МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КІБЕРБЕЗПЕКИ, КОМП'ЮТЕРНОЇ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ
КАФЕДРА КОМП'ЮТЕРИЗОВАНИХ СИСТЕМ УПРАВЛІННЯ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач випускової кафедри
_____Литвиненко О.Є.
"_____"_____ 2021 р.

ДИПЛОМНИЙ ПРОЄКТ

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

ЗА НАПРЯМОМ ПІДГОТОВКИ 6.050102 «КОМП'ЮТЕРНА ІНЖЕНЕРІЯ»

Тема: «Комп'ютерна програма обробки та аналізу даних для вантажних перевезень»

Виконавець: Царук О. В.

Керівник: д.т.н., доц. Вавіленкова А. І.
(науковий ступінь, вчене звання, прізвище, ім'я, по-батькові)

Нормоконтролер: Тупота Є.В.

КИЇВ 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, ком'ютерної та програмної інженерії
Кафедра *комп'ютеризованих систем управління*
Напрям (спеціальність) *6.050102 «Комп'ютера інженерія»*
(шифр, найменування)

ЗАТВЕРДЖУЮ
Завідувач кафедри
О.Є. Литвиненко
«_____» _____ 2021 р.

ЗАВДАННЯ
на виконання дипломного проєкту
Царука Олександра Володимировича
(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. Тема дипломного проекту «Комп'ютерна програма обробки та аналізу даних для вантажних перевезень» затверджена наказом ректора від « 04 » лютого 2021 р. № 135/ст

2. Термін виконання проєкту: з 17 травня 2021 р. по 20 червня 2021 р.

3. Вихідні дані до проєкту: реляційні таблиці даних, мова програмування *Python*, середовище програмування *PyCharm 2021.1.1*, система управління базами даних *SQLite*

4. Зміст пояснювальної записки:

1) Обробка та аналіз даних

2) База даних для комп'ютерної програми обробки та аналізу даних для вантажних перевезень

3) Програма обробки та аналізу даних для вантажних перевезень

5. Перелік обов'язкового графічного (ілюстративного) матеріалу:

1) Процес побудови впорядкованої гістограми (схема алгоритму)

2) Діаграма розмаху досвіду роботи працівників

3) Діаграма відношення витрат до часу перевезення

4) Таблиця бази даних «Перевезення»

6. Календарний план-графік

№ з/п	Завдання	Термін виконання	Підпис керівника
1.	Аналіз джерел за темою дослідження. Розроблення плану дипломного проєкту	17.05.2021-19.05.2021	
2.	Затвердження плану дипломного проєкту та проведення консультацій з науковим керівником, щодо наповнення пояснювальної записки	20.05.2021-21.05.2021	
3.	Дослідження засобів для роботи з наборами даними	22.05.2021-24.05.2021	
4.	Розробка архітектури програми по обробці та аналізу даних	25.05.2021-26.05.2021	
5.	Створення програми обробки та аналізу даних	01.06.2021-10.06.2021	
6.	Написання пояснювальної записки	11.06.2021-15.06.2021	
7.	Підготовка демонстраційного матеріалу	16.06.2021 20.06.2021	

7. Дата видачі завдання: « 17 » травня 2021 р.

Керівник дипломного проекту _____ д.т.н., доц. Вавіленкова А.І.
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Царук О.В.
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до дипломного проєкту «Комп'ютерна програма обробки та аналізу даних для вантажних перевезень»: 76 с., 41 рис., 2 табл., 15 літературних джерел, 1 додаток.

ОБРОБКА ДАНИХ, АНАЛІЗ ДАНИХ, ВІЗУАЛІЗАЦІЯ ДАНИХ, СИСТЕМА УПРАВЛІННЯ БАЗОЮ ДАНИХ, ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ, НАБІР ДАНИХ, ДІАГРАМА.

Об'єктом дослідження даного дипломного проєкту є процес обробки та аналізу даних.

Предметом дослідження є системи для роботи з наборами даних.

Метою даного проєкту є проектування та розробка комп'ютерної програми для обробки та аналізу даних для вантажних перевезень на основі дослідження способів обробки структурованих наборів даних та принципів візуалізації інформації.

Методи дослідження – технології систем управління базами даних, порівняльний аналіз, методи об'єктно-орієнтованого програмування, методи візуалізації наборів даних.

Здійснено огляд теорії електронної обробки даних та існуючих програм обробки даних для вантажних перевезень; здійснено порівняльний аналіз їх функціональних можливостей; проаналізовано структуру модулів бібліотеки *matplotlib*; здійснено огляд принципів коректного графічного представлення даних; реалізовано комп'ютерну програму обробки і аналізу даних для вантажних перевезень.

Матеріали дипломного проекту **рекомендується використовувати** при проведенні досліджень з маніпулювання даними, у навчальному процесі фахівців з аналізу даних та системного програмування, а також у проєктах, пов'язаних з візуалізацією наборів даних.

Прогнозні припущення про розвиток об'єкту та предмету дослідження – застосування в якості системи для роботи з базами даних у сферах, пов'язаних з проєктною діяльністю.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

API – application programming interface

ЗМІ – засоби масової інформації

IT – information technology

БД – база даних

СУБД – система управління базами даних

SQL – structured query language

ЛКМ – ліва кнопка миші

MS – Microsoft

ОС – операційна система

ВСТУП

Актуальність. Ми знаходимось в інформаційній епосі, де об’єм даних в цифровому вигляді весь час збільшуються. Коректна обробка інформації є важливим чинником в сучасних робочих процесах, адже вона має за мету удосконалити прийняття рішень, зробити їх більш точними і надійними. Також при правильному поводженню з даними можна досягти збільшення продуктивності систем, підвищити прибуток, знизити часові, людські та матеріальні затрати. Важливими складовими роботи з даними є простота їх зберігання та розподілення, формування звітів для аналізу та графічного представлення. В теперішній час необхідність обробки даних широко усвідомлюється і віддзеркалюється у всіх сферах діяльності. Виконується робота в діловій обстановці чи в дослідницьких навчальних цілях − системи управління даними використовуються в кожному бізнесі. Це багатоетапний процес, що присутній практично у всіх сферах людського життя.

Іноді доводиться збирати інформацію з різних джерел. Тоді потрібно упорядковувати її так, щоб вона була зручно представлена для досягнення поставленої цілі. Це одне з завдань роботи з зібраними даними − синхронізувати їх та організувати в зручну форму. Існують різні методи обробки інформації, такі як ручна, механічна та електронна обробка даних. Всі сфери, де ми працюємо з доступними великими об’ємами знань, наприклад, освітні заклади, банківська установа чи транспортна компанія, розуміють важливість обробки та аналізу даних. З появою таких областей як наука про дані, аналітика даних та наука про великі дані, методи якісної обробки та аналізу інформації стали надзвичайно важливими. На сьогодні все більше знань збирається для академічних, наукових досліджень, для приватного або загального користування, для освітнього чи комерційного використання. Зібрані дані потребують зберігання, сортування, розподілення, аналізу та візуалізації. Ці процеси можуть бути нескладними або, навпаки, будуть вимагати пошуку рішень, залежно від масштабу збору знань та точності результатів, які потрібно отримати на виході. Час, що витрачається на

досягнення таких цілей, залежить від операцій, які потрібно виконати з наявною інформацією, та від характеру вихідних файлів. Це іноді може стати проблемою при роботі з дуже великим обсягом даних. Кожен етап, починаючи від пошуку і збирання інформації, безпосередньо впливає на результуючі оброблені дані. Для деяких наборів даних при передачі їх третій стороні повинна бути присутня письмова угода про легальність передачі. Це є варіантом запобігання крадіжці та неправильному використанню даних, або їх втраті. Проаналізовані дані набагато легше впорядкувати, що дозволяє зекономити багато місця.

Для спеціалізованих потреб компанії створюють вакансії спеціалістів з обробки даних. Обробка даних має безліч застосувань у різноманітних галузях, таких як освіта, бізнес, наукові дослідження, охорона природи та медицина, тощо. Також вона є необхідною при підготовці даних до їх графічного представлення. Важливість наявності спеціалістів у сфері аналітики зростає з поступовим прогресом у таких галузях, як машинне навчання, системи штучного інтелекту, нейронні мережі, наука про дані, наука про обробку великих даних (*Big Data*), якість та безпеку даних. Доречними тут стануть знання про збір даних, бізнес аналітику, правильне представлення та візуалізацію даних, якісну інформацію, тощо.

Метою даного проєкту є проектування та розробка комп'ютерної програми для обробки та аналізу даних для вантажних перевезень на основі дослідження способів обробки структурованих наборів даних та принципів візуалізації інформації.

Об'єктом дослідження даного дипломного проєкту є процес обробки та аналізу даних.

Предметом дослідження є системи для роботи з наборами даних, які використовуються у різноманітних сферах, де присутня проєктна діяльність; їх можливості роботи з даними; їх типова структура; методи по редагуванню і підготовці даних для їх подальшого графічного представлення.

Методи дослідження. Технології систем управління базами даних використано при створенні об'єктів для доступу, редагування та відображення баз

даних проєкту; порівняльний аналіз застосовується у першому та третьому розділах для виділення особливостей або переваг того чи іншого програмного продукту; методи об'єктно-орієнтованого програмування – використано при розробці функціоналу, що відповідає за роботу вікон програми, а також при створенні методу з'єднання вікон з базою даних; методи роботи з реляційною базою даних застосовано у функціях для редагування вмісту *SQL* файлів та функціях, що відповідають за створення діаграм; методи візуалізації наборів даних були використані при реалізації алгоритмів побудови діаграм, що використовують набори інформації з бази даних програми.

Здійснено огляд теорії електронної обробки даних та існуючих програм обробки даних для вантажних перевезень; здійснено порівняльний аналіз їх функціональних можливостей; розроблену структуру моделі баз даних проєкту; розглянуто можливості методів стандартного інтерфейсу мови програмування *Python*; досліджено особливості запитів на мові *SQL*; досліджено засоби і методи роботи з наборами даних в мові *Python*; досліджено методи мови *Python* для роботи з файлами *MS Word*; проаналізовано структуру модулів бібліотеки *matplotlib*; здійснено огляд принципів коректного графічного представлення даних; розроблено структуру програми управління базами даних; реалізовано комп'ютерну програму обробки і аналізу даних для вантажних перевезень.

Практичне значення отриманих результатів. Результуючі методи можуть використовуватися для ведення обліку процесів; при проведенні досліджень з маніпулювання даними; у навчальному процесі фахівців з аналізу даних та аналітики; при роботі з управлінням даних засобами мови *SQL*; а також у проєктах, пов'язаних з візуалізацією наборів даних. В результаті дипломного проєктування отримано багатоплатформну комп'ютерну програму для обліку вантажних перевезень, що використовує сучасні потужні інструменти для роботи з даними та їх графічного представлення, а також надійну систему управління базами даних. Таким чином, отриманий додаток є сумісним з більшим числом операційних систем та має набагато більше засобів для візуалізації інформації на

відміну від аналогів. Функціонал, описаний у програмних модулях проєкту може бути модифікований або розширений без значних зусиль.

Особистий внесок випускника. Всі результати, представлені у дипломній роботі, отримані випускником особисто. Були прийняті рішення про алгоритми обробки інформації для кожного з наборів даних у структурах баз даних програми.

Апробація отриманих результатів. Теоретичні аспекти отриманих у дипломному проєкті результатів проходили апробацію на міжнародній науково-технічній конференції “Інтелектуальні технології лінгвистичного аналізу”.

Публікації. Царук О.В. Модель кодера-декодера на основі представлень у вигляді тензорних добутків: міжнар. науково-техн. конф. «Інтелектуальні технології лінгвістичного аналізу», 20 – 21 жовтня 2020 р.: тези доп. – К., 2020. – С. 28.

Прогнозні припущення про розвиток об’єкту та предмету дослідження – застосування в якості системи для роботи з базами даних у сферах, пов’язаних з проєктною діяльністю та використанням інформаційних технологій.

1. Обробка та аналіз даних

1.1 Роль обробки та аналізу даних

Сучасний світ вимагає автоматизованої обробки даних. Це досягається за допомогою різного програмного інструментарію. Інформація – це не просто числа і таблиці, якою може володіти організація, а й перевага над своїми конкурентами. Дані можуть оброблятися в різних формах, щоб отримати необхідну інформацію для прийняття того чи іншого рішення.

Обробка даних − це дії, що приводять дані у придатну для використання в тому чи іншому завданні форму. Ці перетворення здійснюються за допомогою заздалегідь спланованої послідовності операцій вручну, автоматизовано чи автоматично. Вихідні дані можуть бути представлені в різних формах, наприклад: рисунок, зображення, графік, таблиця, файл вектору, аудіо, анімація та інші формати. Отримана форма може залежати від програмного забезпечення, що буде її використовувати. Центри обробки даних є важливим компонентом будь-якої системи, адже вони забезпечують обробку, зберігання, доступ, обмін та аналіз даних. Зараз доступні різні послуги з обробки даних. Компанії, що цим займаються, допомагають іншим компаніям створювати документацію щодо безпеки, запобігати втраті чи крадіжці персональних даних, прискорювати виробничий процес, виконувати функції управління. Обробка даних стала популярною темою в тому числі через різні способи використання та закони, пов’язані з розповсюдженням та зберіганням конфіденційною інформацією. Організації та установи збирають різноманітні дані про своїх працівників та клієнтів. Для збору таких речей як особисті рахунки, інформація про доходи, медичні дані, інформація про будівельні об’єкти та інше розроблені спеціальні закони, що регулюють обробку і збір таких даних для конфіденційності

користувачів. Також важливість обробки даних полягає в тому, що представники влади можуть використовувати проаналізовані дані з опитувань, що економить час, а також моніторити економічну ситуацію в регіоні. Аналітики в області урбаністики збирають дані з великої кількості джерел. Наприклад, дані про переміщення громадського транспорту, дані телефонних операторів, соціальні мережі використовують для вивчення функціонування середовища (хто і коли користується, які вулиці найбільш завантажені, і тому подібне) та для передбачення її поведінки після нововведень. В архітектурі спрощують роботу задачі автоматичного проектування. Завдяки надійним даним можна складати звіти, що будуть корисними в роботі. Системи транзакцій залежать від обробки даних у режимі реального часу. Правильно оброблена інформація також допоможе регулювати страхові послуги, вести управління медичними документами. Обробка текстів допомагає зробити документи більш читабельними на зручними для сприйняття.

В електронній обробці дані аналізуються з допомогою комп'ютера: дані та набір інструкцій передаються як вхідні дані, і комп'ютер автоматично обробляє їх відповідно заданого алгоритму. Такий метод є найшвидший та найбільш надійний та точний. Також це дозволяє вагомо зменшити людські затрати. Обробка важливих даних у цифровій формі має відповідати стандартам захисту інформації.

В сучасному світі бізнесу робота з даними є критично важливою. Технології науки про великі дані (*Big Data*) застосовуються для аналізу цих даних, щоб на основі них отримати певні висновки, що допоможуть у прийнятті стратегічних рішень. Важливішим тут є не лише сам обсяг даних, а й те, як вони оброблюються − це мають бути економічно вигідні або, іноді, інноваційні методи обробки інформації. Концепція про великі дані склалася на початку 21 століття. Зараз кожен технологічний гігант використовує її технології та інструменти. Мова йде про об'ємні набори даних, які можуть бути структуровані або не структуровані. Це дані, що кожного дня створюються підприємствами та користувачами різноманітних сервісів та мереж. Аналіз великих даних є процесом вивченням

наборів інформації та знаходження в них закономірностей. Область аналізу є досить широка.

Загалом, аналіз даних - це область математики й інформатики, що займається побудовою і вивченням найбільш загальних математичних методів і обчислювальних алгоритмів отримання знань з експериментальних (в широкому сенсі) даних; процес дослідження, фільтрація, перетворення і моделювання даних з метою знаходження корисної інформації для прийняття рішень. Таким чином, ця сфера являє собою набір методів і додатків, що пов'язані з алгоритмами обробки даних. Іноді вони не мають чітко фіксованої відповіді на кожен вхідний об'єкт. Це відрізняє їх від звичайних алгоритмів, наприклад, від тих, що реалізують сортування чи словник. Від конкретної реалізації класичного алгоритму залежить час його виконання і об'єм витраченої пам'яті, але очікуваний результат його використання строго визначений. В противагу цьому ми очікуємо від нейромережі, що займається аналізом даних і розпізнає цифри, відповіді 5 при вхідному зображенні, на якому знаходиться рукописна п'ятірка, але не можем вимагати такого результату. Більше того, нейромережа здатна помилятися на тих чи інших варіантах коректних вхідних даних. Така постановка задачі з її методами рішення і алгоритмами є недетермінованими, або нечіткими.

Аналіз даних може бути якісний або кількісний. В залежності від типу визначається метод аналізу. В якісних дослідженнях всі нечислові дані, такі як текст або окремі слова, аналізуються. Кількісні ж, на противагу, фокусуються на обчисленні даних і можуть використовувати статистику для досягнення кінцевого результату. До того ж, обидві форми аналізу використовуються пліч-о-пліч. Наприклад, кількісний аналіз може допомогти підтвердити висновок якісного аналізу. Ця сфера допомагає в структуруванні результатів від різних джерел даних.

Методи аналізу великих даних *Big data* допомагають організаціям використовувати свої дані для знаходження нових можливостей, перспектив та шляхів у робочому процесі. Ця технологія стала популярною завдяки таким перевагам:

1. Можливість зниження витрат. Такі технології як *Hadoop* та хмарна аналітика дозволяють зменшити витрати, коли мова йдеться про зберігання великих обсягів даних.

2. Швидкість прийняття рішень. Завдяки технологіям *Hadoo*р та *in-memory* (аналітика з використанням оперативної пам'яті для зберігання даних) компанії можуть миттєво аналізувати інформацію і потім приймати необхідні рішення.

3. Нові послуги. Завдяки здатності оцінювати потреби і вдоволеність споживачів за допомогою аналітики, з'являється можливість надавати клієнтам саме те, що вони хочуть. Таким чином завдяки методам аналітики *Big data* компанії створюють продукти більш відповідні для потреб своїх клієнтів.

Завдяки великому інтересу розробників та інвестицій у технології великих даних спеціалісти, що володіють навичками аналітики, пов'язані з цією сферою, користуються великим попитом. Професії в таких галузях як *Data Analytics* та *Data Engineering* зараз є сильно важливими. IT-менеджери, бізнес-аналітики та розробники програмного забезпечення вивчають інструменти та методи обробки даних, щоб розвиватись в можливостях та створювати нові робочі місця. Багато інструментів використовується в мовах *Python* та *Java*, тому програмісти працюють над новими бібліотеками і додатками, щоб зробити аналіз даних простішим. За допомогою інструментів візуалізації, таких як *Power Bi, Qlikview, Tableau* тощо, люди, що працюють в сфері науки про дані, можуть зручно аналізувати дані і представляти нові маркетингові стратегії.

Одна з компаній, що займається технологіями *Big Data* є *Kyivstar*. Вона представляє свої послуги в цій сфері іншим компаніям, таким як Сільпо, ФК Шахтар, *Uber, Monobank, MOYO* та іншим. Наприклад, для банку Monobank було зменшено кількість проблемних позичальників шляхом розроблення предикативних моделей фінансового скорингу та проведення інтеграції через *API*. На рахунку компанії біля 30 предикативних моделей, що використовують машинне навчання. Вони здатні передбачати стать, вік, сімейний стан, наявність дітей, заплановані подорожі, особисті інтереси та інше. Kyivstar має власну команду, що займається наукою про великі дані, а також свою *Big data-*

платформу. Вона надає доступ до *API* з усім необхідним для швидкої інтеграції з бізнесами партнерів. В 2015 році компанія створила власний курс підготовки спеціалістів у сфері великих даних та машинного навчання.

