

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО”**

Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем

**КУРСОВА РОБОТА
ПОЯСНЮВАЛЬНА ЗАПИСКА**
з дисципліни “Бази даних”

спеціальність 121 – Програмна інженерія

на тему: **Система аналізу кліматичних показників**

**Студент
групи КП-83**

Симонюк В. П.
(ПІБ)

(підпис)

Викладач

Радченко К.О.

(підпис)

Київ – 2021

АНОТАЦІЯ

Дана курсова робота включала в себе здобуття практичних навичок у створенні прикладних програмних додатків, які взаємодіють із базою даних PostgreSQL. Було запропоновано наступні етапи розробки додатку:

- створення системи отримання / генерації та фільтрації даних;
- створення системи реплікації даних;
- створення системи аналізу даних предметної галузі;
- створення системи резервування / відновлення даних;
- створення системи візуалізації даних.

Результатом виконання курсової роботи стала реалізація усіх пунктів, описаних вище, та отримання кінцевої інформаційно-аналітичної системи, яка виконує аналіз даних, отриманих із зовнішніх ресурсів, та взаємодіє із реляційною СУБД PostgreSQL.

ЗМІСТ

Анотація.....	2
Зміст.....	3
Вступ.....	4
1. Аналіз інструментарію для виконання курсової роботи.....	6
2. Структура бази даних.....	8
3. Опис програмного забезпечення.....	9
3.1. Загальна структура програмного забезпечення.....	9
3.2. Опис модулів програмного забезпечення.....	9
3.3. Опис основних алгоритмів роботи.....	10
4. Аналіз функціонування засобів реплікації.....	11
5. Аналіз функціонування засобів резервування / відновлення.....	14
6. Аналіз результатів підвищення швидкодії запитів.....	16
7. Опис результатів аналізу предметної галузі.....	17
Висновки.....	18
Література.....	20
Додатки.....	22

ВСТУП

Основною метою даної курсової роботи є створення інформаційно аналітичної системи за допомогою мови програмування Python у середовищі розробки PyCharm, яка взаємодіє із базою даних PostgreSQL.

Призначенням програмного додатку є здійснення аналізу кліматичних показників.

Створення інформаційно-аналітичного додатку є актуальною задачею, оскільки він дозволяє значно прискорити, а також забезпечити процес аналізу великих обсягів даних, неможливий для здійснення людиною.

У процесі розробки було постановлено наступні основні етапи:

- створення системи генерації / отримання та фільтрації даних, основним призначенням якої є генерація/отримання даних для подальшої їх фільтрації та аналізу;
- створення системи реплікації даних, яка передбачає розділення функціональності отримання даних та їх передачі програмному додатку, для розподілення навантаження на базу даних та зменшення можливості втрати даних;
- створення системи аналізу даних, яка необхідна для аналізу отриманих раніше даних та їх візуалізації у вигляді графіків та діаграм;
- створення системи резервування / відновлення даних, необхідної для збереження стану бази даних та можливості відновлення даних на основі обраного користувачем файлу стану.

Усі ці пункти є важливими для програмного додатку, оскільки забезпечують стабільну роботу системи, а також надійність збереження даних.

Окрім пунктів, наведених вище, варто відзначити і процес підвищення швидкодії отримання даних із бази даних на основі застосування індексів БД, який також було виконано.

I. АНАЛІЗ ІНСТРУМЕНТАРІЮ ДЛЯ ВИКОНАННЯ КУРСОВОЇ РОБОТИ

Для виконання курсової роботи було обрано наступний інструментарій:

- СУБД – PostgreSQL 11. Даний вибір базується на тому, що PostgreSQL має безліч можливостей: тригери, індекси, функції, вбудовані методи реплікації та резервування. Крім того, дана СУБД дозволяє розгортання БД на декількох машинах та має широку підтримку;
- мова програмування – Python 3.9. Ключовим фактором у виборі мови програмування був її широкий спектр застосування у сфері Data Science та Big Data завдяки широкому спектру бібліотек;
- середовище розробки – PyCharm, оскільки дане середовище забезпечує комфортний процес написання коду, має вбудовані засоби його налагодження та виконання. Крім того, PyCharm має комфортний інтерфейс для пошуку та встановлення необхідних бібліотек;
- засоби генерації даних – отримання даних виконувалося шляхом здійснення requests запитів до Visual Crossing Weather API. Даний метод генерації даних забезпечує отримання достовірних даних і отримання адекватної статистики у подальшому при аналізі;
- засоби для роботи з СУБД – psycopg2 та sqlalchemy. Перша бібліотека є написаною на мові C, в результаті чого є ефективною та надійною при завантаженні великих обсягів даних (отримання даних із API), тим часом як sqlalchemy надає повний набір добре відомих шаблонів корпоративного рівня стабільності, сконструйованих для високопродуктивного

доступу до бази даних;

- засоби аналізу даних – Pandas. Ця бібліотека написана на основі бібліотеки numpy, реалізованої мовою програмування С, що забезпечує високу її високу швидкодію при аналізі великих обсягів даних;
- візуалізація даних – бібліотека matplotlib.

