**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**

з дисципліни “Бази даних”

спеціальність 121 – Програмна інженерія

на тему: Система аналізу цін споживчих товарів

(назва теми)

|  |  |  |
| --- | --- | --- |
| **Студент групи** КП-93 | **Долгов О. Ю.**  (ПІБ) | \_\_\_\_\_\_\_\_\_  (підпис) |
| **Викладач**  **к.т.н, доцент кафедри СПіСКС** | **Петрашенко А.В.** | \_\_\_\_\_\_\_\_\_  (підпис) |

Захищено з оцінкою \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Київ – 2020

**Анотація**

В ході виконання курсової роботи було створено додаток для аналізу цін на споживчі продукти. Підґрунтям для створення додатку стала інформація, розміщена в вільному доступі на торгівельній площадці rozetka.com.

**Зміст**

1. Анотація.
2. Зміст.
3. [Вступ](#_ezgj41728qgk).
4. [Аналіз інструментарію для виконання курсової роботи.](#_98prtg2r4jok)
5. [Структура бази даних.](#_yhn8pmj1zz2s)
6. [Опис програмного забезпечення](#_q3gdketapocm).

6.1 Загальна структура програмного забезпечення.

6.2 Опис модулів програмного забезпечення.

6.3 Опис основних алгоритмів роботи.

1. [Аналіз функціонування засобів реплікації.](#_ygx40wq30sd2)
2. [Аналіз функціонування засобів резервування/відновлення.](#_b6pwbjudgo4p)
3. [Аналіз результатів підвищення швидкодії запитів.](#_4vawyc26280a)
4. [Опис результатів аналізу предметної галузі.](#_k3w7oyivlmo4)
5. [Висновки.](#_aoc8it6mo547)
6. [Література](#_oupi48hwwpvq).
7. [Додатки](#_ip4lfyiugqj4)

## **Вступ**

Під час аналізу ринку споживчих товарів професійними аналітиками

виникає необхідність в системі аналізу цін на споживчі товари, яка здатна

зберігати велику кількість даних даної природи та надавати інструменти

для аналізу збереженої інформації. Основною проблемою цього класу

користувачів є незручність у використанні базового інструментарію баз

даних через необхідність виконувати багато складних і великих

пошукових запросів до бази, що зменшує швидкість праці та призводить

до появи людських помилок, а також відсутність вбудованих засобів

візуалізації даних.

Метою розробки цього програмного забезпечення є надання зручної

у використанні програми з консольним інтерфейсом для виконання цього

спектру задач.

## **Аналіз інструментарію**

Для виконання курсової роботи було обрано такі інструменти:

Мова програмування - Python3

Бібліотека для роботи з базою даних - Sqlalchemy

Бібліотека для візуалізації даних - Pyplot

Бібліотека для статистичного аналізу даних - Pandas

Фреймворк для скрепінгу - Scrapy

База даних - Postgresql12

Мова python3 була обрана через наявність в ній зручних інструментів для збору, аналізу та візуалізації даних, а також простотою синтаксису, що робить її зручною для використання і підвищує швидкість розробки.

Вибір sqlalchemy, pyplot, pandas був зумовлений тим, що ці інструменти можна вважати стандартними в мові python для роботи з базою даних, візуалізації даних, аналізу даних відповідно.

Для здійснення скрепінгу був здійснений вибір між фреймворком Scrapy та бібліотекою Beautiful Soup на користь Scrapy, через те, що Scrapy більш пристосований для збору інформації з великої кількості сторінок, ніж Beautiful Soup, який більше підходить для збору інформації з обмеженої кількості сторінок. Scrapy досягає такої ефективності при роботі з великим об’ємом роботи через свою асинхронну природу.

Увесь код був написаний за допомогою IDE Pycharm.

## **Структура бази даних**

див. Додаток 1

## **Опис програмного забезпечення**

### **Загальна структура**

Основна програма побудована згідно з шаблоном MVC.

