	Артем Захарченко Мартинович
4		4	Ярема Бевзенко Максимович
5		5	Іван Баранник Геннадійович
6		6	Камілла Дубас Дем'янівна
7		7	Адам Литвин Симонович
8		8	Марта Кириленко Ігнатівна
9		9	Михайло Баранець Трохимович
10		10	Михайлина Свистун Дем'янівна
11		11	Володимир Шухевич Дем'янович
12		12	Веніямін Гресь Русланович
13		13	Орина Канівець Францівна
14		14	Василь Семенченко Прохорович
15		15	Хома Бевзенко Варфоломійович
16		16	Марія Чміль Трохимівна
17		17	Вікторія Туркало Омелянівна

Рисунок 7.3.6.1 – Приклад роботи функції `get_citizen_full_name`

7.3.7 Функція `get_plate_type`

```
CREATE OR REPLACE FUNCTION
```

```
get_plate_type(registration_number VARCHAR)
```

```
RETURNS VARCHAR AS
```

```
$$
```

```
BEGIN  
    IF registration_number ~ '^[A-Z]{2}[0-9]{4}[A-Z]{2}$' THEN  
        RETURN 'regular';  
    ELSIF LENGTH(registration_number) BETWEEN 3 AND 8  
    THEN  
        RETURN 'personalized';  
    ELSE  
        RETURN 'unknown';  
    END IF;  
END;  
$$ LANGUAGE plpgsql;
```

На рис. 7.3.7.1 наведено приклад роботи функції get_plate_type.

functions.sql - functions.sql

Result 1

	<input type="checkbox"/> id	<input type="checkbox"/> registration_number	<input type="checkbox"/> get_plate_type
1	1	CC4568T0	regular
2	2	BI3952XC	regular
3	3	002119TP	regular
4	4	XM3416HB	regular
5	5	XO5097TA	regular
6	6	XI4643HO	regular
7	7	TT3442PX	regular
8	8	BA8563HE	regular
9	9	A03941XI	regular
10	10	HX2494IE	regular
11	11	JLY97YP0	personalized
12	12	AI0619CO	regular
13	13	XM8848CA	regular
14	14	HK1939BH	regular
15	15	KT5593AI	regular
16	16	IH3974EE	regular
17	17	AX4558XM	regular
18	18	VXQ8708	personalized
19	19	AE0997BE	regular
20	20	AX4840IK	regular
21	21	80T	personalized
22	22	HC3690XA	regular
23	23	XE8457OA	regular
24	24	XT3722HK	regular
25	25	HC7546CO	regular
26	26	OQADAQ25	personalized
27	27	HM2489EX	regular
28	28	BB4905CC	regular

Рисунок 7.3.7.1 – Приклад роботи функції get_plate_type

7.3.8 Функція get_total_penalty_fees

```
CREATE OR REPLACE FUNCTION
get_total_penalty_fees(driver_id INT)
    RETURNS DECIMAL AS
$$
DECLARE
    total_fees DECIMAL;
BEGIN
    SELECT SUM(ao.penalty_fee)
    INTO total_fees
    FROM violations v
    JOIN vehicles veh ON v.vehicle_id = veh.id
    JOIN administrative_offenses ao ON v.administrative_offense_id =
        ao.id
    WHERE veh.owner_id = driver_id;

    RETURN COALESCE(total_fees, 0);
END;
$$ LANGUAGE plpgsql;
```

На рис. 7.3.8.1 наведено приклад роботи функції `get_total_penalty_fees`

functions.sql - functions.sql

Result 3 ×

1	1 Максим Вишняк Костянтинович		93840
2	2 Теодор Гоголь Венедиктович		850
3	3 Артем Захарченко Мартинович		3740
4	4 Ярема Бевзенко Максимович		0
5	5 Іван Баранник Геннадійович		9180
6	6 Камілла Дубас Дем'янівна		0
7	7 Адам Литвин Симонович		0
8	8 Марта Кириленко Ігнатівна		0
9	9 Михайло Баранець Трохимович		42313
10	10 Михайліна Свистун Дем'янівна		0
11	11 Володимир Шухевич Дем'янович		0
12	12 Веніямін Гресь Русланович		1615
13	13 Орина Канівець Францівна		41140
14	14 Василь Семенченко Прохорович		42245
15	15 Хома Бевзенко Варфоломійович		0
16	16 Марія Чміль Трохимівна		1088
17	17 Вікторія Туркало Омелянівна		1700
18	18 Олекса Єщенко Федорович		37145
19	19 Святослава Дурдинець Данилівна		0
20	20 Ярослав Артимович Миколайович		0
21	21 Анастасія Якименко Вікторівна		0
22	22 Богуслава Черненко Євгенівна		1955
23	23 Зиновій Кальченко Орхипович		0
24	24 Алевтин Ґерус Миколайович		153

Рисунок 7.3.8.1 – Приклад роботи функції get_total_penalty_fees

7.3.9 Функція get_vehicle_owner

```

CREATE OR REPLACE FUNCTION get_vehicle_owner(vehicle_id
INT)
RETURNS VARCHAR AS
$$
DECLARE
    owner_name VARCHAR;

```

```
BEGIN  
    SELECT get_citizen_full_name(c.id)  
    INTO owner_name  
    FROM vehicles v  
    JOIN citizens c ON v.owner_id = c.id  
    WHERE v.id = vehicle_id;  
  
    RETURN owner_name;  
END;  
$$ LANGUAGE plpgsql;
```

На рис. 7.3.9.1 наведено приклад роботи функції `is_citizen_older_than`.

functions.sql - functions.sql

Result 4

	<input type="checkbox"/> id	<input type="checkbox"/> registration_...	<input type="checkbox"/> get_vehicle_owner
1		1 CC4568TO	Едита Чарниш Леопольдівна
2		2 BI3952ХС	Панас Рева Зиновійович
3		3 002119TP	Орест Хмара Юхимович
4		4 XM3416HB	Василина Полтавець Тарасівна
5		5 X05097TA	Богдан Дем'янчук Орхипович
6		6 XI4643НО	Ганна Стельмах Вадимівна
7		7 TT3442PX	Ада Остапчук Андріївна
8		8 BA8563НЕ	Теодор Стець Ігнатович
9		9 A03941XI	Онисим Башполченко Леопольдович
10		10 HX2494IE	Альбіна Пелех Артемівна
11		11 JLY97YP0	Василина Худяк Едуардівна
12		12 AI0619CO	Степан Бевз Аврелійович
13		13 XM8848CA	Охрім Юрчишин Олесьович
14		14 HK1939BH	Розалія Зінченко Олексandrівна
15		15 KT5593AI	Едита Яремків Іgnatіvna
16		16 IH3974EE	Юхим Голяш Петрович
17		17 AX4558XM	Михайлина Швачко Петрівна
18		18 VXQ8708	Ірина Чуприна Остапівна
19		19 AE0997BE	Марина Баклан Трохимівна
20		20 AX4840IK	Гордій Їжакевич Леонтійович
21		21 80T	Віра Швачко Стефанівна
22		22 HC3690XA	Наталія Засядько Ааронівна

Рисунок 7.3.9.1 – Приклад роботи функції get_vehicle_owner

7.3.10 Функція get_total_violations_for_vehicle

CREATE OR REPLACE FUNCTION

get_total_violations_for_vehicle(veh_id INT)

RETURNS INT AS \$\$

DECLARE

total_violations INT;

```
BEGIN  
    SELECT COUNT(*)  
        INTO total_violations  
        FROM violations  
        WHERE violations.vehicle_id = veh_id;  
  
    RETURN total_violations;  
END;  
$$ LANGUAGE plpgsql;
```

На рис. 7.3.10.1 наведено приклад роботи функції
get_total_violations_for_vehicle.

functions.sql - functions.sql

Result 5 ×

The screenshot shows a database query result titled "functions.sql - functions.sql" with a sub-tab "Result 5 ×". The results are displayed in a grid with three columns: "id", "registration_number", and "get_total_violations_for_vehicle". The data consists of 17 rows, each containing an ID number from 1 to 17, a registration number, and a count of violations. The registration numbers are CC4568TO, BI3952XC, 002119TP, XM3416HB, X05097TA, XI4643HO, TT3442PX, BA8563HE, AO3941XI, HX2494IE, JLY97YP0, AI0619CO, XM8848CA, HK1939BH, KT5593AI, IH3974EE, and AX4558XM. The violation counts are 1, 0, 2, 0, 3, 1, 1, 2, 2, 1, 0, 0, 0, 2, 4, 1, and 3 respectively.

	id	registration_number	get_total_violations_for_vehicle
1	1	CC4568TO	1
2	2	BI3952XC	0
3	3	002119TP	2
4	4	XM3416HB	0
5	5	X05097TA	3
6	6	XI4643HO	1
7	7	TT3442PX	1
8	8	BA8563HE	2
9	9	AO3941XI	2
10	10	HX2494IE	1
11	11	JLY97YP0	0
12	12	AI0619CO	0
13	13	XM8848CA	0
14	14	HK1939BH	2
15	15	KT5593AI	4
16	16	IH3974EE	1
17	17	AX4558XM	3

Рисунок 7.3.10.1 – Приклад роботи функції
get_totalViolationsForVehicle

7.3.11 Процедура transfer_vehicle_ownership

```
CREATE OR REPLACE PROCEDURE transfer_vehicle_ownership(
    vehicle_id INT,
    new_owner_id INT
)
LANGUAGE plpgsql
AS $$

DECLARE
    current_owner_id INT;
BEGIN
```

```

SELECT owner_id INTO current_owner_id
FROM vehicles
WHERE id = vehicle_id;

IF NOT FOUND THEN
    RAISE EXCEPTION 'Vehicle with ID % does not exist.',

vehicle_id;
END IF;

IF NOT EXISTS (
    SELECT 1 FROM citizens WHERE id = new_owner_id
) THEN
    RAISE EXCEPTION 'New owner with ID % does not exist.',

new_owner_id;
END IF;

UPDATE vehicles
SET owner_id = new_owner_id
WHERE id = vehicle_id;

RAISE NOTICE 'Ownership of vehicle ID % transferred to citizen
ID %', vehicle_id, new_owner_id;

END;
$$;

```

На рис. 7.3.11.1 - 7.3.11.3 показано работу процедуры transfer_vehicle_ownership.

