Лабораторна робота № 5. СУБД часових рядів: TimescaleDB

МЕТА РОБОТИ: отримати навички роботи з базами даних часових рядів.

ПОСТАНОВКА ЗАДАЧІ

Розробити додаток, який буде відправляти дані з датчиків до бази даних, яка зберігається в СУБД TimescaleDB. Пропонується 2 варіанти клієнтського застосунку, які оцінюються різною кількістю балів. Варіант 1 передбачає генерування даних з датчиків безпосередньо у застосунку за допомогою генератору випадкових чисел. Варіант 2 передбачає використання реальних даних з датчиків, що присутні в мобільному телефоні студента. Ці дані надсилаються до додатку, який зберігає їх у базі даних TimescaleDB. Для аналізу часових рядів використовується Grafana.

ПОРЯДОК ВИКОНАННЯ

Варіант 1 (достатній рівень, оцінюється у **5 балів**).

1. Якщо студент не має у себе встановленої СУБД PostgreSQL, то встановити її з офіційного сайту https://www.postgresql.org/download/

2. Встановити СУБД TimescaleDB, яка є розширенням СУБД PostgreSQL, з офіційного сайту: https://docs.tigerdata.com/self-hosted/latest/install/. Нижче наведено послідовність кроків для встановлення TimescaleDB на ОС Windows:

2.1. Завантажте останню стабільну версію інсталлятора з офіційного сайту: https://docs.tigerdata.com/self-hosted/latest/install/installation-windows/#supported-platforms

2.2. Запустіть інсталятор та виконуйте його вказівки.

Якщо установник пише, що не може знайти утиліту pg_config, то треба додати шлях до цієї утиліти (зазвичай це щось подібне до C:\Program Files\PostgreSQL\17\bin\) до списку шляхів системної змінної PATH.

Коли під час установки установник попросить вказати шлях до postgresql.conf, вкажіть повний шлях до цього файлу, який зазвичай знаходиться в папці data Вашої установки PostgreSQL.

Під час установки установник буде пропонувати деякі зміни в конфігурації PostgreSQL, які, на його думку, сприятимуть продуктивності використовуваної СУБД. Найкритичнішими з цих змін є ті, що стосуються оперативної пам’яті. Для потреб поточної лабораторної роботи рекомендується відмовитись від запропонованих змін. Для цього достатньо натиснути “s” (skip).

Решту пропозицій можна прийняти, вони є менш критичними з точки зору продуктивності самого комп'ютера.

2.3. Після того, як розширення буде встановлене, Ви можете ним користуватись. Для цього в базі даних, в якій Ви плануєте працювати з часовими рядами, виконайте команду:

```
CREATE EXTENSION IF NOT EXISTS timescaledb;
```

Перелік доступних розширень можна подивитись за допомогою команди:

```
SELECT * FROM pg_extension;
```

3. Створити базу даних, в якій Ви будете зберігати Ваші дані.

3.1. Спочатку створюєте звичайну таблицю, наприклад:

```
CREATE TABLE sensor_readings (
    time TIMESTAMPTZ NOT NULL,
    device_id INTEGER NOT NULL,  -- ID пристрою
    temperature NUMERIC,         -- Температура
    humidity NUMERIC             -- Вологість
    PRIMARY KEY (time, device_id)
);
```

3.2. На базі звичайної таблиці створюєте гіпертаблицю:

```
SELECT create_hypertable('sensor_readings', 'time');
```

3.3. Якщо необхідно, створюєте інші таблиці та гіпертаблиці.

Примітка! Гіпертаблиці потрібні виключно для зберігання часових рядів. Для зберігання інших даних використовуйте звичайні таблиці, які, за необхідності, можуть бути пов'язані, через зовнішні ключі, з гіпертаблицями.

4. Розробити додаток, який буде відправляти дані по температурі, вологості повітря та швидкості вітра до TimescaleDB.

Дані з датчиків "отримувати" за допомогою генератора випадкових чисел. Приклад коду на Python:

```python
import psycopg2
import random
import datetime
import time
import sys

try:
    # Параметри підключення до локальної бази даних TimescaleDB
    conn = psycopg2.connect(
        dbname="iotdb",  # назва вашої БД
        user="postgres",  # користувач
        password="password",  # пароль
        host="localhost",
        port="5432"
```

```
    )
except psycopg2.OperationalError as e:
    print("Помилка підключення до бази даних:", e)
    sys.exit(1)

# Використовуємо блок `with` для автоматичного керування ресурсами.
# Це гарантує, що з'єднання та курсор будуть закриті, навіть якщо
# виникне помилка.
with conn:
    with conn.cursor() as cur:
        try:
            # Генерація та запис випадкових значень температури
            for _ in range(100):
                temp = round(random.uniform(-10, 35), 2)
                now = datetime.datetime.now()
                # Запис у таблицю (яка вже існує)
                cur.execute(
                   "INSERT INTO temperatures (time, value) VALUES (%s, %s);",
                   (now, temp)
                )
                # Підтверджуємо транзакцію, щоб зберегти дані в БД
                conn.commit()
                print(f"Збережено: {now} -> {temp} °C")
                time.sleep(0.1)  # пауза 0.1 секунди між вимірюваннями
        except psycopg2.Error as e:
            print("Помилка під час виконання SQL-запиту:", e)
            conn.rollback()
```

5. Оформити звіт, в який включити розроблений в межах лабораторної роботи код, скріншоти результатів роботи застосунку та відповіді на контрольні запитання.

