**Київський національний університет імені**

**Тараса Шевченка**

**Факультету комп’ютерних наук та кібернетики**

Лабараторна робота №1

З дисципліни «**Сучасні технології баз даних**»

Виконав студент 3-го курсу

Групи ТТП-32 Пелих Олександр

**Київ-2024**

**Тема лабараторної роботи**

Розробка реляційної бази даних для управління електронною комерцією

**Опис проекту**

Метою проекту є створення реляційної бази даних для підтримки роботи електронного магазину. Система забезпечує управління товарами, категоріями, клієнтами, замовленнями, постачальниками, купонами та іншими елементами, що характерні для платформи e-commerce. База даних включає мінімум 15 сутностей, кожна з яких реалізує функціонал, необхідний для ефективного управління магазином.

Проект орієнтований на демонстрацію сучасних підходів до моделювання та реалізації баз даних, таких як використання "м’якого видалення" (Soft Delete), збереження історії змін (дата оновлення та користувач, що змінив дані), тригерів, збережених процедур, а також створення користувацьких функцій та віртуальних таблиць (View).

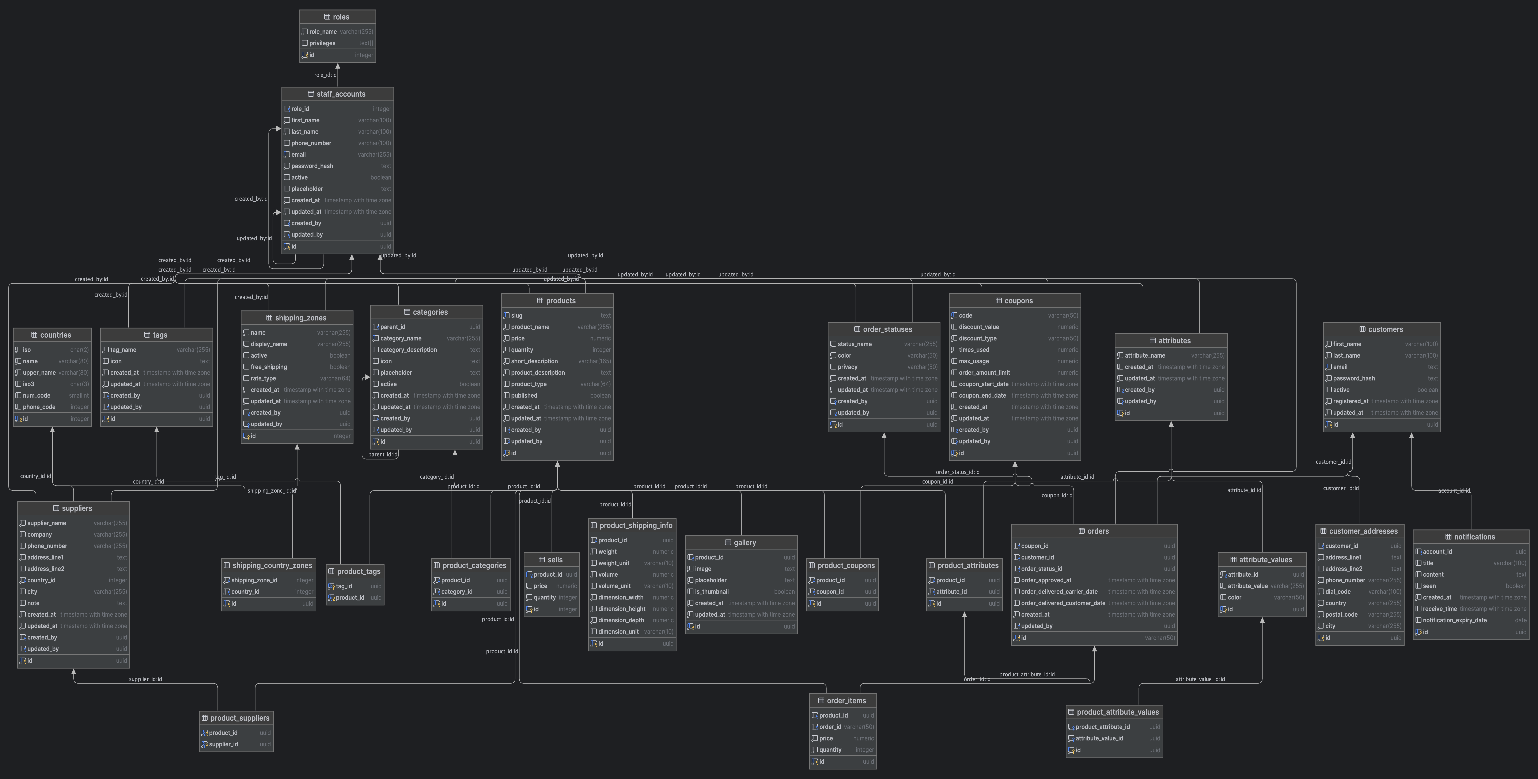
**Реалізація**

Проект було реалізовано в СУБД PostgreSQL. SQL-скрипти структуровані за категоріями:

* **Init.sql**: створення таблиць.
* **Functions.sql**: збережені процедури та функції.
* **Triggers.sql**: тригери.
* **DefaultData.sql**: початкові дані.
* **Permissions.sql**: права доступу.
* **Testing.sql**: сценарії тестування.

Цей проект демонструє глибоке розуміння принципів проєктування реляційних баз даних та їх реалізації у сучасних СУБД.

**Схема реалізованої БД**

****

**Опис сутностей реалізованої БД**

1. **roles**

Зберігає інформацію про ролі користувачів та їхні привілеї.

1. **staff\_accounts**

Інформація про працівників магазину. Поля включають ідентифікатор ролі, ім’я, електронну пошту, пароль, активність.

