Лабораторні роботи

Зміст

Загальні вимоги

Всі лабораторні роботи мають бути виконані, як еволюційний розвиток однієї системи, мати ту саму кодову базу та перевикористовувати максимальну кількість коду з попередніх робіт (студенти мають уникати дублювання коду).

Лабораторні мають бути оформлені у вигляді єдиного Github-репозиторію. На момент здачі, всі лабораторні роботи мають бути в одній основній гілці. Кожна лабораторна робота має бути відмічена відповідним тегом, щоб мати можливість швидко переключатись на відповідну версію, зберігаючи працездатність системи.

Студент відповідає за контроль доступу до свого коду та надання відповідних прав викладачу. У разі виявлення плагіату або схожого коду у різних студентів, код в усіх таких студентів вважається списаним та роботи будуть оцінені відповідно до РСО.

Лабораторна робота 1

Мета

Отримати навички роботи з Docker. Отримати навички роботи з РСКБД із Python коду та побудови якісних data ingestion програм на основі Python.

Цілі

1. Навчитись описувати Dockerfile. Навчитись описувати інфраструктуру в форматі docker-compose.
2. Навчитись завантажувати дані в РСКБД з використанням інструментів доступних в популярних мовах програмування (опанувати DML та викликати з бібліотек-драйверів)
3. Навчитись робити запити до даних у РСКБД з використанням інструментів доступних в популярних мовах програмування(опанувати SQL та викликати з бібліотек-драйверів)
4. Отримати навички роботи з транзакціями з використанням інструментів доступних в

популярних мовах програмування

Завдання

Написати програму, що завантажує результати ЗНО з https://zno.testportal.com.ua/opendata за декілька років у таблицю (1 таблицю) в реляційній базі даних. Структуру таблиці (колонки та їх типи) студенти мають визначити на основі набору даних. Програма має поновлювати свою роботу у разі помилки (наприклад, помилки в роботі програми, розриву мережевого з‘єднання або помилки в роботі СУБД). Програма не має породжувати дублікати записів. Студенти мають продумати та продемонструвати сценарій "падіння" бази, та те як програма поновлює свою роботу (важливо мінімізувати час завантаження навіть у випадку постійних падінь). Виконати запити, що повертають порівняльну статистику за кілька років (див. Варіанти завдань). Результат запиту має бути записаний у CSV-файл (засобами обраного стеку технологій). Але студент має бути готовим виконати запити з клієнта до БД.

Додаткові вимоги:

- Реалізація має запускатись на Лінукс, МасОС та Віндовс.
- Конфігурація з’єднання з БД має виконуватись без внесення змін у код програми.
- Схема БД має створюватись із коду програми (Python-скрипта) і описаною на мові SQL. У користувача не має бути потреби виконувати будь-які ручні дії.
- Система має розгортатись через docker-compose. Для сервера СУБД можна використовувати готові образи з https://hub.docker.com/search?category=database&source=verified&type=image

Рекомендований стек технологій:

- Мова імплементації — Python (модуль psycopg2)
- РСКБД — PostgreSQL
- Клієнт БД — pgAdmin

Стек може бути змінений за погодженням з викладачем.

Репозиторій має містити:

- Код програми
- Dockerfile для процесу що заповнює базу та робить запит, docker-compose.yaml для всієї інфраструктури.
- Інструкцію по запуску
- Файл з результатами виконання запиту та легенду до нього.
- Файл з вимірами часу роботи завантаження даних у БД

Лабораторна робота 2

Мета

Отримати базові навички міграції схеми РБД та модифікації даних.

Цілі

1. Навчитись планувати міграцію БД
2. Навчитись переміщати дані між таблицями БД без втрати цілісності
3. Освіжити навички створення ERD та проектуванням БД

Завдання

Змінити структуру створеної у лаб1 бази, так щоб вона відповідала щонайменше 3 нормальній формі. Імена сутностей мають відображати об’єкти реального світу. **Міграції мають дозволяти, як створити базу з нуля, так і мігрувати існуючу базу (з лаб1). При міграції бази дані не мають бути втрачені. Модифікувати програми з лаб1 так, щоб вони могли працювати з новою схемою** (писати дані та виконувати запити)

Додаткові вимоги:

- Реалізація має запускатись на Лінукс, МасОС та Віндовс.
- Конфігурація з’єднання з БД має виконуватись без внесення змін у вихідних кодів програми.

Рекомендований стек технологій:

- Мова опису міграцій — SQL
- РСКБД — PostgreSQL
- Інструмент міграції — flyway (https://flywaydb.org/)
- Клієнт БД — pgAdmin

Стек може бути змінений за погодженням з викладачем. Для запуску міграцій рекомендується використовувати Docker образ flyway та підключати його в той же docker-compose файл де і БД та скрипти.

Репозиторій має містити:

- Код програми та скрипти міграції БД
- Інструкцію по запуску
- Файл з результатами виконання запиту та легенду до нього.
- Логічна діаграма "сутність-зв’язок"
- Фізична діаграма "сутність-зв’язок"

Лабораторна робота 3

Мета

Отримати навички побудови багатошарової архітектури, використання ОРМ та кешування даних.

