Опис домашнього завдання

1. Переведіть початкову таблицю в першу нормальну форму.
2. Переведіть нові таблиці в другу нормальну форму.
3. Переведіть нові таблиці в третю нормальну форму.
4. Розробіть ER-діаграму отриманих таблиць.
5. Використовуючи ER-діаграму, створіть таблиці в базі даних. Оформіть ці таблиці без конкретних значень, тільки з урахуванням колонок та їхніх зв'язків, вручну або автоматично.

Початкова таблиця

Номер_замовлення	Назва_товару і кількість	Адреса_клієнта	Дата_замовлення	Клієнт
101	Лептоп: 3, Мишка: 2	Хрещатик 1	2023-03-15	Мельник
102	Принтер: 1	Басейна 2	2023-03-16	Шевченко
103	Мишка: 4	Комп'ютерна 3	2023-03-17	Коваленко

Рішення

1. Переведення в Першу Нормальну Форму (1НФ)

💡 Концепція 1HF:

Два головних правила:

1. **Атомарність:** Кожна комірка (cell) таблиці повинна містити одне, неподільне значення.
2. **Відсутність груп, що повторюються:** Усі рядки мають бути унікальними, і не має бути стовпців, які містять "списки" значень.

Аналіз початкової таблиці:

Таблиця одразу порушує 1НФ. Стовпець Назва_товару і кількість є проблемою:

- Він неатомарний.
- У рядку 101 він містить групу значень, що повторюється ("Лептоп: 3" і "Мишка: 2").

Рішення:

Розгортаємо цей стовпець, створюючи новий рядок для кожного окремого товару в замовленні. Також логічно розділяємо "Назву" та "Кількість".

Результат (Таблиця в 1НФ):

Первинний ключ тут стає складеним: (Номер_замовлення, Назва_товару)

Номер_замовлення	Назва_товару	Кількість	Адреса_клієнта	Дата_замовлення	Клієнт

101	Лептоп	3	Хрещатик 1	2023-03-15	Мельник
101	Мишка	2	Хрещатик 1	2023-03-15	Мельник
102	Принтер	1	Басейна 2	2023-03-16	Шевченко
103	Мишка	4	Комп'ютерна 3	2023-03-17	Коваленко

Проблеми, які тепер видно:

Ця таблиця в 1НФ, але вона неефективна. У ній багато надлишкових даних:

- Адреса, дата та ім'я клієнта повторюються для кожного товару в замовленні 101.
- Якщо Мельник змінить адресу, нам доведеться оновити її в *кожному* рядку, що стосується замовлення 101. Це "аномалія оновлення".
- Якщо клієнт Коваленко видалить своє замовлення (103), ми втратимо інформацію про самого клієнта Коваленка. Це "аномалія видалення".

Ці проблеми вирішує 2НФ.

---

2. Переведення в Другу Нормальну Форму (2НФ)

💡 **Концепція 2НФ:**

1. Має бути в 1НФ.
2. Кожен не-ключовий атрибут (стовпець) має повністю залежати від **усього** первинного ключа. (Тобто **жодних часткових залежностей**).

Аналіз 1НФ таблиці:

- Первинний ключ (PK): (Номер_замовлення, Назва_товару).
- Не-ключові атрибути: Кількість, Адреса_клієнта, Дата_замовлення, Клієнт.

Шукаємо часткові залежності:

- Кількість — залежить від **обох** частин ключа? Так. Щоб знати кількість "2", нам потрібен і номер замовлення (101), і назва товару (Мишка). Цей стовпець залишається.
- Адреса_клієнта — залежить від **обох** частин ключа? Ні. Адреса "Хрещатик 1" залежить *тільки* від Номер_замовлення (101). Їй байдуже, "Лептоп" там чи "Мишка". **Це часткова залежність.**
- Дата_замовлення — те саме. Залежить *тільки* від Номер_замовлення. **Часткова залежність.**
- Клієнт — те саме. Залежить *тільки* від Номер_замовлення. **Часткова залежність.**

Рішення:

Розділяємо таблицю на дві, щоб прибрати часткові залежності.

1. **Таблиця Orders (Замовлення):** Містить стовпці, що залежать тільки від Номер_замовлення.
2. **Таблиця Order_Items (Позиції замовлення):** Містить стовпці, що залежать від повного складеного ключа.

