Кросплатформне програмування Python

Змістовий модуль 2.

Прикладне програмування мовою Python: від обробки даних до веб-розробк

Лабораторна робота № 9

Корисні посилання

1. Docker: Accelerated Container Application Development *https://www.docker.com/*
2. Docker Hub Container Image Library https://hub.docker.com/
3. *Найкращі бібліотеки Python для створення баз даних SQL* *https://eternalhost.net/blog/razrabotka/python-mysql-postgresql-sqlite*
4. Welcome to Python.org https://www.python.org/
5. PyCharm: IDE для професійної розробки на Python https://www.jetbrains.com/ru-ru/pycharm/

Робота повинна складатися з текстового звіту, що завантажений на eLearn і проекту, що завантажений на GitHub.

Структура звіту:

- титульний аркуш (приклад наведено нижче),
- текст завдання,
- скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
- текст Python коду програм і SQL запитів,
- скріншоти виконання програм в власному віртуальному середовищі
- скріншоти всіх таблиць і запитів БД в графічному клієнті
- посилання на проект на GitHub

Завдання:

- На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
- В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями і заповнити їх даними (*вимоги, опис БД та мінімальний об'єм даних, який необхідно ввести в таблиці, вказані в кожному варіанті*).
- На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
- На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
- Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
- В текстовому редакторі створити звіт наступної структури і завантажити на Elearn
  - титульний аркуш (приклад наведено вище),
  - текст завдання,
  - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
  - текст Python коду програм і SQL запитів,
  - скріншоти виконання програм в власному віртуальному середовищі
  - скріншоти всіх таблиць і запитів БД в графічному клієнті
  - посилання на проект на GitHub

Зразок титульного аркушу:

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ
І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ЛАБОРАТОРНА РОБОТА №9

З дисципліні "Кросплатформне програмування Python"

Виконав студент 4-го курсу
Микитин Юрій Романович
Група ІПЗ-22009б,
Варіант № 5

Перевірив:
к.ф.-м.н. доцент кафедри КН
Кириченко Віктор Вікторович

_____

Київ, 2025

Варіанти завдань

Варіант 0

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (*предметна область та дані наведені нижче*).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні та зовнішні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (3 склади, 7 клієнтів, 17 товарів, 22 покупки)

Предметна область: Магазин одягу (*складається з 4 класів*)
Сутності та дані:
Склади[Номер складу, адреса, завідувач складом, телефон (маска вводу)],
Товари[Код товару, тип (*жіночий, чоловічий, дитячий*), назва одягу, виробник, номер складу, на якому знаходиться, кількість на складі, ціна],
Клієнти[Код клієнта, назва клієнта, адреса клієнта, телефон клієнта (маска вводу), контактна особа],
Продаж[Код продажу, дата продажу, код клієнта, код товару, кількість купленого одягу, знижка].

8. **Створіть наступні запити:**
   Відобразити інформацію по покупкам, яка містить дату покупки, назву товару, назву клієнта, кількість купленого товару, ціну товару. Відсортувати назви клієнтів за алфавітом;
   Відобразити увесь одяг за вибраним типом (*запит з параметром*);
   Порахувати кількість покупок, яку зробив кожний клієнт (*підсумковий запит*);
   Порахувати вартість кожної покупки без урахування та з урахуванням знижки (*запит з обчислювальним полем*);
   Порахувати загальну суму грошей, яку витратив кожен клієнт на купівлю одягу (*підсумковий запит*)
   Відобразити кількість кожного виду одягу на кожному складі (*перехресний запит*).

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 1

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (*предметна область та дані наведені нижче*).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (11 студентів, 3 предмета)

Предметна область: Університет (*складається з 3 класів*)
Сутності та дані:
Студенти[Код студента, Прізвище студента, Ім'я студента, По батькові студента, Адреса студента, телефон студента(маска вводу), курс(1-4, обмеження, повідомлення про помилку), факультет(*аграрного менеджменту, економіки, інформаційних технологій*), група, чи є старостою],
Предмети[Код предмету, назва, кількість годин за семестр, кількість семестрів, протягом яких вивчається предмет],
Складання іспитів[Код складання, дата складання, код студента, код предмету, отримана оцінка(2-5, обмеження, повідомлення про помилку).