Аналізом рейтингів переглядів телеканалів серед комерційно привабливої аудиторії в Україні займається компанія *BIG DATA UA*. Вона спеціалізується на вимірюванні поведінки перегляду сервісів *IPTV* (*Internet Protocol Television*) і *OTT* (*Over the Top*, частина технології *IPTV*). Триває постійне вивчення клієнтів, їх потреб і інтересів. Таким чином можна підвищити якість існуючих сервісів. Партнерами компанії є провайдер «ВОЛЯ» і Національна рада України з питань телебачення і радіомовлення.

Послуги обробки даних компанії *CIENCE* допомогли сотням організацій створити важкодоступну цільову аудиторію з високою точністю. Це зарубіжна компанія, один з офісів якої знаходиться в Києві. Організація використовує новітні методи роботи з штучним інтелектом, машинне навчання, програмні засоби для аналітики. З боку персоналу це спеціалізовані працівники та висококваліфіковані куратори. Як результат ретельне, точне та ефективне дослідження продажів. Підхід аналізу даних, керований даними, відображає можливості машинного навчання платформи *CIENCE*. Було відібрано новітні алгоритми в сферах семантичного аналізу, обробки природньої мови, і технологій великих даних. Це дозволяє отримати дані, які нелегко знайти і яких немає в інших платних сервісах по обробці даних. Сфери досліджень включають наступні: демографічні дані, фірмографічні дані, техногрфічні дані, психологічні дані, соціальні графіки.

IT-компанія *INTELLIAS* є однією з найбільших в Україні: вона містить понад 1500 професіоналів. Крім України також є представництва, розташовані у Німеччині та США. Інженери та експерти організації вирішують проблеми глобального ринку за допомогою інноваційних технологічних рішень. Важливою частиною цього є аналітичний відділ. Також *INTELLIAS* забезпечує повний цикл реалізації IT продуктів: ідеї та бачення клієнта перетворюються в продукт, що потім виходить на ринок і розвивається далі.

1.2 Візуалізація даних

Візуалізація даних − це графічне представлення інформації. Використовуючи візуальні елементи, такі як діаграми, графіки, гістограми та карти, візуалізація забезпечує доступний спосіб для наглядного розуміння тенденцій, викидів (частина набору даних, що сильно відрізняється від всього набору) та закономірностей в наборах даних. Існує декілька типів візуалізації:

- Звичайне візуальне представлення кількісної інформації в схематичній формі (кругові і лінійні діаграми, гістограми і спектрограми, таблиці і різні точкові графіки);
- Концептуальна візуалізація дозволяє розробляти складні концепції, ідеї і плани за допомогою концептуальних карт, діаграм Ганта, графів з мінімальним шляхом та інших схожих видів діаграм;
- Стратегічна візуалізація переводить у візуальну форму різні дані про аспекти роботи організацій. Це різноманітні діаграми продуктивності, життєвого циклу і графіки структур організацій;
- Метафорична візуалізація допомагає графічно організувати структурну інформацію за допомогою пірамід, дерев і карт даних. Прикладом є карта київського метрополітену;
- Комбінована візуалізація дозволяє об’єднати декілька важких графіків в одну схему, як в карті з прогнозом погоди.

Візуальна інформація краще сприймається і дозволяє швидко і ефективно донести до глядача думки та ідеї. Фізіологічно, сприйняття візуальної інформації є основою для людини. Більшість інформації людина сприймає саме через зір, вагома частина нейронів головного мозку виконує обробку візуальної інформації, продуктивність людини, що працює з візуальною інформацією трохи вище, на більш ніж 300 відсотків краще людина сприймає інструкцію, якщо вона містить ілюстрації. Наприклад, алгоритм на рис. 1.1 буде легше запам’ятати, ніж відповідне речення «Спочатку натиснути круглу кнопку, потім квадратну, потім трикутну, потім знов круглу».

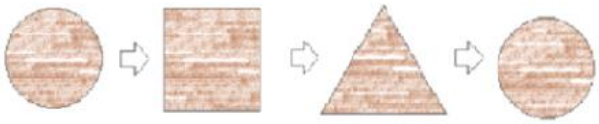


Рис. 1.1. Візуальне представлення алгоритму

Окрім того, що ця інформація чудово сприймається, візуалізація даних має ще декілька переваг:

- акцентування уваги на різних аспектах набору даних;
- виділення зв’язків та відносин, що містить інформація;
- аналіз великого набору даних з важкою структурую;
- зменшення інформаційного завантаження людини;
- однозначність і конкретність демонстрованих даних.

Візуалізація даних представляє факти та цифри чітко, візуально привабливо. Це цінний інструмент у поточному процесі освоєння великих обсягів інформації. Досить складно розібрати набір великих даних для знаходження відповідної корисної інформації, не кажучи вже про пошук закономірностей та тенденцій. Ось чому візуалізація даних є критично важливою для сучасних аналітиків даних та інших користувачів - вона допомагає простіше передавати результати аналізу даних та дозволяє читачам легко бачити тенденції та закономірності. Багато галузей отримують вигоду від цього, наприклад:

- Наука про дані (*Data Science*). Візуалізація даних допомагає спеціалістам виконувати складний аналіз даних, знаходити закономірності та розуміти набори даних. Люди, що працюють в цій галузі, мають багатий інструментарій для графічного представлення даних. Часто для цього використовується мова програмування *Python*;
- Фінанси. Візуалізація є зручним інструментом для інвестиційного світу. Можна прослідкувати популярність товарів, облігацій та акцій;
- Охорона здоров’я. На тепловій карті можна відобразити географічні райони різними кольорами для ілюстрації певних величин. Це зручно для лікарів та епідеміологів, що відстежують розповсюдження різних вірусів;

- Логістика. Візуалізація даних допомагає транспортним компаніям у визначенні найкращих маршрутів доставки;
- Політика. Виборці та працівники передвиборчої кампанії можуть отримати чітке уявлення про те, хто і скільки голосів віддав за конкретного кандидата в кожному регіоні.
- Дослідження. *SciVis*, або «наукова візуалізація», надає вченим краще розуміння експериментальних даних.

Одним з найвідоміших розробників, що працюють в сфері візуалізації, є Альберто Кайро. Це іспанський дизайнер інформації та професор, директор програми візуалізації в Центрі обчислювальних наук організації *UM*. Він очолював сферу інформаційної графіки в програмах та виданнях ЗМІ в Іспанії та Бразилії. Є автор кількох підручників та книг про візуалізацію. Кайр в даний час консультується з компаніями та установами, такими як *Google* і Бюджетне бюро Конгресу, проводить навчання з візуалізації для навчальних закладів країн Європейського Союзу, Євростату, Центрів контролю та профілактики захворювань, Національної гвардії та багатьох інших. Наразі він живе в Маямі, штат Флорида. У своїй новій книзі «*How Charts Lie*» розробник показує не лише те як знаходити оману в графіках, але і як користуватися перевагами хорошої візуалізації, щоб зробити людей, які навчаються аналізу даних, більш підготовленими до роботи у світі даних.

Вагомий внесок у світ статистики та візуалізації вніс заслужений професор Едвард Тафті (*Edward R. Tufte*). Його дослідження стосується статистичних даних та наукової візуалізації. Він є членом Американської статистичної асоціації, Американської академії мистецтв і наук та Товариства технічних комунікацій, має сім докторських ступенів. Є одним з кращих авторів книг по візуалізації даних. Професор описує візуалізацію як інструмент для показу даних; бажання глядача задуматись про суть, а не про методологію; уникнення спотворення того, що мають розповісти дані; відображення багатьох чисел на невеликому просторі;

представлення великого набору даних, пов’язаного в єдине ціле; бажання глядача порівнювати фрагменти даних; служіння достатньо чітким цілям: опису, дослідженню, сортуванню і прикрашанню.

Тафті пропонує шість основних принципів візуалізації: демонстрація порівняння, демонстрація причини, використання різноманітних даних, вдале поєднання частин (тексту, цифр, зображень), встановлення довіри та зосередження уваги на головному. Значна частина популярності Тафті завдячує його першій книзі «*The Visual Display of Quantitative Information*» . Там він висвітлює кілька аспектів того, як створювати привабливі та переконливі графіки даних. Основним пунктом філософії дизайну Тафті є досягнення своїх цілей, а не поверхневі графічні прикраси. Він стверджує, що перше, що слід зробити при завданні відобразити дані даних, це визначити, що саме потрібно показати і яку ідею відобразити. Тафті неодноразово згадує, що використання правдивої інформації є одним з найважливіших аспектів створення переконливого візуального зображення або переконливої презентації. Дозвіл глядачам отримати доступ до вихідного матеріалу додасть їм впевненості у результатах.

Цілі візуалізації − це реалізація основної ідеї інформації. І яким би гарним графік не був, якщо він не допомагає донести ідею − він не потрібен. Процес створення графіка можна поділити на чотири кроки: формування головної думки, вибір типу діаграми, розставлення акцентів, видалення надлишкових елементів. В залежності від мети і датасету можна підібрати найбільш зручний для них графік. Краще прагнути простоти і лаконічності, ніж нагромаджувати зображення графічною різноманітністю. Для специфічних даних можна використати специфічні графіки, але в інших випадках використовуються стандартні: лінійні (*line*), з областями (*area*), з стовпцями або гістограми (*bar*), полярні (*radar*), точкові (*scatter, bubble*), мапи (*map*), дерева (*tree, mental map, tree map*), часові діаграми (*time line, waterfall*).

Важливо зрозуміти, який ефект потрібен для візуалізації даних − виявлення відношень в інформації, демонстрація розподілу даних, композиція чи порівняння даних. Відношення між даними визначається залежностями між ними. Якщо

основне питання полягає у показі того як одна величина змінюється при зміні іншої, то варто показати цю залежність. Для цього підходить лінійний або точковий графік. На рисунку 2 величина точок змінюється в залежності від того скільки значень попадає в неї. Графік показує відношення розходу палива на трасі від розходу в місті за даними датасета, що містить інформацію про популярні автомобілі 1999-2008 років.

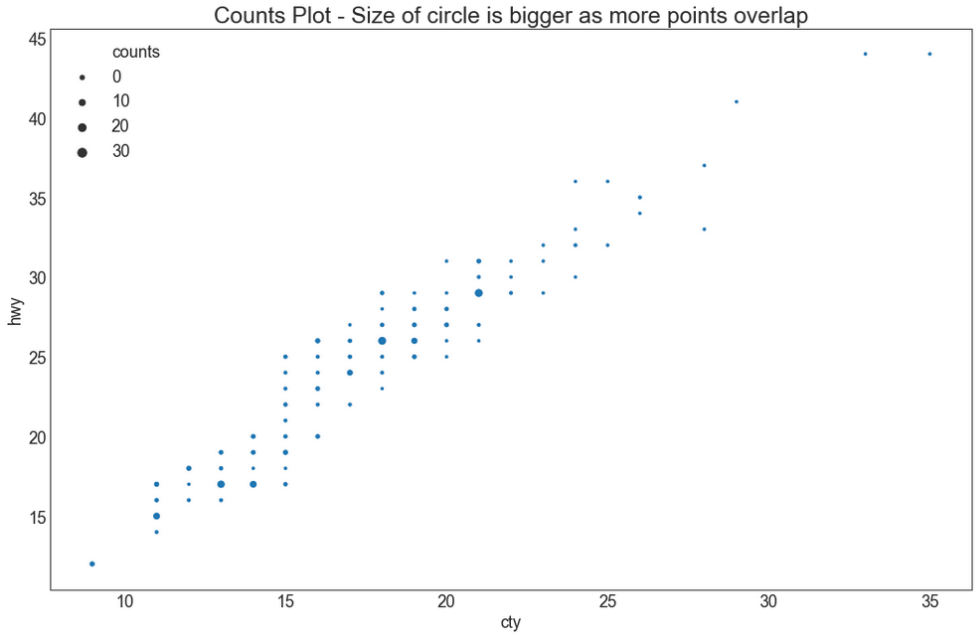


Рис. 1.2. Приклад візуалізації залежності між даними

Розподіл показує те, як дані розподіляються відносно якоїсь величини, скільки об’єктів потрапляє в певні послідовні області числових значень. Це можуть бути задачі з частотою, концентрацією або в якомусь діапазоні. Прикладом є графік щільності, що широко використовується для візуалізації розподілення безперервної змінної. На рисунку 3 показано, як змінюється розподілення пробігу по місту в залежності від кількості циліндрів.

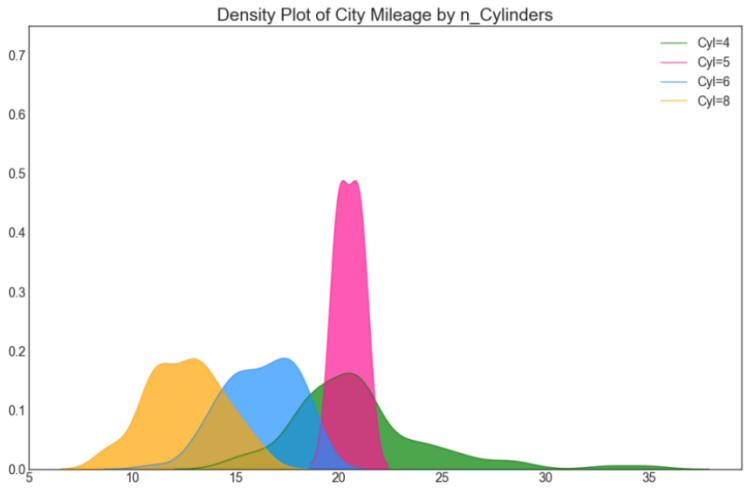


Рис. 1.3. Приклад візуалізації розподілення даних

Композиція даних − це об’єднання даних з метою аналізу загальної картини, порівняння компонентів, що складають якийсь відсоток від цілого. Гістограма категоріальної змінної на рисунку 4 показує розподілення частоти цієї змінної. Було проаналізовано попередній датасет та відібрано типи кузова, що зустрічаються в ньому відносно певних мануфактур. Композицію даних можна побачити за допомогою різних кольорів стовпчиків гістограми.

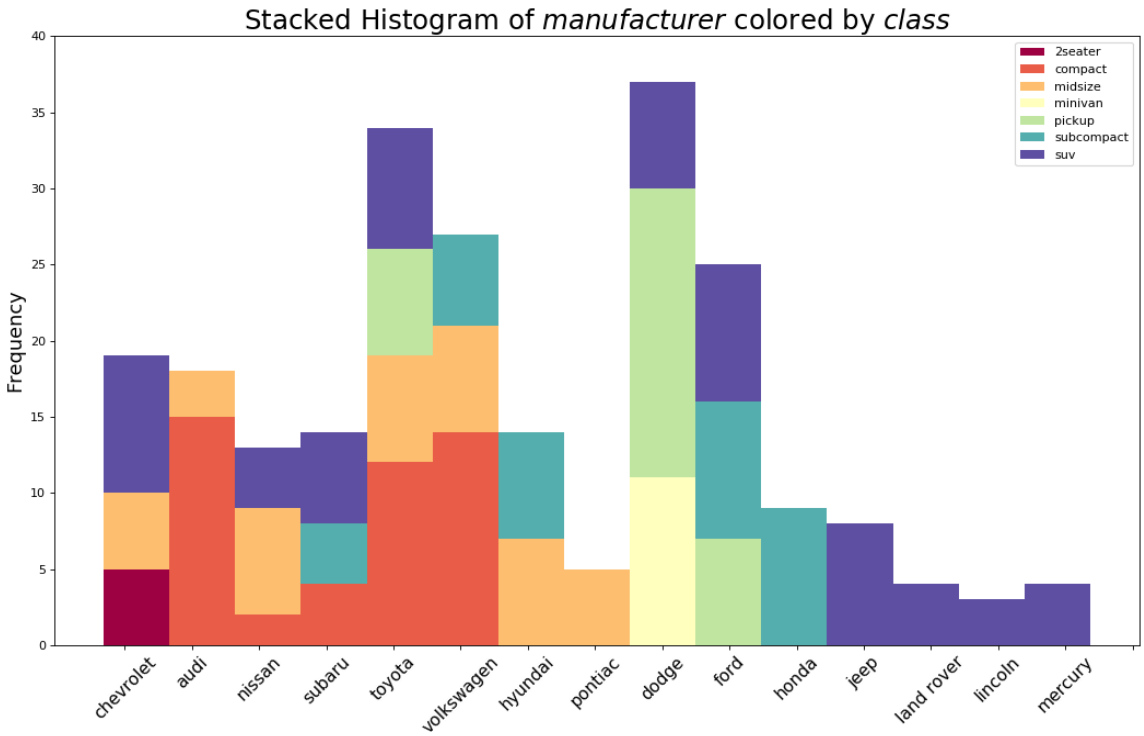


Рис. 1.4. Приклад візуалізації композиції даних

Порівняння даних використовується щоб об’єднати дані з метою знаходження відмінностей між певними показниками, визначення того, як об’єкти відносяться один до одного. Також це порівняння для компонентів, що змінюються з часом.

Ключові фрази для ідей при порівнянні — «більше/менше», «рівні», «змінюються». Прикладом служить графік часового ряду на рисунку 5. На ньому можна побачити, як змінюється потік пасажирів з 1949 по 1969 рік.

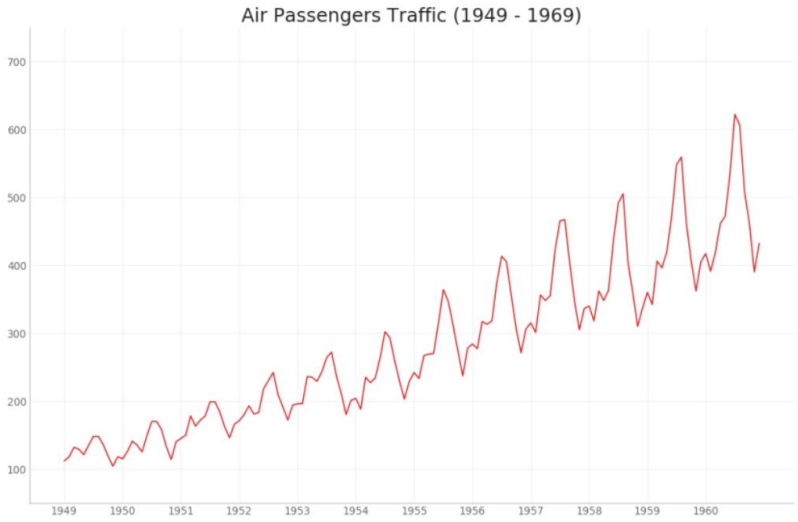


Рис. 1.5. Приклад візуалізації порівняння даних

Рекомендації щодо використання тих чи інших видів графіків для різних цілей наведено в таблиці 1.

Таблиця 1.1

Використання графіків

Ціль візуалізації / Тип даних	Відношення даних	Розподіл даних	Порівняння даних	Композиція даних
Безперервні числові	*line, area, scatter, bubble*	*scatter, bubble*	*line, area, radar*	*stacked line, stacked area*
Безперервні часові	*line, area, radar, scatter, bubble*	*time line, waterfall, radar*		*stacked line, stacked area*
Дискретні	*bar, scatter, bubble*		*bar, pie, doughnut*	*pie, doughnut, stacked bar*
Географічні	*map, line, area*	*map, scatter*	*map, bar*	*map, stacked bar*
Логічні	*tree, mental tree*		*tree map*	

При неправильних мотивах або при неумілому використанні, статистика і візуалізація можуть давати не точні результати. В соціальних мережах чи політиці це може використовуватись для введення в оману, адже цілком можливо хибно показати правильні цифри. Якщо розглянути випадок з стовпчиковою діаграмою, то це може бути випадок, представлений на рисунку 6, де представлені продажі альбому групи *Nirvana* у мільйонах копій. В першому випадку можна побачити, що продажі у США критично малі в порівнянні з продажами по світу, але варто звернути увагу, що відлік стовпців діаграми починається з відмітки в 10 мільйонів. В другому випадку представлені ті ж самі дані, але в коректному поданні.