1. **categories**

Ієрархічна структура категорій товарів. Поля: назва категорії, опис, іконка, активність.

1. **products**

Основна сутність для зберігання інформації про товари. Поля: назва товару, опис, ціна, кількість, статус публікації.

1. **product\_categories**

Зв’язок між товарами та категоріями.

1. **product\_shipping\_info**

Інформація про вагу, об'єм, розміри товару.

1. **gallery**

Зображення товарів, включаючи мініатюри (thumbnail).

1. **attributes та attribute\_values**

Зберігання атрибутів товарів (наприклад, колір, розмір) та їх значень.

1. **product\_attributes та product\_attribute\_values**

Визначають зв’язки між товарами, атрибутами та їх значеннями.

1. **customers**

Дані про клієнтів магазину: ім’я, електронна пошта, пароль.

1. **customer\_addresses**

Адреси клієнтів для доставки.

1. **coupons**

Управління знижками та промокодами.

1. **orders та order\_items**

Замовлення клієнтів і їхні позиції.

1. **suppliers**

Інформація про постачальників товарів.

1. **notifications**

Повідомлення для клієнтів.

**Опис функцій у реалізованій БД**

1. **update\_at\_timestamp()**

* Опис: Оновлює поле updated\_at на поточний час перед оновленням рядка.
* Викликається через тригери для 12 таблиць.

1. **add\_product()**

* Опис: Додає новий продукт до таблиці products.
* Параметри:
  + p\_slug, p\_product\_name, p\_price, p\_quantity, p\_short\_description, p\_product\_description, p\_product\_type, p\_published, staff\_email.
* Додатково перевіряє, чи існує співробітник із зазначеним email. Якщо ні — викликає помилку.

1. **update\_product()**

* Опис: Оновлює інформацію про продукт за його slug.
* Параметри:
  + p\_slug, p\_product\_name, p\_price, p\_quantity, p\_short\_description, p\_product\_description, staff\_email.
* Аналогічно перевіряє існування співробітника перед оновленням.