8. **Створіть наступні запити:**
   Відобразить всіх студентів, які є старостами, відсортувати прізвища за алфавітом;
   Порахувати середній бал для кожного студента (*підсумковий запит*);
   Для кожного предмета порахувати загальну кількість годин, протягом яких він вивчається (*запит з обчислювальним полем*);
   Відобразити успішність студентів по обраному предмету (*запит з параметром*);
   Порахувати кількість студентів на кожному факультеті (*підсумковий запит*)
   Відобразити оцінки кожного студента по кожному предмету (*перехресний запит*).

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 2

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (*предметна область та дані наведені нижче*).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (14 книг, 9 читачів, 11 видач)

Предметна область: Бібліотека інституту (*складається з 3 класів*)
Сутності та дані:
Книги[інвентарний номер книги, автор книги, назва, розділ (*технічна, художня, економічна*), рік видання, кількість сторінок, ціна, вид (*посібник, книга, періодичне видання*), кількість примірників, максимальний термін находження у читача (*кількість днів*)],
Читачі[номер читацького квитка, прізвище читача, ім'я читача, телефон читача(маска вводу), адреса, курс(1-4, обмеження, повідомлення про помилку), група],
Видача книжок[Код видачі, дата видачі, номер квітка читача, номер книги].

8. **Створіть наступні запити:**
   Відобразити всі книги, які були видані після 2001 року. Відсортувати назви за алфавітом;
   Порахувати кількість книг кожного виду (*підсумковий запит*);
   Відобразити всіх читачів, які брали посібники в бібліотеці. Відсортувати прізвища за алфавітом;
   Відобразити всі книги за указаним розділом (*запит з параметром*);
   Для кожної книги, яка була видана читачу, порахувати кінцевий термін її повернення в бібліотеку (*запит з обчислювальним полем*);
   Порахувати кількість *посібників, книг та періодичних видань* в кожному розділі (*перехресний запит*)

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 3

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (*предметна область та дані наведені нижче*)
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (17 працівників, 8 проектів, відділи: *програмування, дизайну, інформаційних технологій*, посади: *інженер, редактор, програміст*)

Предметна область: Відділ кадрів (*складається з 5 класів*).
Сутності та дані:
Співробітники[Код співробітника, прізвище співробітника, ім'я, по батькові, адреса, телефон(маска вводу), освіта (*спеціальна, середня, вища*), код відділу, код посади],
Відділи[Код відділу, назва відділу, телефон (маска вводу), № кімнати (обмеження 701-710, повідомлення про помилку)],
Посади[Код посади, Посада, оклад, премія(відсоток від окладу)],
Проекти[Номер проекту, назва проекту, термін виконання, розмір фінансування],
Виконання проектів[Код виконання, номер проекту, код відділу в якому виконується проект, дата початку].

8. **Створіть наступні запити:**
   Відобразити всіх робітників, які мають оклад більший за 2000 грн. Відсортувати прізвища за алфавітом;
   Порахувати середню зарплатню в кожному відділі (*підсумковий запит*);
   Відобразити всі проекти, які виконуються в обраному відділі (*запит з параметром*);
   Порахувати кількість працівників у кожному відділу (*підсумковий запит*);
   Порахувати розмір премії для кожного співробітника (*запит з обчислювальним полем*);
   Порахувати кількість робітників які мають *спеціальну*, *середню, вищу* освіту у кожному відділу (*перехресний запит*).

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 4

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (*предметна область та дані наведені нижче*).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (4 постачальника, 22 поставки, матеріали: *деревина, лак, сталеві деталі*)

Предметна область: Відділ поставок (*складається з 3 класів*).
Сутності та дані:
Постачальники[Код постачальника, назва компанії постачальника, контактна особа, телефон(маска вводу), розрахунковий рахунок],
Матеріали, що поставляються[Код матеріалу, назва матеріалу, ціна],
Поставки[номер поставки, дата поставки, код постачальника, код матеріалу, кількість днів, протягом яких здійснюється поставка(1-7; обмеження, повідомлення про помилку), кількість матеріалів, які поставляються].

8. **Створіть наступні запити:**
   Відобразити всі поставки, які здійснюються за 3 або менше днів. Відсортувати назви постачальників за алфавітом;
   Порахувати суму, яку треба сплатити за кожну поставку (*запит з обчислювальним полем*);
   Відобразити всі поставки обраного матеріалу (*запит з параметром*);
   Порахувати кількість кожного матеріалу, що поставляється кожним постачальником (*перехресний запит*);
   Порахувати загальну кількість кожного матеріалу (*підсумковий запит*);
   Порахувати кількість поставок від кожного постачальника (*підсумковий запит*).