	<input type="checkbox"/> id	<input type="checkbox"/> owner_id	<input type="checkbox"/> vehicle_type	<input type="checkbox"/> registration_no	<input type="checkbox"/> vin	<input type="checkbox"/> insurance_policy	<input type="checkbox"/> model	<input type="checkbox"/> brand	<input type="checkbox"/> color	<input type="checkbox"/> engine_capacity
1	1	13799	IS F	Lexus	Темно-королівий	2.42				

Рисунок 7.3.11.1 – Запис про автомобіль до виклику процедури

procedures.sql - procedures.sql										
<input type="checkbox"/> Output										
<input type="checkbox"/>	course_work.public> CALL transfer_vehicle_ownership(1, 2)									
<input type="checkbox"/>	Ownership of vehicle ID 1 transferred to citizen ID 2									
<input type="checkbox"/>	[2024-12-22 18:42:11] completed in 6 ms									

Рисунок 7.3.11.2 – Повідомлення про успішне виконання процедури

	<input type="checkbox"/> id	<input type="checkbox"/> owner_id	<input type="checkbox"/> vehicle_type	<input type="checkbox"/> registration_no	<input type="checkbox"/> vin	<input type="checkbox"/> insurance_policy
1	1	2	IS F	Lexus	Темно-королівий	KHK435576

Рисунок 7.3.11.3 – Запис про автомобіль після виклику процедури

7.3.12 Процедура registerViolation

```
CREATE OR REPLACE PROCEDURE registerViolation(
    p_vehicle_id INT,
    p_location_id INT,
    p_administrative_offense_id INT,
    p_traffic_rule_id INT,
    p_time_ofViolation TIMESTAMPTZ,
    p_description TEXT,
    p_evidence_type EVIDENCE_TYPE DEFAULT NULL,
    p_evidence_url TEXT DEFAULT NULL)
```

```

)
LANGUAGE plpgsql
AS $$

DECLARE
    violation_id INT;

BEGIN
    IF NOT EXISTS (
        SELECT 1 FROM vehicles WHERE id = p_vehicle_id
    ) THEN
        RAISE EXCEPTION 'Vehicle with ID % does not exist.',

p_vehicle_id;
    END IF;

    INSERT INTO violations (
        vehicle_id, location_id, administrative_offense_id,
        traffic_rule_id,
        time_ofViolation, description
    )
    VALUES (
        p_vehicle_id, p_location_id, p_administrative_offense_id,
        p_traffic_rule_id,
        p_time_ofViolation, p_description
    )
    RETURNING id INTO violation_id;

    IF p_evidence_type IS NOT NULL AND p_evidence_url IS NOT
NULL THEN
        INSERT INTO evidences (violation_id, type, url)
        VALUES (violation_id, p_evidence_type, p_evidence_url);
    END IF;

```

```

RAISE NOTICE 'Violation ID % registered successfully.',  

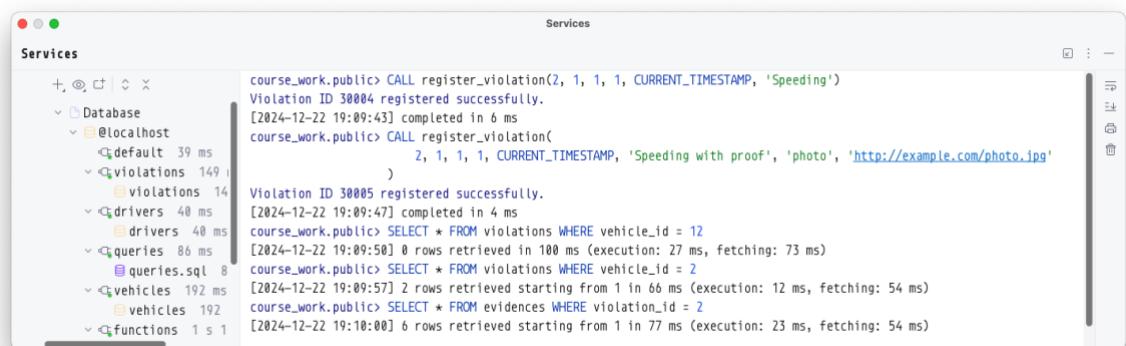
violation_id;  

END;  

$$;

```

На рис. 7.3.12.1 - 7.3.12.3 показано роботу процедурі transfer_vehicle_ownership.

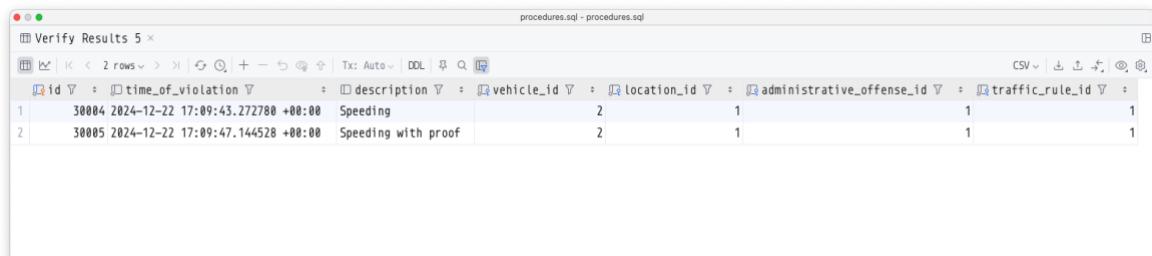


```

Services
course_work.public> CALL registerViolation(2, 1, 1, 1, CURRENT_TIMESTAMP, 'Speeding')
Violation ID 30004 registered successfully.
[2024-12-22 19:09:43] completed in 6 ms
course_work.public> CALL registerViolation(
    2, 1, 1, 1, CURRENT_TIMESTAMP, 'Speeding with proof', 'photo', 'http://example.com/photo.jpg'
)
Violation ID 30005 registered successfully.
[2024-12-22 19:09:47] completed in 4 ms
course_work.public> SELECT * FROM violations WHERE vehicle_id = 12
[2024-12-22 19:09:50] 0 rows retrieved in 100 ms (execution: 27 ms, fetching: 73 ms)
course_work.public> SELECT * FROM violations WHERE vehicle_id = 2
[2024-12-22 19:09:57] 2 rows retrieved starting from 1 in 66 ms (execution: 12 ms, fetching: 54 ms)
course_work.public> SELECT * FROM evidences WHERE violation_id = 2
[2024-12-22 19:10:00] 6 rows retrieved starting from 1 in 77 ms (execution: 23 ms, fetching: 54 ms)

```

Рисунок 7.3.12.1 – Виклик процедури register_violation 2 рази



	<code>id</code>	<code>time_ofViolation</code>	<code>description</code>	<code>vehicle_id</code>	<code>location_id</code>	<code>administrative_offense_id</code>	<code>traffic_rule_id</code>
1	30004	2024-12-22 17:09:43.272780 +00:00	Speeding	2	1	1	1
2	30005	2024-12-22 17:09:47.144528 +00:00	Speeding with proof	2	1	1	1

Рисунок 7.3.12.1 – Результат виклику процедури без опціональних параметрів

	<code>id</code>	<code>violation_id</code>	<code>type</code>	<code>url</code>
1	132632	30005	photo	http://example.com/photo.jpg

Рисунок 7.3.12.1 – Результат виклику процедури із опціональними параметрами

7.4 SQL-запити

7.4.1 Перелік всіх громадян та їх водійських посвідчень

```
SELECT citizens.first_name,
       citizens.last_name,
       drivers.license_number,
       drivers.license_issued_time
  FROM citizens
 LEFT JOIN drivers ON citizens.id = drivers.citizen_id;
```

На рис. 7.4.1.1 наведено результати роботи запиту, який повертає всіх громадян та їх водійські посвідчення.

queries.sql - queries.sql

Перелік всіх громадянсько-водійських посвідчень ×

first_name ▾ last_name ▾ license_number ▾ license_issued_time ▾

	first_name	last_name	license_number	license_issued_time
1	Максим	Вишняк	XEO670013	2013-11-30 12:23:34.000000 +00:00
2	Теодор	Гоголь	НАН829410	2014-07-18 05:48:51.000000 +00:00
3	Артем	Захарченко	<null>	<null>
4	Ярема	Бевзенко	<null>	<null>
5	Іван	Баранник	<null>	<null>
6	Камілла	Дубас	EPE193898	2012-10-07 22:06:59.000000 +00:00
7	Адам	Литвин	<null>	<null>
8	Марта	Кириленко	<null>	<null>
9	Михайло	Баранець	<null>	<null>
10	Михайлина	Свистун	<null>	<null>
11	Володимир	Шухевич	<null>	<null>
12	Веніамін	Гресь	CEK480618	2006-01-24 15:18:51.000000 +00:00
13	Орина	Канівець	XOE354304	2008-01-23 08:15:15.000000 +00:00
14	Василь	Семенченко	<null>	<null>
15	Хома	Бевзенко	<null>	<null>
16	Марія	Чміль	KPX181501	2009-04-07 15:19:20.000000 +00:00
17	Вікторія	Туркало	BEH621983	2021-08-17 04:33:54.000000 +00:00
18	Олекса	Єщенко	<null>	<null>
19	Святослава	Дурдинець	<null>	<null>
20	Ярослав	Артимович	OIM290082	2007-04-28 16:45:19.000000 +00:00
21	Анастасія	Якименко	TKB191453	2007-04-06 12:47:58.000000 +00:00
22	Богуслава	Черненко	MCC416634	2018-07-13 07:13:49.000000 +00:00
23	Зиновій	Кальченко	ECO191288	2023-03-31 10:03:45.000000 +00:00
24	Алевтин	Герус	HOT591760	2001-07-12 16:24:51.000000 +00:00
25	Оксенія	Герета	<null>	<null>
26	Леонід	Давимука	XCO419306	2012-08-12 03:43:57.000000 +00:00
27	Артем	Рябовіл	<null>	<null>
28	Борис	Василенко	TEH106558	2013-12-05 05:54:52.000000 +00:00
29	Ігор	Нестайко	<null>	<null>
30	Альбіна	Пушкар	BAI200569	2007-11-13 12:13:34.000000 +00:00
31	Марта	Редько	<null>	<null>
32				

Рисунок 7.4.1.1 – Результати роботи запиту, який повертає всіх громадян та їх водійські посвідчення

7.4.2 Перелік протоколів для певного громадянина

SELECT

```
citizens.first_name,
citizens.last_name,
accident_protocols.id,
accident_protocols.series,
```

```
accident_protocols.number,  
accident_protocols.defendant_explanation,  
accident_protocols.time_of_drawing_up,  
accident_protocols.violation_id,  
accident_protocols.police_officer_id,  
accident_protocols.defendant_id  
FROM  
accident_protocols  
JOIN citizens_on_protocol ON accident_protocols.id =  
citizens_on_protocol.protocol_id  
JOIN citizens ON citizens_on_protocol.citizen_id = citizens.id  
WHERE  
citizens.first_name = 'Володимир'  
AND citizens.last_name = 'Тарасенко';
```

На рис. 7.4.2.1 наведено результати роботи запиту, який повертає всі протоколи заданого громадянина.

queries.sql - queries.sql

Протоколи для певного громадянина

□ first_name	Володимир	
□ last_name	Тарасенко	
□ id	6268	
□ series	XI	
□ number	496073	
□ defendant_explanation	<null>	
□ time_of_drawing_up	2018-09-09 04:02:54.000000 +00:00	
□ violation_id	7572	
□ police_officer_id	606	
□ defendant_id	6496	

Рисунок 7.4.2.1 – Результати роботи запиту, який повертає всіх громадян та їх водійські посвідчення

7.4.3 Перелік поліцейських, які склали найбільше протоколів та постанов

```
WITH officer_counts AS (
    SELECT
        police_officers.id AS officer_id,
        get_citizen_full_name(police_officers.citizen_id) AS full_name,
        police_officers.rank,
        police_officers.badge_number,
        get_officer_protocol_count(police_officers.id) AS
protocol_count,
        get_officer_resolution_count(police_officers.id) AS
resolution_count
```

```
FROM
    police_officers
)
SELECT
    full_name,
    rank,
    badge_number,
    protocol_count,
    resolution_count
FROM
    officer_counts
ORDER BY
    protocol_count DESC,
    resolution_count DESC;
```

На рис. 7.4.3.1 наведено результати роботи запиту, який повертає перелік поліцейських, які склали найбільше протоколів та постанов.

