



НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

Проектування висконавантажених систем

Лабораторна робота №2

Виконав:
Поночевний Назар ФІ-92
Перевірив:
Родіонов А. М.

Київ – 2022

ЛАБОРАТОРНА РОБОТА №2

Реалізація каунтера з використанням PostgreSQL

Завдання:

- 1) Необхідно декількома способами реалізувати оновлення значення каунтера в СКБД PostgreSQL та оцінити час кожного із варіантів.
 - a) Lost-update (реалізація що втрачатиме значення)
 - b) In-place update
 - c) Row-level locking
 - d) Optimistic concurrency control

Результати

1) Інсталяція PostgreSQL

```
root@MSI: /mnt/c/Users/Nazar/PythonWorkspace
root@MSI:/mnt/c/Users/Nazar/PythonWorkspace# docker run -it --rm -e POSTGRES_USER=nazar -e POSTGRES_PASSWORD=nazar -p 5432:5432 postgres
The files belonging to this database system will be owned by user "postgres".
This user must also own the server process.

The database cluster will be initialized with locale "en_US.utf8".
The default database encoding has accordingly been set to "UTF8".
The default text search configuration will be set to "english".

Data page checksums are disabled.

fixing permissions on existing directory /var/lib/postgresql/data ... ok
creating subdirectories ... ok
selecting dynamic shared memory implementation ... posix
selecting default max_connections ... 100
selecting default shared_buffers ... 128MB
selecting default time zone ... Etc/UTC
creating configuration files ... ok
running bootstrap script ... ok
performing post-bootstrap initialization ... ok
syncing data to disk ... ok

initdb: warning: enabling "trust" authentication for local connections
initdb: hint: You can change this by editing pg_hba.conf or using the option -A, or --auth-local and --auth-host, the next time you run initdb.

Success. You can now start the database server using:

    pg_ctl -D /var/lib/postgresql/data -l logfile start
```

- 2) Код (для економії часу інкремент до 1000 в 10 потоках):

```
import time
import psycopg2
import threading

# Docker: docker run -it --rm -e POSTGRES_USER=nazar -e
POSTGRES_PASSWORD=nazar -p 5432:5432 postgres
config = {
    "host": "localhost",
    "port": 5432,
```

```

    "dbname": "postgres",
    "user": "nazar",
    "password": "nazar",
}
conn = psycopg2.connect(**config)
cursor = conn.cursor()
inc_value = 1000

def reset_db(conn, cursor):
    cursor.execute("DROP TABLE IF EXISTS user_counter")
    cursor.execute("CREATE TABLE user_counter (user_id INT PRIMARY
KEY, counter INT, version INT)")
    conn.commit()

    cursor.execute("INSERT INTO user_counter (user_id, counter,
version) VALUES (1, 0, 1)")
    conn.commit()

def get_counter(conn, cursor):
    cursor.execute("SELECT counter FROM user_counter WHERE user_id =
1")
    counter = cursor.fetchone()[0]
    return counter

def inference(func):
    threads = []
    for t in range(10):
        thread = threading.Thread(target=func)
        thread.start()
        threads.append(thread)

    for thread in threads:
        thread.join()

def lost_update():
    with psycopg2.connect(**config) as conn:
        conn.autocommit = True
        cursor = conn.cursor()
        for _ in range(inc_value):
            cursor.execute("SELECT counter FROM user_counter WHERE
user_id = 1")
            counter = cursor.fetchone()[0]

```

```

        counter += 1
        cursor.execute(f"UPDATE user_counter SET counter =
{counter} WHERE user_id = 1")

def in_place_update():
    with psycopg2.connect(**config) as conn:
        conn.autocommit = True
        cursor = conn.cursor()
        for _ in range(inc_value):
            cursor.execute("UPDATE user_counter SET counter = counter
+ 1 WHERE user_id = 1")

def row_level_locking():
    with psycopg2.connect(**config) as conn:
        conn.autocommit = True
        cursor = conn.cursor()
        for _ in range(inc_value):
            cursor.execute("SELECT counter FROM user_counter WHERE
user_id = 1 FOR UPDATE")
            counter = cursor.fetchone()[0]
            counter += 1
            cursor.execute(f"UPDATE user_counter SET counter =
{counter} WHERE user_id = 1")

def optimistic_concurrency():
    with psycopg2.connect(**config) as conn:
        conn.autocommit = True
        cursor = conn.cursor()
        for _ in range(inc_value):
            while True:
                cursor.execute("SELECT counter, version FROM
user_counter WHERE user_id = 1")
                counter, version = cursor.fetchone()
                counter += 1
                cursor.execute(f"UPDATE user_counter SET counter =
{counter}, version = {version + 1} WHERE user_id = 1 AND version =
{version}")
                if cursor.rowcount > 0:
                    break

reset_db(conn, cursor)
start_time = time.time()

```

```

inference(lost_update)
print("--- Lost-update ---")
print(f"Final value of counter: {get_counter(conn, cursor)}")
print(f"Execution time: {round(time.time() - start_time, 2)}")

reset_db(conn, cursor)
start_time = time.time()
inference(in_place_update)
print("--- In-place update ---")
print(f"Final value of counter: {get_counter(conn, cursor)}")
print(f"Execution time: {round(time.time() - start_time, 2)}")

reset_db(conn, cursor)
start_time = time.time()
inference(row_level_locking)
print("--- Row-level locking ---")
print(f"Final value of counter: {get_counter(conn, cursor)}")
print(f"Execution time: {round(time.time() - start_time, 2)}")

reset_db(conn, cursor)
start_time = time.time()
inference(optimistic_concurrency)
print("--- Optimistic concurrency control ---")
print(f"Final value of counter: {get_counter(conn, cursor)}")
print(f"Execution time: {round(time.time() - start_time, 2)}")

conn.close()

```

3) Знімки екрана з логами виконання:

```

C:\Windows\system32\cmd.exe
((dataint)) C:\Users\Nazar\PythonWorkspace\Programming-Labs\Data Intensive Apps\Lab2>python lab2.py
--- Lost-update ---
Final value of counter: 1000
Execution time: 9.45
--- In-place update ---
Final value of counter: 10000
Execution time: 5.11
--- Row-level locking ---
Final value of counter: 10000
Execution time: 9.27
--- Optimistic concurrency control ---
Final value of counter: 10000
Execution time: 9.13

```