II. СТРУКТУРА БАЗИ ДАНИХ

База даних містить у собі наступні таблиці:

- Details – для опису всіх показників погоди і їх умови, містить наступні поля id_det, min_tem, max_tem, wind_speed, wind_dir, dew_point, precipitation, visibility, conditions, could_cov, relative_hum;
- Sity Date – сутність для опису країн і їх міст в певний час і деталями погоди, містить наступні поля id_cd, id_loc, id_det, date;
- Location – для опису сутності локації, містить наступні поля id_loc, address, longitude, latitude.

Зв'язки між сутностями:

- Location – Sity Date: ONE to MANY;
- Details – Sity Date: ONE to MANY.

III. ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1. Загальна структура програмного забезпечення

Програмний додаток було реалізовано у консольному варіанті із застосуванням шаблону ORM на базі операційної системи Windows 10.

2. Опис модулів програмного забезпечення

Програмний додаток складається із наступних модулів:

1. Models – директорія, яка містить моделі для роботи з ORM:
 - a. City Date;
 - b. Location;
 - c. Details.
2. View – модуль для генерування виду консольного меню для взаємодії з користувачем.
3. Controller – виконує збір даних шляхом запитів до модуля database і view.
4. Api_data – модуль, який виконує запити до Visual Crossing Weather API з метою отримання актуальних даних, та передає в database для запису.
5. Database – модуль, який виконує функції з контролера для отримання даних з slave-сервера або запису у master-сервер.
6. Main – ядро програмного додатку.
7. Data – модуль забезпечує фільтрацію та аналіз зібраних даних.

3. Опис основних алгоритмів роботи

Данні беруться за допомогою Visual Crossing Weather API, так як це найкращий сервер, де є все, що потрібно для отримання детальних даних у будь якій точці світу і з необхідною точністю.

Для аналізу двох країн був використаний алгоритм, який фіксує три точки щодоби і визначає середнє арифметичне значення і зрівнює цей же день з іншою країною підставляючи це у графік для детального спостереження.

Інший аналіз бере всі країни за вказаним проміжком і вираховує статистику по якомусь параметру у вигляді співвідношення демонструючи на діаграмі результати.

Робота програмного додатку базується на взаємодії модулів та алгоритмах, реалізованих у бібліотеці pandas. У разі виникнення помилок використані можливості перехоплення помилок за допомогою конструкції try...except.

IV. АНАЛІЗ ФУНКЦІОНУВАННЯ ЗАСОБІВ РЕПЛІКАЦІЇ

PostgreSQL дозволяє реалізацію реплікації за допомогою наступних методів:

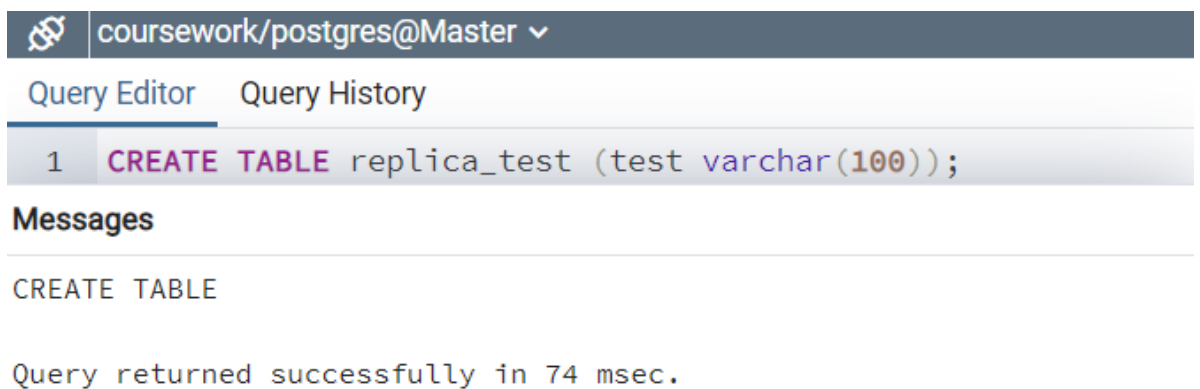
- трансляція файлів (головний сервер працює в режимі постійної архівації змін, тим часом як кожен резервний сервер виконує постійне отримання даних у вигляді WAL файлів від головного сервера);
- потокова реплікація (працює таким ж самим чином, як і при трансляції файлів, проте резервний сервер може працювати із меншими затримками, ніж у першому випадку. Резервний сервер підключається до головного, який передає для нього потік записів WAL у момент їх додавання, не чекаючи на їх повне заповнення);
- каскадна реплікація (резервний сервер приймає підключення реплікації та потоки WAL від інших резервних серверів, які виступають посередниками);
- синхронна реплікація (при такому методі реплікації кожна фіксація транзакції очікує на підтвердження того, що транзакція була записана у журнал транзакції на обох серверах: головному та резервному).

У процесі виконання курсової роботи було обрано реалізацію потокової реплікації (головний **master** сервер працює в режимі постійної архівації змін, тим часом як резервний **slave** сервер виконує постійне отримання даних у вигляді WAL файлів від головного сервера) через нативну утиліту постгреса.

За допомогою утиліти pg_basebackup був створений slave-сервер для реплікації даних. Master-сервер ми використовуємо для запису, оновлення та видалення, а slave - read-only.

