Міністерство освіти і науки України

Національний університет «Одеська політехніка»

Інститут комп’ютерних систем

Кафедра інформаційних систем

Арабаджи Лев Сергійович

студент групи АІ-226

КУРСОВА РОБОТА

Тема роботи : розробити базу даних для підтримки діяльності ательє мод.

Спеціальність:

122 Комп’ютерні науки

Освітня програма: Комп’ютерні науки

Керівник:

Іванов Олексій Володимирович,

PhD, старший викладач

Одеса – 2023

Національний університет «Одеська політехніка»

Інститут комп'ютерних систем

Кафедра інформаційних систем

ЗАВДАННЯ

НА КУРСОВУ РОБОТУ

студенту \_Арабаджи Лев Сергійович\_\_\_\_\_\_\_\_\_\_\_\_\_група АІ-226\_\_

1.Тема роботи:\_Розробити базу даних для підтримки діяльності ательє мод

2. Термін здачі студентом закінченої роботи 01.12.2023\_

3. Початкові дані до проекту (роботи)\_\_\_\_\_ Каталог моделей: тканина, витрата тканини (за розмірами), вартість тканини, вартість фурнітури, вартість роботи, ціна. Каталог тканин: назва, ширини і ціна за 1 метр, наявність тканини на складі. Каталог фурнітури: назва, кількість, ціна за 1 шт. Журнал реєстрації замовлень: ПІБ клієнта, інформація про модель, інформація про тканину, ПІБ виконавця замовлення, дату прийому, дату примірки, дату виконання, відмітка про оплату

4. Зміст розрахунково-пояснювальної записки (перелік питань, які належить розробити) 2 Проектування бази даних «Діяльності ательє мод».

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень

Завдання видано 21.09.2023 \_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис викладача)

Завдання прийнято до виконання 21.09.2023 \_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис студента)

ЗМІСТ

Завдання на курсову роботу………………………………….…...…2

Анотація………………………………………………………….…...4

Вступ………………………….……………………………………….5

1 Аналіз предметної області та постановка задачі……………...….6

1.1Опис предметної області для підтримки діяльності ательє мод.6

1.2Опис користувачів системи та їх історії (User Story)……...……7

1.3Детальний опис функціоналу бази даних, що пропонується.…9

2 Проектування бази даних ательє мод ……………..…………….12

3 Вибір програмного забезпечення…………………..………..……17

4 Створення бази даних………………………………..……………18

4.1 Створення таблиць……………………………………..………..18

4.2 Створення представлень……………………………...…………23

4.3 Створення тригерів…………………………………...…………27

4.4 Створення збережених процедур (функцій)……………..…….35

5 Маніпулювання даними……………………….…………………..43

5.1 Оператори відновлення………………………...……………… 43

5.2 Оператор вибірки……...……………………………...…………47

6 Створення користувачів і призначення прав доступу…….……58

Висновки……………………………………………………………..60

Перелік використаних джерел………………………...……………61

# АНОТАЦІЯ

# У даній роботі розглядається завдання створення бази даних для підтримки діяльності ательє моди з врахуванням різноманітних аспектів, таких як клієнти, моделі, дизайнери, матеріали та процеси виробництва. Запропоновано інтеграцію відомостей про клієнтів, їх замовлення, моделі та використані матеріали. Розроблено функціонал для зручного ведення обліку замовлень, слідкування за статусами виробництва та управління інформацією про моделі та використані матеріали.

# ABSTRACT

# This paper addresses the task of creating a database to support the operations of a fashion atelier, considering various aspects such as clients, models, designers, materials, and production processes. The integration of information about clients, their orders, models, and utilized materials is proposed. Functionality is developed for convenient order management, tracking production statuses, and overseeing information about models and used materials.

# ВСТУП

У сучасному світі, де мода стала не тільки важливим елементом культури, але й визначальною складовою суспільства, необхідність у систематизації та ефективному управлінні інформацією про ательє мод стає все більш актуальною. Саме тому, розробка бази даних для збереження відомостей про модні ательє, колекції, та графік роботи стає важливим етапом в удосконаленні організації та взаємодії в цій сфері.

Зазначимо, що у сучасному інформаційному віці, де швидкість доступу до даних та їх якість мають вирішальне значення, база даних стає ефективним інструментом для вирішення завдань, пов'язаних із світом моди. Вона дозволяє оперативно та зручно отримувати інформацію про ательє, дизайнерів, моделі та графік подій. Велика кількість вхідних даних, що включають в себе інформацію про нові колекції, події та творчих представників, стає більш організованою та доступною завдяки базі даних.

Створення бази даних для ательє мод відповідає потребам спільноти, яка безпосередньо чи опосередковано пов'язана з цією галуззю. Вона полегшить роботу організаторів показів, дизайнерів, журналістів та прихильників моди. Завдяки можливостям пошуку та видачі інформації, база даних стане ефективним інструментом для організації та планування подій, а також для створення історії модних трендів, яка залишиться в пам'яті модної спільноти.

Таким чином, створення бази даних для ательє мод є доцільним та важливим завданням, яке сприятиме покращенню організації та управління цією захоплюючою галуззю.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

* 1. Опис предметної області для ательє мод

Предметною областю для створення бази даних є ательє мод. В даному випадку ми маємо на меті зберегти та керувати інформацією, пов'язаною із сферою моди та діяльністю ательє. Основні завдання бази даних полягають в збереженні та наданні доступу до різних аспектів роботи ательє мод.

Проблеми в предметній області:

* управління великим обсягом даних про ательє, дизайнерів, колекції, тканини та інші деталі, пов'язані із світом моди;
* необхідність ведення історії створених моделей та збереження актуальних даних для створення статистичних звітів;
* потреба в надійному інструменті для пошуку та видачі інформації щодо різних аспектів діяльності ательє мод.

Потреба в базі даних та інформаційній системі:

* база даних допоможе ефективно зберігати та оновлювати дані про ательє, дизайнерів, використані матеріали, колекції і багато іншого, що важливо для ательє мод;
* інформаційна система дозволить використовувати ці дані для створення графіку роботи, виставок, аналізу популярних моделей та багатьох інших завдань, що полегшать роботу учасників та клієнтів ательє мод.

Інформація, яку повинна зберігати база даних:

* вона дозволить здійснювати контроль за різними аспектами діяльності, включаючи інформацію про моделі, наявність тканин, замовлення, майстрів та клієнтів.
* зберігання даних про моделі та їх характеристики, наявність тканин та їх параметри, а також інформацію про замовлення та статус оплати, сприятиме ефективному веденню обліку, організації виробництва та взаємодії з клієнтами.

Як інформаційна система полегшить роботу користувачів:

* користувачі зможуть легко отримувати доступ до актуальної інформації про нові колекції, події та акції ательє;
* можливість генерувати графік роботи та видачі інформації для публічності спростить організацію та споживання інформації про діяльність ательє мод;
* аналіз популярних моделей та статистика роботи ательє допоможуть дизайнерам удосконалювати свої колекції.

Отже, створення бази даних та інформаційної системи для ательє мод допоможе полегшити роботу учасників та клієнтів цього захоплюючого сегменту модної індустрії, забезпечуючи доступ до важливої інформації та сприяючи розвитку галузі моди.

## Опис користувачів системи та їх історії (User Story)

Майстер ательє**:**

1.Як Майстер ательє, я хочу мати можливість переглядати список своїх призначених завдань та їхні деталі, щоб ефективно планувати свою роботу.

2.Як Майстер ательє, я хочу отримувати повідомлення про нові замовлення та оновлення статусу завдань, щоб бути завжди в курсі подій.

3.Як Майстер ательє, я хочу мати можливість реєструвати інформацію про клієнтів, включаючи їхні замовлення та контактні дані, для зручного ведення обліку та взаємодії з клієнтами.

4.Як Майстер ательє, я хочу мати можливість вести історію замовлень кожного клієнта, щоб забезпечити персоналізований підхід та вчасне виконання замовлень.

5.Як Майстер ательє , я хочу мати можливість відзначати завдання як виконані та вказувати час виконання, щоб дотримуватися графіку та забезпечувати якість виготовлення.

Клієнт ательє:

1.Як Клієнт ательє, я хочу мати можливість переглядати історію своїх замовлень та їх статус, щоб слідкувати за процесом виготовлення.

2.Як Клієнт ательє, я хочу мати можливість вести історію своїх замовлень не лише для відстеження, але й для повторного замовлення подібних виробів.

3.Як Клієнт ательє, я хочу мати можливість залишати відгуки про вироблені моделі та взаємодію з ательє, щоб поділитися своїм досвідом та допомогти в покращенні сервісу.

4.Як Клієнт ательє, я хочу мати можливість переглядати доступні моделі одягу з фотографіями та цінами, щоб обрати бажану модель перед замовленням.

5.Як Клієнт ательє, я хочу мати можливість розміщувати замовлення на виготовлення одягу та вказувати свої побажання, розміри та інші деталі.

Дизайнер мод:

1.Як Дизайнер мод, я хочу мати можливість створювати портфоліо своїх робіт та дизайнів, щоб привертати нових клієнтів та представляти свої творіння.

2.Як Дизайнер мод, я хочу отримувати сповіщення про попит на певні моделі або дизайни, щоб адаптувати свою творчість під потреби клієнтів.

3.Як Дизайнер мод, я хочу мати можливість завантажувати свої дизайнерські проекти та колекції в базу даних, включаючи інформацію про матеріали та зразки, щоб представити їх клієнтам та виготовити на їх замовлення.

4.Як Дизайнер мод, я хочу мати можливість отримувати зворотний зв'язок від клієнтів стосовно моїх дизайнів та змінювати їх відповідно до їхніх побажань та вимог.

5. Як Дизайнер мод, я хочу мати можливість аналізувати попит на різні стилі та дизайни, щоб адаптувати свою творчість під потреби ринку.

Керівник ательє:

1.Як Керівник ательє, я хочу мати можливість призначати завдання майстрам та контролювати їхню роботу для ефективного виконання замовлень.

2.Як Керівник ательє, я хочу мати доступ до звітів про продуктивність майстрів та обсяги виготовленого одягу для оптимізації виробничого процесу.

3.Як Керівник ательє, я хочу мати можливість встановлювати ціни на послуги та товари, а також вносити зміни в асортимент, щоб забезпечити конкурентоспроможність та рентабельність ательє.

4.Як Керівник ательє, я хочу мати можливість генерувати звіти щодо фінансової діяльності ательє за певний період, щоб визначити ефективність бізнесу.

5.Як Керівник ательє, я хочу мати можливість приймати та відстежувати оплату за замовлення, щоб забезпечити фінансову стабільність ательє.

## 

## Детальний опис функціоналу бази даних, що пропонується

Детальний опис функціоналу запропонованої бази даних для ательє моди включає в себе ряд ключових елементів, які спрямовані на ефективне ведення бізнесу та полегшення робочих процесів. Відзначимо основні аспекти цього функціоналу:

Запропонована база даних для ательє моди включає в себе каталоги моделей, тканин та фурнітури. У каталозі моделей зберігається інформація про використані матеріали (тканина, фурнітура), витрати на роботу та загальну ціну. Каталог тканин включає дані про назву, ширину, ціну та наявність на складі. Каталог фурнітури містить інформацію про назву, кількість та ціну за одиницю.

Журнал реєстрації замовлень включає особисті дані клієнта, інформацію про обрану модель та використані матеріали, а також дати прийому, примірки та виконання замовлення, з позначкою про оплату. Функціонал бази даних включає в себе ведення каталогів, розрахунок вартості ескізу, облік витрат і залишків матеріалів, друк детальних чеків на оплату та відстеження завантаженості виконавців. Така база даних дозволяє ефективно керувати замовленнями, відстежувати наявність матеріалів, а також забезпечує докладну інформацію для операційного управління ательє моди.

У Каталозі моделей ми створюємо такі атрибути як :тканина, витрата тканини (за розмірами), вартість тканини, вартість фурнітури, вартість роботи, ціна.

Ми використовуємо такий функціонал як, ведення каталогу моделей зі зазначенням типу тканини, витрати тканини в залежності від розмірів, вартості тканини, фурнітури, праці та загальної ціни.

У каталозі тканин ми створюємо такі атрибути як, назва, ширина, ціна за 1 метр, наявність на складі.

Я використовую такий функціонал як, облік різних видів тканин із зазначенням назви, ширини, ціни за метр та наявності на складі.

Каталог фурнітури:

У каталозі фурнітури я стоврюю такі атрибути як, Назва, кількість, ціна за 1 шт.

Використовую такий функціонал як, утримання каталогу фурнітури зі зазначенням назви, кількості та ціни за одиницю.

У базі даних ательеє мод використовую Журнал реєстрації замовлень наших клієнтів.

Атрибути які використано :

* ПІБ клієнта, інформація про модель
* інформація про тканину
* ПІБ виконавця замовлення
* дата прийому
* дата примірки
* дата виконання
* відмітка про оплату.