Також, розроблено додаткову утіліту для скрепінгу товарів з сайту rozetka.com

### **Модулі**

app.py - Зчитує команди користувача до консолі, та передає їх до роутера

router.py - Відповідно до команди користувача делегує задачу до певного метода контролера

controller.py - Збирає необхідні параметри для виконная команд від користувача та делегує роботу до моделі

model.py - Напряму працює з базою даних, відповідно до переданих інструкцій з модуля контроллера; Зчитує .json які надає модуль скрепера.

scraper - модуль для збору інформації с веб-сторінок, та її збереження у .json файли.

Має стандартну структру проекта фреймворка scrapy[1]

**Аналіз основних алгоритмів роботи**

Окрім надання інтерфейсу для та візуалізації необхідних даних, задачею побудованої програми є збереження цілісності бази даних. Для її підтримки було розроблено два тригера before insert для таблиць “Consignment\_arrival” та “Volume\_of\_sales” (див. додаток 2)

## 

## **Аналіз засобів реплікації**

В якості slave-сервера було обрано сервер, якій розміщений на віртуальній машині. Перевірку наявності реплікації було здійснено за допомогою:

1. Sql запиту SELECT client\_addr, state FROM pg\_stat\_replication;

(див. Додаток 3)

1. Емпірично, шляхом вставки нових даних до master-серверу та перевірки нового змісту slave-сервера

Побудована реплікація працює в асинхронному режимі.

Однієї з особливостей конфігурації реплікації є створення додаткового користувача, з правами на реплікацію.

Робота програми при виході з ладу master-сервера (див. Додаток 4)

## **Аналіз функціонування засобів резервування/відновлення даних**

Для здійснення резервування використовується команда pg\_dump, яка додана до /etc/crontab, через що команда автоматично виконується в системі раз на день. Час виконання команди ~2 секунди, на виході отримуємо файл розміром ~110 МБ.

## **Аналіз результатів підвищення швидкодії**

Через відносно великий об’єм даних в таблиці Products (~240 000 рядків) пошук продуктів з певної категорії виконується відносно довго, через те, що база даних змушена послідовно сканувати усі записи, щоб знайти ті, що відповідають заданій ціновій категорії. Для пришвидшення запитів було побудована індекс “б-дерево” для колонки expected\_cost, що помітно пришвидшує швидкість виконання запитів, що продемонстровано на графіку (див. Додаток 5)

## **Опис результатів аналізу предметної галузі**

Побудовні алгоритми для збереження цілісності бази даних (див. додаток 2) пройшли перевірку на рандомізованих даних, тому можуть бути використані на реальних даних. Але їх наявність помітно сказується на швидкості додавання рядків до таблиці.

Текстовий опис алгортмів:

1. Отримати інформацію про розмір партії, що прибуває на склад.

Отримати інформацію про заповнене місце на складі.

Перевірити, чи не перевищить максимальне можливе місце на складі нова партія товару.

Оновити інформацію про кількість продукту, що прийшов на склад у таблиці “Volume\_of\_product”

1. Перевірити чи було достатньо товару на складі, щоб продати задану кількість товару. Якщо ні - генерувати виключну ситуацію.

## **Висновки**

На даній курсовій роботі було розроблено додаток для аналізу цін на споживчі товари.

Для сбору інформації про основні категорії, підкатегорії, продукти та їх вартість на сайті rozetka.ua було розроблено три павуки. Окрім цього, до функціоналу основної програми було додано методи для роботи з .json файлами, за допомогою яких зібрана павуками інформація була занесена до таблиць бази даних.

Інформація, яку неможливо знайти в відкритому доступі (дані про партії товарів, об’єм продажів товарів) імітується за допомогою засобів рандомізації мови sql, частково на основі даних цін що зібрано скрепером (ринкова ціна товарів генерується як +-10% від ціни, яку зібрано з розетки). Інформація, яка додається в базу даних проходить фільтрацію за допомогою тригерів before insert.

В якості засобу статистичного аналізу даних використано бібліотеку pandas за допомогою якої будується гістограма цін на певну категорію товарів (див. додаток 6)

Налагоджено реплікацію між master-сервером та slave-сервером, що знаходиться на віртуальній машині. В разі неполадок на основному сервері, запроси на зчитування інформації перенаправляються на slave-сервер.