The screenshot shows a MySQL Workbench interface with a results grid titled "queries.sql - queries.sql". The grid displays the following columns: full_name, rank, badge_number, protocol_count, and resolution_count. The data consists of 28 rows, each representing a police officer. The columns are ordered by resolution_count in descending order.

	full_name	rank	badge_number	protocol_count	resolution_count
1	Вадим Червоненко Федорович	sergeant	POX510705	16	8
2	Анастасія Онищенко Назарівна	lieutenant_colonel	CAP633057	16	5
3	Дан Засядько Ростиславович	senior_sergeant	AII197349	15	5
4	Камілла Шахрай Феофанівна	junior_lieutenant	OPB053865	15	4
5	Онисим Оробець Глібович	captain	OHE846493	15	2
6	Ігнат Лемешко Пилипович	senior_lieutenant	HTP280385	14	6
7	Аніта Валенка Тарасівна	colonel	BTE133898	14	3
8	Одарка Цимбалюк Тарасівна	sergeant	TXP495765	14	0
9	Ілля Девдюк Валерійович	senior_sergeant	HAK467659	13	7
10	Степан Перебійніс Теодорович	captain	CMI925512	13	5
11	Альбіна Верменич Геннадіївна	senior_sergeant	BKM689978	13	4
12	Августин Щицьо Давидович	sergeant	BHX422346	13	4
13	Леопольд Запорожець Августинович	senior_lieutenant	TAM081187	13	3
14	Аркадій Гавриленко Богданович	lieutenant	ECX469231	13	2
15	Леон Даньків Захарович	senior_sergeant	KXH777326	12	9
16	Герман Кабалюк Омелянович	senior_lieutenant	TME473341	12	8
17	Соломія Чумаченко Дмитрівна	senior_sergeant	XCH735793	12	7
18	Емілія Вертипорх Романівна	sergeant	AAO468347	12	7
19	Алла Гузь Георгіївна	senior_lieutenant	EOB891895	12	7
20	Даром Шаповал Леонідовна	senior_sergeant	TTCR74455	12	7

Рисунок 7.4.3.1 – Результати роботи запиту, який повертає перелік поліцейських, які склали найбільше протоколів та постанов

7.4.4 Перелік постанов, де автомобіль знаходиться у власності поліцейського

SELECT

```
accident_resolutions.id AS resolution_id,
accident_resolutions.series,
accident_resolutions.number,
accident_resolutions.time_of_consideration,
accident_resolutions.time_of_entry_into_force,
get_citizen_full_name(police_officers.citizen_id) AS officer_full_name,
vehicles.registration_number,
vehicles.vin
```

FROM

```
accident_resolutions
```

```
JOIN vehicles ON accident_resolutions.violation_id = vehicles.id
```

```
JOIN police_officers ON vehicles.owner_id = police_officers.citizen_id
```

ORDER BY

```
accident_resolutions.time_of_consideration DESC;
```

На рис. 7.4.4.1 наведено результати роботи запиту, який повертає перелік постанов, де автомобіль знаходиться у власності поліцейського

постанови, де автомобіль знаходиться у власності поліцейського									
	resolution_id	series	number	time_of_consideration	time_of_entry_into_force	officer_full_name	registration_number	vln	
1	12189 BD	792389	2024-12-21 11:06:45.000000 +0:00	2025-01-12 17:58:29.000000 +0:00	Гліб Рудик Станіславович	BB88129KA	KURFBEB8BHPZ5WAN		
2	12445 OH	665555	2024-12-19 18:54:52.000000 +0:00	2025-01-19 07:18:25.000000 +0:00	Хома Іненко Кlementович	IE9408XX	XH97GANNX1JGL1HGU		
3	11641 MB	548923	2024-12-17 06:24:35.000000 +0:00	2025-01-11 17:42:37.000000 +0:00	Златослава Юрчшин Соломонівна	BC5511TP	LHCYUCG526WADVP36		
4	3558 AI	158693	2024-12-14 15:31:41.000000 +0:00	2025-01-03 23:05:38.000000 +0:00	Ефросій Алексік Семенович	XI8752AP	IB37RVA3502GBVB13		
5	12164 TT	861438	2024-12-12 14:57:08.000000 +0:00	2025-01-17 20:58:14.000000 +0:00	Венедикт Сагаль Адамович	AC00080E	0772NHRV575D0B5Z		
6	5380 MP	168969	2024-12-09 08:09:48.000000 +0:00	2024-12-11 13:32:41.000000 +0:00	Клавдія Дейсун Захаріївна	AB9428OB	YBZ6Mw0XPX5W54V4B		
7	9859 PI	142382	2024-12-04 02:27:54.000000 +0:00	2025-01-18 19:59:16.000000 +0:00	Анаїз Ванчук Альбертіна	AA1988OC	MNPKG312V7JLGLW		
8	7949 EE	487747	2024-11-28 11:29:20.000000 +0:00	2025-01-08 05:02:43.000000 +0:00	Захар Гавриленко Аврелийович	XI737250A	WNEZB53B2PWAy1DU		
9	1585 ZM	589323	2024-11-27 07:06:43.000000 +0:00	2024-12-04 22:43:35.000000 +0:00	Гліб Рудик Станіславович	BB88129KA	KURFBEB8BHPZ5WAN		
10	12385 XX	241748	2024-11-27 04:57:39.000000 +0:00	2025-01-15 17:36:35.000000 +0:00	Святослав Радченко Геннадійович	KC5594TP	4ZMD595373ABP3Y5		
11	4279 KA	248457	2024-11-24 04:19:43.000000 +0:00	2024-11-07 05:24:27.000000 +0:00	Валентин Негода Демидович	AХ7642EH	K1A8F1TGBE8BHN4NO		
12	1653 KM	649223	2024-11-23 20:43:07.000000 +0:00	2024-12-07 08:44:04.000000 +0:00	Остан Голіш Эмілович	НН8821TX	CPVVT79UJFHFGZEKA		
13	3888 MT	269811	2024-11-19 23:27:58.000000 +0:00	2024-11-24 13:17:28.000000 +0:00	Валентин Негода Демидович	AХ7642EH	K1A8F1TGBE8BHN4NO		
14	4877 OO	427587	2024-11-17 15:54:32.000000 +0:00	2025-01-17 22:46:27.000000 +0:00	Ева Грищенко Єлизавета	AP2527EX	YESUNFAK82H4GEYT		
15	18287 KO	787983	2024-11-15 10:12:06.000000 +0:00	2025-01-09 19:33:46.000000 +0:00	Остап Голіш Эмілович	НН8821TX	CPVVT79UJFHFGZEKA		
16	1765 CO	384532	2024-11-13 08:57:32.000000 +0:00	2024-11-27 10:58:15.000000 +0:00	Нестор Михайличенок Остимович	HC4246TB	V25GPDXY3UDNY7ZH		
17	8582 BD	276802	2024-11-12 09:42:27.000000 +0:00	2025-01-11 03:31:34.000000 +0:00	Ева Грищенко Єлизавета	AP2527EX	YESUNFAK82H4GEYT		
18	7788 HI	685251	2024-11-11 15:41:44.000000 +0:00	2024-11-07 16:29:59.000000 +0:00	Гліб Рудик Станіславович	BB88129KA	KURFBEB8BHPZ5WAN		
19	7684 OC	943818	2024-11-06 07:00:53.000000 +0:00	2024-12-10 23:43:33.000000 +0:00	Борислав Данченко Арсенович	005287MK	XE215C9P73GYZPNN8		
20	8458 EB	915888	2024-11-05 23:28:54.000000 +0:00	2024-12-26 04:17:24.000000 +0:00	АЗаз Забіла Денидович	HK3821HC	TXPMF2B21B2DHFZ		
21	1748 EA	583284	2024-11-02 19:38:32.000000 +0:00	2024-11-29 08:24:15.000000 +0:00	Артем Гуляяк Орестович	A01513AX	7PQYCXM5L5RZDNLK		
22	6684 TK	525108	2024-11-01 04:52:49.000000 +0:00	2024-11-14 23:12:19.000000 +0:00	Степан Дунін Наразович	HT9654TH	VH04W3D2FEA2FVY		
23	18990 MI	169542	2024-10-30 07:42:17.000000 +0:00	2024-12-24 14:36:18.000000 +0:00	Ева Удовиченко Артеміна	AI1819CB	SDCRUD0B1016T6LRD		
24	826 BH	688585	2024-10-29 08:57:29.000000 +0:00	2024-12-19 19:16:19.000000 +0:00	Ірина Зубко Романіана	H08952DH	TLBYFVG8L8XPX368M		
25	573 AC	816244	2024-10-27 15:37:42.000000 +0:00	2025-01-20 02:38:38.000000 +0:00	Галина Архипенко Франціана	CB1940EI	8X3WPFJL0L21F5N4		

Рисунок 7.4.4.1 – Результати роботи запиту, який повертає перелік постанов, де автомобіль знаходиться у власності поліцейського

7.4.5 Перелік громадян, які порушували ПДР найчастіше за останній рік

SELECT

citizens.id,

get_citizen_full_name(citizens.id),

COUNT(violations.id)

FROM

citizens

JOIN vehicles ON citizens.id = vehicles.owner_id

JOIN violations ON vehicles.id = violations.vehicle_id

WHERE

violations.time_ofViolation >= CURRENT_DATE - INTERVAL '1 year'

GROUP BY

citizens.id

ORDER BY

COUNT(violations.id) DESC

LIMIT

10;

На рис. 7.4.4.1 наведено результати роботи запиту, який повертає перелік громадян, які порушували ПДР найчастіше за останній рік.

1	8504	Єва Теліженко Демидівна	4
2	5807	Юстина Семенченко Ігорівна	3
3	5478	Марина Корбут Гаврилівна	3
4	16925	Святослава Єременко Петрівна	3
5	8867	Аркадій Чередник Прохорович	3
6	12263	Панас Рева Зиновійович	3
7	16323	Тереза Чаленко Леонідівна	3
8	2	Теодор Гоголь Венедиктович	3
9	270	Зорян Лукаш Романович	3
10	17175	Сніжана Фесенко Несторівна	3

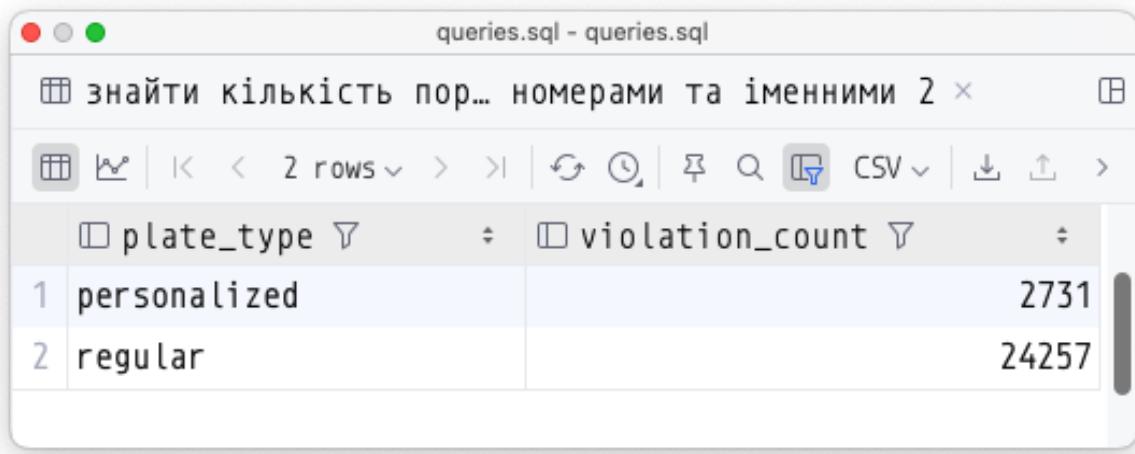
Рисунок 7.4.5.1 – Результати роботи запиту, який повертає перелік громадян, які порушували ПДР найчастіше за останній рік

7.4.6 Кількість порушників та порушень, які скоєні на автомобілях із звичайними номерами та іменними

```
WITH categorized_vehicles AS (
    SELECT
        vehicles.id,
        vehicles.registration_number,
        get_plate_type(vehicles.registration_number) AS plate_type
    FROM
        vehicles
```

```
),
violations_summary AS (
    SELECT
        categorized_vehicles.plate_type,
        COUNT(violations.id) AS violation_count
    FROM
        categorized_vehicles
    JOIN violations ON categorized_vehicles.id = violations.vehicle_id
    WHERE
        categorized_vehicles.plate_type IN ('regular', 'personalized')
    GROUP BY
        categorized_vehicles.plate_type
)
SELECT
    plate_type,
    violation_count
FROM
    violations_summary;
```

На рис. 7.4.6.1 наведено результати роботи запиту, який повертає кількість порушників та порушень, які сконцентровані на автомобілях із звичайними номерами та іменними.