Використанно такий функціонал замовлень :

* Реєстрація замовлень із зазначенням особистих даних клієнта
* інформації про модель і тканину
* Виконавця
* дат прийому
* примірки та виконання замовлення
* також відмітка про оплату.

Нашими вхідними даними програми будуть такими, Залишки тканини та фурнітури на складі, вибірка тканини, моделі та фурнітури в каталозі за назвою, поточні замовлення, навантаження виконавців замовлень та чек на оплату замовлення

У роботі було також було використано додаткові функції такі як, ведення каталогів моделей, тканин та фурнітури , Розрахунок вартості ескізу, облік витрат і залишків тканини та фурнітури, друк чеків на оплату, відстеження завантаженості виконавців.

1. ПРОЕКТУВАННЯ БАЗИ ДАНИХ «ДІЯЛЬНІСТЬ АТЕЛЬЄ МОД»

Метою концептуального проектування є створення повної і точної моделі предметної області «Ательє мод», яка є основою для побудови бази даних.

У предметній області «Діяльність видавництва» було виділено 6 сутностей: замовлення, клієнт, модель , тканина , фурнітура , майстер . Для кожної сутності були виділені свої атрибути, які дають можливість реалізувати усю множину запитів до БД. Виділені суттєві сутності, що характеризують предметну область, та їх характеристика представлені в таблиці 2.1.

Таблиця 2.1 – Суттєві сутності, що характеризують предметну область

|  |  |
| --- | --- |
| Сутність | Характеристика |
| Модель | представляє різні дизайнерські моделі одягу. Кожна модель має атрибути, такі як ідентифікатор, назва, опис, вартість роботи та інші характеристики, які визначають вигляд і вартість виготовлення моделі. |
| Тканина: | містить інформацію про різні види тканин, їх назву, ширину, ціну за метр та кількість на складі. Вона використовується для обчислення вартості матеріалів для кожної моделі та контролю за наявністю тканин. |
| Замовлення | включає інформацію про кожне замовлення включаючи ПІБ клієнта, обрану модель, використану тканину та фурнітуру, ПІБ виконавця, дату прийому, дату примірки та дату виконання. Також, |

Продовження таблиці 2.1

|  |  |
| --- | --- |
| Сутність | Характеристика |
|  | ця сутність може містити інформацію про статус замовлення та оплату. |
| Майстер | представляє інформацію про майстрів ательє, включаючи їхні прізвища, імена, контактні дані та інші характеристики. |
| Клієнт | містить інформацію про клієнтів, які замовляють одяг у ательє. Вона включає ПІБ клієнта, адресу, контактні дані та інші атрибути для ідентифікації клієнта та комунікації з ним |

Виділені властивості об’єктів (атрибути), їх типи даних та ключі представлено в таблиці 2.2.

Таблиця 2.2 – Виділені атрибути сутностей та їх характеристики

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Сутність | Властивість | Тип даних | Ключ | |
|  |  |  |  | |
| Замовлення | ID замовлення | числовий | | первинний |
| обрана модель | символьний | | зовнішній |
| ПІБ виконавця | символьний | |  |
| Дата виконання | Дата | |  |
| Статус замовлення | символьний | |  |
| Модель | ID моделі | числовий | первинний | |
| ID Тканини | числовий |  | |
| ID Фурнітури | числовий |  | |

Продовження таблиці 2.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Сутність | Властивість | | Тип даних | Ключ |
|  | Назва моделі | | символьний | зовнішній |
| Вартість роботи | | числовий |  |
| Тканина  Сутність | ID тканини | | числовий | первинний |
| Назва тканини | | символьний |  |
| Ширина тканини | | числовий |  |
| Ціна за метр тканини | | числовий |  |
| Кількість на складі | | числовий |  |
| Властивість | | Тип даних | Ключ |
| Фурнітура | ID фурнітури | | числовий | первинний |
| Назва фурнітури | | символьний |  |
| Ціна за одиницю фурнітури | | дійсний |  |
| Кількість на складі | | числовий |  |
| Майстер | ID майстра | | числовий | первинний |
| ПІБ майстра | | символьний |  |
| Контактні дані майстра | | символьний |  |
| Клієнт | ID клієнта | числовий | | первинний |

Продовження таблиці 2.2

|  |  |  |  |
| --- | --- | --- | --- |
| Сутність | Властивість | Тип даних | Ключ |
|  | ПІБ клієнта | символьний |  |
| Контактні дані клієнта | числовий |  |

Також була виконана формалізація зв’язків відповідно до правил формалізації шляхом додавання зовнішніх ключів та, за необхідності, асоціативних сутностей, які були згадані раніше при описі об’єктів. Для кожного атрибута було створено власний первинний ключ у вигляді id. Дані id були додані як зовнішні ключі до таблиць інших атрибутів для зображення зовнішніх зв’язків. Було виділено дві асоціативні сутності для типу зв’язку багато-до-багатьох між об’єктами «замовлення» і «видання», та об’єктами «замовлення» і «друкарня». Визначені типи зв’язків між сутностями представлено в таблиці 2.3.

Таблиця 2.3 – Типи зв’язків між сутностями

|  |  |  |
| --- | --- | --- |
| сутність | Тип зв’язку | Сутність |
| Замовлення | N : 1 | Клієнт |
| Замовлення | N : N | Модель |
| Замовлення | N : 1 | Майстер |
| Модель | N: N | Тканина |
| Модель | N: N | Фурнітура |

На основі виділених сутностей, їх атрибутів та зв'язків між сутностями було спроектовано графічну модель у вигляді схеми даних для "Діяльності ательє мод". Ця схема даних відображає таблиці, що відповідають окремим сутностям, а також зв'язки між ними, забезпечуючи зручне використання цих зв'язків при обробці даних та роботі з базою даних. Також в схемі даних визначені зовнішні ключі, що вказують на зв'язки між різними сутностями.

На основі виділених сутностей, їх атрибутів та зв’язків між сутностями створено інформаційну модель в нотації MS Access, яка представлена на рис. 2.1.

Графічна модель у вигляді схеми даних дозволяє чітко відобразити структуру бази даних та її складові частини. Вона є важливим інструментом для розуміння взаємозв'язків між сутностями та забезпечує зручний спосіб для організації та відображення інформації в базі даних "Діяльності ательє мод ". Інформаційну модель представлено на рис. 2.1.

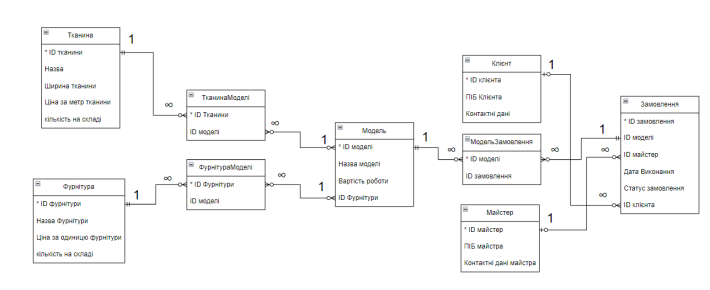


Рисунок 2.1 – Інформаційна модель БД

# ВИБІР ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Для забезпечення надійного та ефективного зберігання і обробки даних використовується СУБД PostgreSQL. Обрана СУБД відома своєю стабільністю, високою продуктивністю та розширюваністю, що робить її відмінним вибором для проектів різної складності. PostgreSQL також підтримує розширення для роботи з геоданими та великими обсягами інформації, що може бути корисним для відстеження різних аспектів ательє мод.

Основним інструментом у вашому випадку є pgAdmin 4, який є адміністративним інтерфейсом для управління базою даних PostgreSQL.

Вибір PostgreSQL може бути обгрунтований його високою надійністю, розширюваністю та відкритістю, що дозволяє вам вільно користуватися та змінювати його відповідно до своїх потреб.

Характеристики персонального комп'ютера:

Для нормального функціонування програми та СУБД, персональний комп'ютер повинен відповідати наступним характеристикам:

* операційна система: ОС Windows 7 або новіше;
* процесор: процесор 2-ядерний 1,6 Ггц;
* оперативна пам'ять (RAM): ОЗУ 2 Гб;
* місце на жорсткому диску: 1 Гб вільного місця на жорсткому диску (HDD);
* інші зовнішні залежності: встановлений .Net Framework 4.7.

1. СТВОРЕННЯ БАЗИ ДАНИХ

На підставі отриманого проекту засобами мови SQL було створено базові елементи бази даних (її сутностей та зв’язків між ними): послідовності, домени, таблиці, приклади наборів даних для кожної таблиці і додаткові елементи бази даних: представлення, тригери і процедури (або призначені для користувача функції).

Для перевірки що ім'я та прізвище введені правильно та не перевищують обмеження у 255 символів було створено домен який визначає тип даних для імен і прізвищ клієнтів та майстрів

CREATE DOMAIN PIB AS VARCHAR(255); -- Домен для ПІБ

CREATE DOMAIN Contact AS VARCHAR(255); -- Домен для контактних даних

Далі було створено таблиці з набором їх властивостей та, за наявності, зовнішніми ключами.

Створення таблиці Тканина (Рисунок 4.1)

CREATE TABLE Тканина (

ID\_тканини SERIAL PRIMARY KEY,

Назва VARCHAR(255),

Ширина\_тканини DECIMAL(10, 2),

Ціна\_за\_метр DECIMAL(10, 2),

Кількість\_на\_складі INT

);



Рисунок 4.1 – Таблиця «тканина»

Створення таблиці Фурнітура (Рисунок 4.2)

CREATE TABLE Фурнітура (

ID\_Фурнітури SERIAL PRIMARY KEY,

Назва\_Фурнітури VARCHAR(255),

Ціна\_за\_одиницю DECIMAL(10, 2),

Кількість\_на\_складі INT

);

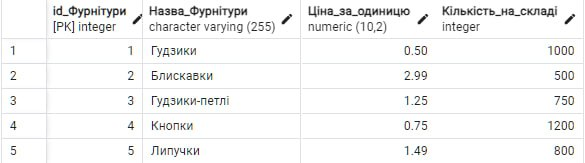


Рисунок 4.2 – Таблиця «фурнітура»

Створення таблиці Клієнт (Рисунок 4.3)

CREATE TABLE Клієнт (

ID\_клієнта SERIAL PRIMARY KEY,

ПІБ\_клієнта PIB,

Контактні\_дані Contact

);

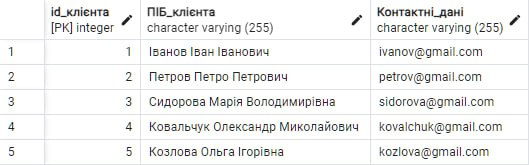


Рисунок 4.3 – Таблиця «клієнт»

Створення таблиці Майстер (Рисунок 4.4)

CREATE TABLE Майстер (

ID\_майстер SERIAL PRIMARY KEY,

ПІБ\_майстра PIB,

Контактні\_дані\_майстра Contact

);



Рисунок 4.4 – Таблиця «майстер»

Створення таблиці Модель (Рисунок 4.5)

CREATE TABLE Модель (

ID\_моделі SERIAL PRIMARY KEY,

Назва\_моделі VARCHAR(255),

Вартість\_роботи DECIMAL(10, 2),

ID\_Фурнітури INT,

FOREIGN KEY (ID\_Фурнітури) REFERENCES Фурнітура(ID\_Фурнітури)

);

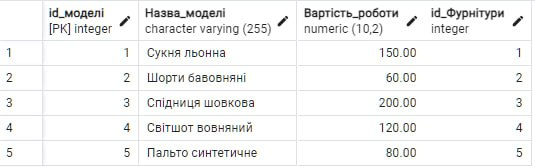


Рисунок 4.5 – Таблиця «модель»

Створення таблиці Замовлення (Рисунок 4.6)

CREATE TABLE Замовлення (

ID\_замовлення SERIAL PRIMARY KEY,

ID\_моделі INT,

ID\_майстер INT,

Дата\_виконання DATE,

Статус\_замовлення VARCHAR(255),

ID\_клієнта INT,

FOREIGN KEY (ID\_моделі) REFERENCES Модель(ID\_моделі),

FOREIGN KEY (ID\_майстер) REFERENCES Майстер(ID\_майстер),

FOREIGN KEY (ID\_клієнта) REFERENCES Клієнт(ID\_клієнта)

);

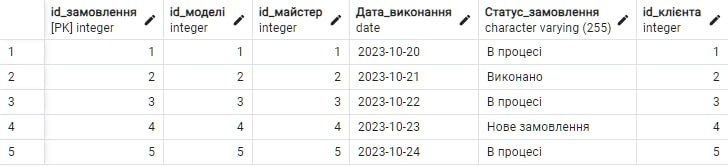


Рисунок 4.6 – Таблиця «замовлення»

Також через необхідність наявності зв’язку багато-до-багатьох було створено три таблиці для асоціативних сутностей між об’єктами «Замовлення» та «Модель» , між «Модель» та «Тканина» і «Фурнітура»

Створення таблиці Замовлення\_моделі (асоціативна сутність)

(Рисунок 4.7)

CREATE TABLE Замовлення\_моделі (

ID\_моделі INT,

ID\_замовлення INT,

PRIMARY KEY (ID\_моделі, ID\_замовлення),

FOREIGN KEY (ID\_моделі) REFERENCES Модель(ID\_моделі),

FOREIGN KEY (ID\_замовлення) REFERENCES Замовлення(ID\_замовлення)

);



Рисунок 4.7 – Асоціативна таблиця Замовлення\_моделі

Створення таблиці Тканина\_Моделі (асоціативна сутність)

(Рисунок 4.8)

CREATE TABLE Тканина\_Моделі (

ID\_Тканина INT,

ID\_моделі INT,

PRIMARY KEY (ID\_Тканина, ID\_моделі),

FOREIGN KEY (ID\_Тканина) REFERENCES Тканина(ID\_тканини),

FOREIGN KEY (ID\_моделі) REFERENCES Модель(ID\_моделі)

);



Рисунок 4.8 – Асоціативна таблиця Тканина\_Моделі

Створення таблиці Фурнітура\_Моделі (асоціативна сутність)

(Рисунок 4.9 )

CREATE TABLE Фурнітура\_Моделі (

ID\_Фурнітура INT,

ID\_моделі INT,

PRIMARY KEY (ID\_Фурнітура, ID\_моделі),

FOREIGN KEY (ID\_Фурнітура) REFERENCES Фурнітура(ID\_Фурнітури),

FOREIGN KEY (ID\_моделі) REFERENCES Модель(ID\_моделі)

);



Рисунок 4.9 – Асоціативна таблиця Фурнітура\_Моделі

## Створення представлень

В рамках виконання курсової роботи створено наступні представлення:

1) Основна мета цього запиту - створити вигляд, який містить інформацію про замовлення, включаючи дані про клієнта, майстра, модель, тканину та фурнітуру. Вибираємо колонки, які будуть включені в вигляд.