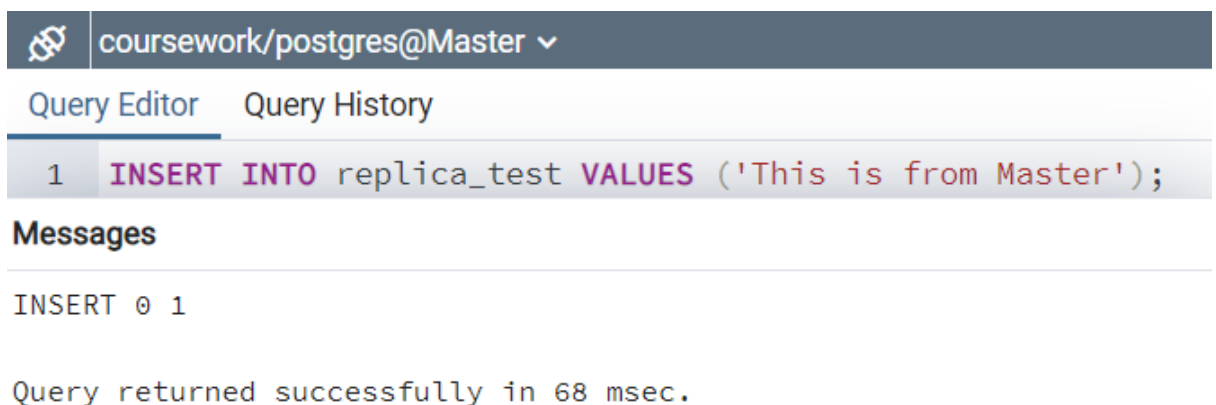
Протестуємо їх роботу:

1. Створимо тестову таблицю на master-сервері



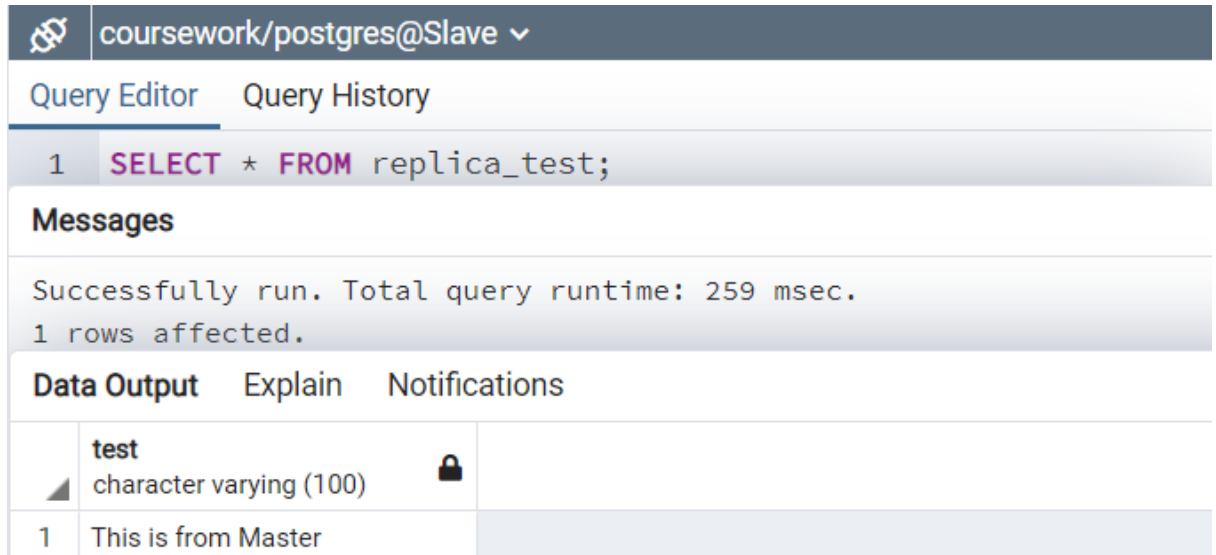
```
coursework/postgres@Master ▾  
Query Editor Query History  
1 CREATE TABLE replica_test (test varchar(100));  
Messages  
CREATE TABLE  
  
Query returned successfully in 74 msec.
```

2. Додамо рядок в таблицю



```
coursework/postgres@Master ▾  
Query Editor Query History  
1 INSERT INTO replica_test VALUES ('This is from Master');  
Messages  
INSERT 0 1  
  
Query returned successfully in 68 msec.
```