The screenshot shows a window titled "queries.sql - queries.sql" containing a table with two rows. The columns are labeled "plate_type" and "violation_count". The first row has "personalized" in the first column and "2731" in the second. The second row has "regular" in the first column and "24257" in the second.

	plate_type	violation_count
1	personalized	2731
2	regular	24257

Рисунок 7.4.6.1 – Результати роботи запиту, який повертає кількість порушників та порушень, які скоєні на автомобілях із звичайними номерами та іменними

7.4.7 Кількість порушень по регіону реєстрації машини

SELECT

```

get_region_by_code(
    SUBSTRING(
        vehicles.registration_number
    FROM
        1 FOR 2
    )
),
COUNT(violations.id)
FROM
violations
JOIN vehicles ON violations.vehicle_id = vehicles.id
WHERE
LENGTH(vehicles.registration_number) = 8
AND SUBSTRING(
    vehicles.registration_number

```

```
FROM
    1 FOR 2
) IN (
    SELECT
        DISTINCT LEFT(code_2004, 2)
    FROM
        regions
    UNION
    SELECT
        DISTINCT LEFT(code_2013, 2)
    FROM
        regions
    UNION
    SELECT
        DISTINCT LEFT(code_2021, 2)
    FROM
        regions
)
GROUP BY
    get_region_by_code(
        SUBSTRING(
            vehicles.registration_number
        FROM
            1 FOR 2
        )
    )
ORDER BY
    COUNT(violations.id) DESC;
```

На рис. 7.4.7.1 наведено результати роботи запиту, який повертає кількість порушень по регіону реєстрації машини.

queries.sql - queries.sql

знайти кількість пор...ону реєстрації машини

	get_region_by_code	count
1	Cherkaska oblast	990
2	Luhanska oblast	967
3	Kyivska oblast	965
4	Donetska oblast	958
5	Ternopilska oblast	953
6	Ivano-Frankivska oblast	938
7	Lvivska oblast	937
8	Volynska oblast	932
9	Khersonska oblast	921
10	Mykolaivska oblast	915
11	Kharkivska oblast	914
12	Chernihivska oblast	907
13	Sumska oblast	905
14	Chernivetska oblast	905
15	Dnipropetrovska oblast	900
16	Rivnenska oblast	900
17	AR Krym	895
18	Zaporizka oblast	893
19	Odeska oblast	885
20	Zhytomyrska oblast	885
21	Zakarpatska oblast	875
22	Khmelnitska oblast	862
23	Poltavska oblast	856
24	misto Sevastopol	829
25	Kirovohradska oblast	806
26	Vinnytska oblast	802
27	misto Kyiv	795

Рисунок 7.4.7.1 – Результати роботи запиту, який повертає кількість порушень по регіону реєстрації машини

7.4.8 Кількість порушень машинами із та без страхового полісу

```

SELECT
    CASE
        WHEN vehicles.insurance_policy_number IS NOT NULL THEN 'With
Insurance'
        ELSE 'Without Insurance'
    END AS insurance_status,
    COUNT(violations.id) AS violation_count
FROM
    vehicles
    LEFT JOIN violations ON vehicles.id = violations.vehicle_id
GROUP BY
    CASE
        WHEN vehicles.insurance_policy_number IS NOT NULL THEN 'With
Insurance'
        ELSE 'Without Insurance'
    END
ORDER BY
    violation_count DESC;

```

На рис. 7.4.8.1 наведено результати роботи запиту, який повертає кількість порушень машинами із та без страхового полісу.

The screenshot shows a software interface for viewing database query results. The title bar says 'queries.sql - queries.sql'. The main area displays a table with two rows. The columns are labeled 'insurance_status' and 'violation_count'. The first row, 'With Insurance', has a value of 26984. The second row, 'Without Insurance', has a value of 4.

	insurance_status	violation_count
1	With Insurance	26984
2	Without Insurance	4

Рисунок 7.4.8.1 – Результати роботи запиту, який повертає кількість порушень машинами із та без страхового полісу

7.4.9 Перелік транспортних засобів, що порушили ПДР за останній місяць

SELECT

```

full_vehicle_info.vehicle_id,
full_vehicle_info.registration_number,
full_vehicle_info.model,
full_vehicle_info.brand,
full_vehicle_info.color,
full_vehicle_info.year_of_manufacture,
full_vehicle_info.vin,
full_vehicle_info.insurance_policy_number,
full_vehicle_info.engine_capacity,
full_vehicle_info.seating_capacity,
full_vehicle_info.owner_full_name,
full_vehicle_info.region,
COUNT(violations.id) AS violation_count

```

FROM

```
full_vehicle_info
JOIN violations ON full_vehicle_info.vehicle_id = violations.vehicle_id
WHERE
    violations.time_ofViolation >= CURRENT_DATE - INTERVAL '1
month'
GROUP BY
    full_vehicle_info.vehicle_id,
    full_vehicle_info.registration_number,
    full_vehicle_info.model,
    full_vehicle_info.brand,
    full_vehicle_info.color,
    full_vehicle_info.year_of_manufacture,
    full_vehicle_info.vin,
    full_vehicle_info.insurance_policy_number,
    full_vehicle_info.engine_capacity,
    full_vehicle_info.seating_capacity,
    full_vehicle_info.owner_full_name,
    full_vehicle_info.region
ORDER BY
    violation_count DESC;
```

На рис. 7.4.9.1 наведено результати роботи запиту, який повертає перелік транспортних засобів, що порушили ПДР за останній місяць.

знайти перелік транс.ДР за останній місяць											
vehicle_id	registration_number	model	brand	color	year_of_manufacture	vin	insurance_policy_number	engine_capacity	seating_capacity	owner	
1	2 BE3952HC	Sedan	Chevrolet	Помаранчевий	2004	Y004AB4A9F1G3620A	BB050557	8,50	7 Павло Рева		
2	1515 ZQZC	C-MAX Hybrid	Ford	Бордо	2013	EB2B9772ZDZ9000W	P0710384	4,40	1 Арк Герауд		
3	18284 PR72TM	Continent	Lincoln	Лімузинний	1997	E1000000000000000000	H00301818	4,42	2 Микола Тарас		
4	1C1E000000000000	GT	Cessna	Темно-зелений	2000	EB2B9772ZDZ9000P	K0030179	2,42	3 Іванна Григор		
5	299 K02874HK	Sedona	Pontiac	Темно-сірий	1996	PS5GOT1U00026798	PA031745	8,10	2 Георгій Світ		
6	544 K02913BE	Sundance	Pontiac	Лісковий	1994	PS5GOT1U00026798	KA021742	6,00	5 Володимир Хар		
7	393 XE756R8B	Sedona	Kia	Сірий (ІКС)	2012	PF22538000000000	XH225395	6,00	6 Ольга Григор		
8	437 AH9898MP	Eos	Volkswagen	Бірюза білувата	2014	WPW9Y5747V7EN000C	BH162018	1,50	2 Надія Гумен		
9	685 9M925W00	XS	BMW	Темно-персиковий	2015	E4D29791F11111111	BR077095	8,71	1 Оксана Ярі		
10	723 1PMB7YKX	3500 Extended Cab	Chevrolet	Оливковий	1992	C1250K9194200000	AC1774078	9,44	5 Володимир С		
11	723 1PMB7YKX	Hevyduty	Mitsubishi	Мідністий	1992	CB1000000000000000000	AK031848	2,00	10 Марина Григор		
12	762 AA111AAZ	Stativa	Dodge	Темно-жасминово-зелений	1999	CCCCCCCCCCCCCCCCCCCC	CT234339	8,40	1 Елена Івон		
13	788 K02913HM	Delta	Volkswagen	Темно-зелений	2007	SNR741391326000	K0050015	5,36	2 Валентин І		
14	811 U02	Grand Vitara	Suzuki	Горчицевий	2004	GRV1000000000000000000	PA070986	2,10	4 Андрій Вітал		
15	837 0B058ME	QE	Infiniti	Лісковий	2007	VKR0000000000000000000	K0060471	4,70	2 Павло Урбан		
16	1818 TP7939HM	3 Series	BMW	Блакитно-блакитний	2012	KF0000000000000000000000	KAC09582	1,51	1 Наталія Гед		
17	1140 BK7951SD	Frontier King Cab	Nissan	Темно-зелений	2008	1G6T31400000000000000000	TK0637924	8,30	3 Святослав Вен		
18	1349 K02913HM	Yaris	Toyota	Білий	2012	GRY100000000000000000000	TK0637912	5,00	4 Ольга Григор		
19	1524 BP7494CP	Canyon Extended Cab	GM	Білий	2011	CH0700000000000000000000	BT030004	5,70	13 Альона Денис		
20	1525 NAVY4J2	C150	Cadillac	Темно-зелений	2012	PN0107000000000000000000	HP030002	6,00	1 Тетяна Кап		
21	1538 Q40300K	Pacifica	Chrysler	Білий	2008	512300000000000000000000	K10700208	3,10	5 Арина Хомя		
22	1541 TO4995K	SS	Chevrolet	Темно-блакитний	2010	NE0000000000000000000000	PK041402	12,70	6 Анастасія І		
23	1587 AE0481SH	Safari Cargo	GM	Блакитно-персиковий	2000	IC0400000000000000000000	AB056297	4,81	1 Олена Гед		
24	1872 DE5311CC	Outlander	Mitsubishi	Від сірих і біл.	2010	144542520100000000000000	TH0487231	5,47	3 Григорій Анд		
25	1994 K02913HM	Elantra GT	Hyundai	Пурпурний	2010	1G2B10000000000000000000	CR075427	5,10	5 Катерина Гед		
26	2140 K02913HM	TSX	Acura	Чорний	2010	1A0000000000000000000000	TK0637911	4,45	4 Ольга Григор		
27	2180 K02913HM	TSX Sporty Super Cab	GM	Зелений	1998	PA0000000000000000000000	DK070048	2,70	4 Олена Григор		
28	2215 K02913HM	Transit 108 Van	Ford	Сірий	2010	UD0515000000000000000000	BT0700000000000000000000	8,51	2 Марта Миха		
29	2252 HT4799M	NGV900 HD Corp	Mitsubishi	Бірюзовий	2015	78237N950000000000000000	HR0100377	14,20	1 Тетяна Кап		
30	2278 K02913HM	City Express	Chevrolet	Білий	2010	CV5702000000000000000000	BT0710000000000000000000	4,62	18 Марина Сагі		
31	2438 L15	Accent	Hyundai	Бірюзовий	2008	260400000000000000000000	DT0700000000000000000000	8,76	1 Борис Дани		
32	2479 K02913HM	Corvette	Chevrolet	Бірюзовий	2010	000000000000000000000000	XX0010000000000000000000	2,00	1 Ірина Івана		
33	2480 K02913HM	RSX	Acura	Чорний	1997	AD0000000000000000000000	MR0110000000000000000000	5,00	3 Оксана Григор		
34	2508 K02913HM	Coupe Regular Cab	GM	Сірий	2006	ZC0000000000000000000000	MR011212	4,20	1 Марина Ваг		
35	2540 XE7140C	Express 2500 Passenger	Chevrolet	Пурпурно-зелений	2008	540000000000000000000000	Z0050019	5,82	2 Дарина Ром		
36	2692 TP7949C1	Q	Infiniti	Лісковий	1998	FL0000000000000000000000	OKC030051	0,00	8 Марта Теті		
37	2731 K02913HM	Grand Prix	Pontiac	Темно-зелений	1995	3B0000000000000000000000	CIO010001	6,20	2 Богдан Єрем		
38	2739 HE2550HK	Tundra Double Cab	Toyota	Пісочний кольор	2017	TUC000000000000000000000	K007100000000000000000000	8,77	2 Тимофій Гал		
39	2786 0E0549PP	Feast	Ford	Зелений пастораль	2010	SD0000000000000000000000	BA0000000000000000000000	0,00	1 Елена Борис		
40	2843 K02913HM	Sedanum 2500	Chevrolet	Зелений	2004	000000000000000000000000	DK0700000000000000000000	1,41	1 Ірина Івана		
41	2847 HE251382	Park Avenue	Buick	Кільцевий	1996	VE7A00000000000000000000	WB0230000000000000000000	7,00	2 Валентина К		
42	2862 HE25019C	Seatra	Kia	Помаранчевий	2010	PUV000000000000000000000	GP0700000000000000000000	3,86	1 Анастасія І		
43	2979 HE04819H	MG	BMW	Ніско-зелений	2015	200130000000000000000000	WD065716	8,40	2 Борислав Ві		
44	3224 T08510P	ST02	Mitsubishi	Темно-зелений	2009	000000000000000000000000	OK0000000000000000000000	5,48	3 Михаїла І		
45	3311 TK0000E	Yukon	GM	Сірий (ІКС)	2002	L70000000000000000000000	XK0000000000000000000000	5,54	1 Еліна Лес		