Цілі

- Навчитись використовувати ОРМ для роботи з даними
  - Організації CRUD-операцій
  - Роботу з сутностями, що містять вкладені колекції інших сутностей
- Отримати навички роботи з веб-фреймворками
  - Організації бізнес логіки на сервері
  - Побудови веб-інтерфейсу для відображення списків сутностей
  - Побудови веб-інтерфейсу для CRUD-операцій
- Отримати навички розбиття системи на шари
- Навчитись коректно працювати з кешованими даними

Завдання

Розробити простий веб-інтерфейс для CRUD-операцій усіх основних сутностей ідентифікованих в попередніх лабораторних роботах. Всі операції роботи з даними мають бути реалізовані з використанням ОРМ-фреймворку, використання SQL має бути мінімізоване та обґрунтоване (використання SQL може бути причиною незарахування роботи).

Має бути реалізовано веб-інтерфейс для відображення результатів запитів, відповідно до варіанту роботи. Інтерфейс має надавати можливість зміни року, предмету та регіону. Дані запитів мають кешуватись.

Робота з кешем має бути реалізована через низькорівневі бібліотеки. У разі використання інтеграцій фреймворків непогоджених з викладачем, робота може бути незарахована.

Рекомендований стек технологій:

- РСКБД — PostgreSQL
- Мова реалізації — Python
- Веб-фреймворк — Flask (with Jinja2)
- ОРМ — SQLAlchemy
- UI-фреймворк — Bootstrap, vanilla js (лише вбудовані можливості джаваскріпт)
- Кеш - Redis (Python модулі redis-py чи redis-om-python)

Стек може бути змінений за погодженням з викладачем. Рекомендується студентам познайомитись з одним з наступних UI-фреймворків (на оцінку не впливає):

- https://reactjs.org/
- https://angular.io/
- https://vuejs.org/

Репозиторій має містити:

- Код системи
- Інструкцію по розгортанню
- Логічна або фізична діаграма "сутність-зв’язок"

Лабораторна робота 4

Мета

Отримати базові навички роботи з документно-орієнтованими базами даних. Закріпити розуміння принципів та переваг багатошарової архітектури.

Цілі

- Навчитись працювати з декількома джерелами даних в рамках однієї програми
- Навчитись оперувати з даними у MongoDB
- Навчитись користуватись mongodb aggregation framework

Завдання

Модифікувати систему, розроблену в попередніх лабораторних роботах, таким чином, щоб вона могла працювати як з реляційною (PostgreSQL) так і з документно-орієнтованою СКБД (MongoDB)

Додаткові вимоги:

- Реалізація має запускатись на Лінукс, МасОС та Віндовс.
- Конфігурація з’єднання з БД має виконуватись без внесення змін у код програми.

Рекомендований стек технологій:

- Мова імплементації — Python (модуль pymongo)
- СКБД — MongoDB

Стек може бути змінений за погодженням з викладачем.

Репозиторій має містити:

- Код програми
- Dockerfile для процесу що заповнює базу та робить запит, docker-compose.yaml для всієї інфраструктури.
- Інструкцію по запуску
- Файл з результатами виконання запиту та легенду до нього.
- Файл з вимірами часу роботи завантаження даних у БД

Варіанти завдань

1. Порівняти найкращий бал з Української мови та літератури у кожному регіоні у 2021 та 2019 роках серед тих кому було зараховано тест
2. Порівняти середній бал з Української мови та літератури у кожному регіоні у 2020 та 2019 роках серед тих кому було зараховано тест
3. Порівняти найгірший бал з Української мови та літератури у кожному регіоні у 2020 та 2021 роках серед тих кому було зараховано тест
4. Порівняти найкращий бал з Історії України у кожному регіоні у 2021 та 2019 роках серед тих кому було зараховано тест
5. Порівняти середній бал з Історії України у кожному регіоні у 2020 та 2019 роках серед тих кому було зараховано тест
6. Порівняти найгірший бал з Історії України у кожному регіоні у 2020 та 2021 роках серед тих кому було зараховано тест
7. Порівняти найкращий бал з Математики у кожному регіоні у 2021 та 2019 роках серед тих кому було зараховано тест
8. Порівняти середній бал з Математики у кожному регіоні у 2020 та 2019 роках серед тих кому було зараховано тест
9. Порівняти найгірший бал з Математики у кожному регіоні у 2020 та 2021 роках серед тих кому було зараховано тест
10. Порівняти найкращий бал з Фізики у кожному регіоні у 2021 та 2019 роках серед тих кому було зараховано тест

11. Порівняти середній бал з Фізики у кожному регіоні у 2020 та 2019 роках серед тих кому було зараховано тест
12. Порівняти найгірший бал з Фізики у кожному регіоні у 2020 та 2021 роках серед тих кому було зараховано тест
13. Порівняти найкращий бал з Англійської мови у кожному регіоні у 2021 та 2019 роках серед тих кому було зараховано тест
14. Порівняти середній бал з Англійської мови у кожному регіоні у 2020 та 2019 роках серед тих кому було зараховано тест
15. Порівняти найгірший бал з Англійської мови у кожному регіоні у 2020 та 2021 роках серед тих кому було зараховано тест