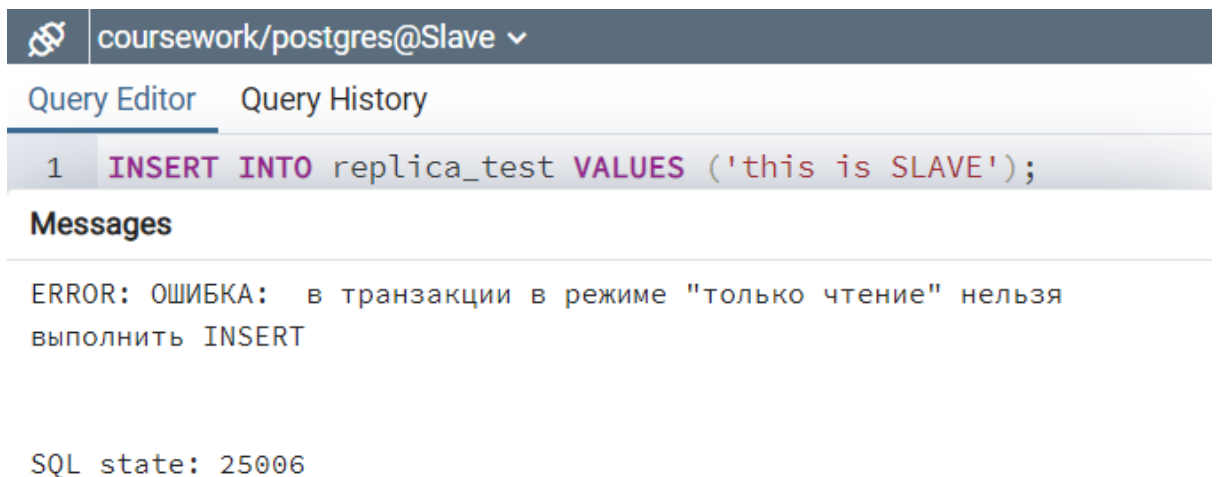
3. Дістанемо дані з slave-серверу



The screenshot shows a PostgreSQL client interface with the following components:

- Header:** A dark blue bar with a database icon and the text "coursework/postgres@Slave".
- Tabs:** "Query Editor" (active) and "Query History".
- Query:** A single line in the editor: `1 SELECT * FROM replica_test;`
- Messages:** A section showing the execution result: "Successfully run. Total query runtime: 259 msec. 1 rows affected."
- Data Output:** A table with one column named "test" of type "character varying (100)". It contains one row with the value "This is from Master".

4. Спробуємо записати дані в slave-server



The screenshot shows the same PostgreSQL client interface as before, but with an error:

- Query:** A single line in the editor: `1 INSERT INTO replica_test VALUES ('this is SLAVE');`
- Messages:** An error message: "ERROR: ОШИБКА: в транзакции в режиме "только чтение" нельзя выполнить INSERT".
- SQL state:** "SQL state: 25006".

Як можемо бачити з наведених вище операцій - засоби реплікації працюють справно.

V. АНАЛІЗ ФУНКЦІОНУВАННЯ ЗАСОБІВ

РЕЗЕРВУВАННЯ/ВІДНОВЛЕННЯ ДАНИХ

СУБД PostgreSQL пропонує наступні методи реалізації резервування та відновлення даних:

- використання періодичного резервного копіювання за допомогою вбудованої утиліти `pg_dump`;
- резервне копіювання на основі базових копій та архівів WAL.

Перший метод є достатньо швидким і простим у реалізації, проте забезпечує виконання відновлення лише за наявності backup-файла, тим часом як другий метод є складнішим у реалізації, потребує більше ресурсів та пам'яті.

У процесі розробки програмного додатку було застосовано перший метод резервування та відновлення, що забезпечує вибір користувачем файлу резервної копії та відновлення у будь-який із раніше збережених станів.

У програмному додатку резервне копіювання та відновлення можуть бути використані у разі виникнення наступних ситуацій:

- у випадку втрати як даних бази, так і самої бази;
- при виникненні потреби відновити один з попередніх станів сховища даних, збережених раніше.




Засоби резервування та відновлення даних впроваджені в консольному інтерфейсі. За його допомогою ми можемо створити файл відновлення поточного стану бази даних і виконувати відновлення.

Зробити це можна наступним чином:

1. Створимо файл відновлення поточного стану бази даних

```
Choose what you want to do:
1. Get data.
2. Graphs(line) of details.
3. Graphs(pie) of details.
4. Backup.
5. Restore.
6. Analysis countries.
7. Top 10.
8. Exit.
>> 4
-----
Successful backup!
```

2. Перевіримо наявність файлу у відповідній директорії

 1.dump	20.05.2021 8:53	Файл "DUMP"	29 825 КБ
 2.dump	20.05.2021 8:58	Файл "DUMP"	31 110 КБ
 next_backup_id.txt	20.05.2021 8:58	Текстовый документ	1 КБ

3. Виконаємо відновлення

```
Choose what you want to do:
1. Get data.
2. Graphs(line) of details.
3. Graphs(pie) of details.
4. Backup.
5. Restore.
6. Analysis countries.
7. Top 10.
8. Exit.
>> 5
-----
PostgreSQL ORM connection is closed
PostgreSQL psycopg2 connection is closed
-----
      filename
0         1.dump
1         2.dump
2         3.dump
3         4.dump
4 close this window
Enter file index: 1
SELECT pg_catalog.set_config('search_path', '', false);
DROP DATABASE coursework;
Drop
-----
SELECT pg_catalog.set_config('search_path', '', false);
CREATE DATABASE coursework ENCODING 'UTF-8';
Create
-----
SET
SET
SET
SET
SET
set_config
-----
(1 ♦♦♦♦♦♦)
SET
SET
SET
SET
SET
CREATE TABLE
ALTER TABLE
ALTER TABLE
CREATE TABLE
ALTER TABLE
ALTER TABLE
CREATE TABLE
ALTER TABLE
ALTER TABLE
COPY 172746
COPY 172746
ALTER TABLE
ALTER TABLE
ALTER TABLE
CREATE INDEX
CREATE INDEX
CREATE INDEX
ALTER TABLE
ALTER TABLE
-----
Successful restore!
-----
Connected to Slave server
Connected to Master server
-----
```

VI. АНАЛІЗ РЕЗУЛЬТАТІВ ПІДВИЩЕННЯ ШВИДКОДІЇ ВИКОНАННЯ ЗАПИТІВ

SELECT запити мають нижчу швидкодію при збільшенні обсягів даних та фільтрації за полями, які не входять до складу РК. Це спричинено тим, що у такому випадку виконується послідовне сканування усіх даних, що є досить неефективним. Саме тому у таких випадках (в середньому при наявності більш ніж 10000 рядків сутностей) застосовується індексація.