Рисунок 7.4.9.1 – Результати роботи запиту, який повертає перелік транспортних засобів, що порушили ПДР за останній місяць

7.4.10 Перелік постанов із штрафами

SELECT

accident_resolutions.id AS resolution_id,

accident_resolutions.series,

accident_resolutions.number,

accident_resolutions.time_of_consideration,

accident_resolutions.time_of_entry_into_force,

accident_resolutions.violation_id,

accident_resolutions.police_officer_id,

accident_resolutions.location_id,

administrative_offenses.penalty_fee

FROM

accident_resolutions

JOIN violations ON accident_resolutions.violation_id = violations.id

JOIN administrative_offenses ON violations.administrative_offense_id =

administrative_offenses.id

WHERE

administrative_offenses.penalty_fee > 0

ORDER BY

accident_resolutions.time_of_consideration DESC;

На рис. 7.4.10.1 наведено результати роботи запиту, який повертає перелік постанов із штрафами.

	resolution_id	series	violation_id	police_office_id	location_id	penalty_fee
1	6302 CE	58519	2824-12-22 11:05:29...	2025-01-13 08:39:43...	23988	1293 1529 850.00
2	12051 ВІ	458058	2824-12-22 09:12:14...	2025-01-06 16:23:07...	1537	174 4211 17800.00
3	1879 PT	947474	2824-12-22 07:45:01...	2024-12-25 23:25:27...	15957	1429 4276 348.00
4	10896 PC	521938	2824-12-22 07:38:16...	2025-01-01 03:23:47...	18974	248 1524 510.00
5	18935 KH	178741	2824-12-22 06:57:34...	2025-01-06 08:23:37...	22599	1457 3913 3480.00
6	3293 НО	757829	2024-12-22 04:51:12...	2025-01-17 22:27:13...	317	1114 3921 680.00
7	6558 KH	778486	2824-12-22 03:27:23...	2025-01-05 13:57:03...	25993	13 1828 850.00
8	3571 НО	538562	2824-12-21 23:15:54...	2025-01-09 19:50:38...	28794	1279 145 510.00
9	9746 ЕК	244810	2824-12-21 21:17:27...	2025-01-09 19:54:06...	29665	1321 2884 3480.00
10	6737 МТ	637866	2024-12-21 20:34:40...	2025-01-11 00:42:41...	13742	1496 1986 17800.00
11	4189 ЕА	633593	2824-12-21 19:54:49...	2025-01-17 08:19:14...	3934	165 2772 1788.00
12	9884 СМ	905384	2824-12-21 19:43:20...	2024-12-23 07:06:16...	13399	1893 2568 1045.00
13	3877 KB	291384	2824-12-21 18:28:09...	2025-01-08 13:51:23...	28549	418 5834 3480.00
14	5654 HP	844914	2824-12-21 17:19:31...	2025-01-13 15:28:16...	27825	1178 2138 348.00
15	4888 IT	928126	2824-12-21 16:58:28...	2025-01-11 08:42:36...	19627	1511 57 510.00
16	10853 ЕК	171247	2824-12-21 15:52:24...	2025-01-01 23:55:22...	28367	1879 3668 17800.00
17	12457 KT	665358	2024-12-21 13:56:28...	2024-12-22 07:08:25...	17870	178 3584 1275.00
18	12189 BO	792389	2824-12-21 11:08:45...	2025-01-12 17:58:29...	12518	1522 935 5108.00
19	1684 IX	865792	2824-12-21 09:07:47...	2025-01-10 10:49:52...	27074	1324 2324 34800.00
20	966 BB	286408	2824-12-21 08:46:44...	2024-12-30 12:07:58...	24294	1786 1491 34.00
21	254 OT	250889	2024-12-21 07:18:05...	2025-01-17 10:46:26...	21113	1700 73 170.00
22	2489 AO	971166	2824-12-21 05:08:45...	2024-12-22 19:55:42...	1854	873 1521 1845.00
23	5260 OK	900457	2824-12-21 04:20:14...	2024-12-23 07:14:31...	3545	1418 1539 850.00
24	5611 IO	939260	2824-12-21 03:34:18...	2025-01-15 09:46:08...	24848	1395 784 255.00
25	3437 PT	078899	2024-12-20 19:29:09...	2025-01-09 13:01:55...	9864	501 3924 5100.00
26	2133 BH	414637	2824-12-20 19:04:19...	2025-01-19 08:45:53...	9721	1714 2298 850.00
27	2993 XC	822326	2824-12-20 14:39:38...	2025-01-13 22:11:13...	8955	268 981 1780.00
28	18987 AB	641685	2824-12-20 14:28:02...	2024-12-25 21:16:36...	13248	250 4129 595.00
29	5844 KH	655275	2824-12-20 09:47:36...	2024-12-23 15:22:37...	25559	1106 4646 51.00
30	545 BP	610824	2824-12-20 08:17:46...	2025-01-15 04:07:23...	20132	1914 1855 510.00

Рисунок 7.4.10.1 – Результати роботи запиту, який повертає перелік постанов із штрафами

7.4.11 Перелік адміністративних порушень із кількістю порушень

SELECT

administrative_offenses.id AS offense_id,

get_administrative_offense_info(administrative_offenses.id),

administrative_offenses.description,

administrative_offenses.penalty_fee,

COUNT(violations.id) AS violation_count

FROM

```

administrative_offenses
    LEFT JOIN violations ON administrative_offenses.id =
        violations.administrative_offense_id
    GROUP BY
        administrative_offenses.id,
        administrative_offenses.article,
        administrative_offenses.part,
        administrative_offenses.description,
        administrative_offenses.penalty_fee
    ORDER BY
        violation_count DESC;

```

На рис. 7.4.11.1 наведено результати роботи запиту, який повертає перелік адміністративних порушень із кількістю порушень.

	offense_id	get_administrative_offense_id	description	penalty_fee	violation_count
1	32 125	Інші порушення правил дорожнього руху	<null>	557	
2	39 127.2	Порушення Правил дорожнього руху обов'язкової накладки	340.00	542	
3	33 126.1	Керування транспортним засобом осо... бистрашником	425.00	537	
4	23 122.4	Перевищення водіями транспортних засобів дозволеної швидкості	1700.00	532	
5	52 133(1).4	Здійснення перевезень пасажирів та вантажів без дозволу	51.00	525	
6	46 130.3	Дії, передбачені частиною першою статті	51000.00	518	
7	5 121(3).1	Порушення вимог законодавства щодо обов'язкової накладки	1190.00	515	
8	14 121.5	Порушення правил користування ременем безпеки	510.00	515	
9	37 126.5	Повторне протягом року вчинення по-порушення	40800.00	514	
10	15 121.6	Керування водієм транспортним засобом особистою рукою	850.00	514	
11	17 122	Невиконання водіями вимог поліціянту	153.00	514	
12	34 126.2	Керування транспортним засобом осо... бистрашником	3400.00	513	
13	50 133(1).2	Порушення правил надання послуг з транспортним засобом	595.00	512	
14	19 122	Порушення вимог законодавства щодо обов'язкової накладки	8500.00	512	
15	53 133(1).5	Перевезення пасажирів чи вантажів без дозволу	510.00	511	
16	2 121(2).1	Перевезення водіями транспортних засобів без дозволу	170.00	510	
17	28 123.2	В'їзд на залізничний переїзд особою	850.00	510	
18	41 127.4	Порушення, передбачені частиною п'яткою статті	850.00	509	
19	24 122.5	Порушення, передбачені частинами п'ятою та шостою статті	1445.00	506	
20	38 127.1	Непокора пішоходів сигналам регулювання руху	255.00	506	
21	7 121(3).3	Повторне протягом року вчинення по-порушення	5100.00	504	
22	16 121.7	Повторне протягом року вчинення будь-якого іншого порушення	1700.00	504	
	2 121.1	Інші порушення правил дорожнього руху	710.00	503	

Рисунок 7.4.11.1 – Результати роботи запиту, який повертає перелік адміністративних порушень із кількістю порушень

7.4.12 Перелік доказів для конкретного порушення

SELECT

```
evidences.id AS evidence_id,
evidences.violation_id,
evidences.type AS evidence_type,
evidences.url AS evidence_url
```

FROM

```
evidences
```

WHERE

```
violation_id = 1
```

ORDER BY

```
evidences.violation_id,
evidences.id;
```

На рис. 7.4.12.1 наведено результати роботи запиту, який повертає перелік доказів для конкретного порушення.

evidence_id	violation_id	evidence_type	evidence_url
1	1	video	https://dummyimage.co...
2	2	video	https://placekitten.c...
3	3	photo	https://dummyimage.co...
4	4	video	https://picsum.photos...
5	5	photo	https://placekitten.com/...
6	6	video	https://picsum.photos...
7	7	video	https://picsum.photos...
8	8	photo	https://picsum.photos...