Варіант 2 (підвищений рівень, оцінюється у **10 балів**).

Загальна архітектура системи, яку потрібно побудувати, представлена на рисунку 1.

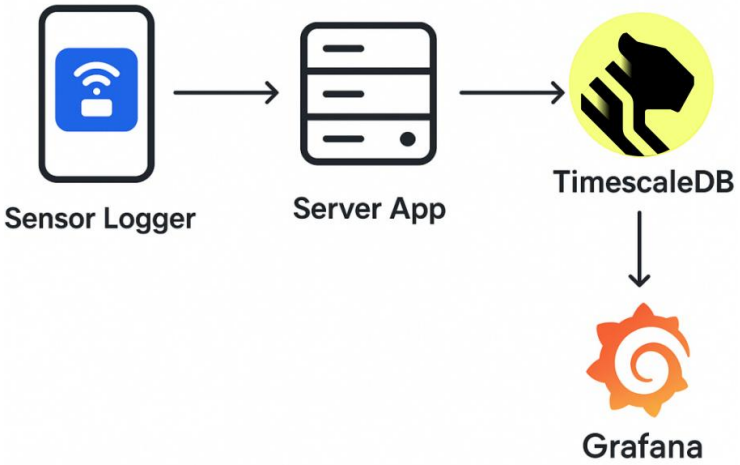


Рис. 1. Схема системи аналізу показів датчиків телефону

Етапи 1-3 ідентичні етапам з Варіанту 1.

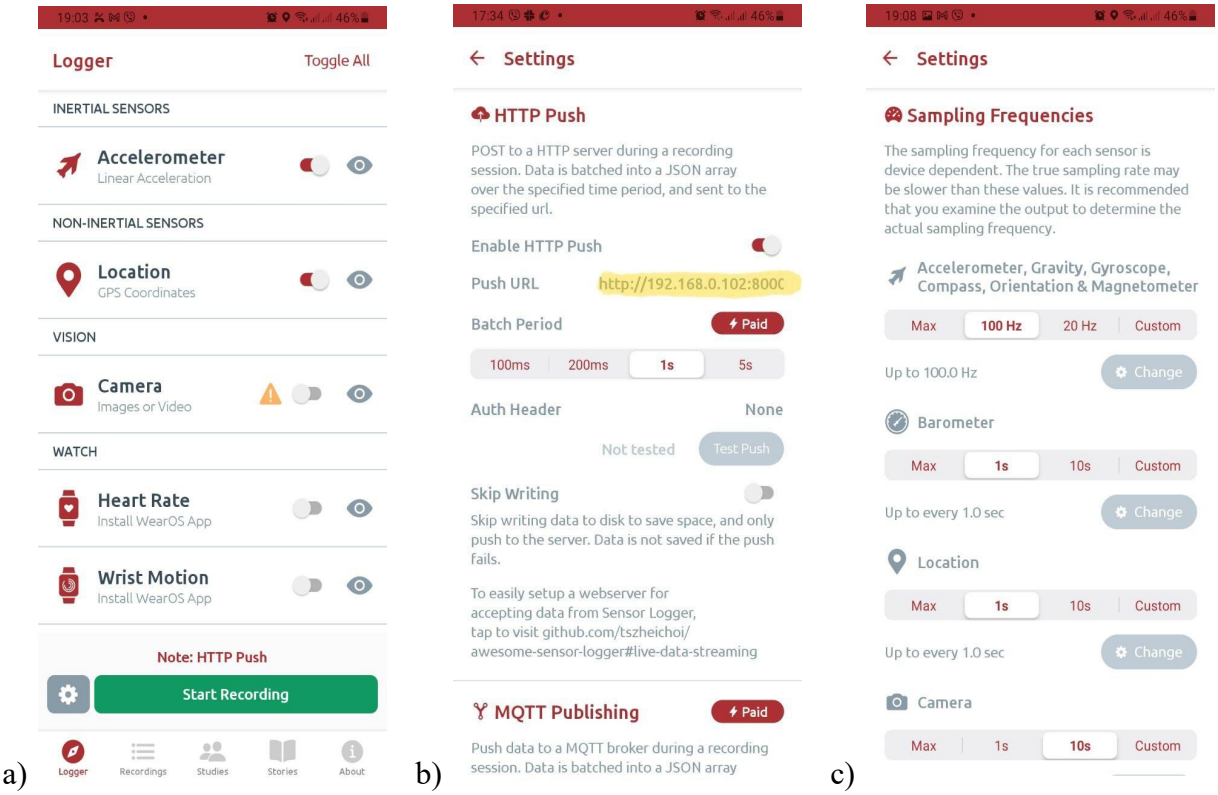
4. Встановити на мобільний телефон додаток, який збирає покази з датчиків та передає їх на вказаний ендпоінт. У якості додатка пропонується використовувати Sensor Logger (кросплатформовий, доступний і в Apple Store і в Google Play). В межах потрібного для лабораторної роботи функціоналу додаток безкоштовний. За бажанням можна обрати інший додаток або написати власний.