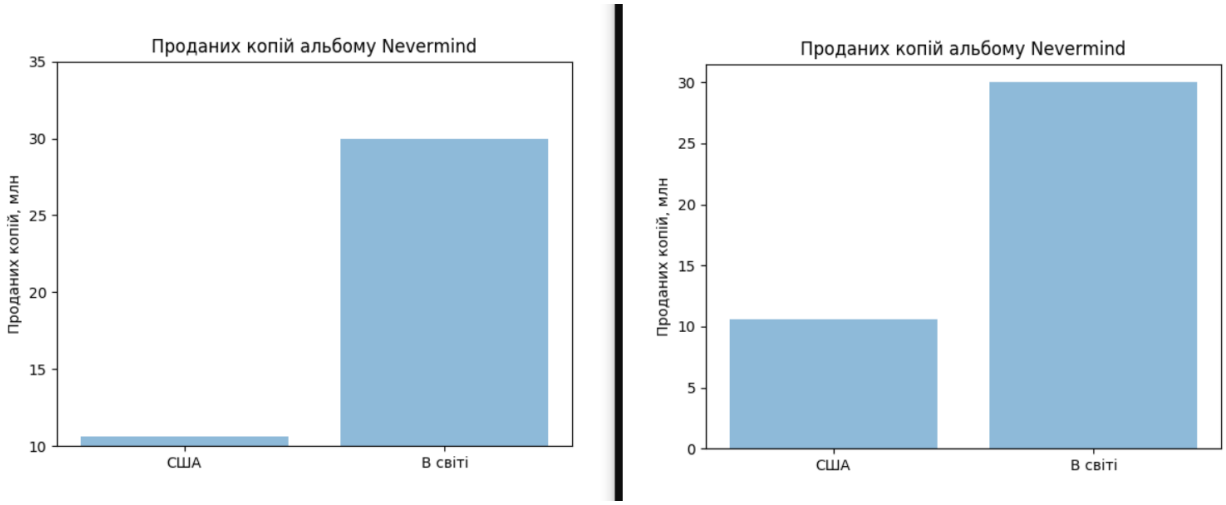


Рис. 1.6. Приклад хибної і правильної візуалізації

1.3. Аналіз існуючих сервісів для вантажних перевезень

Програмне забезпечення «АвтоПеревозки 4» від компанії *AutoSoft* розроблена для обліку роботи автотранспорту підприємства. За допомогою програми можна формувати дорожні листи в більш ніж десяти формах, створювати обліки ремонтних робіт, роботи водіїв, стану шин і акумуляторів, контрагентів і дій з ними, зберігання документів у різних форматах, таких як «*.doc*», «*.xls*», «*.pdf*». База даних програми включає марки і моделі транспортних засобів, державні номери, колір, дата виготовлення та інше. З інформації по організації доступні дані про працівників, їх посади, зміни, друковані форми документів та

прострочені документи. Також доступна система безпеки, що перевіряє логін з паролем та надає інформацію відповідно посади працівника, що користується програмою. Інтерфейс програми показаний на рисунку 7.

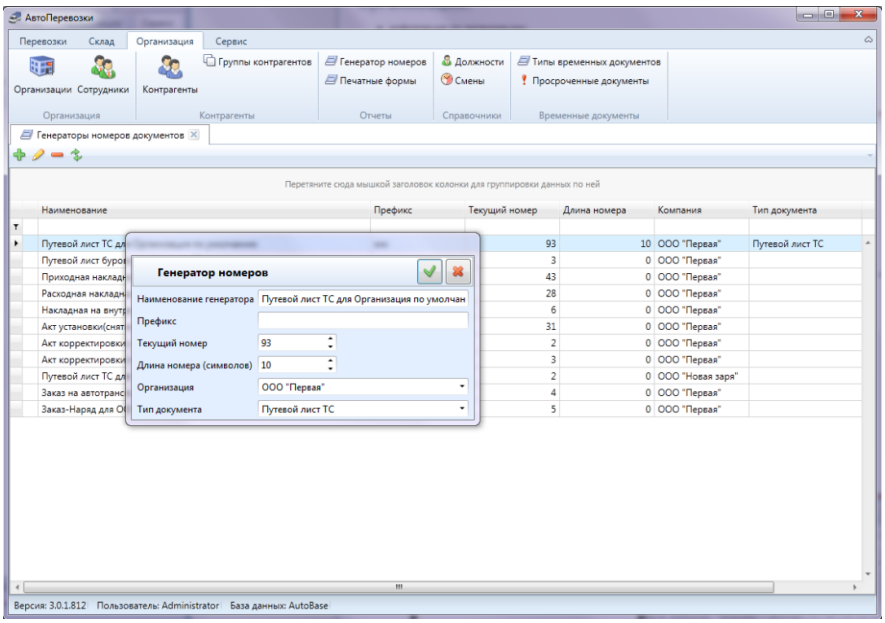


Рис. 1.7. Інтерфейс програми «АвтоПеревозки 4»

Програма «Авто-Поезд» компанії *TEUWORLD* призначена для створення рахунків за вантажні перевезення (контейнери, вантажі), для обліку доходів і витрат для кожного перевезення або угоди. Інтерфейс програми представлений на рисунку 8.

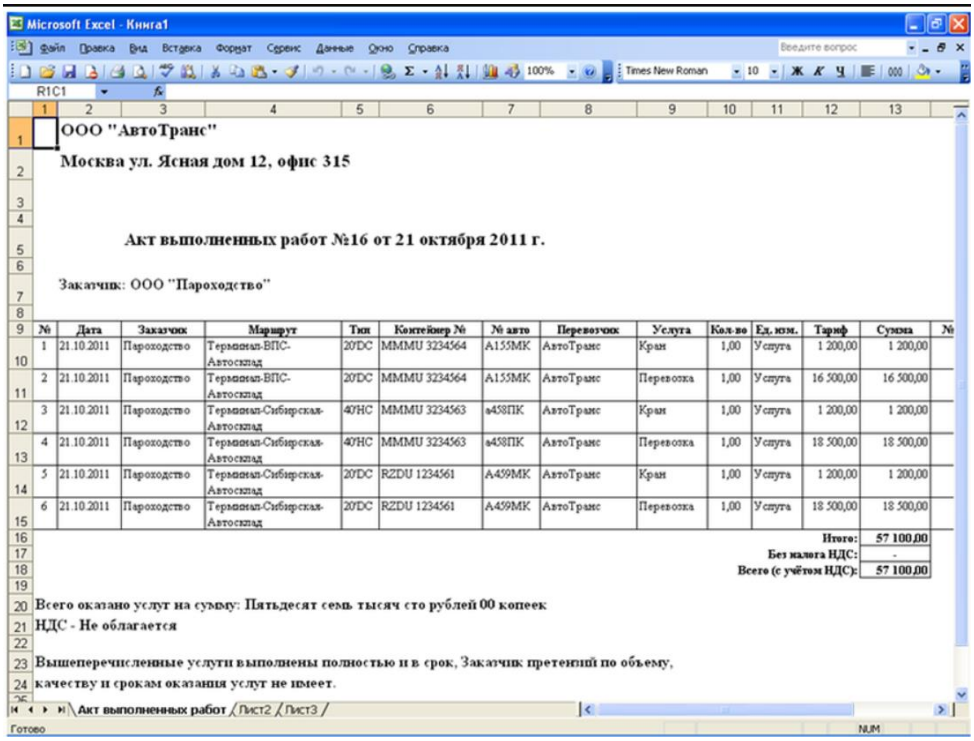


Рис. 1.8. Інтерфейс програми «Авто-Поезд»

В програмі доступні такі дії з документами: друк акту виконання робіт, вивід даних в *Exel*, друк рахунку на оплату послуг, зберігання в базі даних програми копії документа «Міжнародно товарно-транспортна накладна». Також можлива реєстрація різноманітних подій, пов'язаних з автомобільними перевезеннями (оформлення документів, митниця, пропуск, зважування і таке інше).

«Корс Автопредприятие» — професіональна програма обліку автотранспорту, дорожніх листів, витрати палива і мастильних матеріалів, створення маршруту руху транспортних засобів на автотранспортних підприємствах. Типові форми дорожніх листів для різних видів автотранспорту (легкові, вантажні, автобуси, крани, і т.д.). Існує три різних пакети програм: для маленького автопідприємства, для середнього розміру, та для великих автопідприємств. Для більших підприємств доступно набагато більше функцій, наприклад, вивід всіх полів у формі друку, імпорт довідників в файл, облік причепів, розрахунок платежів по кредиту та інше. Інтерфейс представлений на рисунку 9.

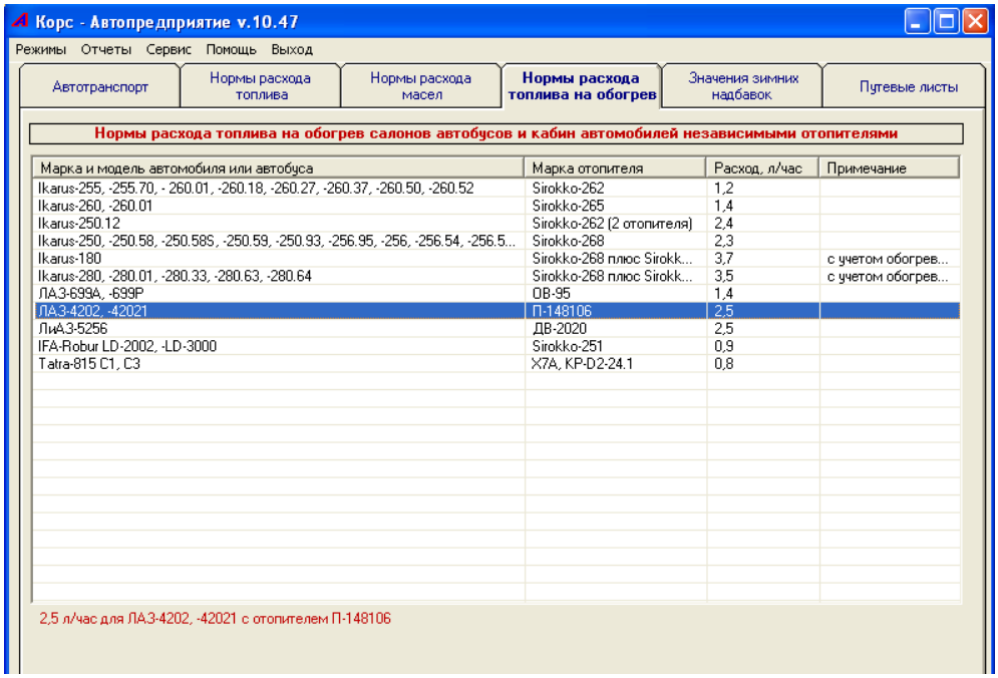


Рис. 1.9. Інтерфейс програми «Корс Автопредприятие»

Було проведено порівняння характеристик трьох програм для роботи з даними вантажоперевезень, в ході яких з'ясовано головні переваги та недоліки кожного з програмних забезпечень, а також розглянуто їх як з боку зручності інтерфейсу, так і з боку нюансів програмної реалізації. Кожне з них має різні форми документів. Результати зіставлення особливостей програм показані в таблиці 2.

Таблиця 1.1

Порівняння програм з обробки та аналізу даних

Характеристики	АвтоПеревозки 4	Авто-Поезд	Корс Автопредприятие
Облік елементів компанії в базі даних	Транспорт, документи, клієнти, ремонт, праця водіїв, пальне, шини та акумулятори, контрагенти, склад	Транспорт, документи, клієнти, праця водіїв, контрагенти, контейнери та грузи, митниця, пробіг, робочі години	Транспорт, документи, клієнти, пальне, мастило, маршрути, замовлення, прибутки і витрати по аналітичним признакам
Наявність документів	15 видів дорожніх листів, документи обліку автопарку	Товарно-транспортна накладна, рахунки на оплату послуг, акти виконаних робіт	Дорожні листи для всіх видів автотранспорту, форми документів для обліку, можливість створення користувацьких шаблонів
Можливість взаємодії з *Excel*	відсутня	Вивід даних в *Excel*	Створення довідників та дорожніх листів на основі файлу *Excel*
Система прав різних користувачів	Налаштування безпеки та прав, список користувачів	відсутня	Призначення паролів та прав користувачів
Різні формати документів	*.doc, .xls, .odf, .ods, .pdf*	*.xls*	*.xls*
Аналітика	відсутня	відсутня	Формування аналітичних звітів та діаграм по витратах матеріалів, прибутках та часу роботи водіїв

З інформації, що наведена в таблиці 2, можна побачити, що кожна з програм має різноманітну базу даних та велику кількість дорожніх листів та інших документів. Найбільше форматів, в яких подаються звіти, має ПЗ «АвтоПеревозки 4». Важливим висновком є те, що розглянуті програми і їх альтернативи мають

невеликий аналітичний модуль, або не мають його взагалі.

В ході дипломного проекту ставиться за мету покращити аналітичний модуль обробки та аналізу даних. Окрім збереження різноманітності корисних полів бази даних, побудови простого інтерфейсу та створення файлів у різних форматах, увага приділяється тому, щоб програма була аналітично корисною. Використовуючи різні прийоми візуалізації даних, можна давати зручніше для сприйняття зображення становища справ у компанії на основі відомої інформації.

1.4. Висновки до розділу

Отже, обробка даних, починаючи від їх відбору, сортування і закінчуючи їх представленням, дуже допомагає у організації робочих процесів у всіх сферах. Робота з аналізом даних була властива людству протягом всього його існування. Зараз аналітика допомагає організаціям аналізувати всі свої дані, щоб знаходити закономірності або показники, що вирізняються з-поміж інших, та генерувати інформацію для автоматизації рішень. Можливості аналітики даних всередині компаній весь час стрімко збільшуються, адже IT-технології розвиваються далі, і організації, що керують даними, стають все більш важливими.

Візуалізація даних важлива для їх групування, розуміння взаємозв’язку даних між собою або їх розподіл відносно певної величини. Це допомагає робітникам в галузі наук про дані виявляти приховані закономірності і те, як вони виникають. В свою чергу бізнес-аналітики застосовують методи графічного представлення знань для знаходження сфер, що потребують вдосконалення, факторів, що впливають на поведінку клієнтів, і для передбачення обсягів продажу. Візуалізація — це потужний інструмент для донесення ідей до кінцевого споживача. Вона є невід’ємною складовою сприйняття і аналізу даних. Як і всі інструменти, її варто використовувати в правильних цілях і місцях. У іншому випадку інформація може сприйматися повільно, або навіть хибно. При коректному застосуванні візуалізація допомагає легше запам’ятати матеріал, дозволяє робити його лаконічним та цікавим.

2. База даних для комп’ютерної програми обробки та аналізу даних для вантажних перевезень

2.1. Структура бази даних комп’ютерної програми обробки та аналізу даних для вантажних перевезень

База даних – це організована структура даних, що використовується для зберігання інформації. Зазвичай бази даних подаються у вигляді групи взаємопов’язаних файлів чи таблиць, що призначені для вирішення конкретного завдання. З цим поняттям тісно пов’язане поняття системи управління базою даних (СУБД). СУБД – це комплекс програмних засобів, що призначається для створення структури нової бази, додавання в неї вмісту, видалення і редагування записів, демонстрації якоїсь групи даних з таблиці. Основним елементом бази даних є таблиця. Стовпчики таблиці називають полями, а рядки записами. Ключ в базі даних представляє собою атрибут або їх набір, по яким можна однозначно визначити потрібний запис. Також ключі служать для зв’язків між таблицями та прискорення обробки даних.

SQL, або мова структурованих запитів, використовується для роботи з даними, що зберігаються в базі даних. *SQL* використовує реляційну алгебру та кортежне реляційне числення. В наш час ця мова сильно утвердилася в роботу бізнес-аналітиків і вимоги до розробників програмного забезпечення по обробці даних. Найбільш популярними СУБД, що використовують *SQL*, є *MS SQL*, *MySQL*, *SQLite*, *PostgreSQL*.

NoSQL-СУБД не використовують реляційну модель структурування даних. Існує багато реалізацій, що вирішують це питання, іноді досить нестандартно.

Ці безсхемні рішення не мають чіткої табличної структури і допускають необмежене формування записів і зберігання даних у вигляді «ключ-значення». *NoSQL* бази даних не використовують загальний формат запитів (як *SQL* в реляційних базах даних). Кожне рішення використовує свою власну систему запитів. Прикладами таких баз даних є *MongoDB, CouchDB, Redis*.

База даних комп'ютерної програми обробки та аналізу даних для вантажних перевезень складається з трьох файлів формату *SQL*: *employees*, *vehicles* та *tasks*. Таким чином, це три окремих бази даних для обліку працівників, транспортних засобів та перевезень. Поле *ID* служить первинним ключем, за яким визначається індивідуальність всіх записів у кожній базі даних.

Структура бази даних employees представлена на рис. 2.1. Таблиця містить таку особисту інформацію про співробітників як П.І.Б., дату народження, номер мобільного телефону, поштову скриньку.

ID	Прізвище	Ім'я	По-батькові	Дата народження	Телефон	Пошта	Дата прийняття	Стаж	Посада
46	Парков	Петро	Петрович	29.03.1991	0987749002	parkov1@gmail.cor	13.01.2021	9	Водій
47	Симончук	Олег	Маркович	13.04.1989	0985576231	symonchyk33@gm	14.09.2016	4	Вантажник
48	Карпенко	Денис	Олексійович	29.10.1994	0932828228	karpenko1@gmail.c	10.12.2020	2	Вантажник
49	Пекорін	Филип	Андрійович	02.02.1992	0965587800	pekorin@gmail.cor	09.02.2018	3	Лікар
50	Нечуй	Анатолій	Олегович	19.11.1989	0567898022	nechyi7@gmail.cor	22.05.2018	10	Диспетчер
51	Довбуш	Олександр	Сергійович	01.12.1975	0969912331	dovbush0@gmail.c	20.03.2015	17	Водій
52	Ботусов	Марк	Панасович	20.08.1995	0959009296	botusov@gmail.co	23.09.2019	3	Менеджер
53	Непалов	В'ячеслав	Глібович	09.12.1990	0928983222	nepalov89@gmail.c	09.11.2016	8	Менеджер
54	Кісєльов	Олег	Ігорович	02.04.1995	0980122018	kiselev007@gmail.c	18.05.2020	4	Водій
55	Цапенко	Микита	Анатолієвич	15.02.1997	0955787992	tsapenko1@gmail.c	20.02.2021	1	Вантажник
56	Рентов	Сергій	Костянтинович	05.03.1997	0663432309	rentov1@gmail.cor	20.02.2021	1	Вантажник
57	Цибуля	Олег	Максимович	30.05.1986	0759822304	tsubylia99@gmail.c	06.06.2020	9	Водій
58	Печаль	Симон	Глібович	13.12.1990	0950962999	pechal11@gmail.cc	03.02.2019	3	Водій
59	Гарна	Валерія	Владиславівна	11.10.1996	0931111888	garna123@gmail.cc	19.04.2020	1	Менеджер
60	Друкаренко	Віталій	Карпович	18.03.1993	0929898791	drukarenko1@gma	09.08.2020	4	Водій

Рис. 2.1 Таблиця бази даних працівників

Окрім цього є поля, пов'язані з роботою: дата працевлаштування, стаж роботи і професія. Дати необхідно вводити у форматі «ДД.ММ.РРРР», інакше поле в записі залишиться пустим. Стаж являє собою ціле число. Інші поля текстові. У вікні додавання та редагування записів для поля «Посада» можна обрати наступні варіанти: водій, механік, вантажник, диспетчер, лікар, охорона, бухгалтер, менеджер. При необхідності можна додати інші варіанти професій. Структура бази даних *vehicles* представлена на рис. 2.2. і містить поля з описом транспортних засобів. Це державний номер, бренд та модель, рік випуску моделі, колір та вид транспорту.