У таблиці Details було використано btree індекс для поля conditions, внаслідок чого вдалося отримати приріст швидкодії при отриманні умов погоди. У середньому, час планування операції зазнав або незначного приросту, або зменшення, тим часом як час виконання операції став у середньому меншим за 1 секунду. Візуальне представлення цих даних було наведено у додатку.

VII. ОПИС РЕЗУЛЬТАТІВ АНАЛІЗУ ПРЕДМЕТНОЇ ГАЛУЗІ

Аналіз даних предметної галузі було виконано для наступних сценаріїв:

- аналіз будь-якого поля з сутності Details може бути представлений у вигляді графіка;
- аналіз будь-якого поля з сутності Details може бути представлений у вигляді діаграми;
- вивід у вигляді стовбчастих діаграм порівняння двох країн на будь-яке поле з сутності Details, для деталізації абсолютної статистики;
- вивід діаграми статистики TOP 10 країн за останній місяць будь-якого поля з сутності Details.

Візуальне представлення отриманих результатів аналізу наведено у додатку.

ВИСНОВКИ

Підіб'ємо підсумки по кожному з етапів виконання курсовою роботи:

- для виконання курсової роботи було підібрано вдалий інструментарій у вигляді різних програмних бібліотек та програмного забезпечення, що дало змогу реалізувати усі програмні пункти роботи;
- структуру бази даних було спроектовано таким чином, щоб вона відповідала необхідним нормальним формам та забезпечувала найоптимальніший шлях для отримання даних із зовнішнього API;
- програмний додаток було реалізовано за допомогою мови програмування Python 3.9 на базі операційної системи Windows 10. Програмний код було розбито на невеликі та компактні модулі. Надлишкова або невідповідна функціональність в модулях відсутня;
- було проаналізовано усі засоби реплікації, доступні у СУБД PostgreSQL, внаслідок чого механізм реплікації було реалізовано за допомогою потокового методу, що забезпечує швидку асинхронну взаємодію між головним та резервним серверами. Тестування реплікації показало, що вона працює належним чином та забезпечує швидкий процес налагодження системи у разі відмови головного сервера;
- внаслідок аналізу методів резервування / відновлення було обрано метод, який забезпечує менше використання ресурсів системи та дозволяє вибір точки відновлення серед створених користувачем раніше. У відповідному пункті було проаналізовано роботу відповідного механізму та підтверджено його коректну роботу;
- за допомогою механізму індексації в PostgreSQL вдалося отримати підвищення швидкодії виконання запитів;
- було реалізовано механізм аналізу та візуалізації даних у вигляді діаграм та графіків за допомогою зовнішніх бібліотек мови

програмування Python, таких як pandas, matplotlib, що забезпечує швидку роботу даними та їх аналіз;

- засоби генерації та фільтрації даних було реалізовано за допомогою звернень до зовнішнього серверу даних Visual Crossing Weather API, що дозволяє отримання достовірних для обраної предметної галузі даних та їх оновлення.

У процесі виконання курсової роботи було набуто практичних навичок у створенні програмних додатків мовою програмування Python, які взаємодіють із реляційною СУБД, здобуто практичний досвід технологіях аналізу даних.

Відповідно до наведених виконаних пунктів роботи можна стверджувати, що основна мета роботи (аналізу кліматичних показників) була досягнута – додаток готовий до використання.

ЛІТЕРАТУРА

1. What is PostgreSQL? educba: веб-сайт. URL:
<https://www.educba.com/what-is-postgresql/>.
2. Advantages of PostgreSQL. bitnine: веб-сайт. URL:
<https://bitnine.net/blog-postgresql/advantages-of-postgresql/?ckattempt=1>.
3. Advantages of PostgreSQL. Cybertec PostgreSQL: веб-сайт. URL:
<https://www.cybertec-postgresql.com/en/postgresql-overview/advantages-of-postgresql/>.
4. PyGreSQL vs psycopg2. stackoverflow: веб-сайт. URL:
<https://stackoverflow.com/questions/413228/pygresql-vs-psycopg2>.
5. What is the best python module to use with PostgreSQL?
<https://www.quora.com/What-is-the-best-python-module-to-use-with-PostgreSQL>.
6. TOP 5 PYTHON LIBRARIES FOR VISUALIZATION. Tech Shenanigans: веб сайт. URL:
<https://www.techshenanigans.com/post/top-5-python-libraries-for-visualization>.
7. How Not To Sort By Average Rating. Evanmiller: веб-сайт. URL:
<https://www.evanmiller.org/how-not-to-sort-by-average-rating.html>.
8. Bayesian Average Ratings. Evanmiller: веб-сайт. URL:
<https://www.evanmiller.org/bayesian-average-ratings.html>.
9. Резервное копирование и восстановление в PostgreSQL [Электронный ресурс] – Режим доступа до ресурсу:
<https://habr.com/ru/post/178567/>.