Перейменовуємо деякі колонки для зручності, наприклад, "ПІБ\_клієнта" перейменовується в "Клієнт".З'єднуємо таблиці "Замовлення", "Клієнт", "Майстер" та "Модель" за допомогою JOIN операцій на основі відповідних ідентифікаторів.

Використовуємо LEFT JOIN для таблиць "Тканина\_Моделі", "Тканина", "Фурнітура\_Моделі" та "Фурнітура" для забезпечення включення інформації навіть для тих замовлень, де можуть бути відсутні дані про тканину чи фурнітуру.Вказуємо, які колонки повинні збігатися для з'єднання таблиць.Вказуємо, що цей SELECT-запит повинен бути використаний для створення нового вигляду "OrderDetails". (Рисунок 4.10)

CREATE VIEW OrderDetails AS

SELECT

Замовлення.ID\_замовлення,

Замовлення.Дата\_виконання,

Замовлення.Статус\_замовлення,

Клієнт.ПІБ\_клієнта AS Клієнт,

Майстер.ПІБ\_майстра AS Майстер,

Модель.Назва\_моделі AS Модель,

Тканина.Назва AS Тканина,

Фурнітура.Назва\_Фурнітури AS Фурнітура

FROM Замовлення

JOIN Клієнт ON Замовлення.ID\_клієнта = Клієнт.ID\_клієнта

JOIN Майстер ON Замовлення.ID\_майстер = Майстер.ID\_майстер

JOIN Модель ON Замовлення.ID\_моделі = Модель.ID\_моделі

LEFT JOIN Тканина\_Моделі ON Модель.ID\_моделі = Тканина\_Моделі.ID\_моделі

LEFT JOIN Тканина ON Тканина\_Моделі.ID\_Тканина = Тканина.ID\_тканини

LEFT JOIN Фурнітура\_Моделі ON Модель.ID\_моделі = Фурнітура\_Моделі.ID\_моделі

LEFT JOIN Фурнітура ON Фурнітура\_Моделі.ID\_Фурнітура = Фурнітура.ID\_Фурнітури;

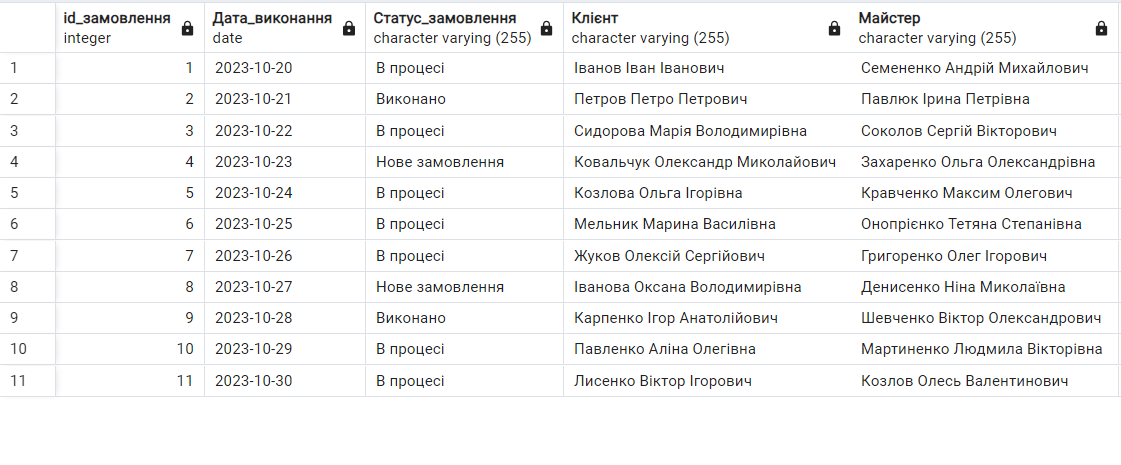


Рисунок 4.10 – Приклад відображення інформації про замовлення з деталями моделі

2)Представлення у результаті вигляду "FabricStock" буде містити інформацію про наявний залишок кожного виду тканини на складі. Вигляд полегшує доступ до цієї інформації і дозволяє використовувати його для подальших операцій та запитів. Рядок починається з команди CREATE VIEW, що вказує, що ми створюємо новий вигляд. FabricStock - назва нового вигляду. У цьому розділі вибираються колонки, які будуть включені у вигляд.

Обираємо колонки з таблиці Тканина. Використовується для перейменування колонок для зручності. Наприклад, Тканина.Назва перейменовується в Тканина, а Тканина.Кількість\_на\_складі перейменовується в Залишок\_тканини. Звідки ми беремо дані для нового вигляду.

У цьому випадку вибираємо дані з таблиці Тканина. (Рисунок 4.11)

CREATE VIEW FabricStock AS

SELECT

Тканина.ID\_тканини,

Тканина.Назва AS Тканина,

Тканина.Кількість\_на\_складі AS Залишок\_тканини

FROM Тканина;

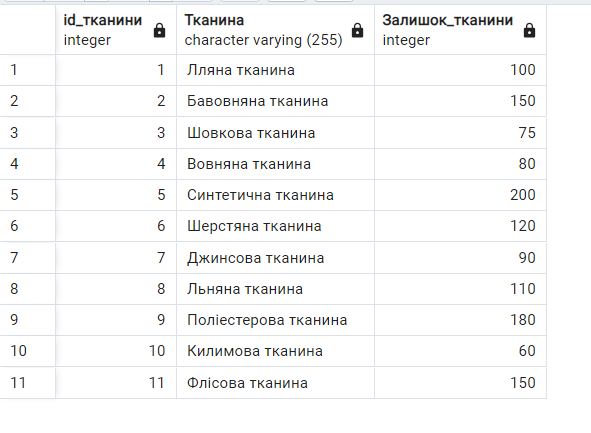


Рисунок 4.11 – Приклад відображення інформації кількості залишку тканини на складі

3)Представлення результаті вигляду "HardwareStock" буде містити інформацію про наявний залишок кожного виду фурнітури на складі. Вигляд полегшує доступ до цієї інформації і дозволяє використовувати його для подальших операцій та запитів. Цей рядок починається з команди CREATE VIEW, що вказує, що ми створюємо новий вигляд. HardwareStock - назва нового вигляду. У цьому розділі вибираються колонки, які будуть включені у вигляд.

Обираємо колонки з таблиці Фурнітура. Використовується для перейменування колонок для зручності. Наприклад, Фурнітура.Назва\_Фурнітури перейменовується в Фурнітура, а Фурнітура.Кількість\_на\_складі перейменовується в Залишок\_фурнітури. Звідки ми беремо дані для нового вигляду.

У цьому випадку вибираємо дані з таблиці Фурнітура. (Рисунок 4.12)

CREATE VIEW HardwareStock AS

SELECT

Фурнітура.ID\_Фурнітури,

Фурнітура.Назва\_Фурнітури AS Фурнітура,

Фурнітура.Кількість\_на\_складі AS Залишок\_фурнітури

FROM Фурнітура;

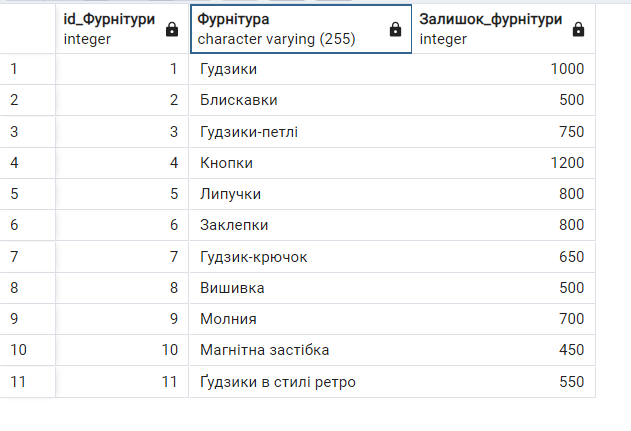


Рисунок 4.12 – Приклад відображення інформації про кількості залишку фурнітури на складі

## Створення тригерів

В рамках виконання курсової роботи створено наступні тригери :

1) Цей код створює функцію та тригер для автоматичного оновлення залишків тканини на складі при додаванні нових записів у таблицю "Замовлення\_моделі". Коли вставляється новий запис у "Замовлення\_моделі", тригер викликає функцію, яка зменшує кількість тканини на складі для тих моделей, які були додані до замовлення.

CREATE OR REPLACE FUNCTION updateFabricStock()

RETURNS TRIGGER AS $$

BEGIN

UPDATE Тканина

SET Кількість\_на\_складі = Кількість\_на\_складі - 1

WHERE ID\_тканини IN (SELECT ID\_Тканина FROM Тканина\_Моделі WHERE ID\_моделі = NEW.ID\_моделі);

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER update\_fabric\_stock\_trigger

AFTER INSERT ON Замовлення\_моделі

FOR EACH ROW

EXECUTE FUNCTION updateFabricStock();

Показуємо таблицю Тканина , перед додаванням нового замовленя

**(**Рисунок 4.13**):** :

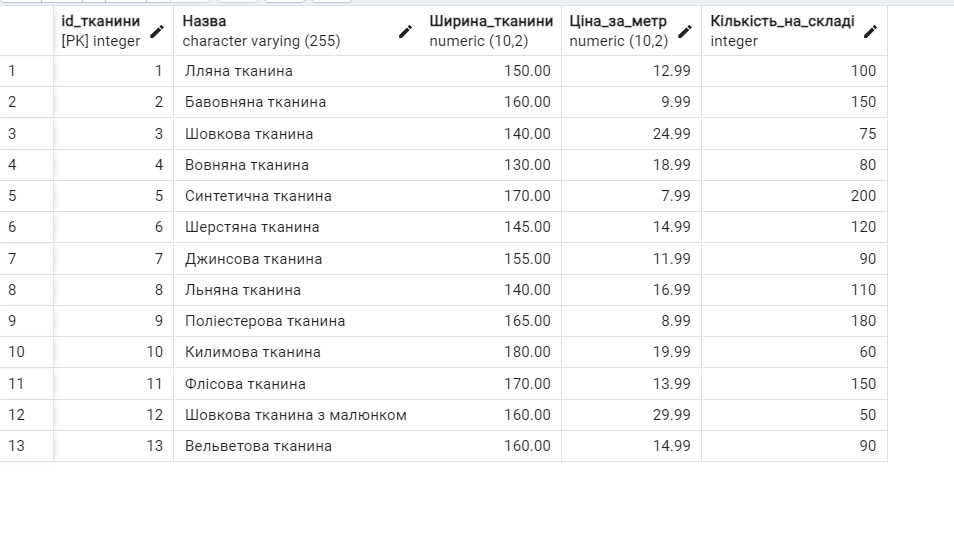


Рисунок 4.13 – Приклад відображення інформації про автоматичне оновлення кількості тканини на складі

INSERT INTO Замовлення\_моделі (ID\_моделі, ID\_замовлення) VALUES (1, 14);

Перевірка, чи було автоматично оновлено залишок тканини за допомогою тригера

SELECT \* FROM Тканина;

Після того як ми додали замовлення кількість тканини «лляна тканина» зменишилась на 1 одиницю (Рис 4.14) :



Рисунок 4.14 – Приклад дії коду

2) Цей код створює функцію та тригер для автоматичного оновлення залишків фурнітури на складі при додаванні нових записів у таблицю "Замовлення\_моделі". Коли вставляється новий запис у "Замовлення\_моделі", тригер викликає функцію, яка зменшує кількість фурнітури на складі для тих моделей, які були додані до замовлення.