ID	Держ. номер	Марка	Модель	Дата випуску	Колір	Група
1	AA 0451 IX	DAF	XF 440	2014	синій	Фура
2	AM 3251 AA	DAF	XF 105	2014	синій	Фура
3	AM 4156 BI	DAF	XF 480	2006	чорний	Фура
4	AC 0455 AB	DAF	XF 106	2015	зелений	Фура
5	AC 2932 AX	DAF	XF 105	2012	білий	Фура
6	AA 8417 CB	DAF	XF 480	2017	синій	Фура
7	BC 3220 II	SCANIA	R 500	2011	червоний	Фура
8	AE 8599 KC	SCANIA	114 P340	2006	синій	Фура
9	AA 6356 CA	SCANIA	G 420	2014	червоний	Фура
10	AA 5196 CO	SCANIA	G	2019	білий	Фура
11	AA 1300 SE	SCANIA	R 420	2012	білий	Фура
12	AA 3963 AB	SCANIA	R 420	2014	синій	Фура
13	AA 7888 CB	SCANIA	R 400	2011	білий	Фура
14	AA 4643 BX	FORD	FOCUS	2010	чорний	Легковий авт.
15	AA 4988 CA	FORD	FOCUS	2010	чорний	Легковий авт.
16		JCB	TLT	2007		Спецтехніка
17		JCB	TLT	2007		Спецтехніка

Рис. 2.2 Таблиця бази даних транспорту

При додаванні або редагуванні запису з бази даних транспортних засобів надаються наступні варіанти поля «Група»: фура, вантажівка, фургон, легковий автомобіль, спецтехніка, автобус. При необхідності можна ввести інакший варіант групи.

Таблиця бази даних *tasks* показана на рис. 2.3 і рис. 2.4, і має найбільшу кількість інформації. Поле статус має три варіанти: «Виконаний», «В процесі», «Скасований». Поля дати відправки вантажу і дати його прибуття заповнюються по аналогії з форматом дати у базі даних працівників. Також наявні поля про локацію перевезення: місто і країна початку вантажоперевезення, місто і країна кінцевого пункту. В полі «Компанія» записується назва компанії, для якої надаються послуги по перевезенню. Для поля «Тип вантажу» в програмі пропонуються 7 значень: насипний, порошкоподібний, наливний, газоподібний, штучний, швидкопсувний, негабаритний. Всі вони є формі створення запису. У полі «Вантаж» конкретизується, який саме вантаж перевозиться. Поле «Примітка» має за мету показати додаткову важливу інформацію про вантаж. Заповнити його можна одним із значень: вибухонебезпечний, вогненебезпечна рідина, вогненебезпечна речовина, окислююча речовина, токсичний, радіоактивний, корозійна речовина, небезпечний (інше), високоцінний вантаж, довгомірний, надважкий, або написати власний варіант. В полі «Транспорт» міститься державний номер вантажівки, а в полі «Водій» ім’я особи, що перевозить вантаж. Час поїздки, витрати і прибутки задаються у форматі числа з плаваючою комою.

ID	Статус	Дата відправ.	Дата прибуття	Місто відправ.	Місто прибуття	Країна відправ.	Країна прибуття	Компанія
1	Виконаний	13.05.2021	13.05.2021	Київ	Львів	Україна	Україна	ROSHEN
2	Виконаний	09.05.2021	10.05.2021	Київ	Білосток	Україна	Польща	ER-TEC
3	Виконаний	20.05.2021	20.05.2021	Харків	Київ	Україна	Україна	TRADEAUX
4	Виконаний	03.05.2021	04.05.2021	Львів	Будапешт	Україна	Угорщина	Cosy&Trendy
5	Виконаний	19.05.2021	20.05.2021	Львів	Печ	Україна	Угорщина	Cosy&Trendy
6	Виконаний	20.05.2021	21.05.2021	Київ	Брно	Україна	Чехія	TRADEAUX
7	Виконаний	01.05.2021	02.05.2021	Харків	Львів	Україна	Україна	УКРБУД
8	Виконаний	12.05.2021	12.05.2021	Варшава	Львів	Польща	Україна	ER-TEC
9	Виконаний	06.05.2021	06.05.2021	Будапешт	Львів	Венгрія	Україна	TRADEAUX
10	Виконаний	21.05.2021	21.05.2021	Печ	Ужгород	Венгрія	Україна	УКРБУД
11	Виконаний	23.05.2021	23.05.2021	Брно	Львів	Чехія	Україна	TRADEAUX
12	Виконаний	15.05.2021	15.05.2021	Київ	Одеса	Україна	Україна	СВІТ МЕБЛІВ
13	Виконаний	22.05.2021	22.05.2021	Київ	Кривий Ріг	Україна	Україна	TRADEAUX
14	Виконаний	03.05.2021	04.05.2021	Харків	Білосток	Україна	Польща	ER-TEC
15	Виконаний	06.05.2021	07.05.2021	Варшава	Київ	Польща	Україна	PharCos
16	Виконаний	23.05.2021	23.05.2021	Київ	Кривий Ріг	Україна	Україна	ROSHEN
17	Виконаний	02.05.2021	03.05.2021	Миколаїв	Варна	Україна	Болгарія	Епіцентр К
18	Виконаний	04.05.2021	04.05.2021	Бухарест	Одеса	Румунія	Україна	TRADEAUX

Рис. 2.3 Таблиця перевезень (частина перша)

Тип вантажу	Вантаж	Примітка	Транспорт	Час маршруту	Водій	Витрати	Прибутки
Штучний	Солодощі		AA 7957 CE	8.5	Лапенко І.Т.	2398.0	8218.0
Насипний	Інфікована сирови		AA 5071 EH	18.5	Комар Д.Я.	4289.0	43400.0
Насипний	Цемент		AA 0451 IX	5.0	Дуб Т.П.	1992.0	12200.0
Штучний	Посуд		AB 1007 BE	7.5	Чуб П.О.	3267.25	27100.0
Штучний	Посуд		AA 1201 CK	9.75	Крачевський Л.П.	3892.75	28116.0
Насипний	Картопля		AO 4004 BO	16.5	Вандін М.О.	6178.25	39666.0
Насипний	Вугілля		BC 3220 II	16.5	Комар Д.Я.	5982.0	31250.0
Наливний	Натрій	Вогненебезпечна р	AA 5071 EH	6.5	Комар Д.Я.	2178.5	25111.0
Насипний	Деревна стружка		AB 1007 BE	9.5	Чуб П.О.	3498.0	31200.0
Штучний	Металеві труби		AA 1201 CK	8.0	Крачевський Л.П.	3661.0	28000.0
Швидкопсувний	М'ясо		AO 4004 BO	10.25	Вандін М.О.	4112.0	31200.0
Штучний	Меблі		AA 0451 IX	9.0	Храпач Д.А.	2910.0	24000.0
Штучний	Автомобілі	Високоцінний груз	AA 7957 CE	9.0	Храпач Д.А.	3279.0	28900.0
Штучний	Важкі метали	Токсичний	AE 8599 KC	21.0	Довбуш О.С.	8890.0	82645.0
Штучний	Медикаменти		AE 8599 KC	12.0	Довбуш О.С.	3866.0	37250.0
Штучний	Солодощі		AA 1300 SE	6.0	Українець П.В.	2900.0	14200.0
Насипний	Картопля		AO 4004 BO	16.0	Прохоров В.А.	6200.0	41000.0
Насипний	Білий фосфор	Вогненебезпечна р	AO 4004 BO	12.0	Прохоров В.А.	4100.0	43600.0

Рис. 2.4 Таблиця перевезень (частина друга)

2.2 Методи роботи з даними в системі *SQLite*

Бази даних комп'ютерної програми обробки та аналізу даних для вантажних перевезень керуються за допомогою *SQLite*. Це багатоплатформна система, що підтримує багато команд *SQL*. Ця СУБД має дуже високу популярність. Крім того, вона дуже надійна: при випуску нової версії вона проходить через ряд серйозних автоматичних тестів (близько двох мільйонів) і має сто відсоткове покриття коду тестами.

Для роботи з *SQLite* використовується бібліотека *sqlite3*. В кожному з модулів програми, що відповідають за обробку тієї чи іншої бази даних міститься користувацький клас *DB*, в якому створюється одна з трьох баз даних проекту.

Клас містить 2 методи, а саме конструктор *__init__ (self)* і функцію запису *insert(self, columns)*. Вони використовують запити *SQL*. Конструктор класу *DB* в програмному модулі по роботі з даними працівників виглядає наступним чином:

```
    self.conn = sqlite3.connect('employees.sql')
    self.c = self.conn.cursor()
    self.c.execute(
    '''CREATE TABLE IF NOT EXISTS employees (id integer primary key, surname text, name text, patronymic text, birthday text, phone text, email text, enrolment text, experience integer, position text)'''
    )
                                    self.conn.commit()
```

Метод connect створює об’єкт підключення до бази даних, назву якої було передано в його аргументи. Після створення з’єднання з необхідним файлом, потрібно створити об’єкт Cursor, що являє собою основний спосіб роботи з базою даних. Вираз «*CREATE TABLE IF NOT EXISTS employees*» значить створення файлу *SQL* при умові, що цього файлу ще не існує. Після цього в запиті йде перелік полів таблиці та їх типів. У СУБД *SQLite* доступні такі типи даних:

- *NULL* (пусте значення);
- *integer* (цілочисленне значення, зберігається в 1, 2, 3, 4, 6 або 8 байтах залежно від величини значення);
- *real* (числове значення з плаваючою комою, зберігається у 8 байтах);
- *text* (текстове значення, зберігається з використанням кодування);
- *blob* (значення бінарних даних, що зберігаються в такому ж вигляді, в якому вони були введені).

Після кожного редагування бази даних необхідно викликати метод об’єкту підключення commit() для підтвердження та збереження результату. Для запису вмісту в таблицю був створений метод insert:

```
      def insert(self, columns):
           self.c.execute(
               '''INSERT INTO employees (surname, name, patronymic, birthday, phone, email, enrolment, experience, position) VALUES (?,?,?,?,?,?,?,?,?)''', (columns))
           self.conn.commit()
```

Він використовує команду *INSERT INTO* для запису даних зі змінної *columns* в поля таблиці *employees*, які зазначені в дужках. Список columns передається з вікна створення записів. Для редагування записів використовується команда *UPDATE*. Приклад методу редагування даних про співробітників:

```
    def edit_records(self, columns):
        self.database.c.execute(
                                '''UPDATE employees SET surname=?, name=?,
patronymic=?, birthday=?, phone=?, email=?,enrolment=?, experience=?, position=?
WHERE id=?''',
        (columns[0], columns[1], columns[2], columns[3], columns[4], columns[5],
columns[6],columns[7],columns[8], self.tree.set(self.tree.selection()[0], '#1')))
        self.database.conn.commit()
    self.database.conn.commit()
```

Команда *SET* встановлює нове значення, команда *WHERE id=?* знаходить запис, який буде редагуватись. Визначається він за допомогою методу *selection*, який передає виділений користувачем рядок таблиці, який він бажає змінити.

В методах, що відповідають за візуалізацію, у всіх *SQL*-запитах використовуються команди *SELECT* та *FROM*. Наприклад, для виділення всіх записів у таблиці vehicles використовується запит:

```
SELECT * FROM vehicles
```

Для пошуку записів таблиці за певною частиною значення поля застосовується команди *WHERE LIKE*. Пошук, виділення та збереження в змінній записів перевезень у таблиці tasks, які були виконані в поточному місяці, виглядає наступним чином:

```
self.now = datetime.datetime.now()
now_time = '%' + str(self.now.month) + '.' + str(self.now.year)
        sql='SELECT end_date, spending FROM tasks WHERE end_date LIKE ?'
        args=[now_time]
        df = pd.DataFrame(self.database.c.execute(sql,args))
```

Для методу *execute* створюється запит і аргумент для запиту, що буде вставлений замість знаку питання. Виділені будуть такі поля прибутків *profit*, в записах яких у полях кінцевої дати перевезення *end_date* є збіг з текстом змінної *now_time*. В цьому можна впевнитись, вивівши частину вмісту *df* на екран (рис. 2.5).

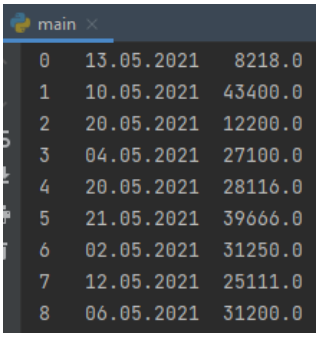


Рис. 2.5 Результат виконання запиту

Для знаходження прибутків від імпорту та експорту в модулі по роботі з базою даних перевезень використовуються такі *SQL* запити:

```
sql_import = "SELECT end_date, spending FROM tasks WHERE end_country LIKE 'Україна' AND end_date LIKE ?"

sql_export = "SELECT end_date, spending FROM tasks WHERE NOT end_country LIKE 'Україна' AND end_date LIKE ?"
```

Як і в попередньому прикладі, в аргументи передається змінна *now_time*, що відповідає за поточний місяць і рік. Додались нові умови: пошук відбувається за двома критеріями, при чому в першому випадку поле *end_country* має набувати значення «Україна», а в іншому навпаки – набувати будь-якого значення, окрім заданого.

2.3. Методи обробки даних засобами мови *Python*

Для відображення вмісту всіх баз даних (див. рис. 2.1-2.4) в проекті використовується модуль *Tkinter* і його розширення (або обгортка) модуль *Ttk*. Перший служить для створення вікон, в яких будуть знаходитись таблиці з даними та засоби для управління ними. За допомогою другого створюється сама таблиця, в яку будуть завантажуватись поля записів з того чи іншого файлу *SQL*. *Tkinter* є багатоплатформною бібліотекою для розробки нескладних графічних інтерфейсів на мові програмування *Python*. Починаючи з версій *Python* 2.7/3.1.2 модуль *Ttk* входить у склад *Tkinter*.

Створення таблиці відбувається за допомогою методу *Treeview*:

```
    self.tree = ttk.Treeview(self, columns=('id', 'number', 'brand', 'model',
'date', 'color',                        'group'), height=17, show='headings')
```

Таким чином задаються змінні колонок і висота таблиці. Щоб задати ширину, оформлення використовується метод *column*, а для тексту назви колонок метод *heading*:

```
    self.tree.column('id', width=40, anchor=tk.CENTER)
    self.tree.heading('id', text='ID')
```

Прогортати вміст таблиць можна за допомогою клавіатури, колеса миші або сенсорної панелі. Так як в таблиці перевезень міститься велика кількість записів і полів, до неї додаються елементи *scrollbar*, що дозволяють зручно керувати елементом *Treeview*. Приклад створення горизонтальної лінії для прогортування:

```
    self.scroll = ttk.Scrollbar(self, orient='horizontal', command=self.tree.xview)
            self.scroll.pack(fill = tk.X)
            self.tree.configure(xscrollcommand=self.scroll.set)
```

Також для зручності можна змінити ширину колонки під час роботи програми, зсунувши розташування грані заголовкового поля таблиці за допомогою миші.

Основними бібліотеками для обробки і візуалізації даних в проекті є *pandas* і *matplotlib*. *Pandas* – це програмний модуль, що створений для аналізу даних та роботи з ними. Випускається з ліцензією *BSD* та має відкритий код. *Pandas* це швидкий, потужний, гнучкий та зручний інструмент, який часто використовується в таких галузях як *data science*, *data analysis* і завданнях машинного навчання. Як один з найпопулярніших модулів в цих сферах, він чудово поєднується з іншими бібліотеками для роботи з даними всередині системи мови *Python*. Однією з головних особливостей цієї бібліотеки є можливість виділення даних з файлів *CSV*, *TSV*, *SQL*, *JSON* та інших, і створення об’єктів мови *Python*, що будуть зберігати рядки і поля з джерел даних. Такий об’єкт має назву «дата фрейм» (*DataFrame*) і містить низку методів для доступу і роботи з наявними записами. Для читання інформації з файлів використовуються методи з префіксом «*read_*», а для запису в файл методи з префіксом «*to_*» (рис. 2.6).

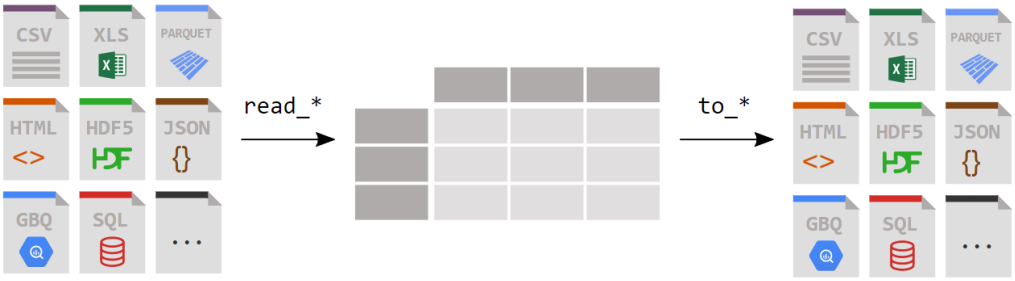


Рис. 2.6 Схема можливостей читання і запису файлів з допомогою pandas

У *DataFrame* можна перетворити список чи словник *Python*, масив модулю *Numpy*, локальний файл або файл за *URL*-адресою. Окрім стандартних засобів виводу вмісту даних на екран, *DataFrame* має змінну shape, що повертає кількість рядків і стовпців, метод *info()*, що повертає таку інформацію як назва класу, діапазон індексів, назви колонок, їх типи і кількість ненульових значень в них, використану пам'ять, методи *head(n)* і *tail(n)* (де *n* – ціле число), що повертає перші n або останні n рядків. Для числових полів є такі методи:

- *describe()* показує зведену статистику;
- *mean()* повертає середнє значення;
- *count()* повертає кількість ненульових значень;
- *max()* і *min()* повертають відповідно найбільше і найменше значення;
- *median()* повертає медіану кожного поля;
- *std()* повертає стандартне відхилення.

Також pandas пропонує методи для індексування, виділення, групування даних, багаторівневого індексування, порівняння, перерізу, та об'єднання об'єктів *DataFrame*, їх переформатування, роботу з текстовими і пропущеними даними, широкі можливості роботи з датами та часовими значеннями, налаштування стилю таблиць для середовища *Jupyter Notebook* та інше.

Для візуалізації даних, оброблених методами pandas, чудово підходить пакет інструментів *matplotlib*. Це бібліотека для мови програмування *Python* з повною підтримкою двовимірного зображення і обмеженими можливостями в створенні тривимірного зображення. Вона широко використовується серед розробників, що працюють з мовою *Python* і цікавляться науковою сферою. *Matplotlib* дозволяє

створювати високоякісні зображення різних форматів: *JPEG*, *PDF*, *PNG*, *PostScript*, *SVG*, *SVGZ*, *TIFF*.

Архітектура модулю *matplotlib* розділена на три шари. Кожен шар вищого рівня має засоби взаємодії з нижчим шаром, але нижній шар не має можливості взаємодіяти з вищим шаром. Нижній шар називається Backend *Layer*, середній *Artist Layer*, і найвищий має назву *Scripting Layer*. Перший реалізовує абстрактні класи інтерфейсу: *FigureCanvas*, що є поняттям поверхні для рисування, *Renderer*, що відповідає за рисування і *Event*, що оброблює дії користувача. Практично все, що відображається на рисунку, є елементом шару *Artist*. Коли відбувається відображення рисунку, всі елементи *Artist* наносяться на основу-полотно (*Canvas*). Рівень скриптів реалізований у вигляді інтерфейсу *matplotlib.pyplot*. *Pyplot* представляє собою набір команд та функцій, що роблять синтаксис графічних команд *matplotlib* схожими на команди, що використовуються в середовищі *MATLAB*.

На користувачу бібліотеки лежить управлінська роль: що саме потрібно створити і якими інструментами. Для складних задач важливо розуміти більш низькорівневі методи модулю, але в більшості випадків зручніше використовувати інтерфейс *pyplot*. Стандарт його виклику наведено нижче:

```
import matplotlib.pyplot as plt
```

Рисунки створюються шляхом послідовного виклику команд: в інтерактивному режимі (в консолі), або в скрипті (файл *python*). Структура типового рисунку та її пояснення показані на рис. 2.7 та рис. 2.8 відповідно. Об’єкти «*ticks*» є позначками на осях *X* та *Y*.