Рисунок 7.4.12.1 – Результати роботи запиту, який повертає перелік доказів для конкретного порушення

7.4.13 Перелік водіїв, які зробили два однакові порушення за перші два роки після отримання водійського посвідчення

```

SELECT
    drivers.id AS driver_id,
    get_citizen_full_name(citizens.id) AS driver_name,
    violations.administrative_offense_id,
    get_administrative_offense_info(administrative_offense_id) AS
offense_article,
    administrative_offenses.description AS offense_description,
    administrative_offenses.penalty_fee,
    COUNT(violations.id) AS violation_count
FROM
    drivers
    JOIN citizens ON drivers.citizen_id = citizens.id
    JOIN vehicles ON vehicles.owner_id = citizens.id
    JOIN violations ON violations.vehicle_id = vehicles.id
    JOIN administrative_offenses ON violations.administrative_offense_id =
administrative_offenses.id
WHERE
    violations.time_ofViolation BETWEEN drivers.license_issued_time
    AND drivers.license_issued_time + INTERVAL '2 years'
GROUP BY
    drivers.id,
    citizens.id,
    violations.administrative_offense_id,
    administrative_offenses.article,
    administrative_offenses.sup,
    administrative_offenses.part,
    administrative_offenses.description,
    administrative_offenses.penalty_fee
  
```

```

HAVING
    COUNT(violations.id) >= 2
ORDER BY
    driver_id,
    violation_count DESC;

```

На рис. 7.4.13.1 наведено результати роботи запиту, який повертає перелік водіїв, які зробили два однакові порушення за перші два роки після отримання водійського посвідчення.

The screenshot shows a database query results window titled "queries.sql - queries.sql". The table has the following columns: driver_id, driver_name, administrative_offense_id, offense_article, offense_description, penalty_fee, and violation_count. The data is as follows:

	driver_id	driver_name	administrative_offense_id	offense_article	offense_description	penalty_fee	violation_count
1	224	Ярослав Рудько Панасович	37	126.5	Повторне протягом року вчинення порушення, передачені частинов первое цієї с...	40800.00	2
2	1572	Руслан Влох Істомович	43	128(1).2	Порушення, передачені частинов первое цієї с...	2550.00	2
3	4879	Евген Верхола Олександрович	36	126.4	Керування транспортним засобом особов, позбав...	20400.00	2
4	6429	Алла Гжицька Денимівна	14	121.5	Порушення правил користування ременями безпек...	510.00	2
5	7840	Ева Шеремет Захарівна	15	121.6	Керування водієм транспортним засобом, не зар...	850.00	2

Рисунок 7.4.13.1 – Результати роботи запиту який повертає перелік водіїв, які зробили два однакові порушення за перші два роки після отримання водійського посвідчення

7.4.14 Топ виробників автомобілів, власники яких мають найбільшу відносну кількість правопорушень

```

SELECT
    vehicles.brand,
    COUNT(vehicles.id) AS total_vehicles,
    COUNT(violations.id) AS total_violations,
    (
        COUNT(violations.id) :: DECIMAL / COUNT(vehicles.id) * 100
    ) AS violation_percentage
FROM
    vehicles
    LEFT JOIN violations ON vehicles.id = violations.vehicle_id
GROUP BY
    vehicles.brand;

```

```

vehicles.brand
HAVING
    COUNT(vehicles.id) > 0
    AND COUNT(vehicles.id) > 200
ORDER BY
    violation_percentage DESC;

```

На рис. 7.4.14.1 наведено результати роботи запиту, який повертає топ виробників автомобілів, власники яких мають найбільшу відносну кількість правопорушень.

	brand	total_vehicles	total_violations	violation_percentage
1	Suzuki	300	287	95.6666666666666667
2	Mercury	389	372	95.629820051413881748
3	Isuzu	237	226	95.358649789029535865
4	Nissan	1326	1261	95.09803921568627451
5	Oldsmobile	256	243	94.921875
6	Dodge	1440	1363	94.6527777777777778
7	Audi	874	826	94.508009153318077803
8	Ram	348	328	94.252873563218390805
9	BMW	920	867	94.239130434782608696
10	INFINITI	446	420	94.170403587443946188
11	Buick	461	434	94.143167028199566161
12	Mitsubishi	669	629	94.020926756352765321
13	Mercedes-Benz	1233	1159	93.9983779399837794
14	Subaru	425	399	93.882352941176470588
15	Cadillac	594	557	93.771043771043771044
16	Pontiac	378	354	93.650793650793650794
17	MAZDA	644	603	93.633540372670807453
18	Honda	795	744	93.584905660377358491
19	Kia	540	505	93.518518518518518519

Рисунок 7.4.14.1 – Результати роботи запиту, який повертає топ виробників автомобілів, власники яких мають найбільшу відносну кількість правопорушень

7.4.15 Кількість порушень без протоколів, без постанов, із протоколами та постановами, загальна кількість порушень

```
SELECT
```

```

COUNT(*) FILTER (
    WHERE
        accident_protocols.id IS NULL
        AND accident_resolutions.id IS NULL
    ) AS violations_without_protocol_or_resolution,
COUNT(*) FILTER (
    WHERE
        accident_protocols.id IS NOT NULL
        OR accident_resolutions.id IS NOT NULL
    ) AS violations_with_protocol_or_resolution,
COUNT(*) AS total_violations

FROM
    violations
    LEFT JOIN accident_protocols ON violations.id =
accident_protocols.violation_id
    LEFT JOIN accident_resolutions ON violations.id =
accident_resolutions.violation_id;

```

На рис. 7.4.15.1 наведено результати роботи запиту, який повертає кількість порушень без протоколів, без постанов, із протоколами та постановами, загальна кількість порушень.

	violations_without_protocol_or_resolution	violations_with_protocol_or_resolution	total_violations
1	12132	18042	30174

Рисунок 7.4.15.1 – Результати роботи запиту, який повертає кількість порушень без протоколів, без постанов, із протоколами та постановами, загальна кількість порушень

7.4.16 Перелік громадян з найбільшою кількістю свідчень на інших осіб

SELECT

 citizens.id,

 citizens.first_name,

 citizens.last_name,

 citizens.patronymic,

 COUNT(citizens_on_protocol.id) AS testimony_count

FROM

 citizens

 JOIN citizens_on_protocol ON citizens.id =

 citizens_on_protocol.citizen_id

WHERE

 citizens_on_protocol.role = 'witness'

GROUP BY

 citizens.id,

 citizens.first_name,

 citizens.last_name,

 citizens.patronymic

ORDER BY

 testimony_count DESC

LIMIT

 10;

На рис. 7.4.16.1 наведено результати роботи запиту, який повертає перелік громадян з найбільшою кількістю свідчень на інших осіб.

	id	first_name	last_name	patronymic	testimony_count
1	18604	Юстина	Ільченко	Опанасівна	5
2	10521	Едита	Байрак	Данівна	4
3	14690	Володимир	Супруненко	Августинович	4
4	10893	Артем	Зінчук	Антонович	4
5	12800	Мирон	Девдюк	Богуславович	4
6	17700	Клавдія	Єщенко	Миколаївна	4
7	16511	Світлана	Батуринець	Микитівна	4
8	16757	Ярослава	Гаврилюк	Олександровна	4
9	15739	Орина	Черненко	Макарівна	3
10	1588	Прохір	Яременко	Богданович	3

Рисунок 7.4.16.1 – Результати роботи запиту, який повертає перелік громадян з найбільшою кількістю свідчень на інших осіб

7.4.17 Перелік громадян які були і свідками і порушниками

SELECT

```

get_citizen_full_name(c.id) AS full_name,
COUNT(
CASE
    WHEN citizens_on_protocol.role = 'witness' THEN 1
END
) AS witness_count,
COUNT(
CASE
    WHEN cop_defendant.role = 'victim' THEN 1
END
) AS victim_count
FROM
citizens c
JOIN citizens_on_protocol ON c.id = citizens_on_protocol.citizen_id
JOIN accident_protocols ON citizens_on_protocol.protocol_id =
accident_protocols.id
  
```

```

JOIN citizens_on_protocol cop_defendant ON
accident_protocols.defendant_id = cop_defendant.citizen_id

WHERE
    citizens_on_protocol.role = 'witness'
    AND cop_defendant.role = 'victim'

GROUP BY
    c.id,
    full_name;

```

На рис. 7.4.17.1 наведено результати роботи запиту, який повертає перелік громадян які були і свідками і жертвами (на різних протоколах).

	full_name	witness_count	victim_count
1	Єва Дем'янюк Альбертівна	1	1
2	Симон Калениченко Трохимович	1	1
3	Володимир Авраменко Олесьович	1	1
4	Данна Онуфрієнко Ігнатівна	1	1
5	Маруся Ребрик Ааронівна	1	1
6	Роман Артем Глібович	1	1
7	Миколай Перешийніс Юхимович	1	1
8	Марко Дейсун Тарасович	1	1
9	Світлана Салій Альбертівна	1	1
10	Одарка Дацюк Валеріївна	2	2
11	Арсен Башполченко Богодарович	1	1
12	Юстина Христенко Мирославівна	1	1
13	Богданна Гайдабура Алевтинівна	1	1
14	Богданна Ісаєвич Юхимівна	3	3
15	Богданна Яценюк Русланівна	1	1
16	Варфоломій Матяш Орестович	2	2
17	Венедикт Базилевич Васильович	1	1
18	Ярина Піддубна Остапівна	1	1
19	В'ячеслав Притула Максимович	1	1

Рисунок 7.4.17.1 – Результати роботи запиту, який перелік громадян які були і свідками і жертвами (на різних протоколах)

7.4.18 Топ громадян за сумою штрафів

```
SELECT
    get_citizen_full_name(citizens.id) AS full_name,
    COALESCE(SUM(administrative_offenses.penalty_fee), 0) AS
total_fines,
    COALESCE(COUNT(violations.id), 0) AS total_violations
FROM
    citizens
        LEFT JOIN vehicles ON citizens.id = vehicles.owner_id
        LEFT JOIN violations ON vehicles.id = violations.vehicle_id
        LEFT JOIN administrative_offenses ON
violations.administrative_offense_id = administrative_offenses.id
GROUP BY
    citizens.id
ORDER BY
    total_fines DESC
LIMIT
    50;
```

На рис. 7.4.18.1 наведено результати роботи запиту, який повертає топ громадян за сумою штрафів.

	full_name	total_fines	total_violations
1	Роман Аврамчук Ростиславович	170323	13
2	Остап Савенко В'ячеславович	157590	7
3	Семен Деряжний Артемович	153255	4
4	Володимира Галаґан Данівна	150705	12
5	Захар Гаврюшenko Опанасович	142800	4
6	Юстина Теліженко Орхипівна	139570	7
7	Альберт Шелест Макарович	137785	6
8	Адам Дейнеко Кирилович	137258	14
9	Тетяна Чайка Григоріївна	134969	11
10	Василь Юрченко Леонідович	133450	5
11	Ліза Артимишина Орхипівна	132719	14
12	Сніжана Фесенко Несторівна	131478	10
13	Лілія Ляшко Миронівна	130611	12
14	Віолетта Мірошниченко Леопольдівна	130220	8
15	Ліза Петлюра Олегівна	129268	12
16	Едита Зайченко Арсенівна	127880	8
17	Михайло Фесенко Олександрович	127823	12
18	Соломія Івасюк Сергіївна	125205	8
19	Тимофій Світличко Панасович	124721	7

Рисунок 7.4.18.1 – Результати роботи запиту, який повертає топ громадян за сумою штрафів

7.4.19 Найчастіше порушувані статті за типом транспортного засобу

SELECT

```

vehicle_types.name AS vehicle_type,
get_administrative_offense_info(administrative_offenses.id) AS
administrative_offense,
administrative_offenses.description,
COUNT(violations.id) AS violations_count
FROM
vehicles
JOIN vehicle_types ON vehicles.vehicle_type_id = vehicle_types.id
JOIN violations ON vehicles.id = violations.vehicle_id
JOIN administrative_offenses ON violations.administrative_offense_id =
administrative_offenses.id