CREATE OR REPLACE FUNCTION updateHardwareStock()

RETURNS TRIGGER AS $$

BEGIN

UPDATE Фурнітура

SET Кількість\_на\_складі = Кількість\_на\_складі - 1

WHERE ID\_Фурнітури IN (SELECT ID\_Фурнітура FROM Фурнітура\_Моделі WHERE ID\_моделі = NEW.ID\_моделі);

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER update\_hardware\_stock\_trigger

AFTER INSERT ON Замовлення\_моделі

FOR EACH ROW

EXECUTE FUNCTION updateHardwareStock();

Показуємо таблицю Фурнітура , перед додаванням нового замовленя

(Рис 4.15):



Рисунок 4.15 – Приклад відображення інформації про автоматичного оновлення кількості фурнітури на складі

Додавання нового замовлення з існуючим ID\_моделі

INSERT INTO Замовлення\_моделі (ID\_моделі, ID\_замовлення)

VALUES (1, 14);

Перевірка, чи було автоматично оновлено залишок фурнітури за допомогою тригера

SELECT \* FROM Фурнітура;

Після того як ми додали замовлення кількість фурнітури «гудзики» зменишилась на 1 одиницю(Рис 4.16) :

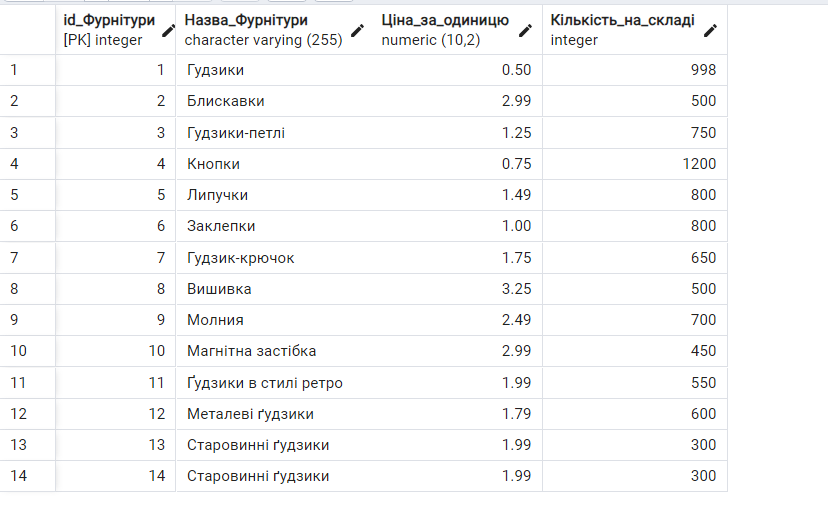


Рисунок 4.16 – Приклад дії коду

3) Цей код створює функцію та тригер для автоматичного оновлення статусу замовлення на "Виконано" при додаванні нових записів у таблицю "Замовлення\_моделі". Коли вставляється новий запис у "Замовлення\_моделі", тригер викликає функцію, яка оновлює статус відповідного замовлення на "Виконано".

CREATE OR REPLACE FUNCTION checkOrderCompletion()

RETURNS TRIGGER AS $$

BEGIN

UPDATE Замовлення

SET Статус\_замовлення = 'Виконано'

WHERE ID\_замовлення = NEW.ID\_замовлення;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER check\_order\_completion\_trigger

AFTER INSERT ON Замовлення\_моделі

FOR EACH ROW

EXECUTE FUNCTION checkOrderCompletion();

Таблиця замовлення перед виконанням тригера (Рисунок 4.17) :

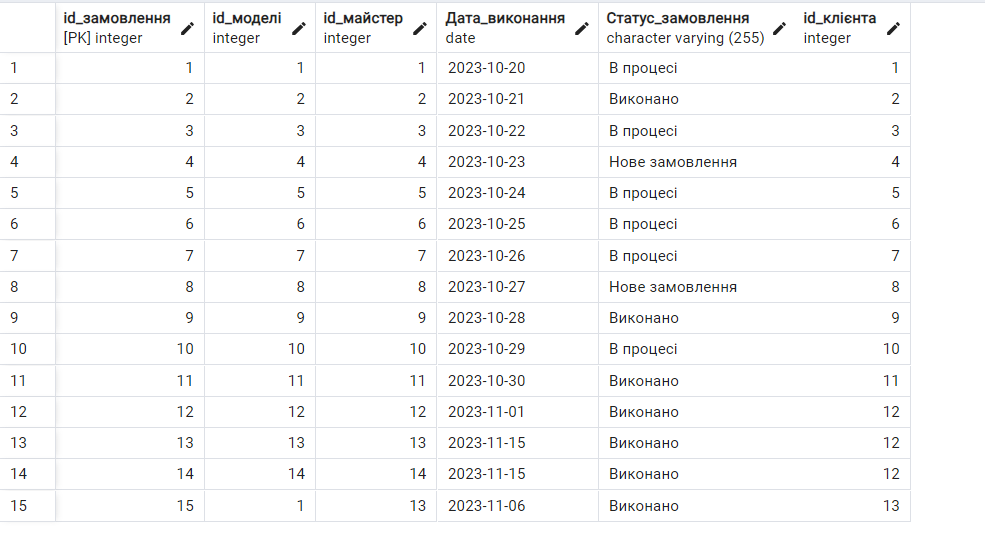


Рисунок 4.17 – Приклад відображення інформації про визначення статусу замовлення "Виконано"

Вставка запису в таблицю Замовлення\_моделі

INSERT INTO Замовлення\_моделі (ID\_моделі, ID\_замовлення)

VALUES (1, 8);

-- Перевірка результату в таблиці Замовлення

SELECT \* FROM Замовлення WHERE ID\_замовлення = 8;

Таблиця замовлення після виконанням тригера (Рис 4.18) після додавання Замовлення\_моделі замовлення під id =8 було змінено з «нове замовлення» у

«виконанно» (Рис 4.18):

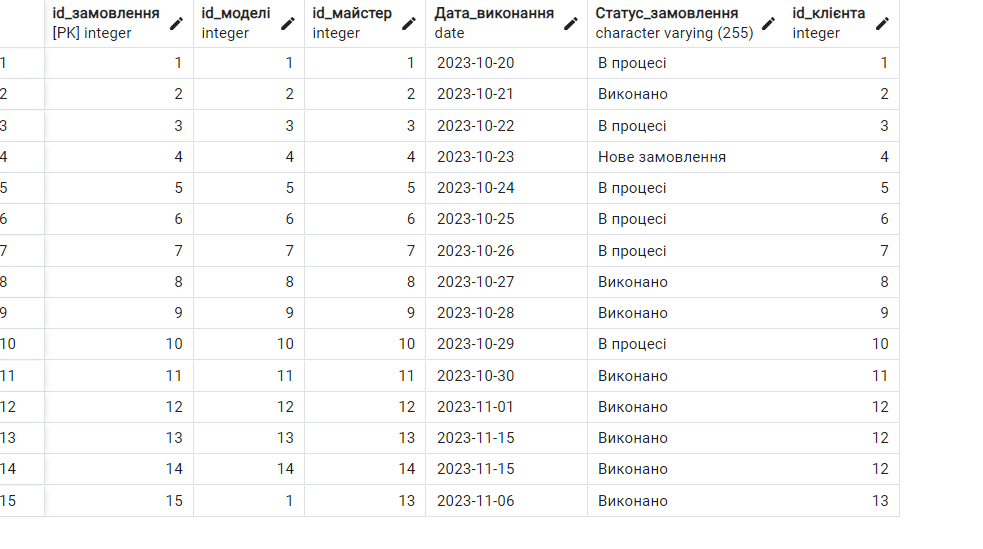


Рисунок 4.18 – Приклад дії коду

4) тригер викликається після видалення запису з таблиці "Клієнт" і призначений для автоматичного видалення всіх замовлень, які пов'язані з цим клієнтом. Давайте розглянемо кожен рядок тригера

CREATE OR REPLACE FUNCTION delete\_orders\_on\_client\_delete()

RETURNS TRIGGER AS $$

BEGIN

DELETE FROM Замовлення WHERE ID\_клієнта = OLD.ID\_клієнта;

RETURN OLD;

END;

$$ LANGUAGE plpgsql;

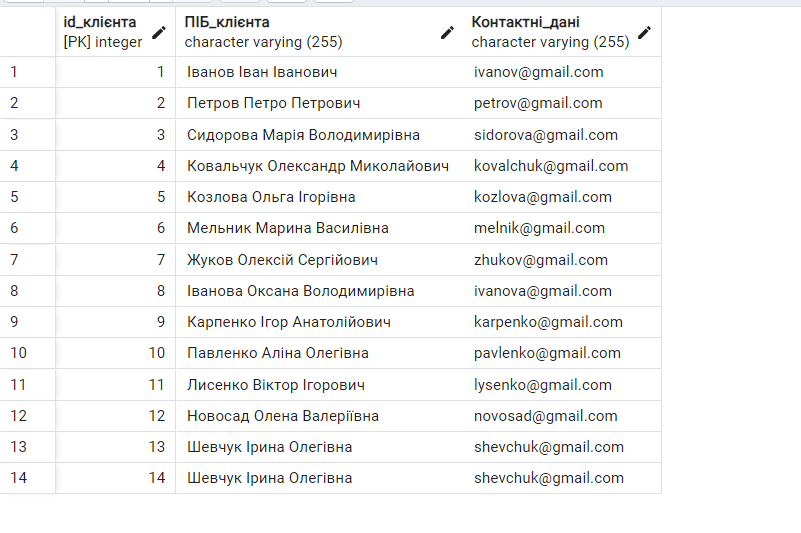
CREATE TRIGGER delete\_orders\_on\_client\_delete\_trigger

AFTER DELETE ON Клієнт

FOR EACH ROW

EXECUTE FUNCTION delete\_orders\_on\_client\_delete();

Таблиця Клієнт перед виконанням дії тригеру (Рисунок 4.19) :

Рисунок 4.19 – Приклад відображення інформації призначений для автоматичного видалення всіх замовлень, які пов'язані з цим клієнтом.

Додавання даних до таблиці Замовлення

INSERT INTO Замовлення (ID\_замовлення, ID\_моделі, ID\_майстер, Дата\_виконання, Статус\_замовлення, ID\_клієнта)

VALUES

(17, 1, 1, '2023-11-01', 'В процесі', 12); -- Додано кому тут

Вивід записів з таблиці Замовлення перед видаленням клієнта

SELECT \* FROM Замовлення WHERE ID\_клієнта = 14;

Видалення клієнта, що спричинить виклик тригера

DELETE FROM Клієнт WHERE ID\_клієнта = 14;

Вивід записів з таблиці Замовлення після видалення клієнта

SELECT \* FROM Замовлення WHERE ID\_клієнта = 14;

Видалення клієнту після виконання тригеру (Рис 4.20) :

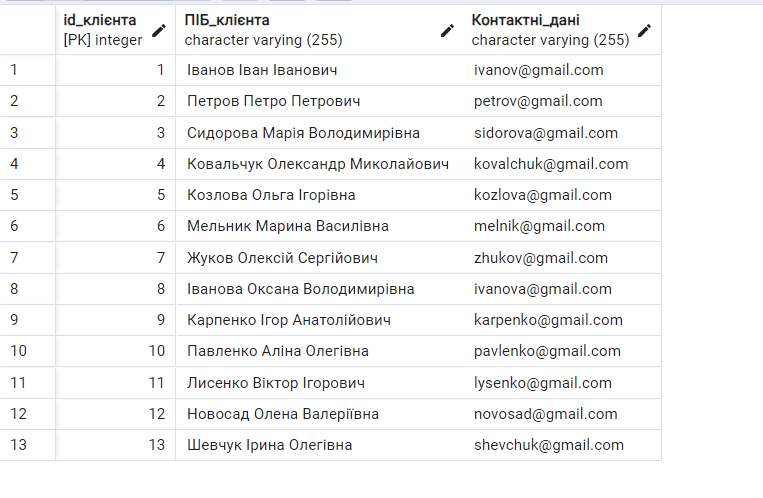


Рисунок 4.20 – Приклад дії коду

5) Цей код створює функцію та тригер для автоматичного оновлення статусу замовлення на "В процесі" при додаванні нових записів у таблицю "Замовлення". Коли вставляється новий запис у "Замовлення", тригер викликає функцію, яка оновлює статус відповідного замовлення на "В процесі".