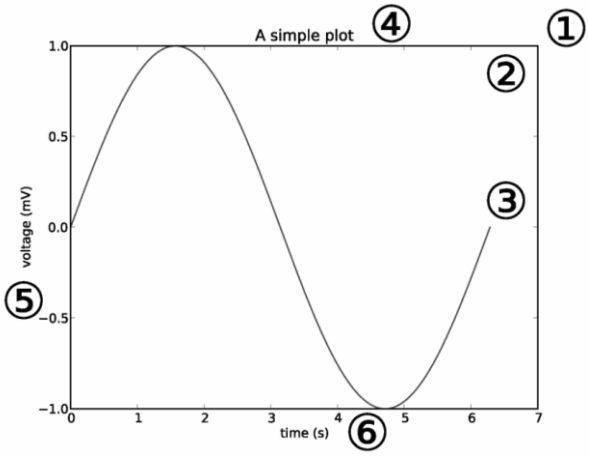


Рис. 2.7 Структура типового рисунку

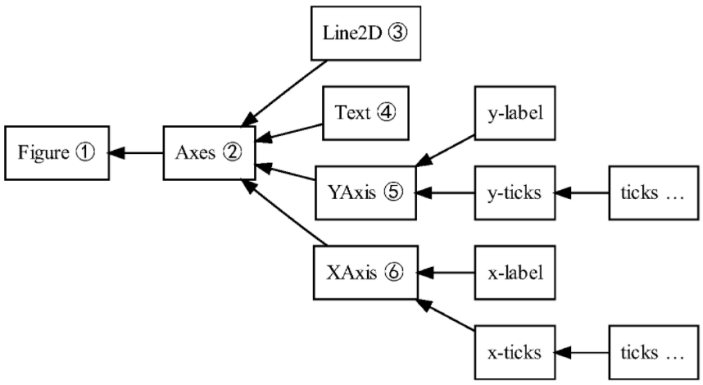


Рис. 2.8 Ієрархія об’єктів на рисунку

В проекті для кожного графіку відведений свій метод в класах вікон з базами даних. Елементи графіки, такі як лінії, точки, фігури накладаються одна на одну послідовно. Таким чином наступні шари перекривають попередні, якщо вони мають спільні розташування на рисунку, але це можна відрегулювати параметром *zorder*. В *matplotlib* існує поняття “поточної області” (*current axes*). Не зважаючи на те, що областей рисування може бути декілька, одна з них завжди буде поточна. Отримати її можна за допомогою методу *plt.gca()* (*get current axes*). Найбільш головним елементом в бібліотеці є рисунок *Figure*. Тому створення наукової графіки необхідно починати саме з створення об’єкту *Figure*. При цьому вказується розміри і властивості основного полотна, на якому буде створюватись графік. Створити об’єкт Figure дозволяє метод *plt.figure()*, а отримати поточний об’єкт метод *plt.gcf()* (*get current figure*). Для відображення поточного графіку використовується метод *plt.show()*. Для збереження рисунку існує метод *plt.savefig()*. Він зберігає поточну конфігурацію поточного рисунку в файл, розширення якого можна задати за допомогою параметру *fmt*.

Проект в загальному містить дванадцять функцій, що створюють відповідне число графіків. Одна з них – це *plot_cargo()*, що працює з базою даних перевезень і демонструє впорядковану гістограму з даними про кількість відправлень кожного з типів вантажів. Загальний алгоритм такого методу представлений на рис. 2.9.

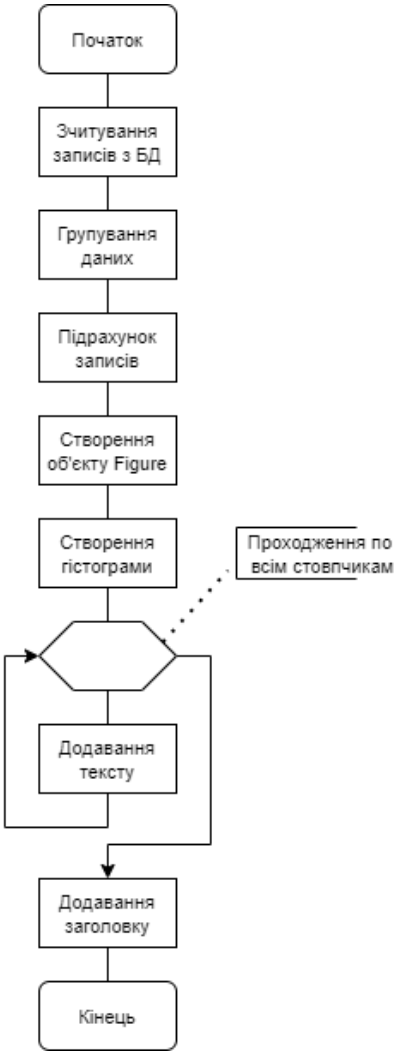


Рис. 2.9 Схему алгоритму побудови впорядкованої гістограми

В метод *read_sql()* передається вираз на мові *SQL* та об’єкт з’єднання з базою даних. За допомогою методу *groupby()* дані розділяються на групи і повертається об’єкт *groupby* для значень поля *type_cargo*. За допомогою методу *size()* проходить підрахунок кількості записів для кожної з груп *DataFrame*. Метод *reset_index()* служить для перезапису індексів об’єкту, і визначає їх як список цілих значень від нуля до числа кількості записів у об’єкті. За допомогою методу *sort_values()* можна відсортувати записи в *DataFrame* за будь-якою віссю. Перший його аргумент – назва поля, за яким проводиться сортування, а другий, *inplace*, показує, що об’єкт змінюється. Значення аргументу *False* використовується тоді, коли об’єкт просто копіюється в інший об’єкт, і для даної задачі сам об’єкт не потребує редагування.

```
df_raw = pd.read_sql('SELECT type_cargo FROM tasks', self.database.conn)
        df = df_raw.groupby('type_cargo').size().reset_index(name='counts')
```

```
df.sort_values('counts', inplace=True)
```

Для візуалізації отриманих даних (рис. 2.10) створюються змінні *fig*, *ax*, що являють собою об'єкти *Figure* і *Axes*. Метод *plt.subplots* розміщує область, на якій створюються графіки, на регулярну сітку. Об'єкт *Figure* може містити декілька контейнерів *Axes*. Аргумент *figsize* задає довжину і ширину об'єкту *Figure*, *facecolor* встановлює фоновий колір, а *dpi* задає розширення графіку, тобто кількість пікселів на дюйм. Всі ці аргументи є необов'язковими. Для створення вертикальних ліній гістограми в даній функції використовується метод *vlines()*. Щоб показати різні групи вантажів на осі *X*, в аргумент *x* передається колонка *DataFrame type_cargo*. По осі *Y* задається мінімальне значення ymin та максимальне ymax, в яке передається колонка counts, що містить значення кількості вантажоперевезень з кожним типом вантажу. Серед інших аргументів – колір стовпчиків (*color*), прозорість (*alpha*) і ширина стовпчика (*linewidth*). Далі, для отримання точної інформації від діаграми, над стовпчиками додаються значення. Це реалізується за допомогою циклу, в якому створюються елементи *plt.text*. Функція *enumerate()* застосовується для ітераційних об'єктів і створює об'єкт-генератор, що формує кортежі, які містять по дві складові – індекс елемента і сам елемент. Таким чином в тілі циклу в змінну і передається індекс, а в змінну counts об'єкт:

```
    fig, ax = plt.subplots(figsize=(8, 4.5), facecolor='white', dpi=80)
    ax.vlines(x=df.type_cargo,    ymin=0,    ymax=df.counts,    color='firebrick',
alpha=0.7, linewidth=30)
    for i, counts in enumerate(df.counts):
        ax.text(i,   counts   +   0.3,   counts,   horizontalalignment='center',
fontsize=11)
```

Рис. 2.10 Впорядкована гістограма

Метод *text()* в циклі приймає аргументи розміщення по осі *X*, по осі *Y*, текст та його оформлення. Для додавання заголовкового тексту використовується метод *plt.set_title()*, в який передається текст заголовку і значення аргументу fontdict, такі як шрифт, колір або розмір. Для того, щоб позначки на осі *Y* не відображались на графіку (адже в даному випадку вони не потрібні), в метод *yticks()* передається пустий список. Для видалення рамок використовується метод *set_alpha(0.0)*, що задає повну прозорість. Опісля всіх налаштувань графік відображається методом *show()*.

```
plt.yticks([])
plt.gca().spines["top"].set_alpha(0.0)
plt.gca().spines["right"].set_alpha(0.0)
plt.gca().spines["left"].set_alpha(0.0)
plt.show()
```

Прикладом графіку щільності є графік розподілу часу перевезень. Він демонструє, з якою частотою та чи інша тривалість перевезення зустрічається у записах бази даних перевезень. При підготовці даних для цього випадку необхідно врахувати, що загалом кривих буде дві: для поточного місяця і для минулого. Таким чином, окрім перегляду статистики по часу перевезення, можна буде порівняти її з минулим місяцем. Для того, щоб модуль міг визначити поточну дату, використовується бібліотека *datetime*. Так як записи в базі даних зберігаються у форматі «ДД.ММ.РРРР», потрібні записи зручно буде знайти за відповідними останніми сімома символами. В змінну *current* записується

поточний місяць і рік, а в змінну previous – минулий місяць. В *DataFrame* записуються поля з бази даних.

```
self.now = datetime.datetime.now()
current = str(self.now.month) + '.' + str(self.now.year)
if self.now.month != 1:
    previous = str(self.now.month-1) + '.' + str(self.now.year)
else:
    previous = '12' + '.' + str(self.now.year)
df = pd.read_sql('SELECT end_date, time FROM tasks', self.database.conn)
```

Щоб створити графік щільності (рис. 2.11), використовується метод *kdeplot()* пакету *seaborn*. Це бібліотека для візуалізації даних в *Python* на основі matplotlib, що забезпечує високорівневий інтерфейс для побудови привабливої інформативної статистики. Її зручно використати в даному проекті, адже вона працює саме з елементами бібліотеки pandas. Метод *kdeplot()* представляє дані у вигляді безперервної кривої щільності в одному вимірі.

```
sbn.kdeplot(df.loc[df['end_date'].str.contains(previous), 'time'], shade=True,
                    color="blue", label="Минулий місяць", alpha=.7)
sbn.kdeplot(df.loc[df['end_date'].str.contains(current), 'time'], shade=True,
                    color="orange", label="Поточний місяць", alpha=.8)
```

В приведеному вище фрагменті коду перед тим як повернути індекс, в метод *loc()* проходить перевірка, чи містить рядок в полі *end_date* в собі підрядок *previous*. Якщо так, значить запис відноситься до минулого місяця, і дані додаються до поточного рисунку. За таким же принципом перевіряються дати поточного місяця. Аргументи *label* застосовуються для опису даних на рисунку. Для їх відображення використовується метод *plt.legend()*.

Рис. 2.11 Розподіл тривалостей перевезень

Діаграма розмаху (*Box Plot*) – це чудовий спосіб представлення груп числових даних через квартилі. У статистиці квартиль - це тип квантилю, який ділить дані на чотири частини, або чверті, приблизно рівного розміру. Всередині кожного ящика лежить лінія медіани, яка не обов'язково має лежати строго по середині висоти ящика. Медіана обчислюється за наступним алгоритмом: числа з стовпця сортуються в порядку зростання, загальна кількість чисел ділиться на 2 і знаходиться відповідна позиція медіани: якщо кількість непарна, то число округлюється до більшого, якщо кількість парна, то медіаною буде наступне значення опісля поточної позиції.

Прямі лінії, що виходять з ящиків, називаються «вусами» (*Whisker*) і відображаються для позначення степені розкиду (дисперсії) за межами верхнього та нижнього квартилів. В даному виді діаграм у вигляді окремих точок на одній лінії з вусами відображаються викиди. Це одиничні значення, що різко відрізняються від інших в наборі даних. Діаграми розмаху можуть бути як вертикальними, так і горизонтальними, а також використовуватись разом з точковим графіком при візуалізації відношень між двома змінними. З допомогою діаграм розмаху можна відповісти на такі питання:

- яких значень набувають медіана та квартилі;
- чи є викиди, і яке вони мають значення;

- симетричність даних;
- чи зміщені дані, і в яку сторону.

Для побудови *Box Plot*, що візуалізує розподіл стажу роботи працівників відносно різних посад, з бази даних співробітників записуються дані у *DataFrame*, створюється об'єкт *Figure* і за допомогою методу бібліотеки *seaborn boxplot()* створюється діаграма. Метод приймає назви полів для осей і змінну, де знаходяться ці дані.

```
df = pd.read_sql('SELECT * FROM employees', self.database.conn)
plt.figure(figsize=(9,5))
sbn.boxplot(x='position', y='experience', data=df)
```

Так як кількість працівників різних груп досить відрізняється, а ящики мають приблизно один розмір, то гарним тоном буде додати інформативності і підписати кількість записів для кожного з ящиків (рис. 2.12). Змінна *medians_dict* є словником, що ініціалізується назвою групи робітників та відповідним їм значенням медіан стажу роботи. За допомогою виразу генератора списків в змінну *xticklabels* записуються значення осі *X* поточної області. Ці значення були створені при виклику методу *boxplot()* і являють собою назви посад. В змінну n записуються значення кількості записів для кожної посади у порядку відповідному порядку ящиків на діаграмі. У метод *text* в якості координат *X* передаються індекси ящиків, в якості координат *Y* значення, трохи більше за медіану, щоб текст не накладався на неї. Заголовок для осей створюються за допомогою методів *set_ylabel()* та *set_xlabel()*.

```
medians_dict = {grp[0]:grp[1]['experience'].median() for grp in df.groupby('position')}
xticklabels = [x.get_text() for x in plt.gca().get_xticklabels()]
counts = df['position'].value_counts(sort=False)
n = [None]*len(xticklabels)
for el in range(len(xticklabels)):
        n[el] = counts[xticklabels[el]]
        for (x, xticklabel), n in zip(enumerate(xticklabels), n):
                plt.text(x, medians_dict[xticklabel]*1.03, "#rec : "+str(n), horizontalalignment='center', fontdict={'size':9}, color='white')
```

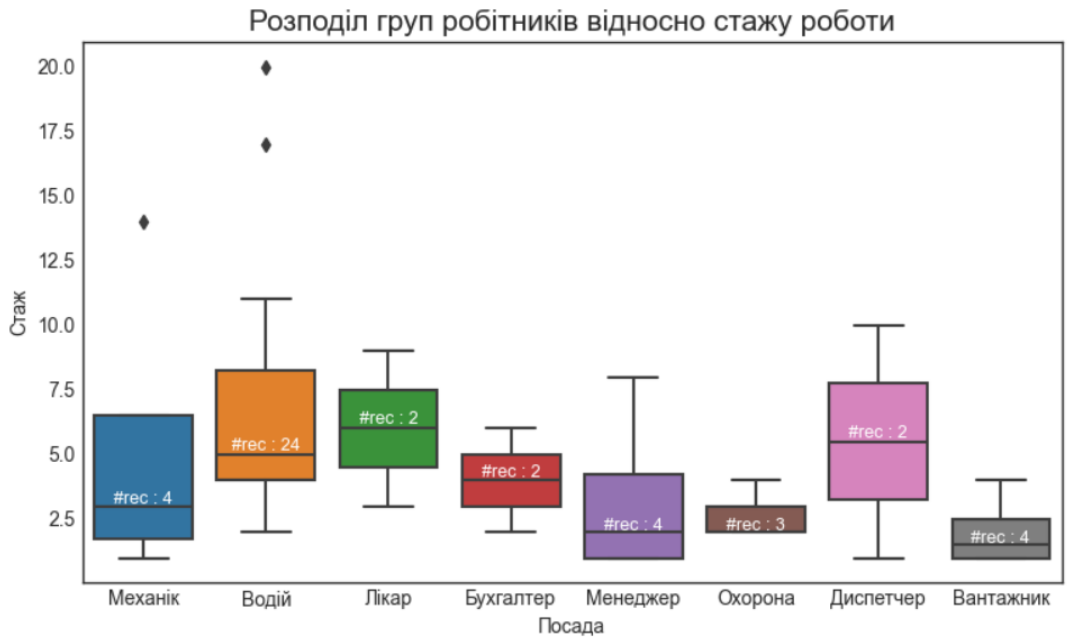


Рис. 2.12 Діаграма розмаху стажу роботи працівників

Точкова діаграма з двома гістограмами використовуються для візуалізації відношення між даними разом з окремим одновимірним розподілом цих даних. Такий графік часто використовується в аналізі даних. В програмі обробки та аналізу даних для вантажних перевезень він використовується для візуалізації даних про тривалість поїздки і витрачені ресурси, і їх залежності. Відмінність побудови рисунку лежить у використанні методу *plt.GridSpec()*. Він створює сітку для розміщення так званих «підграфіків» (*subplots*) в межах *Figure*. Таким чином можна створювати декілька графіків на одному рисунку. Розподіл сітки виглядає наступним чином:

```
grid = plt.GridSpec(4, 4, hspace=0.5, wspace=0.2)
ax_main = fig.add_subplot(grid[:-1, :-1])
ax_right = fig.add_subplot(grid[:-1, -1], xticklabels=[], yticklabels=[])
ax_bottom = fig.add_subplot(grid[-1, 0:-1], xticklabels=[], yticklabels=[])
```

Після цього на *ax_main* створюється точкова діаграма, а на інших підграфіках – гістограми. Далі додаються підписи та значення по осях. Результат представлений на рис. 2.13.

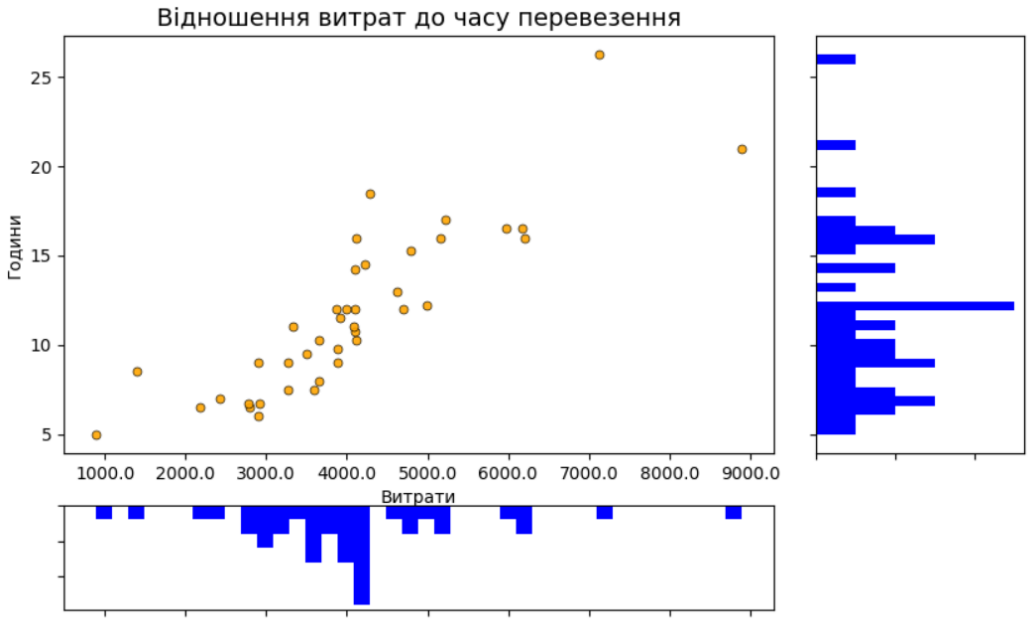


Рис. 2.13 Діаграма відношення витрат до тривалості перевезення

2.4. Висновки до розділу

База даних – це сукупність даних, що організовані відповідно концептуальної структури, яка описує характеристики цих даних і взаємовідносин між ними. Вона розробляється таким чином, щоб користувачеві було максимально зручно отримувати доступ до інформації і керувати нею. Якісна база даних має критично важливе значення для будь-якої компанії, адже в ній зберігаються такі деталі про організацію, як особиста інформація працівників, записи транзакцій, деталі зарплати тощо. Тому в сфері обробки даних є важливим розуміння запитів мови *SQL* та наявність досвіду роботи з різними системами управління базами даних, такими як *PostgreSQL*, *MySQL*, *MS SQL*, *SQLite*. Остання з них чудово підходить для невеликих додатків, вона універсальна, портативна і надійна, тому була обрана для даного проекту.