```

GROUP BY

 vehicle_types.name,
 administrative_offense,
 administrative_offenses.description

ORDER BY

 vehicle_types.name,
 violations_count DESC;

На рис. 7.4.19.1 наведено результати роботи запиту, який повертає перелік найчастіше порушуваних статей за типом транспортного засобу.

vehicle_type	administrative_offense	description	violations_count
150 Bus	121(2).1	перевезення видимими транспортними засобами, що працюють у ре...	55
157 Bus	128(1).2	Порушення, передбачені частиною першою цієї статті, що спр...	34
158 Bus	121.5	Порушення правил користування ременями безпеки або мотошол...	33
159 Bus	128(1).1	Порушення або невиконання правил, норм і стандартів, що ст...	32
160 Bus	130.4	Вживання водієм транспортного засобу після дорожньо-трансп...	30
161 Bus	121.4	Повторне протягом року вчинення будь-якого з порушень, пер...	30
162 Bus	133	Здійснення внутрішніх автомобільних перевезень пасажирів і...	29
163 Car	121.7	Повторне протягом року вчинення будь-якого з порушень, пер...	59
164 Car	126.4	Керування транспортним засобом особою, позбавленою права к...	59
165 Car	122	Порушення вимог законодавства щодо встановлення і використ...	56
166 Car	133(1).4	Здійснення перевезень пасажирів таксі, в яких не встановле...	55
167 Car	126.5	Повторне протягом року вчинення порушень, передбачених час...	54
168 Car	124	Ненадання посадовими особами підприємств, установ, організ...	54
169 Car	121.5	Порушення правил користування ременями безпеки або мотошол...	54
170 Car	121.6	Керування водієм транспортним засобом, не зареєстрованим а...	53
171 Car	133(1).5	Перевезення пасажирів чи вантажів водієм, який не пройшов –	53
172 Car	132	Порушення правил дорожнього перевезення небезпечних вантаж...	52
173 Car	127.1	Непокора пішоходів сигналам регулювання дорожнього руху, п...	52
174 Car	128(1).1	Порушення або невиконання правил, норм і стандартів, що ст...	51
175 Car	121(2).1	Перевезення водіями транспортних засобів, що працюють у ре...	50
176 Car	121.3	Керування водієм транспортним засобом, що підлягає обов'яз...	50
177 Car	133(1).1	Здійснення регулярних перевезень пасажирів на постійних ма...	50
178 Car	121(3).2	Керування або експлуатація транспортного засобу із незакон...	50

Рисунок 7.4.19.1 – Результати роботи запиту, який повертає найчастіше порушувані статті за типом транспортного засобу

7.4.20 Топ водіїв автобусів, які отримали протокол чи постанову за керування у нетверезому стані

```

SELECT
    citizens.id AS citizen_id,
    get_citizen_full_name(citizens.id) AS full_name,
    COUNT(accident_protocols.id) AS protocol_count
FROM
    drivers
JOIN vehicles ON drivers.citizen_id = vehicles.owner_id
JOIN vehicle_types ON vehicles.vehicle_type_id = vehicle_types.id
JOIN accident_protocols ON accident_protocols.defendant_id =
    drivers.citizen_id
JOIN violations ON violations.id = accident_protocols.violation_id
JOIN traffic_rules ON traffic_rules.id = violations.traffic_rule_id
  
```

```

JOIN administrative_offenses ON administrative_offenses.id =
violations.administrative_offense_id
JOIN citizens ON citizens.id = drivers.citizen_id
WHERE
vehicle_types.name = 'Bus'
AND (
--      пункт ПДР або КУпАП про водіння у нетверезому стані
(
traffic_rules.article = 2
AND traffic_rules.part = 9
)
OR (
administrative_offenses.article = 130
AND administrative_offenses.part = 1
)
)
GROUP BY
citizens.id,
full_name
ORDER BY
protocol_count DESC
LIMIT
10;

```

На рис. 7.4.20.1 наведено результати роботи запиту, який повертає топ водіїв автобусів, які отримали протокол чи постанову за керування у нетверезому стані.

	citizen_id	full_name	protocol_count
1	2658	Ігнат Безбородько Альбертович	1
2	6086	Юхим Валенко Стефанович	1
3	10636	Софія Іщенко Аркадіївна	1
4	13313	Камілла Влох Аркадіївна	1
5	14723	Станіслав Якимчук Станіславович	1
6	19667	Сніжана Рубан Пантелеймонівна	1

Рисунок 7.4.2.1 – Результати роботи запиту, який повертає топ водіїв автобусів, які отримали протокол чи постанову за керування у нетверезому стані

7.4.21 Кількість свідків та жертв в протоколах

SELECT

```

accident_protocols.id AS protocol_id,
COUNT(
CASE
    WHEN citizens_on_protocol.role = 'witness' THEN 1
END
) AS witness_count,
COUNT(
CASE
    WHEN citizens_on_protocol.role = 'victim' THEN 1
END
) AS victim_count

```

FROM

```

accident_protocols
LEFT JOIN citizens_on_protocol ON accident_protocols.id =
citizens_on_protocol.protocol_id
GROUP BY
```

```
accident_protocols.id
```

```
ORDER BY
```

```
accident_protocols.id;
```

На рис. 7.4.21.1 наведено результати роботи запиту, який повертає кількість свідків та жертв для протоколів.

queries.sql - queries.sql

знайти кількість свідків та жертв для протоколів

<input type="checkbox"/> protocol_id	<input type="checkbox"/> witness_count	<input type="checkbox"/> victim_count	
1	1	2	0
2	2	1	2
3	3	1	1
4	4	0	0
5	5	1	0
6	6	0	2
7	7	1	0
8	8	0	0
9	9	1	0
10	10	2	1
11	13	0	1
12	14	1	0
13	15	0	1
14	17	0	0
15	18	1	1
16	20	0	0
17	21	0	0

Рисунок 7.4.21.1 – Результати роботи запиту, який повертає кількість свідків та жертв для протоколів

7.4.22 Найпоширеніший тип порушень ПДР по кожній вулиці

SELECT

```

locations.street,
get_administrative_offense_info(violations.administrative_offense_id) AS
administrative_offense,
COUNT(violations.id) AS violation_count
FROM
violations
JOIN locations ON violations.location_id = locations.id
GROUP BY
locations.street,
```

```

administrative_offense
ORDER BY
    locations.street,
    violation_count DESC;

```

На рис. 7.4.22.1 результати роботи запиту, який повертає перелік найпоширеніших типів порушень ПДР по кожній вулиці.

The screenshot shows a table titled "Найпоширеніший тип п... ПДР по кожній вулиці" (Most common type of traffic offense by street). The table has three columns: "street" (улиця), "administrative_offense" (тип порушення), and "violation_count" (кількість порушень). The data is sorted by violation count in descending order. The table contains 19 rows, with the first few rows being: 1-й Академічний тупик (126.2), 1-й Академічний тупик (130.3), 1-й Академічний тупик (133), 1-й Академічний тупик (128(1).2), 1-й Академічний тупик (121.3), etc.

street	administrative_offense	violation_count
1-й Академічний тупик	126.2	2
1-й Академічний тупик	130.3	1
1-й Академічний тупик	133	1
1-й Академічний тупик	128(1).2	1
1-й Академічний тупик	121.3	1
1-й Академічний тупик	130.1	1
1-й Академічний тупик	121.1	1
1-й Академічний тупик	121(3).2	1
1-й Академічний тупик	122.2	1
1-й Академічний тупик	133(1).4	1
1-й Академічний тупик	126.5	1
1-й Академічний тупик	122.6	1
1-й Академічний тупик	133(1).3	1
1-й Академічний тупик	121(1).1	1
1-ша Лінія 4-ї ст. Люстдорфської дороги	126.5	2
1-ша Лінія 4-ї ст. Люстдорфської дороги	121(3).2	2
1-ша Лінія 4-ї ст. Люстдорфської дороги	121.6	2
1-ша Лінія 4-ї ст. Люстдорфської дороги	133(1).5	2
1-ша Лінія 4-ї ст. Люстдорфської дороги	126.4	2

Рисунок 7.4.22.1 – Результати роботи запиту, який повертає перелік найпоширеніших типів порушень ПДР по кожній вулиці

7.4.23 Інформація про водіїв, які мають транспортні засоби різних типів

SELECT

```

citizens.first_name,
citizens.last_name,
COUNT(DISTINCT vehicles.vehicle_type_id) AS vehicle_types
FROM
    citizens
JOIN vehicles ON citizens.id = vehicles.owner_id

```

```

GROUP BY
    citizens.id
HAVING
    COUNT(DISTINCT vehicles.vehicle_type_id) > 1
ORDER BY
    vehicle_types DESC;

```

На рис. 7.4.23.1 наведено результати роботи запиту, який повертає інформацію про водіїв, які мають транспортні засоби різних типів.

	first_name	last_name	vehicle_types
1	Олена	Гаврилишин	5
2	Максим	Вовк	5
3	Дмитро	Піддубний	5
4	Сніжана	Супруненко	5
5	Ганна	Тарасенко	4
6	Климент	Радченко	4
7	Павло	Данилюк	4
8	Нестор	Міщенко	4
9	Орися	Єщенко	4
10	Юхим	Яремчук	4
11	Ігор	Непорожній	4
12	Розалія	Ткач	4
13	Оксенія	Іщак	4
14	Хома	Бебешко	4
15	В'ячеслав	Їжакевич	4
16	Аврелій	Данилюк	4
17	Арсен	Ватаманюк	4
18	Климент	Щорс	4
19	Вікторія	Корсун	4
20	Варфоломій	Остапчук	4

Рисунок 7.4.23.1 – Результати роботи запиту, який повертає інформацію про водіїв, які мають транспортні засоби різних типів

7.5 Індекси

Для забезпечення ефективного доступу до даних та підвищення продуктивності запитів у базі даних були створені індекси. Вони дозволяють оптимізувати виконання операцій вибірки, пошуку та сортування, особливо в умовах великого обсягу даних.

Первинні ключі забезпечують унікальність записів у таблицях і автоматично створюють індекси. Наприклад, у таблицях citizens, drivers та vehicles індекси створені на полях id, що дозволяє швидко знаходити записи за унікальними ідентифікаторами.

Зовнішні ключі дозволяють встановлювати зв'язки між таблицями. Для підвищення продуктивності запитів, що об'єднують дані з кількох таблиць, були створені додаткові індекси на полях зовнішніх ключів, наприклад, на колонці citizen_id у таблицях drivers та police_officers.