CREATE OR REPLACE FUNCTION assignMaster()

RETURNS TRIGGER AS $$

BEGIN

UPDATE Замовлення

SET Статус\_замовлення = 'В процесі'

WHERE ID\_замовлення = NEW.ID\_замовлення;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER assign\_master\_trigger

AFTER INSERT ON Замовлення

FOR EACH ROW

EXECUTE FUNCTION assignMaster();

Таблиця «Замовлення» перед виконанням тригера (Рис 4.21) :

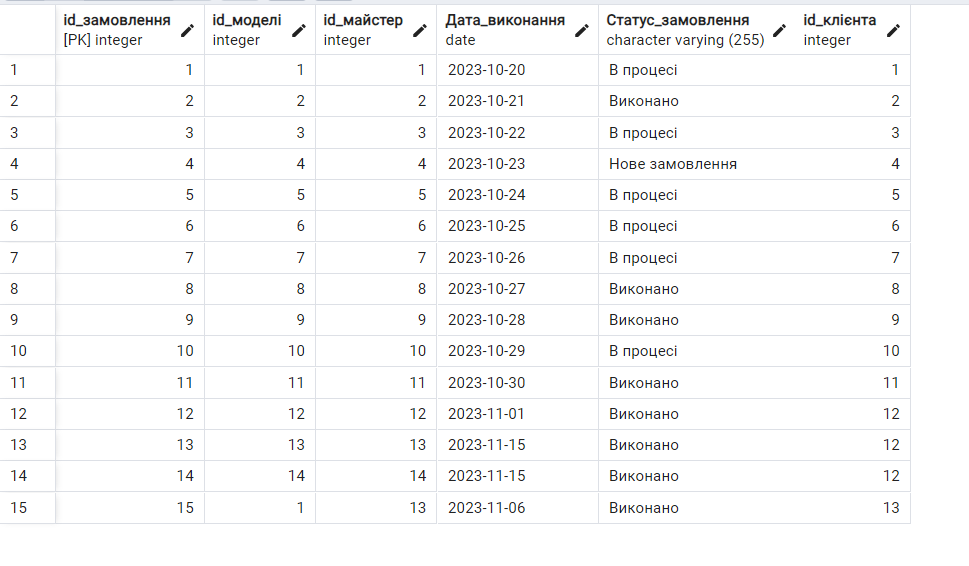


Рисунок 4.21 – Приклад відображення інформації визначення статусу замовлення "В процесі" при призначенні майстра

INSERT INTO Замовлення (ID\_замовлення, ID\_моделі, ID\_майстер, Дата\_виконання, Статус\_замовлення, ID\_клієнта)

VALUES (16, 1, 13, '2023-11-13', 'Нове замовлення', 13);

SELECT \* FROM Замовлення WHERE ID\_замовлення = 15;

Додано було нове замовлення яке оновило статус відповідного замовлення на "В процесі" (Рис 4.22) :

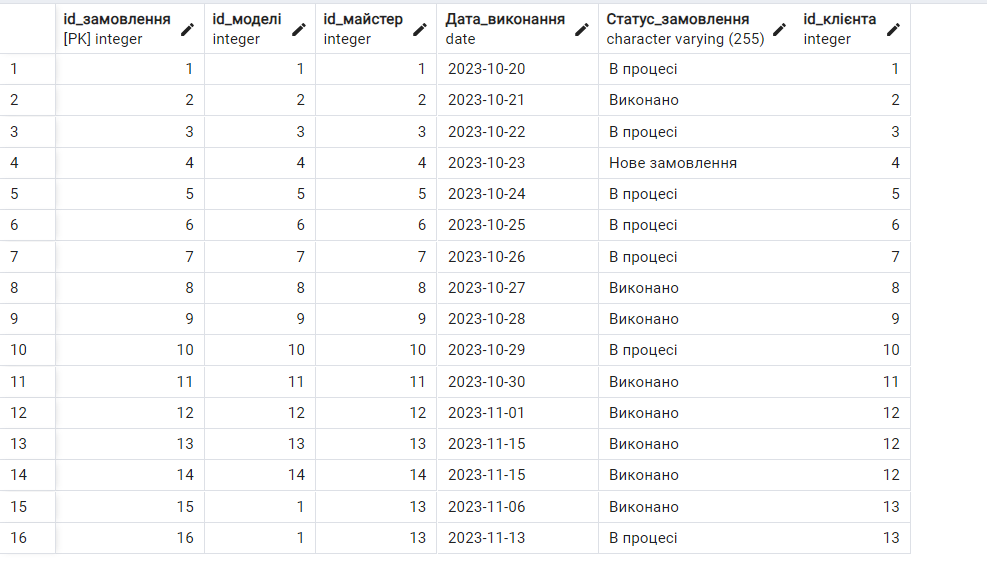


Рисунок 4.22 – Приклад дії коду

## Створення збережених процедур (функцій)

В рамках виконання курсової роботи створено наступні збережені процедури:

1)Отримання інформації про замовлення для конкретного клієнта (Рисунок 4.23.):

CREATE OR REPLACE FUNCTION getOrdersForClient(clientId INT)

RETURNS TABLE (

OrderId INT,

ModelName VARCHAR(255),

Status VARCHAR(255),

CompletionDate DATE

) AS $$

BEGIN

RETURN QUERY

SELECT

З.ID\_замовлення AS OrderId,

М.Назва\_моделі AS ModelName,

З.Статус\_замовлення AS Status,

З.Дата\_виконання AS CompletionDate

FROM

Замовлення З

JOIN Модель М ON З.ID\_моделі = М.ID\_моделі

WHERE

З.ID\_клієнта = clientId;

END;

$$ LANGUAGE plpgsql;

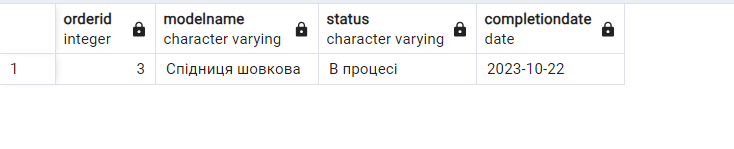


Рисунок 4.23 – Отримання інформації про замовлення для конкретного

клієнта

2)Отримання інформації про доступні тканини ( Рисунок 4.24)

CREATE OR REPLACE FUNCTION getAvailableFabrics()

RETURNS TABLE (

FabricId INT,

FabricName VARCHAR(255),

FabricWidth DECIMAL(10, 2),

PricePerMeter DECIMAL(10, 2),

QuantityInStock INT

) AS $$

BEGIN

RETURN QUERY

SELECT

ID\_тканини AS FabricId,

Назва AS FabricName,

Ширина\_тканини AS FabricWidth,

Ціна\_за\_метр AS PricePerMeter,

Кількість\_на\_складі AS QuantityInStock

FROM

Тканина;

END;

$$ LANGUAGE plpgsql;

SELECT \* FROM getAvailableFabrics();



Рисунок 4.24 – Отримання інформації про доступні тканини

3) Ця функція повертає інформацію про моделі, вказуючи для кожної моделі інформацію про саму тканину та модель, яка використовується для її вироблення (Рисунок 4.25.)

CREATE OR REPLACE FUNCTION getAvailableModels()

RETURNS TABLE (

ModelId INT,

ModelName VARCHAR(255),

WorkCost DECIMAL(10, 2),

FabricName VARCHAR(255),

FabricWidth DECIMAL(10, 2),

PricePerMeter DECIMAL(10, 2),

QuantityInStock INT

) AS $$

BEGIN

RETURN QUERY

SELECT

М.ID\_моделі AS ModelId,

М.Назва\_моделі AS ModelName,

М.Вартість\_роботи AS WorkCost,

Т.Назва AS FabricName,

Т.Ширина\_тканини AS FabricWidth,

Т.Ціна\_за\_метр AS PricePerMeter,

Т.Кількість\_на\_складі AS QuantityInStock

FROM

Модель М

JOIN Тканина\_Моделі ТМ ON М.ID\_моделі = ТМ.ID\_моделі

JOIN Тканина Т ON ТМ.ID\_Тканина = Т.ID\_тканини;

END;

$$ LANGUAGE plpgsql;

-- Приклад виклику функції

SELECT \* FROM getAvailableModels();

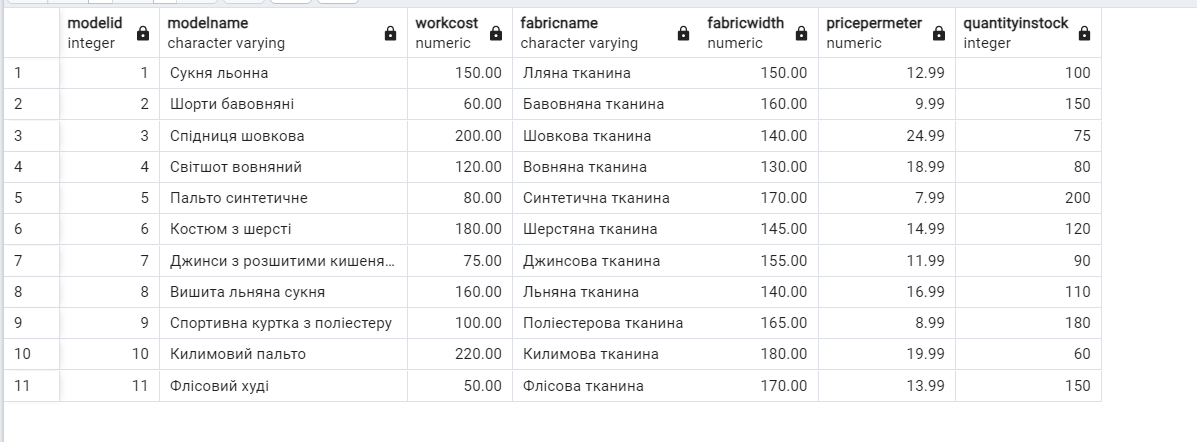


Рисунок 4.25 – Отримання інформації про моделі

4) Ця функція перевіряє наявність клієнта за його ідентифікатором, і якщо його немає, то додає нового клієнта з вказаним ім'ям та контактними даними. Якщо ідентифікатор клієнта вже існує, генерує помилку. Після успішного додавання клієнта статус відповідного замовлення оновлюється на "В процесі".

CREATE OR REPLACE FUNCTION додати\_клієнта(

новий\_id\_клієнта INT,

ім\_я VARCHAR(255),

контакт VARCHAR(255)

)

RETURNS VOID AS

$$

BEGIN

IF EXISTS (SELECT 1 FROM Клієнт WHERE ID\_клієнта = новий\_id\_клієнта) THEN

RAISE EXCEPTION 'Клієнт з ідентифікатором % вже існує', новий\_id\_клієнта;

END IF;

BEGIN

INSERT INTO Клієнт (ID\_клієнта, ПІБ\_клієнта, Контактні\_дані) VALUES (новий\_id\_клієнта, ім\_я, контакт);

EXCEPTION

WHEN unique\_violation THEN

RAISE EXCEPTION 'Помилка додавання клієнта. Можливо, ідентифікатор клієнта вже існує.';

END;

END;

$$

LANGUAGE plpgsql;

Таблиця «Клієнти» до виконання дії функції (Рис 4.26) :

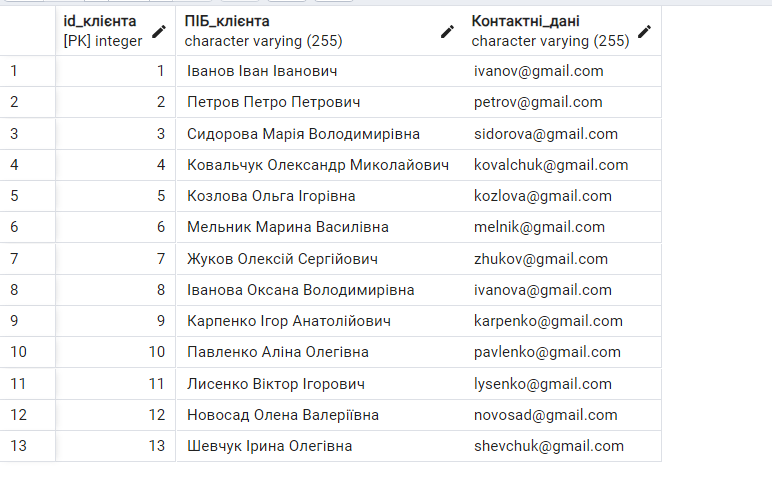


Рисунок 4.26 – таблиця Клієнта до використання функції

Викликаємо функцію і передаємо ідентифікатор клієнта

SELECT додати\_клієнта(14, 'Гербар Ірина Віталівна', 'gerbariyIrina@gmail.com');

Таблиця «Клієнти» після виконання дії функції де ми бачимо нового клієнта ательє :

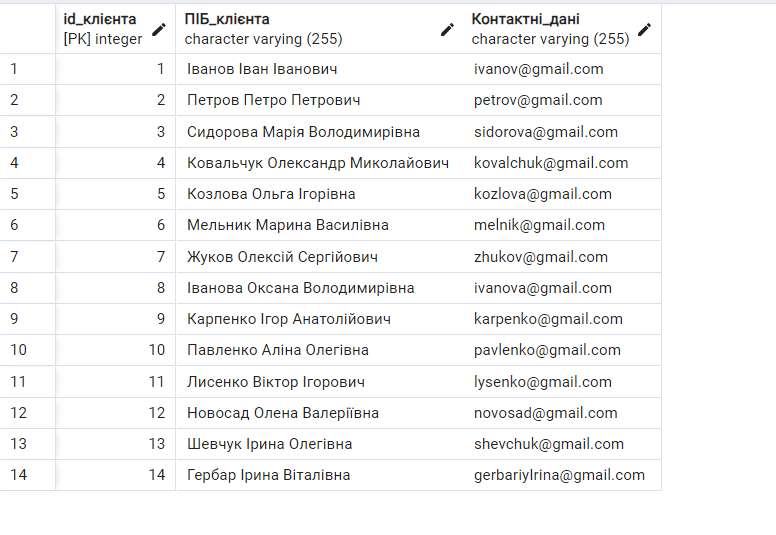


Рисунок 4.27 – таблиця Клієнта після використання функції

5) Ця функція, позначена як додати\_замовлення, призначена для додавання нового замовлення до таблиці "Замовлення". Якщо вставка в таблицю завершиться помилкою (наприклад, через конфлікт унікальності), функція видасть повідомлення про помилку.