В роботі з даними найважливіше те, як їх правильно обробити. Але для цього спочатку потрібно проаналізувати кореляції, виділити потрібні дані, відкинути непотрібні, знайти незаповнені поля і так далі. Для цього чудово підходить бібліотека pandas, яка заслужено є дуже популярним модулем для роботи з

інформацією. Вона дозволяє швидко і зручно працювати з табличними даними. Це методи сортування, фільтрації, групування, з’єднання та багато інших. Для графічного представлення підготовлених даних використовується потужний інструментарій *matplotlib*. Це багатоплатформна бібліотека, що має високоякісну графіку. Також вона містить безліч методів для візуалізації даних та має споріднені бібліотеки, такі як *seaborn*, *squarify*, *pywaffle*, що також використовуються у проекті. Таким чином, названі методи роботи з даними є надійним перевіреним рішенням.

3. Програма обробки та аналізу даних для вантажних перевезень

3.1.Структура комп’ютерної програми обробки та аналізу даних для вантажних перевезень

Загальна структура програми представлена на рис. 3.1. При запуску програми в головному вікні є можливість обрати одну з трьох можливостей: роботу з базою даних співробітників, транспортних засобів чи перевезень.

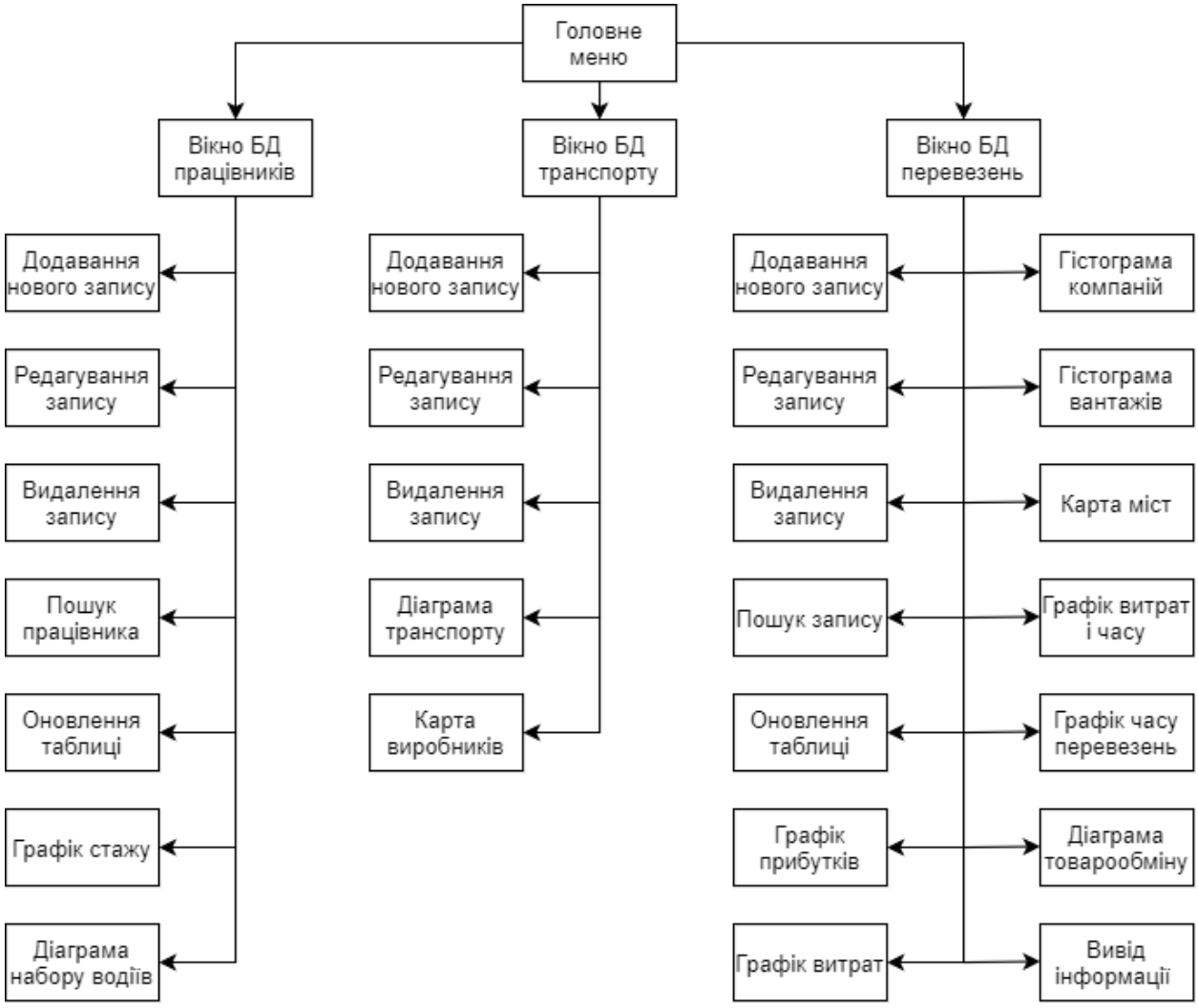


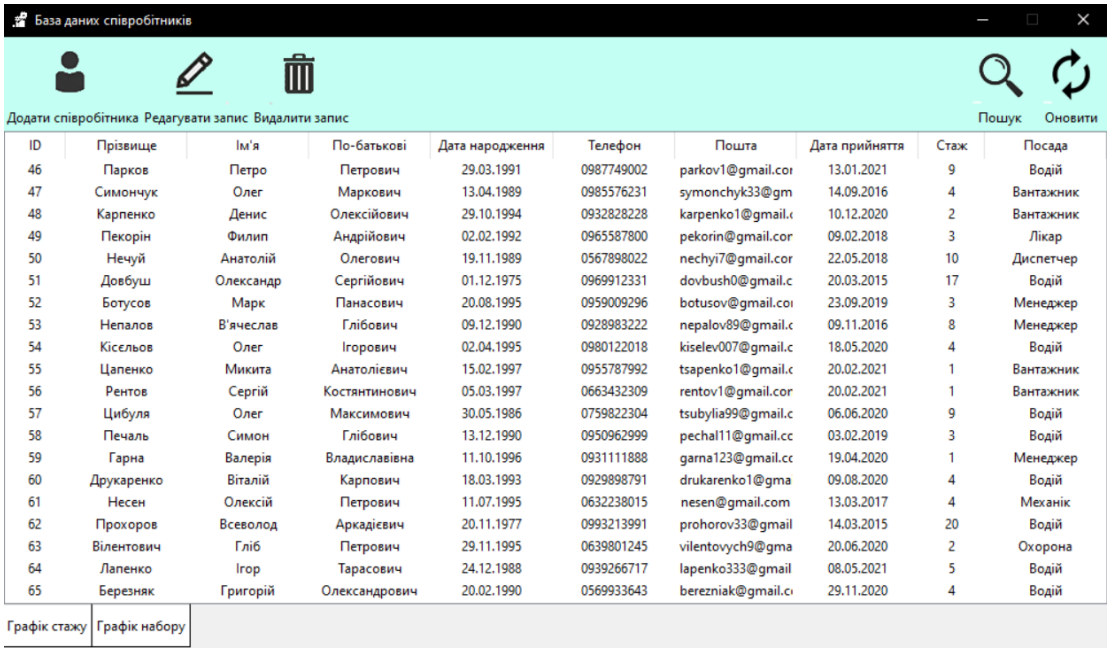
Рис. 3.1 Функціональна схема комп’ютерної програми обробки та аналізу даних для вантажних перевезень

Головне вікно (рис. 3.2) являє собою клас *Main*, що унаслідується від класу *Frame* бібліотеки *tkinter* і створюється при запуску головного скрипта програми. Назва і розмір вікна встановлюється за допомогою стандартних методів класу *Frame*. Також розмір вікон програми не є змінним, що регулюється методом *resizable(False, False)*. Вибір вікон з базами даних виконується за допомогою кнопок, які представляють собою об'єкти *Button* бібліотеки *tkinter*. В параметри приймається об'єкт, на якому вони розташовуються, текст, команда (функція, яка виконується при їх натисканні), колір фону, товщина рамки, опції розташування, фоновий малюнок.

Рис. 3.2 Головне вікно

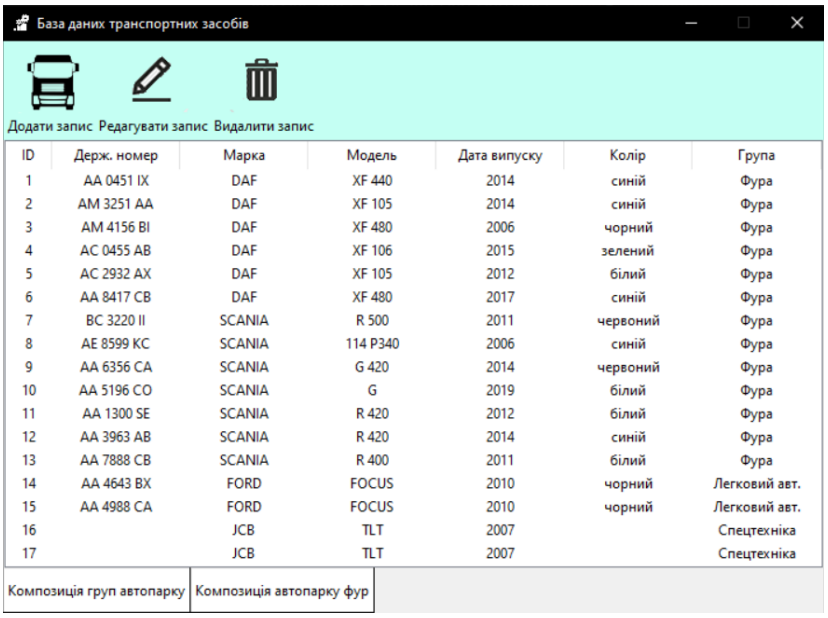
Підрядні вікна (рис. 3.3 – 3.5) наслідуються від класу *Toplevel*. Це класи *Employees*, *Vehicles*, *Tasks*, що є головними вікнами для обробки баз даних, і класи *Add*, *Edit*, *Search*, що є допоміжними вікнами. Кожен з модулів, що відповідає за роботу з певною базою даних, має методи додавання, редагування та видалення запису, а також доступні методи візуалізації даних.

Окрім цього, кожен з трьох модулів містить свій клас *DB*, що відповідає за створення бази даних і доступу до неї через програму. Взаємодія з базою даних в модулі *sqlite3* відбувається за допомогою елементу *cursor*. *SQL*-запити можна виконувати, використовуючи метод *cursor.execute()*.



ID	Прізвище	Ім'я	По-батькові	Дата народження	Телефон	Пошта	Дата прийняття	Стаж	Посада
46	Парков	Петро	Петрович	29.03.1991	0987749002	parkov1@gmail.cor	13.01.2021	9	Водій
47	Симончук	Олег	Маркович	13.04.1989	0985576231	symonchyk33@gm	14.09.2016	4	Вантажник
48	Карпенко	Денис	Олексійович	29.10.1994	0932828228	karpenko1@gmail.(	10.12.2020	2	Вантажник
49	Пекорін	Филип	Андрійович	02.02.1992	0965587800	pekorin@gmail.cor	09.02.2018	3	Лікар
50	Нечуй	Анатолій	Олегович	19.11.1989	0567898022	nechyi7@gmail.cor	22.05.2018	10	Диспетчер
51	Довбуш	Олександр	Сергійович	01.12.1975	0969912331	dovbush0@gmail.c	20.03.2015	17	Водій
52	Ботусов	Марк	Панасович	20.08.1995	0959009296	botusov@gmail.coı	23.09.2019	3	Менеджер
53	Непалов	В'ячеслав	Глібович	09.12.1990	0928983222	nepalov89@gmail.(	09.11.2016	8	Менеджер
54	Кісєльов	Олег	Ігорович	02.04.1995	0980122018	kiselev007@gmail.c	18.05.2020	4	Водій
55	Цапенко	Микита	Анатолієвич	15.02.1997	0955787992	tsapenko1@gmail.(	20.02.2021	1	Вантажник
56	Рентов	Сергій	Костянтинович	05.03.1997	0663432309	rentov1@gmail.cor	20.02.2021	1	Вантажник
57	Цибуля	Олег	Максимович	30.05.1986	0759822304	tsubylia99@gmail.c	06.06.2020	9	Водій
58	Печаль	Симон	Глібович	13.12.1990	0950962999	pechal11@gmail.cc	03.02.2019	3	Водій
59	Гарна	Валерія	Владиславівна	11.10.1996	0931111888	garna123@gmail.cc	19.04.2020	1	Менеджер
60	Друкаренко	Віталій	Карпович	18.03.1993	0929898791	drukarenko1@gma	09.08.2020	4	Водій
61	Несен	Олексій	Петрович	11.07.1995	0632238015	nesen@gmail.com	13.03.2017	4	Механік
62	Прохоров	Всеволод	Аркадієвич	20.11.1977	0993213991	prohorov33@gmail	14.03.2015	20	Водій
63	Вілентович	Гліб	Петрович	29.11.1995	0639801245	vilentovych9@gma	20.06.2020	2	Охорона
64	Лапенко	Ігор	Тарасович	24.12.1988	0939266717	lapenko333@gmail	08.05.2021	5	Водій
65	Березняк	Григорій	Олександрович	20.02.1990	0569933643	berezniak@gmail.c	29.11.2020	4	Водій

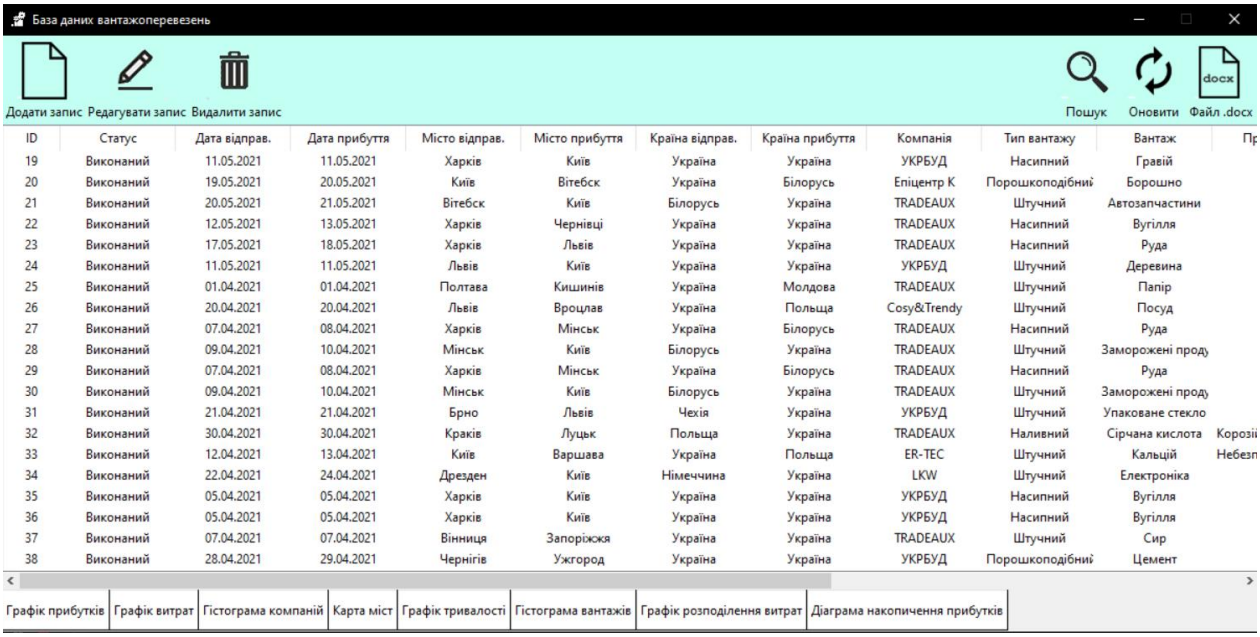
Рис. 3.3 Вікно роботи з БД працівників



ID	Держ. номер	Марка	Модель	Дата випуску	Колір	Група
1	AA 0451 IX	DAF	XF 440	2014	синій	Фура
2	AM 3251 AA	DAF	XF 105	2014	синій	Фура
3	AM 4156 BI	DAF	XF 480	2006	чорний	Фура
4	AC 0455 AB	DAF	XF 106	2015	зелений	Фура
5	AC 2932 AX	DAF	XF 105	2012	білий	Фура
6	AA 8417 CB	DAF	XF 480	2017	синій	Фура
7	BC 3220 II	SCANIA	R 500	2011	червоний	Фура
8	AE 8599 KC	SCANIA	114 P340	2006	синій	Фура
9	AA 6356 CA	SCANIA	G 420	2014	червоний	Фура
10	AA 5196 CO	SCANIA	G	2019	білий	Фура
11	AA 1300 SE	SCANIA	R 420	2012	білий	Фура
12	AA 3963 AB	SCANIA	R 420	2014	синій	Фура
13	AA 7888 CB	SCANIA	R 400	2011	білий	Фура
14	AA 4643 BX	FORD	FOCUS	2010	чорний	Легковий авт.
15	AA 4988 CA	FORD	FOCUS	2010	чорний	Легковий авт.
16		JCB	TLT	2007		Спецтехніка
17		JCB	TLT	2007		Спецтехніка

Рис. 3.4 Вікно роботи з БД транспортних засобів

Класи *Employees*, *Vehicles*, *Tasks* складаються з елементів *Frame*, на яких розміщуються кнопки (*tkinter.Button*) для взаємодії з базою даних, елементу *Treeview* з модулю *ttk* бібліотеки *tkinter*, та ще одного елементу *Frame*, на якому розміщуються кнопки для роботи з графічним представленням даних. Також клас *Tasks* містить елемент *scrollbar* для перегляду таблиці *Treeview*.