1. **get\_product\_counts\_by\_category()**

* Опис: Повертає кількість продуктів у кожній категорії (product\_type).
* Результат: Таблиця з полями:
  + category\_name (назва категорії),
  + product\_count (кількість продуктів).

1. **get\_product\_counts\_by\_user()**

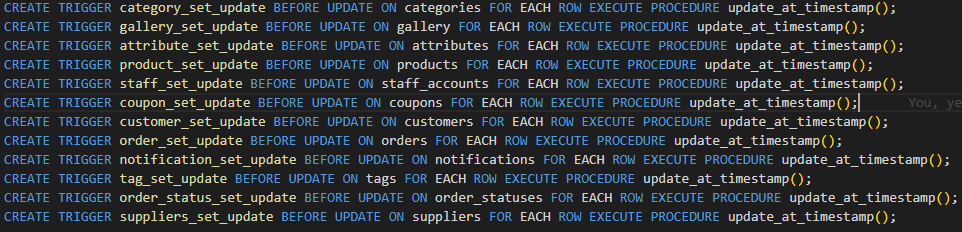
* Опис: Повертає кількість продуктів, створених кожним користувачем (співробітником).
* Результат: Таблиця з полями:
  + user\_email (email співробітника),
  + product\_count (кількість продуктів).

1. **notify\_customers\_about\_coupon()**

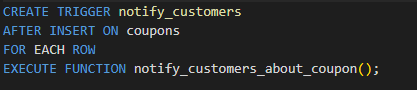
* Опис: Додає сповіщення про новий купон для всіх активних клієнтів (customers).
* Використовується як **тригерна функція**.

**Опис тригерів у реалізованій БД**

* 1. Тригери для автоматичного оновлення поля **updated\_at**

****

* Ці тригери викликають функцію update\_at\_timestamp() перед оновленням даних у відповідних таблицях. Це забезпечує автоматичне оновлення часу останньої модифікації запису.
  1. Тригер для сповіщення клієнтів про новий купон **notify\_customers**



* Цей тригер спрацьовує після додавання нового запису в таблицю coupons. Він викликає функцію notify\_customers\_about\_coupon(), яка створює сповіщення для всіх активних клієнтів (customers) про новий купон.

**Висновок**

У межах лабораторної роботи було реалізовано низку функцій, тригерів та індексів для ефективного управління даними в реляційній базі даних. Виконані завдання демонструють використання механізмів PL/pgSQL для автоматизації бізнес-логіки на рівні бази даних.

**Основні результати:**

1. **Створено функції для роботи з даними:**
   * Функції для додавання та оновлення продуктів із використанням авторизації через електронну пошту працівника.
   * Функції для отримання статистики, такі як кількість продуктів за категоріями та кількість продуктів, створених окремими користувачами.
   * Функція для автоматичного створення сповіщень клієнтам про нові купони.
2. **Реалізовано тригери:**
   * Тригери для автоматичного оновлення часу модифікації (updated\_at) у 12 основних таблицях.
   * Тригер для автоматичного створення записів у таблиці сповіщень після додавання нового купона.
3. **Оптимізовано роботу бази даних за допомогою індексів:**
   * Створено індекси для ключових таблиць і полів, що часто використовуються в запитах (наприклад, індексація поля email у таблиці customers та зв’язків між таблицями через зовнішні ключі).
4. **Використано механізм захисту від помилок:**
   * У функціях передбачено перевірку на існування відповідних записів у пов’язаних таблицях (наприклад, перевірка існування працівника за його email перед додаванням чи оновленням продукту).

Дана лабораторна робота продемонструвала практичне застосування розширених можливостей PostgreSQL, таких як тригери, користувацькі функції та індекси, для вирішення завдань, пов’язаних з автоматизацією процесів у базі даних. Реалізовані механізми покращують ефективність роботи з даними та забезпечують гнучкість і надійність системи.