4.1. Завантажити та встановити додаток.

4.2. Обрати датчики, з яких планується відправка даних до застосунку (рис. 1a). Перелік датчиків обирається самостійно студентом в залежності від доступних в його смартфоні. Це можуть бути дані з акселерометру, місцерозташування, крокоміра, гіроскопа, яскравості тощо.

4.3. Налаштувати URL, на який буде відбуватись відправка даних з датчиків, в якому вказати локальну адресу вашого комп'ютера/ноутбука, порт, на якому буде запущений додаток, та локальниий шлях, наприклад: http://192.168.0.102:8000/sensor_data (рис. 1b).

4.4. Зверніть увагу, що дані відправляються пакетами, в яких можуть бути досить багато замірів. В межах безкоштовної версії відправка даних здійснюється щосекунди, але заміри відбуваються набагато частіше. Можна спробувати налаштувати частоту замірів (рис. 1c).



a) b) c)

Рис. 2. Налаштування Sensor Logger

5. Розробити програмний додаток, який буде приймати дані з мобільного додатку. Мову програмування обирається мтудентом самостійно. У випадку Python це може бути, наприклад, Flask-додаток. Найпростіший варіант додатку наведено в лістингу:

```
from flask import Flask, request, jsonify
import psycopg2
import sys

# Спроба підключення до TimescaleDB
try:
    conn = psycopg2.connect("...")
except psycopg2.OperationalError as e:
    print(f"Failed to connect to the database: {e}")
    sys.exit(1)

app = Flask(__name__)

@app.route('/sensor_data', methods=['POST'])
def receive_data():
    try:
        data = request.json
        # Обробка та запис даних в TimescaleDB
        return "OK", 200
    except Exception as e:
        print(f"An error occurred: {e}")
        return jsonify({"error": "An internal server error occurred."}), 500

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8000)
```

6. За допомогою розробленого додатку здійснити запис даних з датчиків мобільного телефону до TimescaleDB протягом певного проміжку часу.

7. Встановити додаток Grafana та побудувати дашборди з графіками результатів показників з датчиків телефону.

7.1. Установка додатку на ОС Windows складається з наступних кроків:

7.1.1. Завантажити з офіційного сайту https://grafana.com/grafana/download інсталянт (файл з розширенням .msi).

7.1.2. Запустити завантажений .msi файл та виконувати вказівки майстра установки ("Next", "Install", "Finish").

7.2. Запуск додатку:

7.2.1. Відкрийте ваш браузер та перейдіть за адресою: http://localhost:3000.

7.2.2. Ви побачите сторінку входу до Grafana. Використовуйте стандартні облікові дані admin/admin.

7.3. Підключення Grafana до TimescaleDB:

7.3.1. У Grafana зайдіть в меню "Connections" -> "Data Sources". Натисніть "Add data source" та виберіть "PostgreSQL".

7.3.2. Заповніть дані для підключення:
Host: localhost:5432
Database: назва тієї бази, в якій Ви зберігаєте покази датчиків
User: postgres
Password: Пароль, який ви задали під час встановлення PostgreSQL.

7.3.3. Прокрутіть сторінку до розділу "PostgreSQL Details".

7.3.4. Відключіть SSL-захист:
Знайдіть опцію "SSL Mode" (Режим SSL).
Натисніть на меню, що випадає, поруч із нею і виберіть значення disable (вимкнути).

7.3.5. Натисніть "Save & Test". Ви маєте отримати повідомлення "Database Connection OK".

7.4. Побудова графіків: використовуючи інтуїтивно зрозумілий інтерфейс Grafana, створити дашборди з декількома графіками: прискорення, геолокації тощо. Поекспериментувати з різними типами графіків. Приклад графіку наведено на рисунку 3.

Примітка! Зверніть увагу на часовий інтервал, для якого ви будуєте графіки. За необхідності, змініть його на той, який відповіда. Інтервалу запису даних з датчиків.

Рис. 3. Дані з датчиків прискорення

8. Оформити звіт, до якого додати скріншоти, що демонструють виконання всіх етапів завдання. До звіту включити також відповіді на контрольні запитання.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. До якого типу СУБД відноситься TimescaleDB? Які ще СУБД з цього класу є популярними на поточний момент? Аргументуйте свою відповідь, прикріпивши посилання на відповідні рейтинги.
2. Які типові сценарії використання СУБД часових рядів ви знаєте?
3. Яку роль відіграють гіпертаблиці в TimescaleDB? Чим вони відрізняються від звичайних таблиць?
4. Що таке неперервні агрегати в контексті TimescaleDB?
5. Який саме механізм дозволяє TimescaleDB ефективно оперувати величезними обсягами часових даних?