ID	Статус	Дата відправ.	Дата прибуття	Місто відправ.	Місто прибуття	Країна відправ.	Країна прибуття	Компанія	Тип вантажу	Вантаж	Пр
19	Виконаний	11.05.2021	11.05.2021	Харків	Київ	Україна	Україна	УКРБУД	Насипний	Гравій	
20	Виконаний	19.05.2021	20.05.2021	Київ	Вітебск	Україна	Білорусь	Епіцентр К	Порошкоподібний	Борошно	
21	Виконаний	20.05.2021	21.05.2021	Вітебск	Київ	Білорусь	Україна	TRADEAUX	Штучний	Автозапчастини	
22	Виконаний	12.05.2021	13.05.2021	Харків	Чернівці	Україна	Україна	TRADEAUX	Насипний	Вугілля	
23	Виконаний	17.05.2021	18.05.2021	Харків	Львів	Україна	Україна	TRADEAUX	Насипний	Руда	
24	Виконаний	11.05.2021	11.05.2021	Львів	Київ	Україна	Україна	УКРБУД	Штучний	Деревина	
25	Виконаний	01.04.2021	01.04.2021	Полтава	Кишинів	Україна	Молдова	TRADEAUX	Штучний	Папір	
26	Виконаний	20.04.2021	20.04.2021	Львів	Вроцлав	Україна	Польща	Cosy&Trendy	Штучний	Посуд	
27	Виконаний	07.04.2021	08.04.2021	Харків	Мінськ	Україна	Білорусь	TRADEAUX	Насипний	Руда	
28	Виконаний	09.04.2021	10.04.2021	Мінськ	Київ	Білорусь	Україна	TRADEAUX	Штучний	Заморожені проду	
29	Виконаний	07.04.2021	08.04.2021	Харків	Мінськ	Україна	Білорусь	TRADEAUX	Насипний	Руда	
30	Виконаний	09.04.2021	10.04.2021	Мінськ	Київ	Білорусь	Україна	TRADEAUX	Штучний	Заморожені проду	
31	Виконаний	21.04.2021	21.04.2021	Брно	Львів	Чехія	Україна	УКРБУД	Штучний	Упаковане стекло	
32	Виконаний	30.04.2021	30.04.2021	Краків	Луцьк	Польща	Україна	TRADEAUX	Наливний	Сірчана кислота	Корозії
33	Виконаний	12.04.2021	13.04.2021	Київ	Варшава	Україна	Польща	ER-TEC	Штучний	Кальцій	Небезп
34	Виконаний	22.04.2021	24.04.2021	Дрезден	Київ	Німеччина	Україна	LKW	Штучний	Електроніка	
35	Виконаний	05.04.2021	05.04.2021	Харків	Київ	Україна	Україна	УКРБУД	Насипний	Вугілля	
36	Виконаний	05.04.2021	05.04.2021	Харків	Київ	Україна	Україна	УКРБУД	Насипний	Вугілля	
37	Виконаний	07.04.2021	07.04.2021	Вінниця	Запоріжжя	Україна	Україна	TRADEAUX	Штучний	Сир	
38	Виконаний	28.04.2021	29.04.2021	Чернігів	Ужгород	Україна	Україна	УКРБУД	Порошкоподібний	Цемент	

Рис. 3.5 Вікно роботи з БД перевезень

За додавання нового запису до тієї чи іншої бази даних відповідає клас *Add*. Він створюється, коли користувач натискає кнопку для додавання нового рядку до таблиці. При цьому в конструктор класу передається об’єкт *Employees*, *Vehicles*, або *Tasks*, в залежності де створюється вікно для нового запису. Це потрібно для того, щоб у класі *Add* була можливість взаємодії з методами і полями класу, який його викликає. Вікно для додавання нового запису містить всі необхідні блоки вводу для заповнення полів відповідної бази даних. Підписи для полів створюються за допомогою об’єктів *ttk.Label*, в параметри яких передається батьківський клас, текст і опис стилю тексту. Самі блоки вводу є об’єктами *ttk.Entry*. Для таких елементів як посада працівника, група транспортного засобу, тип вантажу і примітка до вантажу використовується об’єкт *ttk.Combobox*. В його аргументи передається список значень. Варіант, який відображається за замовчуванням, визначається методом *current()*. Кнопки підтвердження confirm та скасування *close* реалізовуються за допомогою віджету *ttk.Button*. Для того, щоб вікно додавання запису закривалось автоматично після затвердження запису або його відхилення, в аргумент *command* передається *self.destroy*. В бібліотеці *tkinter* є метод *bind()*, що пов’язує між собою віджет, подію і дію. Для кнопки згоди confirm він використовується, щоб викликати метод запису інформації з полів вводу в базу даних:

```
    self.confirm.bind('<Button-1>', lambda event: self.view.add_records(
    [self.number.get(),self.brand.get(),
self.model.get(),     self.date.get(),
self.color.get(), self.tgroup.get()]))
```

Аргумент *<Button-1>* спрацьовує при натисканні ЛКМ. Для кожного з елементів, розміщеного на вікні, використовується метод *place(x,y)* для розміщення. Клас *Edit* наслідується від класу *Add*, тому розмір вікна та вигляд полів і підписів буде аналогічний. Змінюється функція методу *bind()*: викликається не метод додавання запису, а метод оновлення полів таблиці. Також у вікні редагування запису для зручності всі поля будуть спочатку заповнені попередніми значеннями. Для цього з відповідної БД обирається запис з *ID* виділеного елемента. Якщо жоден елемент не виділений, то виняток перехоплюється за допомогою блоку «*try-except-else*», і вікно не з'являється. Якщо ж користувач натиснув на кнопку редагування після виділення потрібного запису за допомогою ЛКМ або клавіатури, то поля у вікні заповнюються методом *insert()*.

Клас *Search*, що є в вікнах по роботі з записами про працівників і про перевезення, наслідується від класу *Toplevel* бібліотеки *tkinter*. Це вікно (рис. 3.6) містить в собі елементи *ttk.Label*, *ttk.Label* та *ttk.Button*, що були описані вище. Значення поля передається в метод *search_records()* при натисканні ЛКМ на елемент підтвердження *confirm*. Реалізація пошуку наступна:

```
    key = ('%' + key + '%',)
    self.database.c.execute('SELECT * FROM tasks WHERE end_date LIKE ?', key)
    [self.tree.delete(i) for i in self.tree.get_children()]
    [self.tree.insert('', 'end', values=row) for row in self.database.c.fetchall()]
```

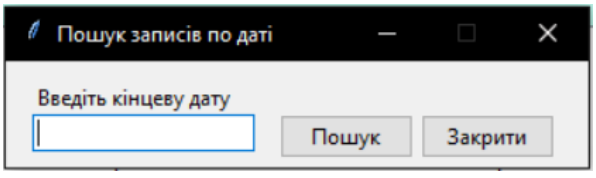


Рис. 3.6 Вікно пошуку запису

Змінна key типу string являє собою слово, яке ввів користувач для пошуку. З бази даних виділяються всі записи що містять цю змінну. Після цього з таблиці, яка дозволяє програмі відображати вміст бази даних, видаляються всі записані в

неї значення. Потім записуються значення, що були виділені в базі даних. Для повернення всіх записів у таблицю використовується кнопка оновлення, що міститься на верхній панелі вікна.

При видаленні виділеного запису з бази даних виконується метод *delete_records()*. В ньому обирається перший з виділених рядків (у випадку коли він не один, для уникнення помилки) і виконується *SQL*-запит по вилученню запису:

```
    for item in self.tree.selection():
                self.database.c.execute('''DELETE FROM tasks WHERE id=?''', 
(self.tree.set(item,'#1'),))
    self.database.conn.commit()
```

Після додавання нового запису, редагування запису і оновлення таблиці, а також при запуску модулю викликається метод *view_records()*. При його виклику з бази даних виділяються всі поля і заносяться в таблицю Treeview.

Вікна для відображення кожного з графіків створюються засобами модулю pyplot бібліотеки *matplotlib*. Крім додаткового модулю seaborn, що описувався в другому розділі, в проекті також використовуються модулі *pywaffle* та *squarify*. Модуль *pywaffle* дозволяє створювати вафельну діаграму: таблицю квадратів, кожен з яких є одиницею величини, яка представляється. За допомогою різних кольорів ці фігури розбиваються на групи. За допомогою аргументів *labels* і *legend* на об’єкті *Figure* створюється пояснення, яку групу представляє кожен з кольорів. Для створення діаграми-карти (*treemap*) потрібен модуль *squarify*. Ця діаграма подібна до попередньої, але композиція створюється не за допомогою окремих елементів, а з об’єднанням їх в одну фігуру. Таким чином роль грає саме відношення площ секторів однин до одного. Діаграма-карта в проєкті використовується для представлення кількості транспорту того чи іншого бренду по відношенню один до одного.

3.2. Результати роботи комп’ютерної програми обробки та аналізу даних для вантажних перевезень

В залежності від вибору категорії користувачем відкривається одне з вікон (див. рис. 3.3, 3.4, 3.5), що відповідає за певну базу даних і відповідні модулі обробки, аналізу і графічного представлення даних. При додаванні запису, наприклад, до бази даних співробітників, відкриється вікно, зображене на рис. 3.7.

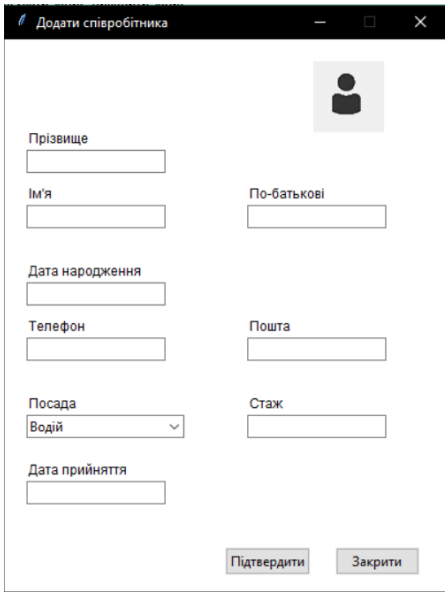


Рис. 3.7 Вікно додавання запису

Приклад роботи пошуку в базі даних перевезень записів з кінцевою датою «20.05» показане на рис. 3.8. Вікно для редагування запису представлене на рис. 3.9. Кнопки для додавання, редагування, видалення і пошуку записів знаходяться у верхній панелі вікон роботи з БД.

ID	Статус	Дата відправ.	Дата прибуття	Місто відправ.	Місто прибуття
3	Виконаний	20.05.2021	20.05.2021	Харків	Київ
5	Виконаний	19.05.2021	20.05.2021	Львів	Печ
20	Виконаний	19.05.2021	20.05.2021	Київ	Вітебск

Рис. 3.8 Результат роботи пошуку

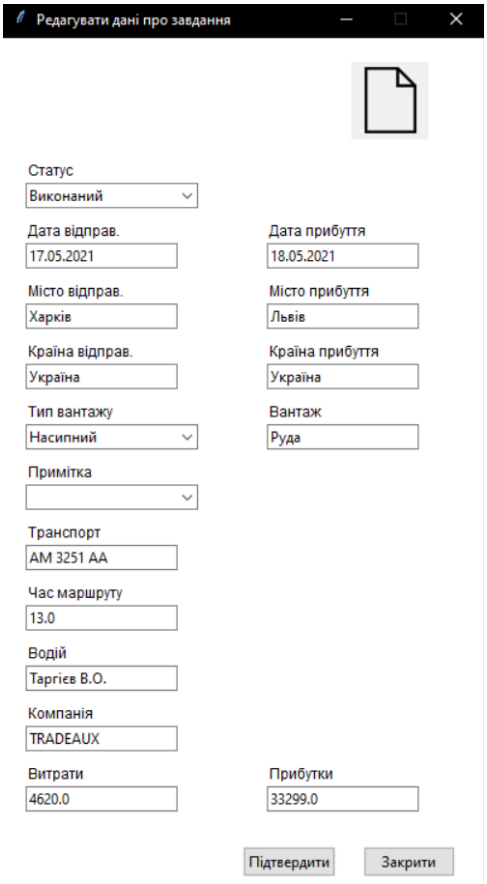


Рис. 3.9 Вікно редагування запису

Можна вивести інформацію про запис в файл формату «*docx*», натиснувши кнопку «Файл .docx». пропонується ввести додаткову інформацію про перевезення, після чого створюється і заповнюється шаблон, показаний на рис. 3.10. Він зберігається у теці з програмою.

Перевезення № 26

Основна інформація

Статус	Дата відправ.	Дата прибуття	Місто відправ.	Місто прибуття	Країна відправ.	Країна прибуття
Виконаний	20.04.2021	20.04.2021	Львів	Вроцлав	Україна	Польща

Компанія	Тип вантажу	Вантаж	Примітка	Транспорт	Час	Водій	Витрати	Прибутки
Cosy	Штучний	Посуд		АС 2932 АХ	9.0	Цибуля О.М.	3888.0	30100.0

Додаткова інформація

Механік	Номер причепу	Вага вантажу, т	Пройдена відстань, км	Кількість придбаного пального, л	Час простою	Причина простою
Несен О.П.	АА 1177 15	6	612	102	-	-

Рис. 3.10 Інформація у створеному файлі

Важливу частину програми складають дванадцять функцій для графічного представлення даних. В кожному з вікон роботи з БД є кнопки для візуалізації певних даних. Ці кнопки знаходяться на нижній панелі, і при їх натисканні спрацьовує відповідний метод класу, після чого готовий графік з'являється у новому вікні (рис. 3.11). Розмір вікна можна змінювати.

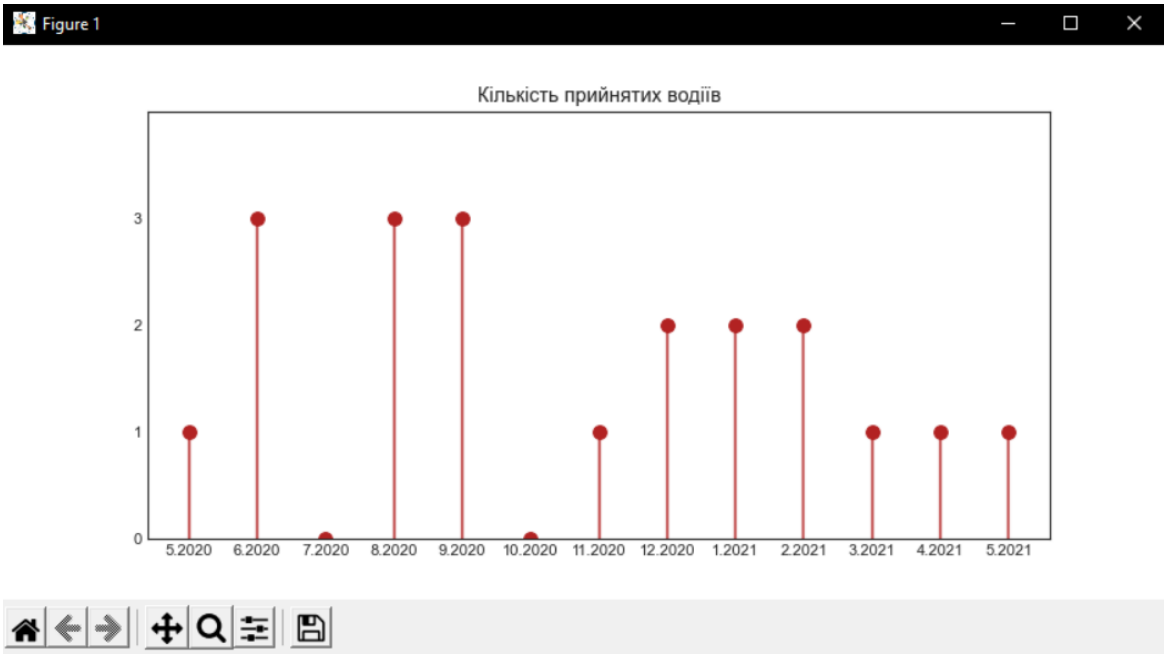


Рис. 3.11 Вікно з графіком

На даному графіку показане число водіїв, що приєдналися до компанії, для кожного місяця за останній рік. Це дозволяє оцінити, чи виконаний бажаний план набору, і в який час. Для виділення потрібних даних з бази даних виконувалась команда «*SELECT enrolment FROM employees WHERE position*='Водій'», для побудови графіку методи *vlines()* і *scatter()*, також було обраховано потрібні місяці на основі поточного, який був визначений за допомогою бібліотеки datetime.

Ще один графік, який працює з базою даних працівників, це діаграма розмаху (див. рис. 2.12). По осі *X* розташовуються назви професій, що наявні у базі даних, по осі Y містяться позначки з роками стажу. Також, при наведенні курсора миші на графік, в правому нижньому кутку вікна можна побачити точні координати. По ящику диспетчерів можна побачити, що досвід робітників цього відділу компанії варіюється від 1до 9 років, але так як кількість записів дорівнює двом, то можна зробити висновок, що працює два диспетчера з досвідом в 1 та 9 років. Досвід

менеджерів лежить в межах від 1 до 8 років, але досвід 75% працівників (верхній квартиль), тобто трьох з чотирьох людей, лежить в межах до 4 років. З першого ящика можна побачити, що розмах досвіду механіків трохи зміщений вниз. Таким чином, більше записів міститься внизу боксу, ніж у верхній частині. Викиди містяться в наборах даних «Механік» та «Водій». Для першого це 14 років, для другого це 16 і 20 років. Ці значення дуже сильно відрізняються у полі від інших.

Вафельна діаграма (рис. 3.12) використовується для наглядної візуалізації груп даних. В даному випадку це групи транспортних засобів в компанії.

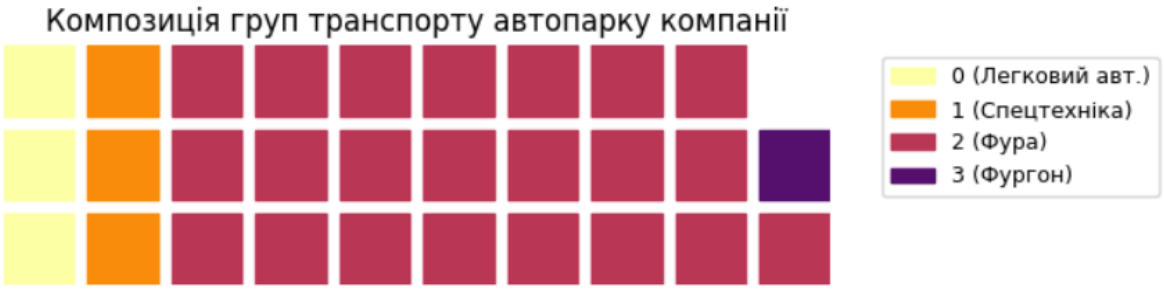


Рис. 3.12 Композиція груп транспорту автопарку компанії

Як видно з графіку, серед автомобілів для вантажних перевезень всі, крім одного, є фурами. Отже, є сенс знаходити завдання з об'ємним вантажем і довгими маршрутами. Графік є результатом роботи з модулем *pywaffle*. Для встановлення кольорів використовувався модуль *cm* бібліотеки *matplotlib*.

Кругова діаграма і деревоподібна карта дозволяють переглянути вклад груп даних в загальне ціле. Але кругова діаграма є не найкращим способом представлення даних. Вона дуже часто використовується для представлення даних, але більш в маркетинговій і журналістській сфері. При візуалізації важливих і точних даних ця діаграма може спотворити інформацію через вигляд площ сегментів[5,15], якщо їх багато. Це ускладнює розуміння ситуації людям, що приймають рішення. Рішенням проблеми кругової діаграми може бути додавання відсотку або числа для кожної групи. В даному проекті використовується деревоподібна карта (рис. 3.13), побудована за допомогою модулю *squarify*. Вона зручніша для сприйняття і площі її сегментів в будь-якому випадку не будуть ускладнювати розуміння графіку. Зараз груп всього три, але

якщо на графіку буде багато груп, числа з кількістю записів додатково допоможуть орієнтуватись в наборі даних.

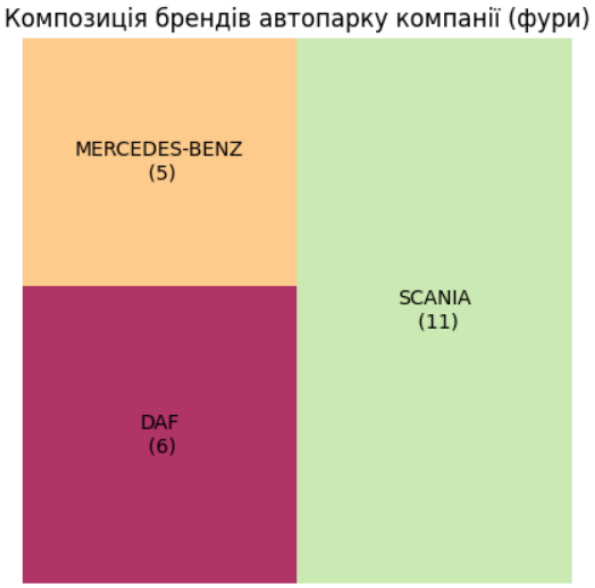


Рис. 3.13 Композиція брендів автопарку компанії

Вікно для роботи з записами про перевезення містить вісім методів для графічного представлення даних. Перші з них це графіки прибутку (рис. 3.14) та витрат (рис. 3.15) за останній місяць.

Рис. 3.14 Графік прибутків за місяць

Рис. 3.15 Графік витрат за місяць

Такі рисунки є прикладом графіків часових рядів, що використовується для представлення того, як певний показник змінюється протягом часу, в даному випадку в межах поточного місяця: з дня першого прибутку чи витрат, і до останньої дати, для якої був записаний прибуток чи витрата. Крім динаміки даних можна побачити максимальне значення для кожного з набору.