CREATE OR REPLACE FUNCTION додати\_замовлення(

id\_замовлення INT,

id\_моделі INT,

id\_майстра INT,

дата\_виконання DATE,

статус\_замовлення VARCHAR(255),

id\_клієнта INT

)

RETURNS VOID AS

$$

BEGIN

BEGIN

-- Вставка запису в таблицю

INSERT INTO Замовлення (ID\_замовлення, ID\_моделі, ID\_майстер, Дата\_виконання, Статус\_замовлення, ID\_клієнта)

VALUES (id\_замовлення, id\_моделі, id\_майстра, дата\_виконання, статус\_замовлення, id\_клієнта);

EXCEPTION

-- Обробка помилок бази даних

WHEN others THEN

-- Виведення повідомлення про помилку

RAISE EXCEPTION 'Помилка при виконанні INSERT: %', SQLERRM;

END;

END;

$$ LANGUAGE plpgsql;

-- Викликаємо функцію і передаємо параметри для нового замовлення SELECT додати\_замовлення(18, 14, 14, '2023-11-15', 'Нове замовлення ', 13); обробка помилок у функції створення запису

На скріншоті зображено таблицю «Замовлення» до виконання функції з додаванням нового замовлення (Рис 4.28) :

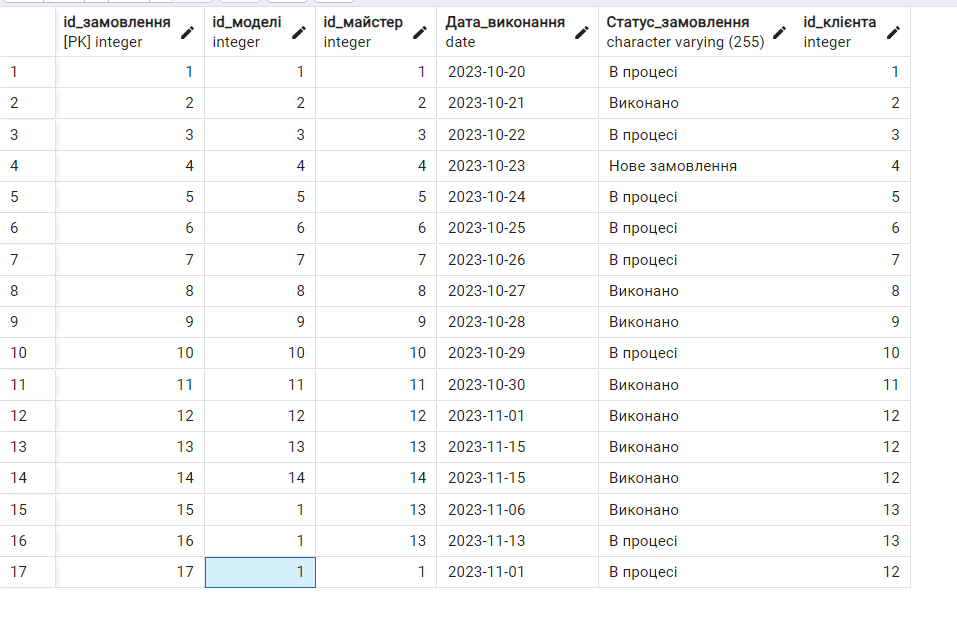


Рисунок 4.28 – таблиця Замовлення до використання функції

На скріншоті показано зміни які відбулися після того як було додано клієнта з id 18 (Рис 4.29):

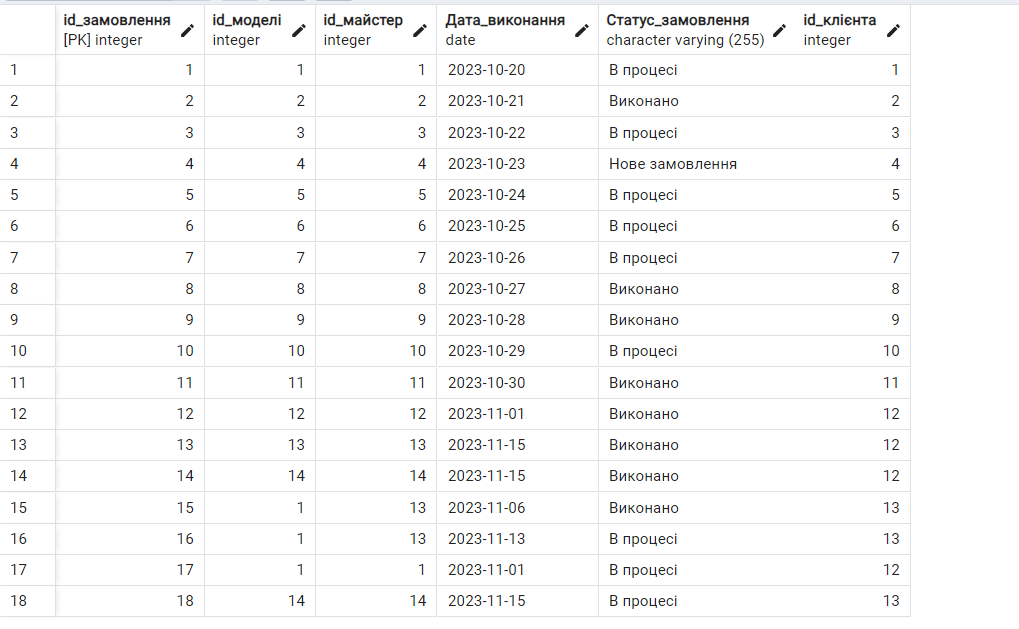


Рисунок 4.29 – таблиця Замовлення після використання функції

# МАНІПУЛЮВАННЯ ДАНИМИ

## 5.1 Оператори відновлення

1. приклад оператора відновлення з підзапитом, в якому ми змінюємо кількість товару на складі для певної тканини **.**

-- Знайдемо ID тканини за назвою

WITH Тканина\_ID AS (

SELECT ID\_тканини FROM Тканина WHERE Назва = 'Льняна тканина'

)

-- Оновимо кількість тканини на складі

UPDATE Тканина

SET Кількість\_на\_складі = Кількість\_на\_складі - 5

WHERE ID\_тканини = (SELECT ID\_тканини FROM Тканина\_ID);

Таблиця «Тканина» до виконання запиту на зменшення кількості тканини за назвою 'Льняна тканина' (Рис 5.1):



Рисунок 5.1 – Таблиця Тканина до зменшення тканини на складі

Оновлення яке відбулося у таблиці , зменшилося кількість на -5(Рис 5.2) :

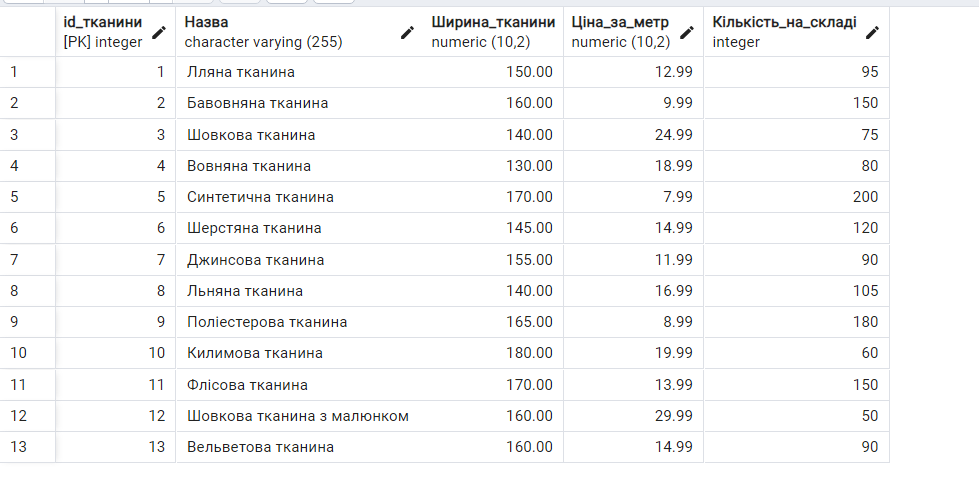


Рисунок 5.2 – Таблиця Тканина до зменшення тканини на складі

2) Вибірка даних з декількох таблиць (за Рис 5.3.)

SELECT Замовлення.ID\_замовлення, Замовлення.Статус\_замовлення, Модель.Назва\_моделі

FROM Замовлення

JOIN Модель ON Замовлення.ID\_моделі = Модель.ID\_моделі;

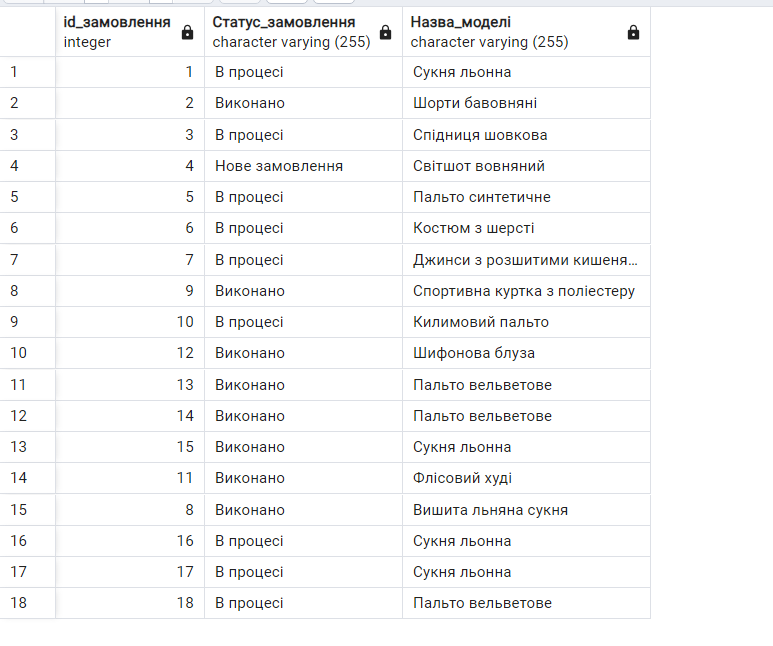


Рисунок 5.3. Вибірка з декількох таблиць

3)Оновлення запису в таблиці Тканина.

UPDATE Тканина

SET Ціна\_за\_метр = 14.99

WHERE ID\_тканини = 1;

Таблиця «Тканина » до оновлення ціни (Рис 5.4) :

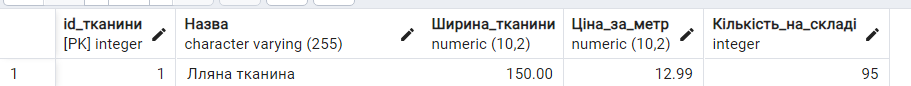


Рисунок 5.4 – Таблиця «Тканина» до зміни ціни

Таблиця «Тканина » після оновлення ціни з номером id=1 (Рис 5.5) :

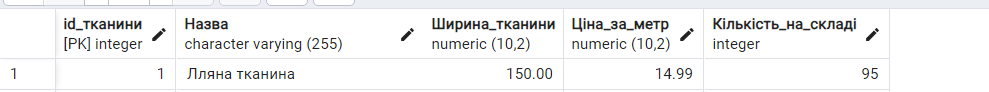


Рисунок 5.5 – Таблиця «Тканина» після зміни ціни

4) SQL-код видаляє рядок (запис) з таблиці "Замовлення", де значення в стовпці "ID\_замовлення".