За допомогою механізму регулярного виконання завдань, який надає операційна система Ubuntu, було створено рутину, що кожен день в 21:30 робить бекап інформації за допомогою команди pg\_dump.

Для підвищення швидкодії звернень до таблиці “Products” було додано індекс “б-дерево” на колонку “expected\_cost”. Швидкодія покращилась в ~3 рази.

### 

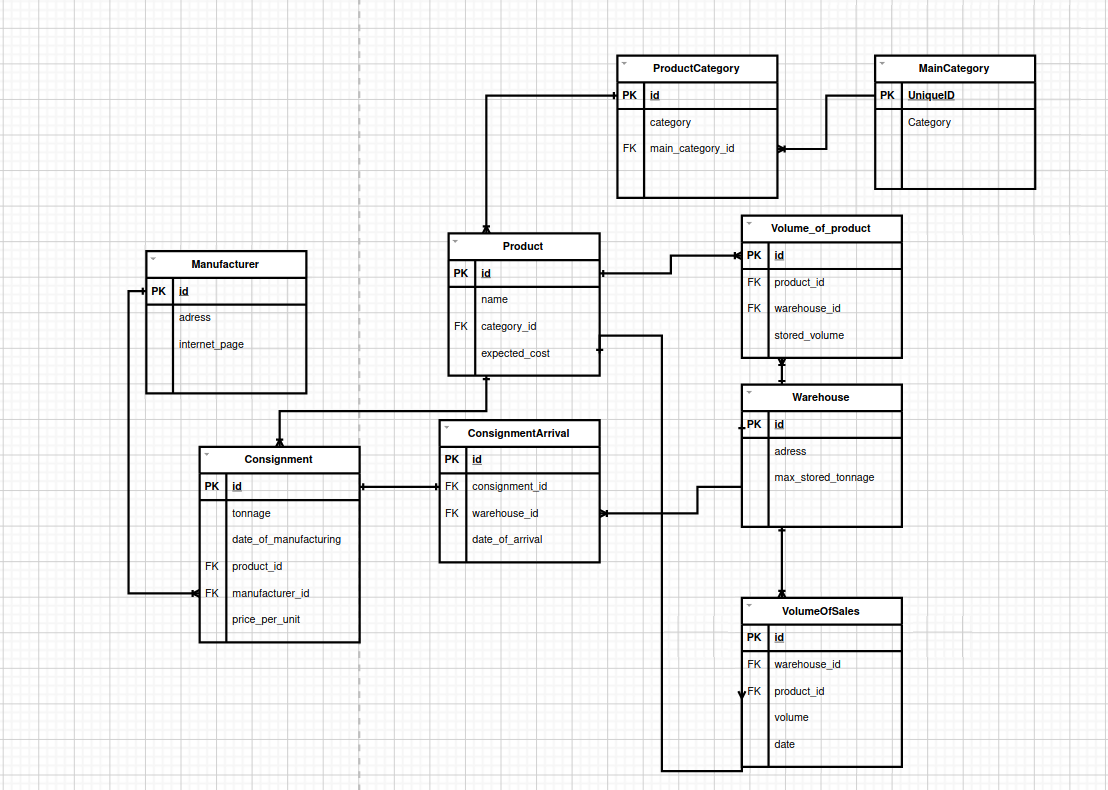