Ще одним графіком є гістограма компаній (рис. 3.16). Це класичний спосіб візуалізації елементів на основі кількості або будь-якої іншої встановленої метрики.

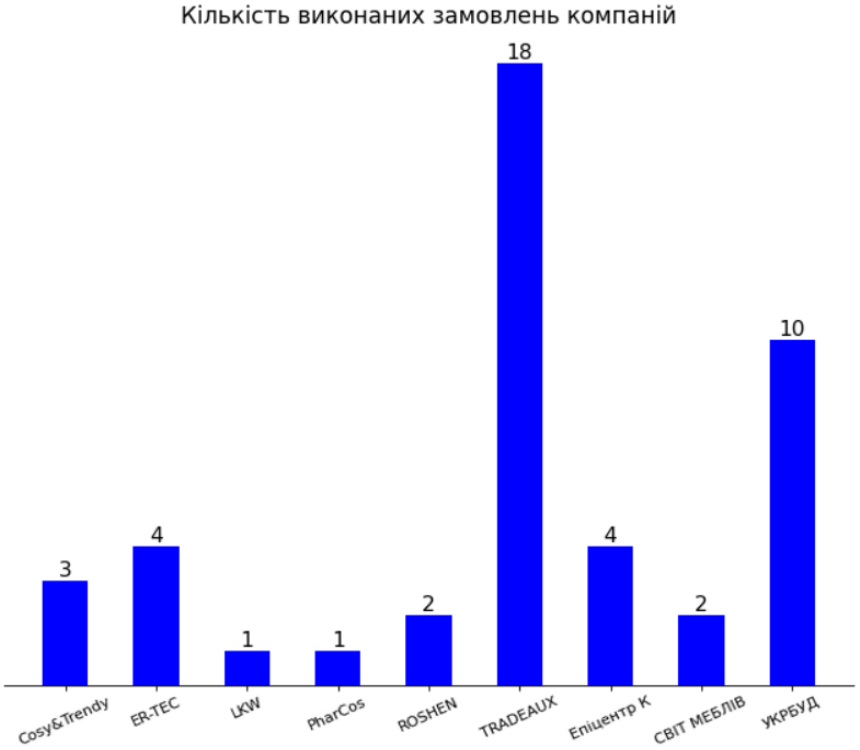


Рис. 3.16 Гістограма компаній

З такого графіка можна зробити висновки про співпрацю з різними компаніями, а також порівняти кількість виконаних перевезень для різних замовників.

Для представлення інформації про те, в які міста найчастіше відбуваються перевезення, в проекті наявний точковий графік з картою в якості фону (рис. А.1.).

Графік розподілу тривалості перевезень (див. рис. 2.11) показує, скільки годин найчастіше займало перевезення в поточному і минулому місяці. Це дозволяє

проаналізувати характер замовлень за останній час. Так, за графіком, більшість поїздок займала близько десяти годин, але невелика їх частина доходила до більш ніж п’ятнадцяти годин. В минулому місяці поїздки найчастіше тривали 10-16 годин, але було перевезення, що тривало більше двадцяти п’яти годин.

Метод, що створює впорядковану гістограму типів вантажів (див. рис. 2.10), доступний у вигляді кнопки «Гістограма вантажів». Окрім побудови графіку, він сортує стовпчики для зручнішого сприйняття інформації.

Діаграма відношення витрат до тривалості перевезення (див. рис. 2.13) окрім самого відношення показує розподіл величин за допомогою бокових гістограм. Можна побачити, що найбільше витрат приходиться на числа від 3-х тис. до 4,5 тис. гривень. Для часу перевезень найвагомішими є значення в 7, 9, 13 та 16 годин. Також видно, що змінна витрат змінюється швидше ніж змінна часу на проміжку до відмітки приблизно в чотири тисячі витрат. Після цього моменту змінна часу починає змінюватись швидше змінної витрат.

Діаграма накопичення прибутків (рис. 3.19) показує вклад імпорту, експорту та перевезень всередині України в загальний прибуток по дням поточного місяця. Діаграма з областями накопичення дає візуальне представлення ступеню вкладу від декількох часових рядів. Дні, в які були отримані прибутки, можна побачити у вигляді вершин. Наприклад, 3-го числа було зароблено в загальному близько 150 тис. гривень, з них 40 тис. – це імпорт, а інші були отримані з перевезення за кордон.

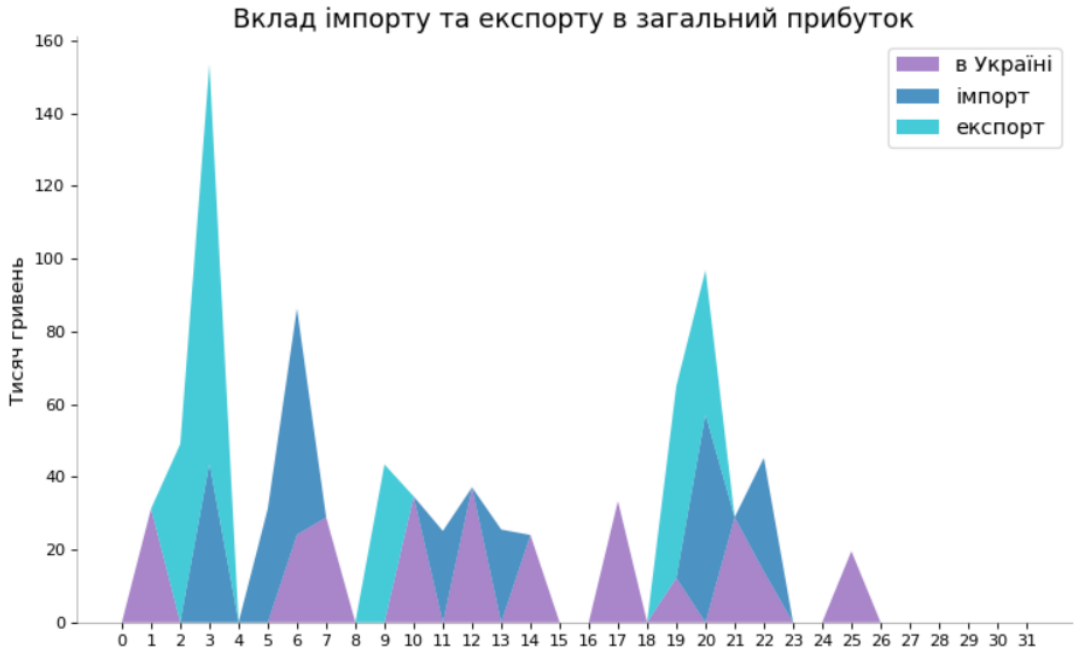


Рис. 3.19 Діаграма накопичення прибутків

3.3. Можливості роботи комп'ютерної програми обробки та аналізу даних для вантажних перевезень

Як і існуючі аналоги «АвтоПеревозки 4», «Авто-Поезд» і «Корс Автопредприятие», комп'ютерна програма обробки та аналізу даних для вантажних перевезень містить велику базу даних. Туди входять такі дані як П.І.Б. водіїв, що здійснюють перевезення та інших працівників, їх контактні дані (номер телефону та пошта), стаж та рік прийняття працівників, номера, марки та моделі транспортних засобів, їх коротка характеристика. В записи про вантажоперевезення входить інформація про дату відправлення та прибуття, міста та країни, компанії та вантажі, прибутки та доходи, також можна додати інформацію про вагу вантажу, довжину маршруту та інше. Також доступний вивід інформації про перевезення в файл *MS Word*. До перегляду і редагування баз даних розроблений зручний, функціональний і водночас простий інтерфейс.

На відміну від аналогів, даний проєкт є багатоплатформний і може працювати не лише на ОС *Windows* 10, а і на *Windows* 7, *Mac OS* і дистрибутивах *Linux*. В розробленому проєкті також міститься багато способів графічного представлення

даних. Це є його перевагою, адже в аналогах він або відсутній, або має декілька функцій. Доступний аналіз стажу робітників, плану набору водіїв, композиції автопарку, витрат і прибутків за місяць, середнього часу перевезень, взаємодії з компаніями та містами та інше.

3.4. Висновки розділу

Отже, розроблений проєкт «Комп'ютерна програма обробки та аналізу даних для вантажних перевезень» має зручну структуру модулів, лаконічний інтерфейс та повну функціональність для роботи з базами даними, що представляють собою файли *SQL*.

Структура програми містить головне вікно, три вікна для роботи з інформацією з файлів *SQL*, і додаткові вікна: для додавання нового запису, редагування, видалення і пошуку записів, а також для заповнення додаткових даних у разі створення файлу *MS Word*. Для роботи з *SQL* було використано СУБД *SQLite* у вигляді модулю *sqlite3*. Це популярна, багатоплатформна і надійна СУБД для невеликих проектів, яка підтримує весь функціонал *SQL*, тому була обрана для роботи. Таблиця, що відображає вміст кожного з файлів *SQL*, була створена засобами стандартного модулю *ttk* бібліотеки *tkinter*. Вона дозволяє зручно переглядати і виділяти записи. Для кожної з кнопок по роботі було додано іконки, що робить їх функції інтуїтивно зрозумілими. Для роботи з графікою використовувалася багатоплатформна бібліотека *matplotlib* з її додатковими модулями. Таким чином було побудовано різноманітні графіки: лінійні, точкові, впорядкована і невпорядкована гістограми, вафельна та секторна діаграма, діаграми відношення, розподілення і розмаху. Всі вони доступні у вигляді кнопок на вікнах по обробці баз даних.

ВИСНОВКИ

Отже, обробка та аналіз даних є важливими процесами у кожній сфері сучасного суспільства. Тому гарні вміння та навички працювати, перетворювати, знаходити і представляти інформацію – це потрібні та корисні можливості для розробника програмного забезпечення, аналітика або іншої професії, яка відноситься до роботи з даними в IT-сфері. Під час розробки проєкту «Комп'ютерна програма обробки та аналізу даних для вантажних перевезень» було використано знання про обробку, аналіз і візуалізацію даних засобами мови програмування *Python*, а також набуто нових знань і практичних навичок в цій сфері. Було розроблено три шаблони баз даних на тематику обліку вантажоперевезень. При використанні системи управління реляційними базами даних *SQLite* вдалося створити зручну структуру для роботи з інформацією, що представляється у вигляді *SQL*-файлів. Важливим чинником є те, що перегляд і робота з записами в базах даних є комфортним завдяки простому і функціональному інтерфейсу. Під час розробки програмних модулів було використано методи та подання з документацій по стандартній графічній бібліотеці мови *Python*, бібліотеці *matplotlib* для візуалізації даних двовимірною 2D графікою та модулю по роботі з даними *pandas*. Таким чином проєкт є поєднанням двох складових: організованою структурою даних та методами її обробки і візуалізації. Зберігши функціональність і практичність системи роботи з файлами *SQL*, було розроблено функції для автоматичного створення графічних звітів по цим даним. В результаті програма отримала вагому перевагу у вигляді надання можливості аналізу і аналітики того чи іншого набору даних за допомогою їх графічного представлення. Важливу роль зіграв підбір інструментарію для побудови проєкту. Мова програмування *Python* в останній час все частіше використовується для аналізу та іншої роботи з даними, як в науці, так і в комерційній сфері. Цьому сприяє лаконічність мови і велике різноманіття бібліотек, що знаходяться у відкритому доступі.

Керуючись інструментарієм модулю *pandas*, в проєкті було застосовано операції по сортуванню, фільтрації, групуванню, об'єднанню і редагуванню полів і записів у структурах даних. Це сучасна бібліотека, яка здатна імпортувати в проєкт дані з різних форматів файлів. До того ж вона використовується спеціалістами в сферах науки про дані, аналітиці даних та розробці штучного інтелекту. Отже, застосування таких методів є хорошою практикою для розробника програмного забезпечення. Було представлено рішення для підготовки даних до їх подальшого графічного відображення. Для створення графіків були використані засоби сучасної бібліотеки *matplolib* та її суміжних модулів. Це дозволило зручно переглядати діаграми, адже вони представляються окремими вікнами з регульованою величиною і можливістю збереження поточного рисунку у файл. Так як кожна з описаних бібліотек для роботи з даними і система управління даними *SQLite* є багатоплатформенними, це робить проект спроможним працювати на різних операційних системах. При побудові графіків були використані принципи правильної і ефективної візуалізації інформації, описані в першому розділі. В результаті отримані рисунки, що уникають спотворення даних, лаконічно відображають багато чисел на невеликому просторі, пов'язують набір записів у єдине ціле, акцентують увагу на різних аспектах даних, виділяють їх зв'язки та відносини, зменшують інформаційне навантаження на користувача. Рішення на основі пакету *matplolib* дозволили урізноманітнити способи подання даних. Це допомагає донести ідею кожного з представлених розподілень, композицій, відношень і порівнянь. Переглядаючи інформацію у подібних формах і рисунках, можна легко зробити висновки про ті чи інші числові дані, або прийняти важливі рішення про ситуацію в певній сфері. Кожен з методів, що викликається за допомогою кнопок на нижніх панелях вікон по роботі з записами, є прикладом потужності та багатофункціональності бібліотек *pandas* та *matplolib* при їх спільному використанні. Таким чином була досягнена мета виконання проєкту – розробка покращеного аналітичного модулю програми по роботі з даними.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бойченко С.В., Іванченко О.В. Положення про дипломні роботи (проекти) випускників Національного авіаційного університету. Київ : НАУ, 2017. 63 с.

2. ДСТУ 3008-95. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення : видання офіційне, Київ : Держстандарт України, 1995. 38 с.

3. Грицюк, П. М. Аналіз даних / П. М. Грицюк, О. П. Остапчук ; Нац. ун-т "Вод. господ. та природокорис." . – Рівне : Вид-во Нац. ун-ту "Вод. господ. та природокорис.", 2008. – 218 с.

4. *McKinney Wes. Python for Data Analysis / Wes McKinney. – Sebastopol : O'Reilly Media, Inc.*, 2013.

5. *Alberto Cairo. How Charts Lie / Cairo Alberto. – New York City : W. W. Norton & Company*, 2019.

6. *Useful Junk? The effects of visual embellishment on comprehension and memorability of charts : proceedings of the 28th International Conference on Human Factors in Computing Systems, April* 10-15, 2010, *CHI* 2010, *Atlanta, Georgia, USA*, 2010.

7. *Geetika Chawla, Big Data Analytics for Data Visualization: Review of Techniques / Chawla Geetika // International Journal of Computer Applications*. – 2018. – *No*. 21, *Vol*. 182.

8. *Tufte Edward R. Visual Display of Quantitative Information / Edward R. Tufte. – Cheshire* : *Graphics Press USA*, 2001.

9. Дейт К. Дж. Введение в системы базы данных. – 8-е изд.: Пер. с англ. – М. : Издательский дом "Вильямс", 2006. – 1328 с.

10. Б. В. Ковтун. Порівняльна характеристика реляційних та NoSQL баз даних / Ковтун Б. В., Манич А. М., Романюк О.В. – 2020. – URL: *http://ir.lib.vntu.edu.ua/bitstream/handle/123456789/29461/9907.pdf?sequence=3* (lastaccess: 28.05.2021),

11. *Graphical User Interfaces with Tk. – 2021. – URL*: *https://docs.python.org/3/library/tk.html* (*lastaccess*: 28.05.2021).

12. *Horvath Reka. Using Pandas and Python to Explore Your Dataset / Reka Horvath. URL*: *https://realpython.com/pandas-python-explore-dataset/#author* (*lastaccess*: 31.05.2021)

12. *Hunter John. Matplotlib / John Hunter, Droettboom Michael– URL*: *http://www.aosabook.org/en/matplotlib.html* (*lastaccess*: 29.05.2021).

13. *Wickham Hadley. 40 years of boxplots / Hadley Wickham, Stryjewski Lisa. URL*: *https://vita.had.co.nz/papers/boxplots.pdf* (lastaccess: 29.05.2021).

15. *E. Bertini, Judgment Error in Pie Chart Variations / Bertini E. URL: https://diglib.eg.org/bitstream/handle/10.2312/eurovisshort20161167/091-095.pdf?sequence=1&isAllowed=y* (*lastaccess*: 30.05.2021).

ДОДАТОК А

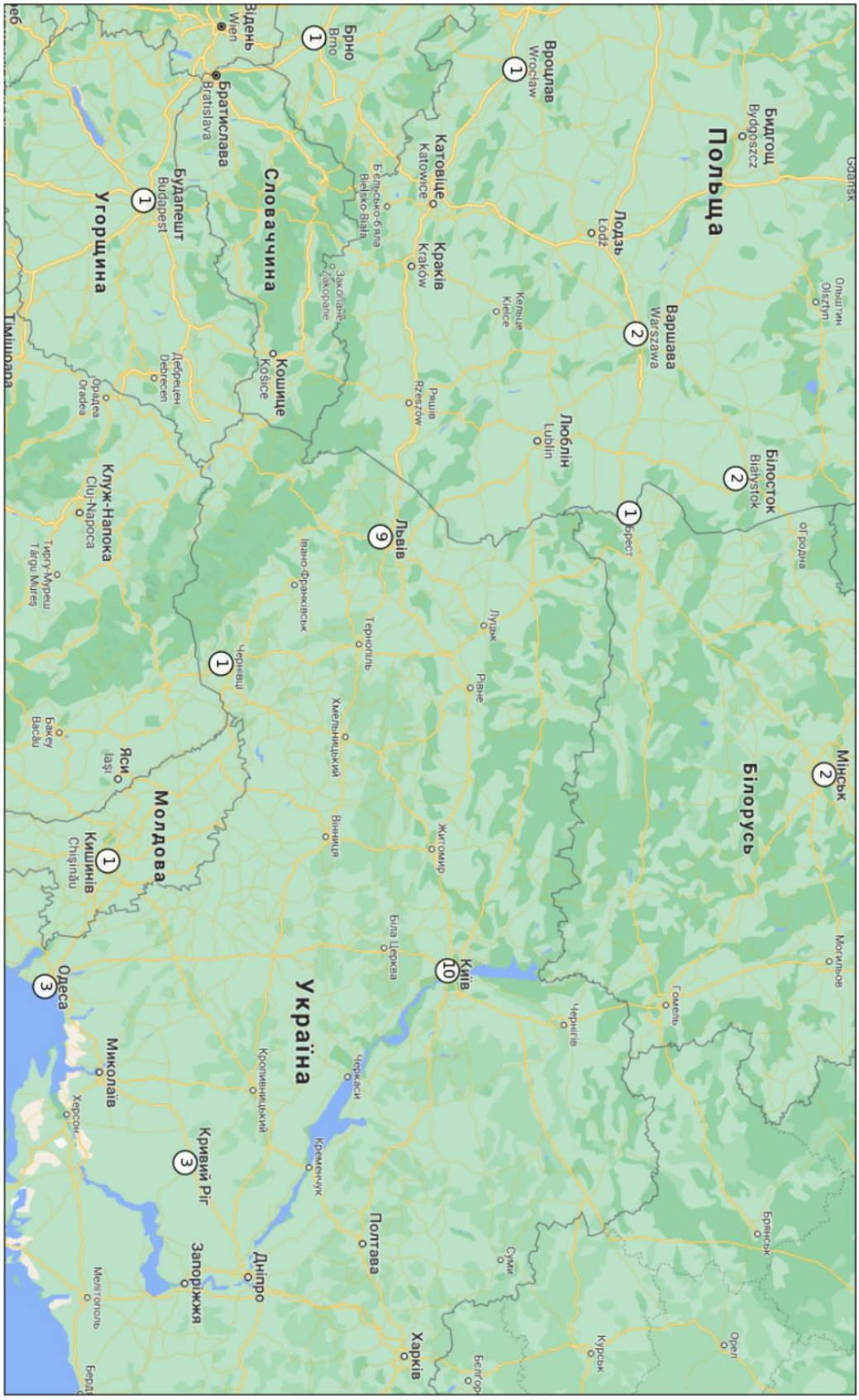


Рис. А.1. Графік міст