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 5

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (*предметна область та дані наведені нижче*).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (6 клієнтів, 4 марки автомобілів, 15 ремонтів)

Предметна область: Авто майстерня Ford (*складається із 4 класів*).
Сутності та дані:
Клієнти[Код клієнта, назва компанії клієнта, розрахунковий рахунок, телефон(маска вводу), контактна особа, адреса],
Автомобілі[Код автомобіля, марка автомобіля(*fiesta, focus, fusion, mondeo*), вартість нової машини, код клієнта]
Ремонт[Код ремонту, дата початку ремонту, код автомобіля,тип ремонту(*гарантійний, плановий, капітальний*), вартість однієї години ремонту, знижка(0%-10%; обмеження, повідомлення про помилку), кількість годин, необхідних для ремонту].

8. **Створіть наступні запити:**
   Відобразити інформацію про всі гарантійні ремонти. Відсортувати назви клієнтів за алфавітом;
   Порахувати вартість ремонту, та вартість з урахуванням знижки, для кожного автомобіля (*запит з обчислювальним полем*);
   Відобразити інформацію по ремонту для всіх авто заданої марки (*запит з параметром*);
   Порахувати загальну суму, яку сплатив кожен клієнт (*підсумковий запит*);
   Порахувати кількість *кожного типу ремонтів* для кожного клієнта (*перехресний запит*);
   Порахувати кількість ремонтів для кожної марки автомобіля.

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 6

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (*предметна область та дані наведені нижче*).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (4 клієнта, 10 товару, 19 продаж)

Предметна область: Магазин (*складається із 3 класів*).
Сутності та дані:
Клієнти[Код клієнта, назва фірми клієнта, юридична або фізична особа(поле зі списком), адреса, телефон(маска вводу), контактна особа, розрахунковий рахунок],
Товари[Код товару, назва товару, ціна, кількість товару в магазині],
Продаж товарів [Код продажу, дата продажу, код клієнта, код товару, кількість проданого товару, знижка(3% – 20%; обмеження, повідомлення про помилку), форма оплати(*готівковий, безготівковий*), необхідність доставки, вартість доставки].

8. **Створіть наступні запити:**
   Відобразити всі продажі, які були оплачені готівкою. Відсортувати їх по назві клієнта за алфавітом;
   Відобразити всі продажі по яких потрібна була доставка;
   Порахувати суму та суму з урахуванням скидки, яку треба сплатити кожному клієнту (*запит з обчислювальним полем*);
   Відобразити всі покупки вказаного клієнта (*запит з параметром*);
   Порахувати кількість покупок, які скоїв кожен клієнт (*підсумковий запит*);
   Порахувати суму , яку сплатив кожен клієнт за готівковим та безготівковим розрахунком (*перехресний запит*);

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 7

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (*предметна область та дані наведені нижче*).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (3 кінотеатри, 11 фільмів, 15 показів)

Предметна область: Кінотеатри (*складається із 3 класів*).
Сутності та дані:
Фільми[Код фільму, назва фільму, жанр(*мелодрама, комедія, бойовик*), тривалість, рейтинг]
Кінотеатри[Код кінотеатру, назва кінотеатру, ціни на квітки, кількість місць, адреса, телефон(маска вводу)],
Транслювання фільмів[Код транслювання, Код фільму, Код кінотеатру, Дата початку показів, Термін показу(*кількість днів*)].

8. **Створіть наступні запити:**
   Відобразити всі комедії. Відсортувати фільми по рейтингу;
   Порахувати останню дату показу фільму для кожного транслювання (Вивести назву фільму, назву кінотеатру, в якому фільм транслюється, дату початку показу, термін показу та кінцеву дату показу фільму) (*запит з обчислювальним полем*);
   Порахувати суму максимального прибутку для кожного кінотеатру від одного показу (*запит з обчислювальним полем*);
   Відобразити всі фільми заданого жанру (*запит з параметром*);
   Порахувати кількість фільмів кожного жанру (*Підсумковий запит*)
   Порахувати кількість мелодрам, комедій, бойовиків, які транслюються в кожному кінотеатрі (*перехресний запит*);