### 

### 

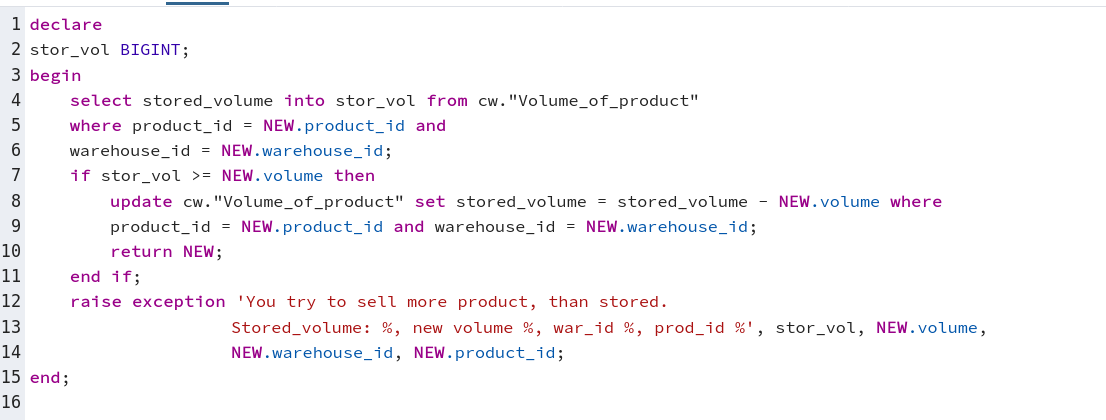
### **Література**

1. Scrapy documentation URL: https://docs.scrapy.org/en/latest/topics/architecture.html (дата обращения: 12.12.2020).

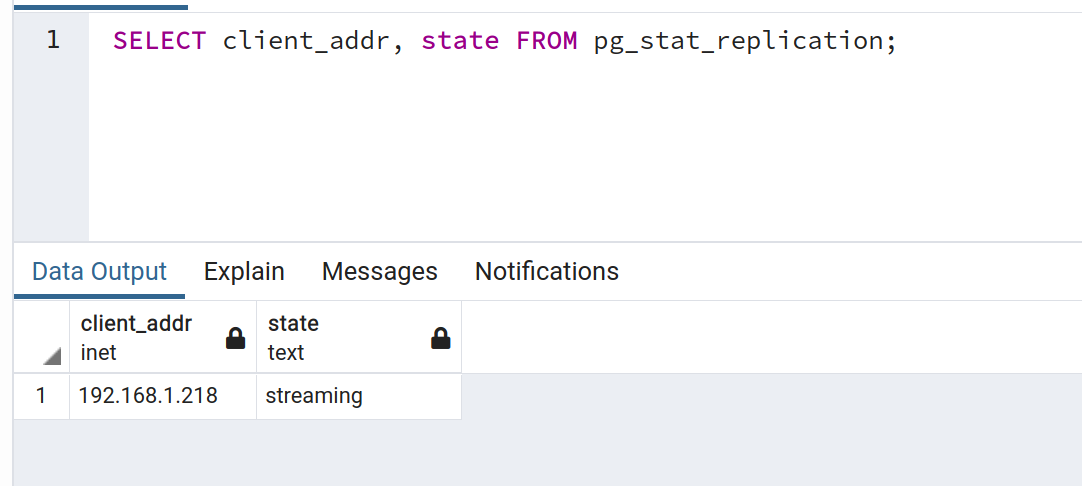
### **Додатки**

1. Структура бази даних 
2. Тексти трігерів для перевірки прибуття Consignment та продажу товару вдповідно