DELETE FROM Замовлення

WHERE ID\_замовлення = 16;

Таблиця «Замовлення» до видалення замовлення (Рис 5.6.):

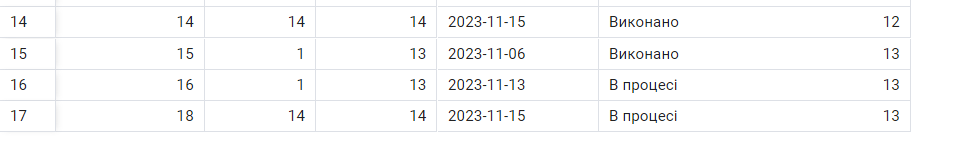


Рисунок 5.6 – Таблиця "Замовлення" до видалення замовлення

Після видалення цього замовлення (Рис 5.7) :



Рисунок 5.7 – Таблиця "Замовлення" після видалення замовлення

5) видалити клієнта за його ім'ям з таблиці "Клієнт"

DELETE FROM Клієнт

WHERE ПІБ\_клієнта = 'Шевченко Елена Георгіївна';

Таблиця «Клієнт» до видалення клієнта(Рис 5.8):

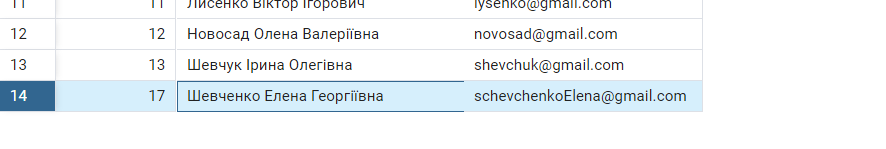


Рисунок 5.8 – Таблиця «Клієнт» до видалення клієнта

Таблиця «Клієнт» після виконання запиту :

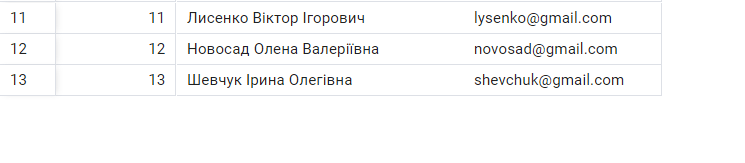


Рисунок 5.9 – Таблиця «Клієнт» після видалення клієнта

6) виведення інформації про замовлення клієнтів разом з їхніми прізвищами, контактами та ціною замовлення, яка менша за середню ціну моделей (за Рис 5.10.)

SELECT

Замовлення.ID\_замовлення,

Клієнт.ПІБ\_клієнта AS Прізвище\_клієнта,

Клієнт.Контактні\_дані AS Контакт,

SUM(Модель.Вартість\_роботи) AS Ціна\_замовлення

FROM Замовлення

JOIN Клієнт ON Замовлення.ID\_клієнта = Клієнт.ID\_клієнта

JOIN Замовлення\_моделі ON Замовлення.ID\_замовлення = Замовлення\_моделі.ID\_замовлення

JOIN Модель ON Замовлення\_моделі.ID\_моделі = Модель.ID\_моделі

GROUP BY Замовлення.ID\_замовлення, Клієнт.ПІБ\_клієнта, Клієнт.Контактні\_дані

HAVING SUM(Модель.Вартість\_роботи) < (SELECT AVG(Вартість\_роботи) FROM Модель);

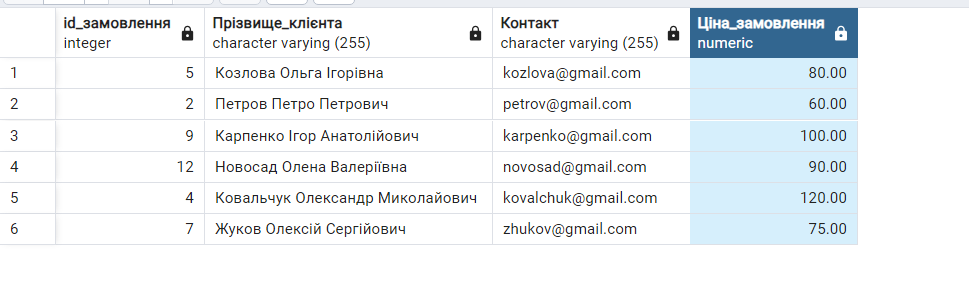


Рисунок 5.10 – Виводимо інформації про замовлення клієнтів разом з їхніми прізвищами, контактами та ціною замовлення

## Оператор вибірки

в рамках виконання курсової роботи створено наступні запити з операторами вибірки :

1)Запит Виводить інформацію про майстра, у якого середня вартість роботи перевищує 150(за Рис 5.11):

SELECT

Майстер.ID\_майстер,

Майстер.ПІБ\_майстра,

AVG(Модель.Вартість\_роботи) AS Середня\_вартість\_роботи

FROM

Майстер

LEFT JOIN

Замовлення ON Майстер.ID\_майстер = Замовлення.ID\_майстер

LEFT JOIN

Модель ON Замовлення.ID\_моделі = Модель.ID\_моделі

GROUP BY

Майстер.ID\_майстер, Майстер.ПІБ\_майстра

HAVING

AVG(Модель.Вартість\_роботи) > 150;

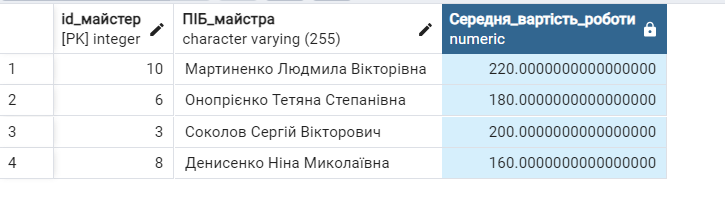


Рисунок 5.11 – Отримання інформації про майстра ціна роботи у якого більше 150

2)Знайти середню ціну тканини для кожної моделі.(за Рис.5.12)

SELECT

Модель.ID\_моделі,

Модель.Назва\_моделі,

AVG(Тканина.Ціна\_за\_метр) AS Середня\_ціна\_тканини

FROM

Модель

LEFT JOIN

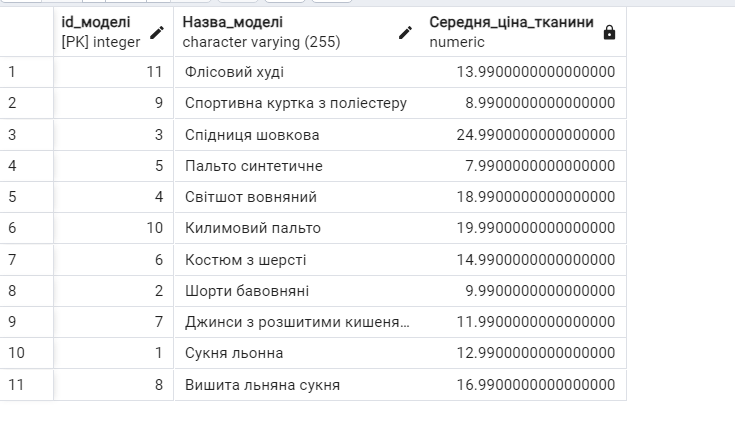
Тканина\_Моделі ON Модель.ID\_моделі = Тканина\_Моделі.ID\_моделі

LEFT JOIN

Тканина ON Тканина\_Моделі.ID\_Тканина = Тканина.ID\_тканини

GROUP BY

Модель.ID\_моделі, Модель.Назва\_моделі;

Рисунок.5.12 – Отримання інформації про середню ціну кожної моделі

3) запит виводить інформацію про клієнтів, які зробили замовлення на моделі з вартістю роботи більше 100, у тому числі їхній ідентифікатор, ПІБ, номер телефону та ідентифікатор і ціну відповідного замовлення, впорядковані за алфавітом за ПІБ клієнта (Рис.5.13)

SELECT

Клієнт.ID\_клієнта,

Клієнт.ПІБ\_клієнта,

Клієнт.Контактні\_дані, -- Додавання поля з номером телефону

Замовлення.ID\_замовлення,

Модель.Вартість\_роботи

FROM

Клієнт

INNER JOIN

Замовлення ON Клієнт.ID\_клієнта = Замовлення.ID\_клієнта

INNER JOIN

Модель ON Замовлення.ID\_моделі = Модель.ID\_моделі

WHERE

Модель.Вартість\_роботи > 100

ORDER BY

Клієнт.ПІБ\_клієнта ASC;

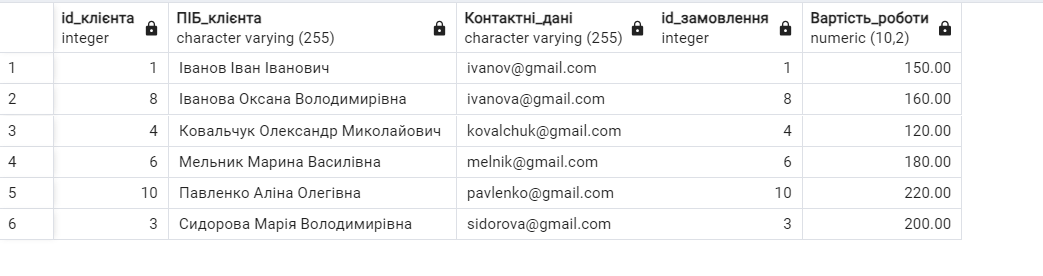


Рисунок.5.13 – Отримання інформації про клієнта де замовлення більше 100

4)Цей запит виведе інформацію про трьох майстрів з найвищою середньою вартістю роботи та їхні моделі замовлень(за Рис.5.14).

WITH AvgCostPerMaster AS (

SELECT

Замовлення.ID\_майстер,

AVG(Модель.Вартість\_роботи) AS Середня\_вартість

FROM

Замовлення

LEFT JOIN

Модель ON Замовлення.ID\_моделі = Модель.ID\_моделі

GROUP BY

Замовлення.ID\_майстер

)

SELECT

Майстер.ID\_майстер,

Майстер.ПІБ\_майстра,

Модель.ID\_моделі,

Модель.Назва\_моделі,

Модель.Вартість\_роботи

FROM

Майстер

LEFT JOIN

AvgCostPerMaster ON Майстер.ID\_майстер = AvgCostPerMaster.ID\_майстер

LEFT JOIN

Замовлення ON Майстер.ID\_майстер = Замовлення.ID\_майстер

LEFT JOIN

Модель ON Замовлення.ID\_моделі = Модель.ID\_моделі

ORDER BY

AvgCostPerMaster.Середня\_вартість DESC

LIMIT 3;

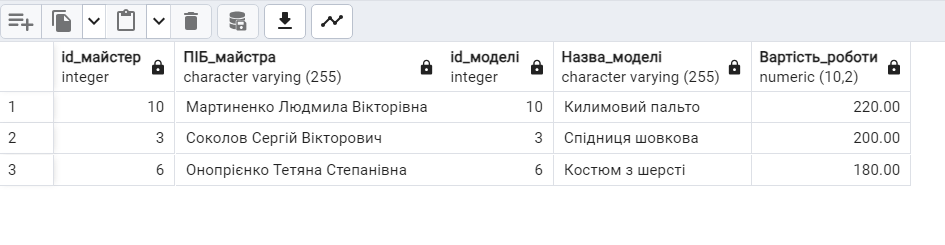


Рисунок.5.14 – Отримання інформаціїпро ціну та модель майстрів з найдорожчим замовленням(Рис.5.15)

5)запит в якому підзапит визначає замовлення в процесі з вартістю роботи більше 100, а зовнішній запит вибирає всі поля цих замовлень та сортує їх за спаданням вартості роботи.

SELECT \*

FROM (

SELECT Замовлення.\*, Модель.Вартість\_роботи

FROM Замовлення

JOIN Модель ON Замовлення.ID\_моделі = Модель.ID\_моделі

WHERE Замовлення.Статус\_замовлення = 'В процесі'

GROUP BY Замовлення.ID\_замовлення, Модель.Вартість\_роботи

HAVING Модель.Вартість\_роботи > 100

) AS Замовлення\_в\_процесі

ORDER BY Замовлення\_в\_процесі.Вартість\_роботи DESC;

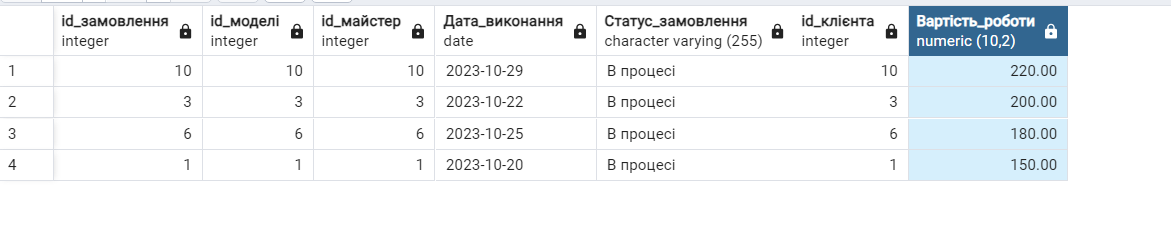


Рисунок.5.15 – Отримання інформації визначає замовлення в процесі та вибирає всі поля цих замовлень та сортує їх

6)запит, який виводить інформацію про майстра з найменшою кількістю замовлень та всі його замовлення, а також клієнтів, які у нього замовляли(Рис.5.16):

WITH МайстерЗамовлення AS (

SELECT

Майстер.ID\_майстер,

COUNT(Замовлення.ID\_замовлення) AS Кількість\_замовлень

FROM

Майстер

LEFT JOIN

Замовлення ON Майстер.ID\_майстер = Замовлення.ID\_майстер

GROUP BY

Майстер.ID\_майстер

)