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 8

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (*предметна область та дані наведені нижче*).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (20 помилок, 4 програміста)

Предметна область: Компанія з розробки і супроводу програмного забезпечення (*складається з 3 класів*).
Сутності та дані:
Помилки[Код помилки, опис помилки, дата надходження інформації про помилку, рівень помилки (*критична, важлива, незначна*), категорія функціональності(*інтерфейс, дані, розрахунковий алгоритм, інше, невідома категорія*), джерело(*користувач, тестувальник*)],
Програмісти, відповідальні за виправлення помилки [Код програміста, Прізвище, ім'я програміста, телефон(маска вводу)],
Виправлення помилок[Код виправлення, код помилки, дата початку виправлення, термін виправлення(*1день, 2 дня, 3дня*), код програміста, вартість роботи 1 дня програміста].

8. **Створіть наступні запити:**
   Відобразити всі критичні помилки. Відсортувати по коду помилки;
   Порахувати кількість помилок кожного рівня (*підсумковий запит*);
   Порахувати вартість роботи програміста при виправленні кожної помилки (*запит з обчислювальним полем*);
   Відобразити всі помилки, які надійшли із заданого джерела (*запит з параметром*);
   Порахувати кількість помилок, які надійшли від користувачів, та тестувальників (*підсумковий запит*)
   Порахувати кількість *критичних, важливих, незначних* помилок, виправлених кожним програмістом (*перехресний запит*);

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 9

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (предметна область та дані наведені нижче).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (5 клієнтів, 7 номерів телефонів, 20 розмов протягом одного місяця, тип дзвінка: *внутрішній, міжміський, мобільний*)

Предметна область: Телефонна станція (*складається із 4 класів*)
Сутності та дані:
Клієнти[Код клієнта, Тип клієнта(*відомство, фізична особа*), Адреса, Прізвище, Ім'я, По батькові],
Телефони[номер телефону абонента, Код клієнта]
Розмови[Код розмови, дата розмови, номер телефону, кількість хвилин розмови, код тарифу]
Тарифи[Код тарифу, тип дзвінка, вартість 1 хвилини розмови].

8. **Створіть наступні запити:**
   Відобразити всіх клієнтів, які є фізичними особами. Відсортувати по прізвищу клієнта;
   Порахувати кількість клієнтів, які є фізичними особами, та кількість клієнтів, які являються відомством (*підсумковий запит*);
   Порахувати вартість кожної розмови (*запит з обчислювальним полем*);
   Відобразити список всіх розмов з обраним типом дзвінка (*запит з параметром*);
   Порахувати загальну вартість всіх розмов для кожного клієнта (*підсумковий запит*);
   Порахувати кількість хвилин кожного типу дзвінків для кожного клієнта (*перехресний запит*).

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 10

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (*предметна область та дані наведені нижче*).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (4 лікаря, 9 пацієнтів, 17 звернень до лікарні)

Предметна область: Приватна поліклініка (*складається із 3 класів*)
Сутності та дані:
Пацієнти[Номер карточки пацієнта, Прізвище, Ім'я, По батькові пацієнта, адреса, телефон(маска вводу), рік народження, категорія(*дитяча, доросла*)],
Прибування в стаціонарі[Код прибування, номер карточки пацієнта, дата надходження в стаціонар, кількість днів проведених стаціонарі, вартість одного дня лікування, пільга(відсоток), код лікаря, який лікував],
Лікарі[Код лікаря, Прізвище, Ім'я, По батькові лікаря, спеціалізація(*лор, терапевт, хірург*), стаж роботи].

8. **Створіть наступні запити:**
   Відобразити всіх пацієнтів, які народилися після 1998 року. Відсортувати по прізвищу пацієнта;
   Порахувати кількість пацієнтів дитячої категорії, та кількість пацієнтів дорослої категорії;
   Порахувати суму лікування, та суму лікування з урахуванням пільги для кожного пацієнта (*запит з обчислювальним полем*);
   Відобразити всі звернення до лікаря заданої спеціалізації (*запит з параметром*);
   Порахувати кількість звернень пацієнтів до кожного лікаря (*підсумковий запит*)
   Порахувати кількість пацієнтів кожної категорії, які лікувалися у лора, терапевта, хірурга (*перехресний запит*);