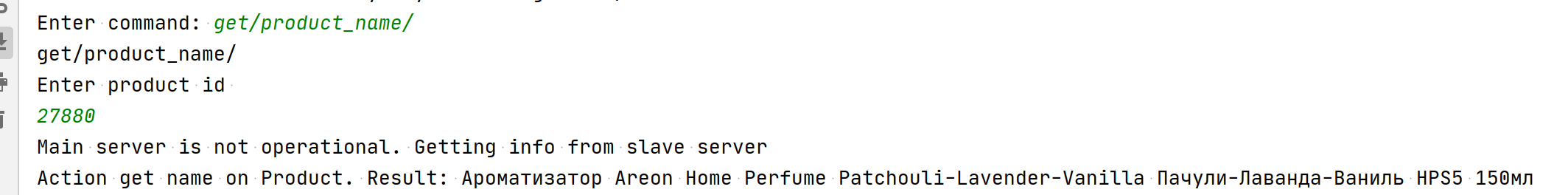




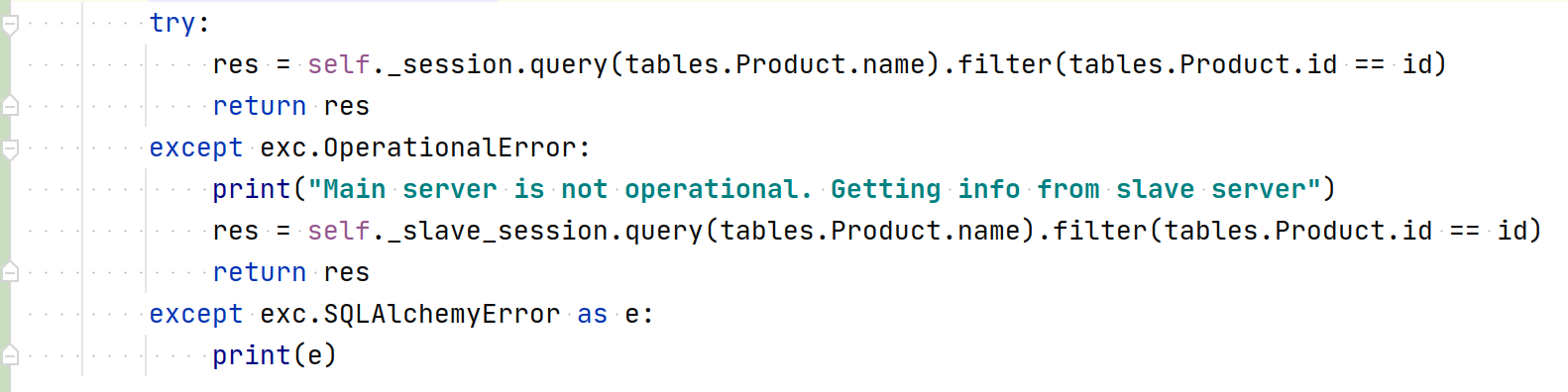
1. Підтвердження роботи реплікації



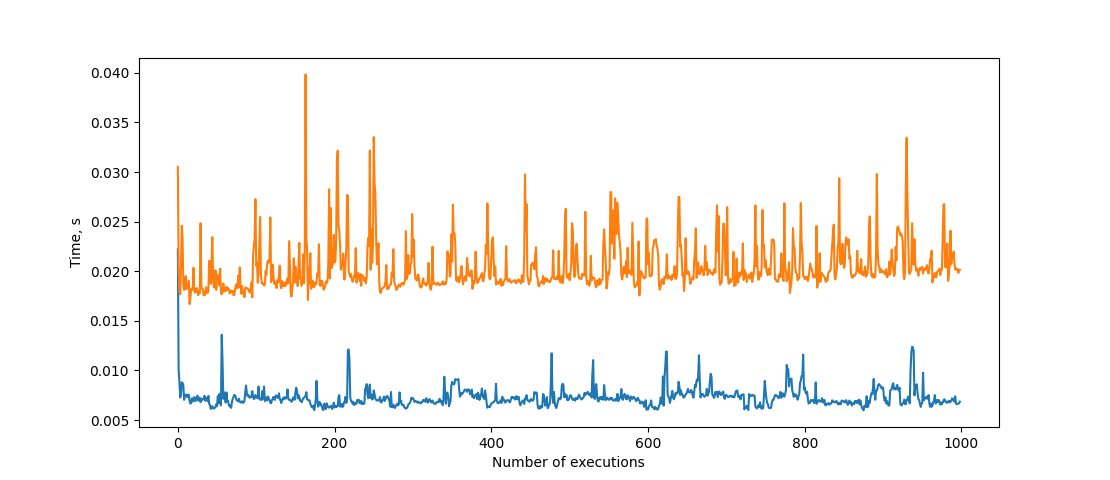
1. Робота при виході з ладу одного з вузлів



Реалізація



Аналогічний патерн try … except наявний і в інших функціях

1. Підвищення швидкодії (синій - графік з індексом б-дерева, помаранчеви - швидкодія лінійного сканування)
2. Результат статистичного аналізу даних на прикладі категорії товарів “Ноутбуки”