SELECT

Майстер.ID\_майстер,

Майстер.ПІБ\_майстра,

Майстер.Контактні\_дані\_майстра,

Замовлення.ID\_замовлення,

Замовлення.Дата\_виконання,

Замовлення.Статус\_замовлення,

Клієнт.ID\_клієнта,

Клієнт.ПІБ\_клієнта,

Клієнт.Контактні\_дані

FROM

Майстер

LEFT JOIN

Замовлення ON Майстер.ID\_майстер = Замовлення.ID\_майстер

LEFT JOIN

Клієнт ON Замовлення.ID\_клієнта = Клієнт.ID\_клієнта

WHERE

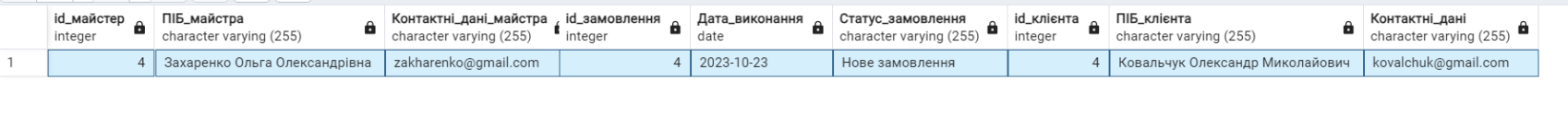
 Майстер.ID\_майстер = (SELECT ID\_майстер FROM МайстерЗамовлення ORDER BY Кількість\_замовлень ASC LIMIT 1);

Рисунок.5.16 – Отримання інформації яка виводить інформацію про майстра його замовлення, і його клієнтів

7)запит, який виводить інформацію про клієнтів, у яких статус замовлення "Нове замовлення", а також виводить ціну їх замовлення та сортує результати за зростанням ціни(Рис.5.17):

SELECT

Клієнт.ID\_клієнта,

Клієнт.ПІБ\_клієнта,

Клієнт.Контактні\_дані,

Замовлення.ID\_замовлення,

Замовлення.Статус\_замовлення,

Модель.Вартість\_роботи AS Ціна\_замовлення

FROM

Клієнт

JOIN

# Замовлення ON Клієнт.ID\_клієнта = Замовлення.ID\_клієнта

# JOIN

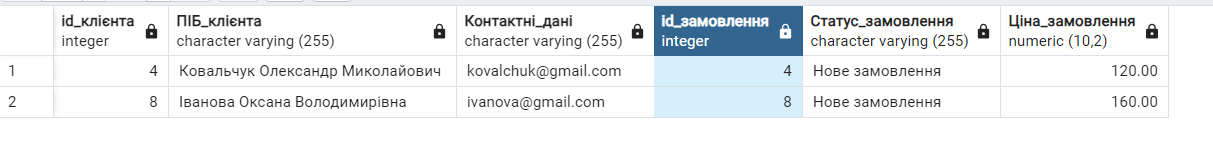
# Модель ON Замовлення.ID\_моделі = Модель.ID\_моделі

# WHERE

# Замовлення.Статус\_замовлення = 'Нове замовлення'

# ORDER BY

# Ціна\_замовлення ASC;

Рисунок.5.17 – Отримання інформації клієнтів, у яких статус замовлення "Нове замовлення"

8)запит який виводить майстрів у яких замовлення виконані та позначимо їх як вільні та виводимо кількість виконаних замовлень цих майстрів та середню ціну за замовлення(Рис.5.18)

SELECT

Майстер.ID\_майстер,

Майстер.ПІБ\_майстра,

COUNT(Замовлення.ID\_замовлення) AS Кількість\_виконаних\_замовлень,

AVG(Модель.Вартість\_роботи) AS Середня\_ціна\_замовлення,

'Вільний' AS Статус\_вільності

FROM

Майстер

INNER JOIN

Замовлення ON Майстер.ID\_майстер = Замовлення.ID\_майстер

INNER JOIN

Модель ON Замовлення.ID\_моделі = Модель.ID\_моделі

WHERE

Замовлення.Статус\_замовлення = 'Виконано'

GROUP BY

# Майстер.ID\_майстер, Майстер.ПІБ\_майстра;

# Рисунок.5.18 – Отримання інформації про вільних майстрів та їх середні ціни

# 9)запит який знайде максимальну кількість фурнітури на складі. Потім виведено замовлення, де використовуються моделі та фурнітура з цією максимальною кількістю(Рис.5.19).

# SELECT

# Замовлення.ID\_замовлення,

# Модель.Назва\_моделі,

# Фурнітура.ID\_Фурнітури,

# Фурнітура.Назва\_Фурнітури,

# Фурнітура.Кількість\_на\_складі

# FROM

# Замовлення

# JOIN

# Замовлення\_моделі ON Замовлення.ID\_замовлення = Замовлення\_моделі.ID\_замовлення

# JOIN

# Модель ON Замовлення\_моделі.ID\_моделі = Модель.ID\_моделі

# JOIN

# Фурнітура\_Моделі ON Модель.ID\_моделі = Фурнітура\_Моделі.ID\_моделі

# JOIN

# Фурнітура ON Фурнітура\_Моделі.ID\_Фурнітура = Фурнітура.ID\_Фурнітури

# WHERE

# Фурнітура.Кількість\_на\_складі = (SELECT MAX(Кількість\_на\_складі) FROM Фурнітура);

# Рисунок.5.19 – Отримання інформації про замовлення, де кількість фурнітури найбільша на складі

# 10)запит для вибору та обмеження три найбільш доступних моделей за ціною в таблиці "Модель"Потім він виводить інформацію про замовлення, в яких використовуються ці моделі, разом з інформацією про майстрів, які виготовили ці замовлення, та виводить результат відсортованим за зростанням ціни моделі(за Рис.5.10).

# WITH Найдешевші\_Моделі AS (

# SELECT

# ID\_моделі,

# Назва\_моделі,

# Вартість\_роботи

# FROM

# Модель

# ORDER BY

# Вартість\_роботи

# LIMIT 3

# )

# SELECT

# Замовлення.ID\_замовлення,

# Модель.Назва\_моделі,

# Модель.Вартість\_роботи AS Ціна\_моделі,

# Майстер.ПІБ\_майстра

# FROM

# Замовлення

# JOIN

# Замовлення\_моделі ON Замовлення.ID\_замовлення = Замовлення\_моделі.ID\_замовлення

# JOIN

# Модель ON Замовлення\_моделі.ID\_моделі = Модель.ID\_моделі

# JOIN

# Майстер ON Замовлення.ID\_майстер = Майстер.ID\_майстер

# JOIN

# Найдешевші\_Моделі ON Модель.ID\_моделі = Найдешевші\_Моделі.ID\_моделі

# ORDER BY

# Ціна\_моделі ASC;

# 

# Рисунок.5.10 – Отримання інформації найбільш доступних моделей за ціною

# СТВОРЕННЯ КОРИСТУВАЧІВ І ПРИЗНАЧЕННЯ ПРАВ ДОСТУПУ

Власник ательє має права доступу до всіх таблиць та зв’язків. Доступна вибірка, оновляння, вставка та видалення таблиць. Тобто має повний доступ до всіх таблиць та послідовністей у схемі public.

Користувач: owner\_atelier

Пароль: 'ownerAtelierr'

SQL-код:

ownerAtelierr

CREATE USER owner\_atelier WITH PASSWORD 'ownerAtelierr';

GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO owner\_atelier;

GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA public TO owner\_atelier;

Майстер ательє має права доступу з правами на вставку і оновлення даних в таблиці «Модель» , «Тканина», «Фурнітура» та Має право тільки на оновлення даних в таблиці «Замовлення». Це означає, що користувач може змінювати існуючі записи в таблиці Замовлення, але не може додавати нові записи.

Користувач: master\_user

Пароль: 'masterpass'

SQL-код:

CREATE USER master\_user WITH PASSWORD 'masterpass';

-- Надаємо права INSERT та UPDATE на конкретні таблиці

GRANT INSERT, UPDATE ON TABLE public.Модель TO master\_user;

GRANT INSERT, UPDATE ON TABLE public.Тканина TO master\_user;

GRANT INSERT, UPDATE ON TABLE public.Фурнітура TO master\_user;

GRANT UPDATE ON TABLE public.Замовлення TO master\_user;

-- Забороняємо доступ до інших таблиць в схемі public

REVOKE ALL PRIVILEGES ON ALL TABLES IN SCHEMA public FROM master\_user;

Продавець ательє має права доступу тільки вибірку даних з таблиці

«Модель», «Клієнт», «Замовлення», «Майстер». Це означає що цей користувач може лише передивляти дані з цих таблиць за цими таблицями що було описано раніше , але не може оновлювати та додавати записи у таблиці.

Користувач : seller\_user\_Atelierr

Пароль : 'sellerpassUsers'

SQL-код:

CREATE USER seller\_user\_Atelierr WITH PASSWORD 'sellerpassUsers';

-- Надаємо право на вибірку тільки з конкретних таблиць

GRANT SELECT ON TABLE public.Модель TO seller\_user\_Atelierr;

GRANT SELECT ON TABLE public.Клієнт TO seller\_user\_Atelierr;

GRANT SELECT ON TABLE public.Замовлення TO seller\_user\_Atelierr;

GRANT SELECT ON TABLE public.Майстер TO seller\_user\_Atelierr;

# -- Забороняємо доступ до інших таблиць в схемі public

# REVOKE ALL PRIVILEGES ON ALL TABLES IN SCHEMA public FROM seller\_user\_Atelierr;

ВИСНОВКИ

У процесі виконання курсової роботи було проведено аналіз, проектування та розробка бази даних для підтримки діяльності ательє моди. Основна мета цієї роботи полягала в створенні ефективної системи для зберігання та обробки інформації про клієнтів, моделі, дизайнерів, матеріали та замовлення. Під час виконання курсової роботи я отримав значущий досвід у розробці баз даних, роботі з СУБД, а також у використанні SQL-запитів та інших інструментів для взаємодії з базою даних.

Виконання проекту поглибило моє розуміння структури баз даних та роботи з СУБД, зокрема PostgreSQL. Я отримав навички проектування таблиць, використання тригерів, функцій та обробки подій у базі даних. Також вдалося реалізувати функціонал для зручного ведення обліку замовлень, слідкування за статусами виробництва та управління інформацією про моделі та використані матеріали.

Отримані результати свідчать про високий рівень ефективності розробленої бази даних для ательє моди. Система дозволяє швидко та ефективно отримувати інформацію про різноманітні аспекти роботи ательє, забезпечуючи зручний доступ до даних. Аналіз негативних аспектів роботи системи показав, що вона вимагає постійного оновлення та підтримки для врахування змін в асортименті тканин, моделей та додаткової функціональності.

Практичне значення розробленої бази даних для ательє моди полягає в полегшенні управління замовленнями, веденні обліку матеріалів та оптимізації виробничих процесів. Система може слугувати універсальним інструментом для ательє моди, дозволяючи зручно вести бізнес, відстежувати модельний асортимент, управляти замовленнями та взаємодіяти з клієнтами.

Отже, виконання курсової роботи дало мені не лише практичні навички в розробці баз даних, але й збагатило мої знання у галузі роботи з інформаційними системами та базами даних. Здобуті в процесі роботи знання та вміння є корисним ресурсом для подальшого розвитку у сфері програмування та адміністрування баз даних.

Перелік використаних джерел

1. Малахов Є.В., Блажко О.А., Глава М.Г. Проектування БД та їх реалізація засобами стандартного SQL та PostgreSQL: Навч. посібник для студ. вищих навч. закладів. – О.: ВМВ, 2012. – 248 с.

2. Глава М.Г. Організація баз даних та знань: Конспект лекцій. [Електронний ресурс] – Режим доступу: https://op.edu.ua/library.

3. Глава М.Г. Методичні вказівки до виконання лабораторних робіт з дисципліни „Організація баз даних та знань “. [Електронний ресурс] – Режим доступу: https://op.edu.ua/library. 4. Глава М.Г. Методичні вказівки до самостійної роботи з дисципліни „Організація баз даних та знань “. [Електронний ресурс] – Режим доступу: <https://op.edu.ua/library>.

5. Басюк Т. М. Основи інформаційних технологій: навч. посібник / Т.М. Басюк, Н.О. Думанський, О.В. Пасічник; за наук. ред. В.В. Пасічника.– Л. : Новий Світ-2000, 2011.– 390 с.

6. Інформаційні системи і технології в економіці: Посiбник для вузiв / В.С. Пономаренко, Р.К. Бутова, I.В. Журавльова та iн.; За ред.В.С.Пономаренка.– К. : Академiя, 2002.– 544 с.

7. Берко А. Ю. Системи баз даних та знань. Кн. 1. Організація баз даних та знань: навч. посібник / А.Ю. Берко, О.М. Верес, В.В. Пасічник; за заг. ред. В.В. Пасічника.– Л.: Магнолія 2006, 2008.– 456 с.