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 11

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (предметна область та дані наведені нижче).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (5 орендарів, 11 приміщень, 11 договорів оренди)

Предметна область: Облік орендної плати (*складається із 3 класів*)
Сутності та дані:
Орендарі[Код орендаря, Назва фірми орендаря, керівник, телефон(маска вводу)],
Приміщення[Номер приміщення, площа, вартість оренди за 1м^2, поверх, на якому розташоване приміщення, телефон в приміщенні (є/ні), оздоблення(*звичайне, поліпшене, євро*)],
Оренда[номер договору оренди, дата початку дії договору, строк дії договору (кількість днів), мета оренди(*офіс, кіоск, склад*), код орендаря, номер приміщення].

8. **Створіть наступні запити:**
   Відобразити всіх орендарів, які орендують приміщення під склад. Відсортувати по назві фірми;
   Порахувати загальну орендну плату за кожне приміщення (*запит з обчислювальним полем*);
   Порахувати загальну площу приміщень з *звичайним*, *поліпшеним* та *євро* оздобленням (*підсумковий запит*);
   Порахувати кінцеву дату дії кожного договору оренди (*запит з обчислювальним полем*);
   Порахувати кількість приміщень які здаються під офіс, кіоск, склад для кожного типу оздоблення (*перехресний запит*);
   Відобразити всі приміщення за обраним типом оздоблення (*запит з параметром*).

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 12

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (предметна область та дані наведені нижче).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (7 гостей, 10 номерів, 3 поверхи, 10 прибувань в готелі)

Предметна область: Готель (*складається із 3 класів*)
Сутності та дані:
Гості[Реєстраційний номер гостя, прізвище, ім'я, по батькові гостя, місто з якого приїхав гість],
Номери[№ номера, кількість кімнат в номері, поверх, телевізор (є/ні), холодильник(є/ні), кількість місць в номері, категорія(*звичайний, полу люкс, люкс*), вартість за добу проживання
Реєстрація гостей[Код реєстрації, реєстраційний номер гостя, дата прибуття гостя, кількість днів проживання, № номера в якому проживатиме гість].

8. **Створіть наступні запити:**
   Відобразити всі номери в яких є телевізор;
   Порахувати кінцеву дату проживання в готелі для кожного гостя (*запит з обчислювальним полем*);
   Порахувати кількість номерів кожної категорії в готелі (*підсумковий запит*);
   Порахувати повну вартість проживання для кожного гостя (*запит з обчислювальним полем*);
   Порахувати кількість номерів кожної категорії на кожному поверсі (*перехресний запит*);
   Відобразити всіх гостей, які проживають(або проживали) в номерах обраної категорії. Відсортувати по прізвищу (*запит з параметром*).

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 13

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (предметна область та дані наведені нижче).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (3 бригади, 9 робітників, 9 локомотивів, 11 ремонтів)

Предметна область: Ремонтно-експлуатаційне локомотивне депо (*складається із 4 класів*)
Сутності та дані:
Локомотиви[Реєстраційний номер локомотива, приписка локомотива до депо (*Фастів, Козятин, П'ятихатки*), тип локомотива (*вантажний, пасажирський*), рік випуску],
Ремонти[Код ремонту, реєстраційний номер локомотива, тип ремонту(*поточний, технічне обслуговування, позаплановий*), дата початку ремонту, кількість днів, які необхідні для ремонту, вартість ремонту одного дня, номер бригади],
Бригади[Номер бригади, телефон(маска вводу)],
Робітники[Код робітника, прізвище, ім'я, по батькові робітника, номер бригади, чи є бригадиром, дата народження].

8. **Створіть наступні запити:**
   Відобразити всі локомотиви, які мають тип вантажний. Відсортувати за роком випуску;
   Порахувати кінцеву дату ремонту для кожного локомотива (*запит з обчислювальним полем*);
   Порахувати кількість ремонтів, які виконала кожна бригада (*підсумковий запит*);
   Порахувати повну вартість ремонту для кожного локомотива, який було відремонтовано (*запит з обчислювальним полем*);
   Порахувати кількість типів ремонтів, які виконала кожна бригада (*перехресний запит*);
   Відобразити всіх локомотиви, які приписані до обраного депо (*запит з параметром*).