10. Введение в pandas: анализ данных на Python [Электронный ресурс] –

Режим доступа до ресурсу:

<https://khashtamov.com/ru/pandas-introduction/>.

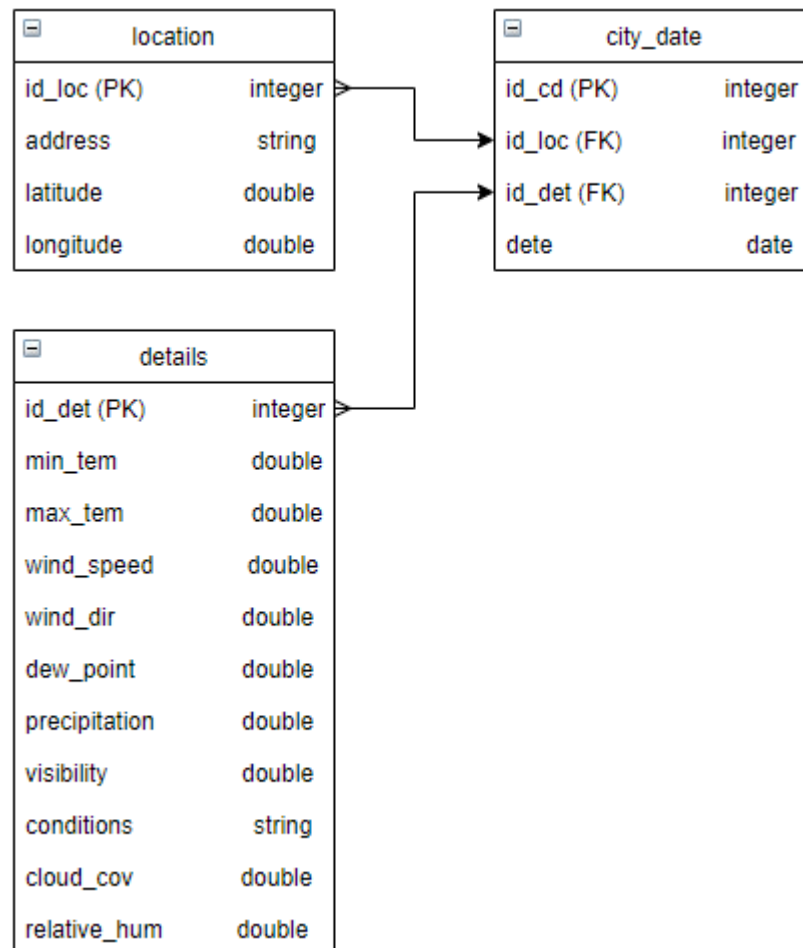
11. PostgreSQL: Документация [Электронный ресурс] – Режим доступа

до ресурсу:

<https://postgrespro.ru/docs/postgresql/9.6/warm-standby>.

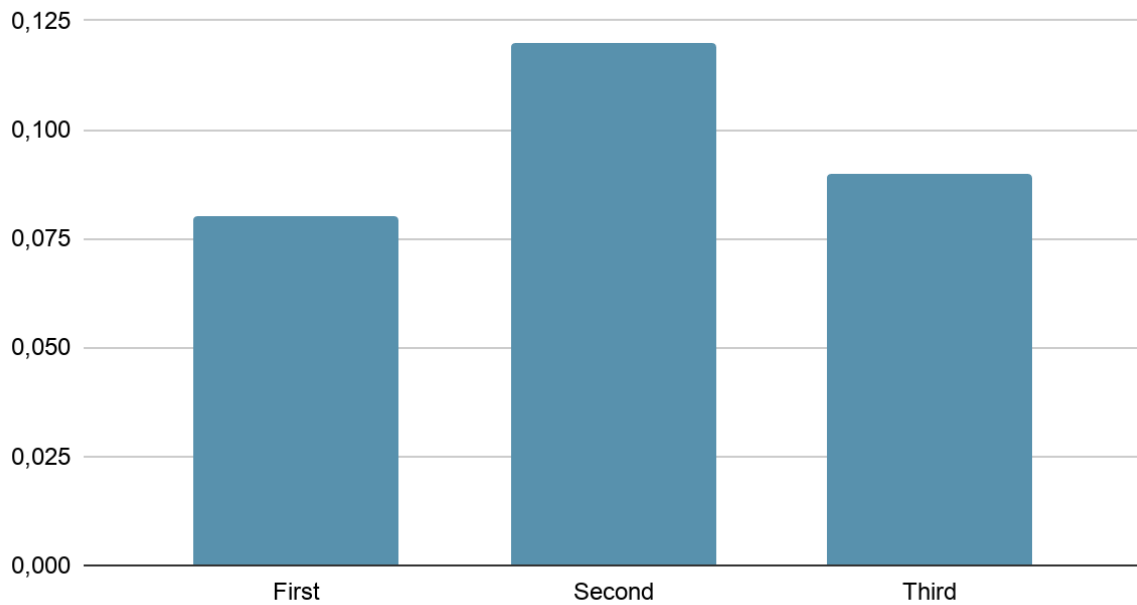
ДОДАТКИ

Структура бази даних:



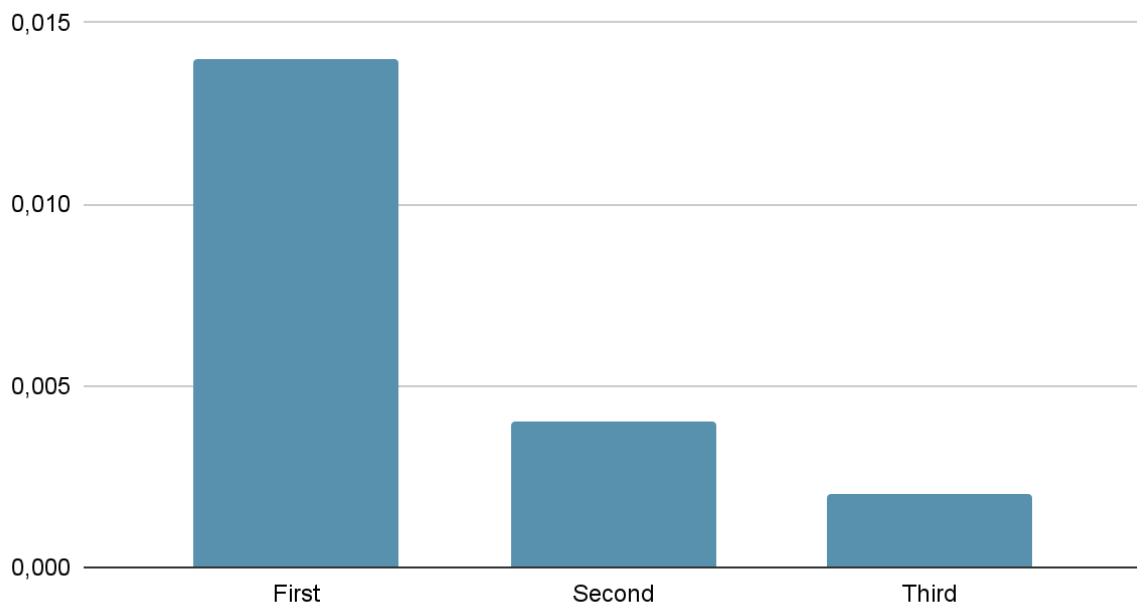
Результати підвищення швидкодії. До:

Requests speed (s)

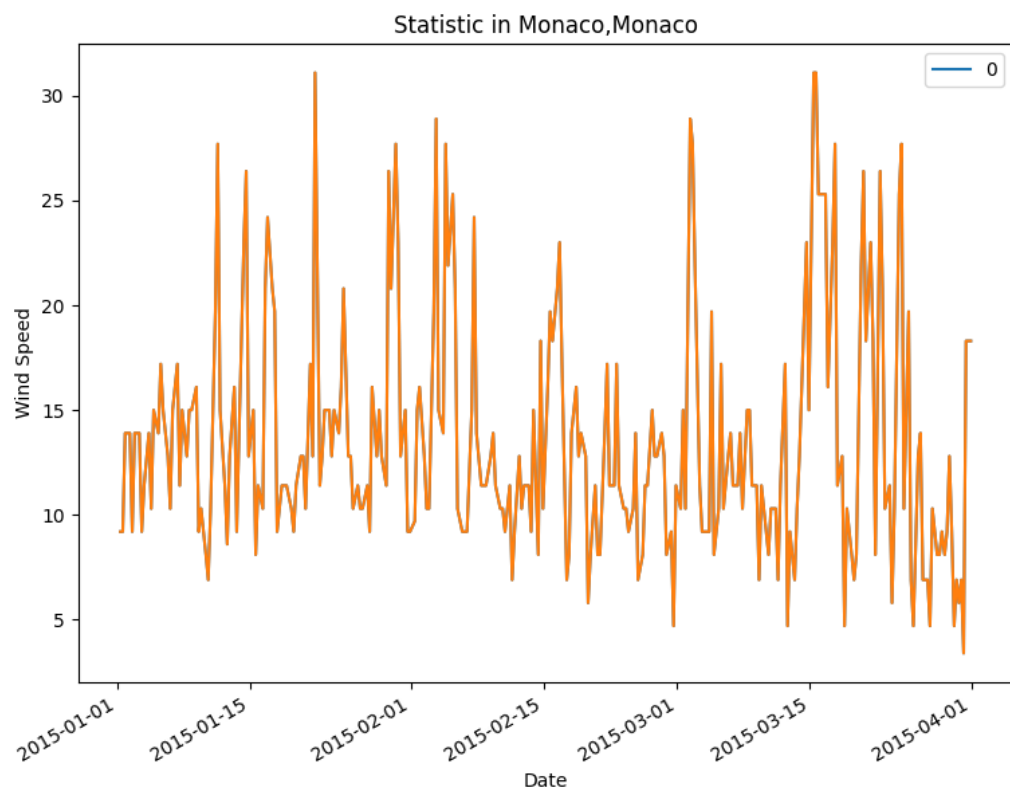


Після:

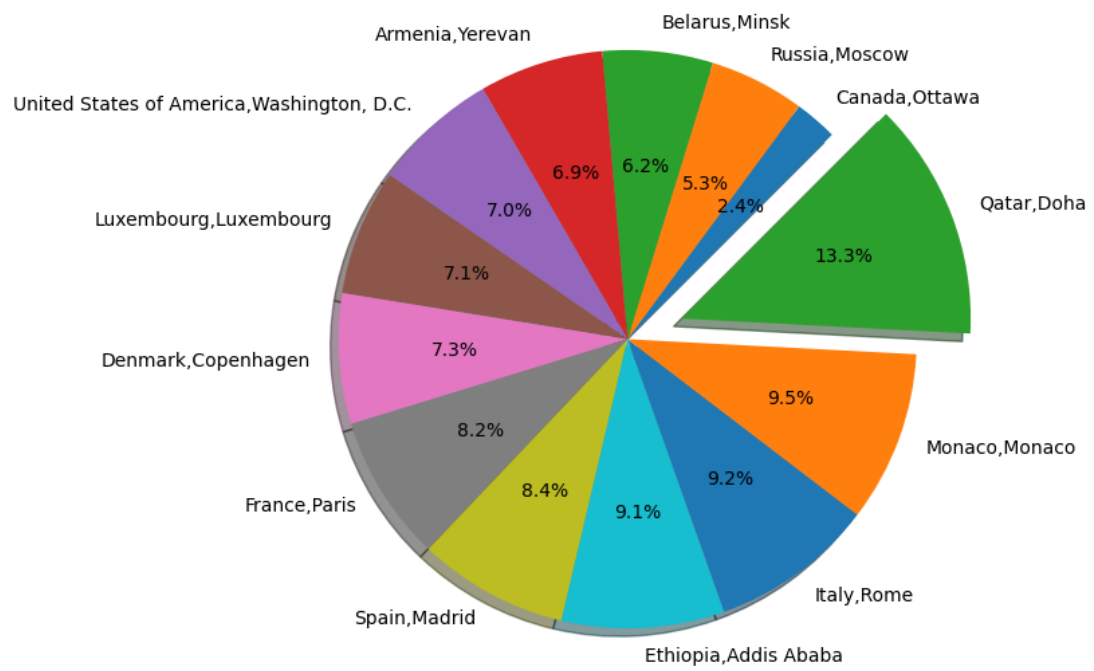
Requests speed (s)

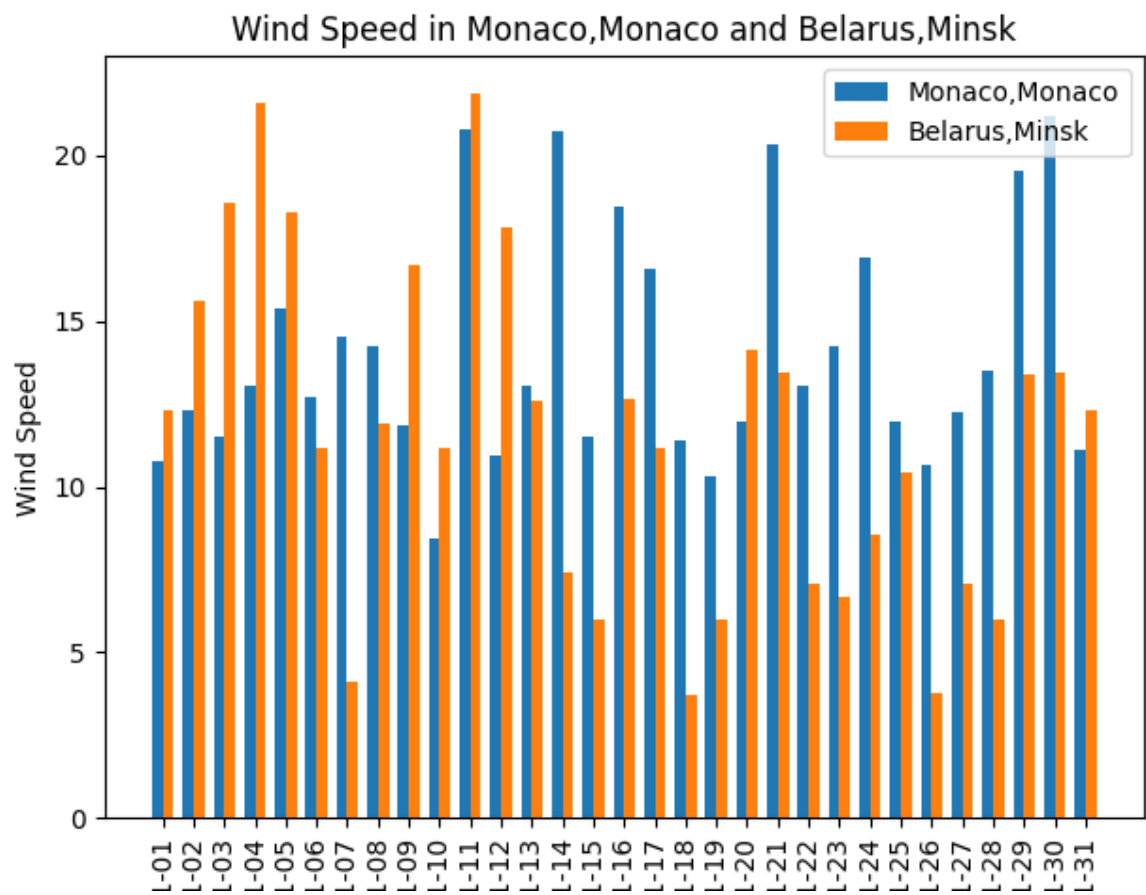


Результати аналізу даних:



Statistic for Minimum Temperature





Statistic for Cloud Cover