Код створення індексів на зовнішні класі для оптимізації:

```
CREATE INDEX idx_drivers_citizen_id ON drivers (citizen_id);
CREATE INDEX idx_police_officers_citizen_id ON police_officers
(citizen_id);
CREATE INDEX idx_violations_vehicle_id ON violations (vehicle_id);
CREATE INDEX idx_violations_location_id ON violations (location_id);
CREATE INDEX idx_violations_administrative_offense_id ON violations
(administrative_offense_id);
CREATE INDEX idx_violations_traffic_rule_id ON violations
(traffic_rule_id);
CREATE INDEX idx_accident_protocolsViolation_id ON
accident_protocols (violation_id);
CREATE INDEX idx_accident_protocolsPolice_officer_id ON
accident_protocols (police_officer_id);
CREATE INDEX idx_accident_protocolsDefendant_id ON
accident_protocols (defendant_id);
```

```
CREATE INDEX idx_citizens_on_protocol_citizen_id ON
citizens_on_protocol (citizen_id);
```

```
CREATE INDEX idx_citizens_on_protocol_protocol_id ON
citizens_on_protocol (protocol_id);
```

На рис. 7.5.1 наведено швидкість запиту до створення індексів, а на рис. 7.5.2, відповідно, після. Можна помітити зменшення часу виконання тестового запиту більше ніж у два рази, що є результатом індексування бази даних.

```

Result 6 × indexes.sql - indexes.sql
QUERY PLAN
1 Hash Join (cost=2378.78..5071.23 rows=10841 width=1094) (actual time=49.705..69.983 rows=10844 loops=1)
2   Hash Cond: (v.administrative_offense_id = ao.id)
3     -> Hash Join (cost=2372.56..5034.32 rows=10841 width=1089) (actual time=49.412..68.124 rows=10844 loops=1)
4       Hash Cond: (v.traffic_rule_id = t.id)
5         -> Hash Join (cost=2334.02..4966.91 rows=10841 width=380) (actual time=49.033..66.010 rows=10844 loops=1)
6           Hash Cond: (v.vehicle_id = ve.id)
7             -> Hash Join (cost=1749.74..4354.16 rows=10841 width=341) (actual time=17.162..31.116 rows=10844 loops=1)
8               Hash Cond: (a.police_officer_id = po.id)
9                 -> Hash Join (cost=1162.92..3618.28 rows=10841 width=313) (actual time=10.219..22.045 rows=10844 loops=1)
10                Hash Cond: (v.id = a.violation_id)
11                  -> Seq Scan on violations v  (cost=0.00..2110.84 rows=26984 width=291) (actual time=0.017..6.325 rows=26988 loops=1)
12                    -> Hash (cost=1027.41..1027.41 rows=10841 width=26) (actual time=10.152..10.152 rows=10844 loops=1)
13                      Buckets: 16384 Batches: 1 Memory Usage: 806kB
14                        -> Seq Scan on accident_protocols a  (cost=0.00..1027.41 rows=10841 width=26) (actual time=0.026..7.066 rows=10844 loops=1)
15                      Hash (cost=565.48..565.48 rows=1707 width=36) (actual time=6.912..6.916 rows=1707 loops=1)
16                        Buckets: 2048 Batches: 1 Memory Usage: 132kB
17                          -> Hash Join (cost=50.41..565.48 rows=1707 width=36) (actual time=0.652..6.478 rows=1707 loops=1)
18                            Hash Cond: (c.id = po.citizen_id)
19                              -> Seq Scan on citizens c  (cost=0.00..423.00 rows=20000 width=32) (actual time=0.011..2.912 rows=20000 loops=1)
20                                -> Hash (cost=29.07..29.07 rows=1707 width=12) (actual time=0.610..0.611 rows=1707 loops=1)
21                                  Buckets: 2048 Batches: 1 Memory Usage: 90kB
22                                    -> Seq Scan on police_officers po  (cost=0.00..29.07 rows=1707 width=12) (actual time=0.008..0.315 rows=1707 loops=1)
23                                    Hash (cost=415.79..415.79 rows=13479 width=47) (actual time=31.788..31.788 rows=13479 loops=1)
24                                      Buckets: 16384 Batches: 1 Memory Usage: 1215kB
25                                        -> Seq Scan on vehicles ve  (cost=0.00..415.79 rows=13479 width=47) (actual time=0.016..26.618 rows=13479 loops=1)
26                                          -> Hash (cost=34.91..34.91 rows=291 width=713) (actual time=0.368..0.368 rows=291 loops=1)
27                                            Buckets: 1024 Batches: 1 Memory Usage: 221kB
28                                              -> Seq Scan on traffic_rules t  (cost=0.00..34.91 rows=291 width=713) (actual time=0.005..0.212 rows=291 loops=1)
29                                              Hash (cost=5.54..5.54 rows=54 width=9) (actual time=0.254..0.255 rows=54 loops=1)
30                                                Buckets: 1024 Batches: 1 Memory Usage: 11kB
31                                                -> Seq Scan on administrative_offenses ao  (cost=0.00..5.54 rows=54 width=9) (actual time=0.013..0.037 rows=54 loops=1)
32 Planning Time: 4.698 ms
33 Execution Time: 71.260 ms

```

Рисунок 7.5.1 – Інформація про виконання тестового запиту до створення індексів

```

Result 6 | Result 10
indexes.sql - indexes.sql
Result 6 Result 10
 QUERY PLAN
12 Planning Time: 1.951 ms
13 Execution Time: 33.015 ms
14 Buckets: 16384 Batches: 1 Memory Usage: 806kB
15      -> Seq Scan on accident_protocols a (cost=0.00..1027.44 rows=10844 width=26) (actual time=0.758..7.51 rows=10844 loops=1)
16      -> Hash (cost=565.48..565.48 rows=1707 width=36) (actual time=5.978..5.979 rows=1707 loops=1)
17      Buckets: 2048 Batches: 1 Memory Usage: 132kB
18      -> Hash Join (cost=50.41..565.48 rows=1707 width=36) (actual time=0.681..5.493 rows=1707 loops=1)
19          Hash Cond: (c.id = po.citizen_id)
20          -> Seq Scan on citizens c (cost=0.00..423.00 rows=20000 width=32) (actual time=0.015..1.886 ..)
21          -> Hash (cost=29.07..29.07 rows=1707 width=12) (actual time=0.656..0.656 rows=1707 loops=1)
22          Buckets: 2048 Batches: 1 Memory Usage: 90kB
23          -> Seq Scan on police_officers po (cost=0.00..29.07 rows=1707 width=12) (actual time=0.729..6.729 rows=13479 loops=1)
24          Buckets: 16384 Batches: 1 Memory Usage: 1215kB
25          -> Seq Scan on vehicles ve (cost=0.00..415.79 rows=13479 width=47) (actual time=0.006..3.103 rows=13479 ..)
26          -> Hash (cost=34.91..34.91 rows=291 width=713) (actual time=0.209..0.209 rows=291 loops=1)
27          Buckets: 1024 Batches: 1 Memory Usage: 221kB
28          -> Seq Scan on traffic_rules t (cost=0.00..34.91 rows=291 width=713) (actual time=0.004..0.099 rows=291 loops=1)
29          -> Hash (cost=5.54..5.54 rows=54 width=9) (actual time=0.043..0.043 rows=54 loops=1)
30          Buckets: 1024 Batches: 1 Memory Usage: 11kB
31          -> Seq Scan on administrative_offenses ao (cost=0.00..5.54 rows=54 width=9) (actual time=0.010..0.031 rows=54 loops=1)
32 Planning Time: 1.951 ms
33 Execution Time: 33.015 ms

```

Рисунок 7.5.1 – Інформація про виконання тестового запиту після створення індексів

7.6 Висновки

У розділі 7 було детально розглянуто механізми оптимізації та управління базою даних, зокрема тригери, представлення, функції, процедури, запити та індекси. Визначено важливість кожного з цих елементів для забезпечення високої ефективності та функціональності системи.

Тригери були використані для автоматизації певних дій, що виконуються при зміні даних, що дозволяє зменшити ризик помилок і зберегти консистентність бази даних. Представлення забезпечують зручний доступ до часто запитуваних або складних даних, дозволяючи зменшити складність запитів. Функції і процедури дозволяють реалізувати складну бізнес-логіку на

рівні бази даних, підвищуючи її гнучкість та зменшуючи навантаження на додатки.

Запити, описані в розділі, продемонстрували використання різноманітних з'єднань таблиць, що забезпечує отримання необхідної інформації з кількох джерел. Індекси допомогли значно підвищити продуктивність системи, зменшуючи час на виконання складних операцій вибірки, що є важливим для роботи з великими обсягами даних.

Загалом, реалізація цих елементів сприяла досягненню необхідної ефективності і зручності в роботі з базою даних, що позитивно впливає на загальну продуктивність системи.

ВИСНОВКИ

У результаті виконаної курсової роботи було успішно розроблено базу даних для підтримки системи фіксації адміністративних правопорушень у сфері забезпечення безпеки дорожнього руху. Всі етапи проектування та реалізації бази даних, починаючи від аналізу предметної області та постановки завдання до розробки реляційної моделі та впровадження індексів і тригерів, були ретельно виконані.

Першим кроком було проведення аналізу предметної області, що дозволило визначити ключові сутності, їх атрибути та взаємозв'язки. Розроблена ER-модель забезпечила чітке уявлення про структуру бази даних і дозволила правильно організувати зберігання інформації. Завдяки цьому було забезпечено відповідність бізнес-правилам та вимогам до бази даних.

Вибір реляційної моделі даних для подальшої реалізації бази був обґрунтований зручністю роботи з таблицями, функціональністю СУБД PostgreSQL і необхідністю підтримки складних зв'язків між сутностями. Усі таблиці були спроектовані таким чином, щоб забезпечити високу нормалізацію та уникнути дублювання даних.

Реалізація бази даних включала не тільки створення структури таблиць, а й написання SQL-скриптів для їх заповнення, а також процедури для управління користувачами та їх правами доступу. Завдяки правильному налаштуванню ролей і привілеїв система забезпечує безпеку та контроль доступу до чутливої інформації.

Окрему увагу було приділено створенню тригерів, представлень, функцій і процедур, що дозволяють автоматизувати виконання операцій і зменшити ймовірність помилок. Це також дозволяє знижувати навантаження на систему та спрощує обробку запитів.

Індекси та оптимізація запитів грають важливу роль у забезпеченні ефективності роботи з базою даних. Всі індекси були створені з урахуванням частоти використання запитів, що значно підвищує швидкість виконання операцій з великою кількістю даних.

Загалом, створена база даних ефективно підтримує систему фіксації адміністративних правопорушень, надаючи зручні інструменти для зберігання, аналізу та обробки даних. Вона відповідає всім вимогам до безпеки, продуктивності та цілісності даних, що забезпечує її подальше використання в реальних умовах.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

- 1) Кабінет водія | Послуги Сервісних центрів МВС онлайн. *Кабінет водія | Послуги Сервісних центрів МВС онлайн.* URL: <https://edriver.mvs.gov.ua/blog> (дата звернення: 22.12.2024).
- 2) Сервіс Перевірки адміністративних порушень у сфері забезпечення безпеки дорожнього руху в автоматичному режиму. URL: <https://bdr.mvs.gov.ua/> (дата звернення: 22.12.2024).
- 3) Штрафи ПДР. App *Store.*
URL: <https://apps.apple.com/us/app/штрафи-пдр/id1541875502> (дата звернення: 22.12.2024).
- 4) Оплата штрафів ПДР | Перевірка штрафів ПДР | Штрафи ПДР | Штрафи UA. *Оплата штрафів ПДР | Перевірка штрафів ПДР | Штрафи ПДР | Штрафи UA.* URL: <https://splata.shtrafy.ua/> (дата звернення: 22.12.2024).
- 5) Contributors to Wikimedia projects. Relational database - Wikipedia. *Wikipedia, the free encyclopedia.*
URL: https://en.wikipedia.org/wiki/Relational_database (дата звернення: 22.12.2024).
- 6) PostgreSQL. *PostgreSQL.* URL: <https://www.postgresql.org/> (дата звернення: 22.12.2024).
- 7) JetBrains. DataGrip: The Cross-Platform IDE for Databases & SQL by JetBrains. *JetBrains.* URL: <https://www.jetbrains.com/datagrip/> (дата звернення: 22.12.2024).