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 14

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (предметна область та дані наведені нижче).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (3 судна, 15 товарів, 5 портів, 5 партій товарів) (Одна партія може складатися із різних товарів)

Предметна область: Судноплавна компанія (*складається із 5 класів*)
Сутності та дані:
Судна[Реєстраційний номер судна, назва судна, ПІБ капітана, тип судна (*танкер, суховантажний*), вантажопідйомність, рік будівлі]
Порти[Номер порту, назва порту, країна]
Товари[Код товару, назва товару, тип товару(*пально-мастильні суміші, побутова техніка, великогабаритний вантаж*), одиниця виміру, ціна за одиницю товару, необхідність митної декларації],
Партії[Номер партії, назва партії, код товару, кількість товару в партії, Номер порту призначення]
Відправка товару[Код відправки, номер партії, дата відправки, кількість діб, необхідних для доставки вантажу, номер судна, на якому відправлено партію товарів].

8. **Створіть наступні запити:**
   Відобразити всі судна, які мають обраний тип (*запит з параметром*);
   Порахувати вартість, кожного товару в кожній партії (*запит з обчислювальним полем*);
   Порахувати вартість кожної партії товарів (*підсумковий запит*);
   Порахувати кількість кожного типу товарів в кожній партії (*перехресний запит*);
   Порахувати дату прибуття в порт призначення для кожної партії товарів (*запит з обчислювальним полем*);
   Відобразити всі товари, які належать до типу «побутова техніка». Відсортувати за назвою товару.

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub

Варіант 15

1. На платформі Docker, за допомогою файлу docker-compose.yml, створити контейнер з СУБД *PostgreSQL* або *MySQL*. Зробити прокидання портів та папок для зберігання БД.
2. В цьому контейнері, використовуючи мову Python, створити базу даних. Створити в ній необхідні таблиці з відповідними полями (предметна область та дані наведені нижче).
3. Визначить типи даних (лічильник, текстовий, числовий тощо) та опис, якщо потрібно.
4. Встановіть необхідні властивості полів (розмір поля, маску вводу, значення за замовченням, обмеження та повідомлення про помилку) створених таблиць.
5. Визначить первинні ключі в створених таблицях.
6. Визначить необхідні зв'язки між таблицями, задайте необхідні параметри забезпечення цілісності даних.
7. Заповнить створені таблиці даними (13 ліків, 5 постачальників, 11 поставок)

Предметна область: Аптека (*складається із 3 класів*)

Сутності та дані:

Ліки[Реєстраційний номер ліки, назва ліки, дата виготовлення, термін зберігання(кількість днів), група(*протизапальне, знеболююче, вітаміни*), ціна, відпускається за рецептом лікаря(так/ні)],

Постачальники[код постачальника, назва постачальника, адреса, телефон(маска вводу), контактна особа, розташування (*Україна, інша країна*)],

Поставки[Код поставки, дата поставки, номер ліки, кількість ліків, які були поставлені, код постачальника].

8. **Створіть наступні запити:**

   Відобразити всі ліки, які відпускаються за рецептом лікаря. Відсортувати за назвою ліків.

   Відобразити всі ліки за обраною групою (*запит з параметром*);

   Порахувати вартість, кожної поставки (*запит з обчислювальним полем*);

   Порахувати загальну суму грошей, яку сплатила аптека кожному постачальнику (*підсумковий запит*);

   Порахувати кількість поставок для кожної групи ліків від вітчизняних та закордонних постачальників (*перехресний запит*);

   Порахувати останню дату придатності для кожної ліки (*запит з обчислювальним полем*).

9. На мові Python написати програму, що підключається до створеної БД, виводить всі таблиці (структура + дані, які в ній зберігаються) та результати виконання запитів в консоль в форматованому вигляді (заголовки стовпців + всі стовпці рівні).
10. На платформі Docker створити контейнер з графічним клієнтом (адмінка) для управління БД. Запустити його і підключитись до створеної БД. Переконатись, що всі таблиці і запити створені вірно.
11. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.
12. В текстовому редакторі створіть звіт наступної структури і завантажити на Elearn
    - титульний аркуш (приклад наведено вище),
    - текст завдання,
    - скріншоти з назвами використаних в проекті docker образів та docker контейнерів,
    - текст Python коду програм і SQL запитів,
    - скріншоти виконання програм в власному віртуальному середовищі
    - скріншоти всіх таблиць і запитів БД в графічному клієнті
    - посилання на проект на GitHub