Результат (Таблиці в 2НФ):

Таблиця 1: Orders (Замовлення)

(PK: Номер_замовлення)

Номер_замовлення	Адреса_клієнта	Дата_замовлення	Клієнт
101	Хрещатик 1	2023-03-15	Мельник
102	Басейна 2	2023-03-16	Шевченко
103	Комп'ютерна 3	2023-03-17	Коваленко

Таблиця 2: Order_Items

(PK: (Номер_замовлення, Назва_товару))

Номер_замовлення	Назва_товару	Кількість
101	Лептоп	3
101	Мишка	2
102	Принтер	1
103	Мишка	4

Проблеми, які тепер видно:

Стало набагато краще! Але в таблиці Orders все ще є проблема. Якщо клієнт "Мельник" зробить 100 замовлень, його адреса "Хрещатик 1" буде повторена 100 разів. А що, якщо ми хочемо зберігати інформацію про клієнтів, які ще не зробили замовлення?

Це "транзитивна залежність", яку вирішує 3НФ.

---

3. Переведення в Третю Нормальну Форму (3НФ)

💡 **Концепція 3НF:**

1. Має бути в 2НФ.
2. Жоден не-ключовий атрибут не повинен залежати від іншого не-ключового атрибута. (Тобто **жодних транзитивних залежностей**).

Аналіз наших 2НФ таблиць:

- Таблиця Order_Items: Тут все добре. Кількість залежить від PK.
- Таблиця Orders:
  - PK: Номер_замовлення.
  - Атрибути Дата_замовлення, Клієнт, Адреса_клієнта залежать від Номер_замовлення.

- **АЛЕ!** Чи Адреса_клієнта залежить від Клієнт? Так! "Хрещатик 1" — це атрибут "Мельника", а не замовлення 101.
- Ми маємо ланцюжок: Номер_замовлення → Клієнт → Адреса_клієнта.
- Це **транзитивна залежність**.

Рішення:

Виносимо Клієнт та Адреса_клієнта в окрему таблицю Clients.

Зауваження: Так само, Назва_товару (наприклад, "Мишка") повторюється в Order_Items. Це теж можна винести в окрему таблицю Products для повної 3НФ. Це найкраща практика.

Результат (Фінальні таблиці в 3НФ):

Таблиця 1: Clients (Клієнти)

(PK: client_id)

client_id	client_name	address
1	Мельник	Хрещатик 1
2	Шевченко	Басейна 2
3	Коваленко	Комп'ютерна 3

Таблиця 2: Products (Товари)

(PK: product_id)

product_id	product_name
1	Лептоп
2	Мишка
3	Принтер

Таблиця 3: Orders (Замовлення)

order_id(PK)	order_date	client_id (FK)
101	2023-03-15	1
101	2023-03-15	1
102	2023-03-16	2
103	2023-03-17	3

Таблиця 4: Order_Items (Позиції замовлення)

order_id	product_id	quantity
101	1	3
101	2	2
102	3	1
103	2	4

Тепер наша схема чиста:

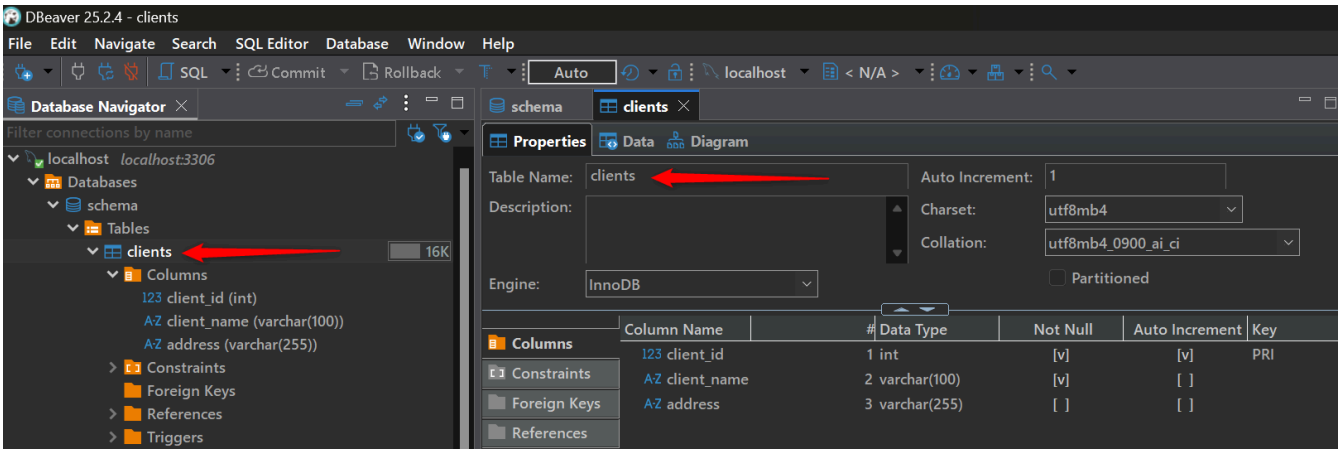
- Немає повторюваних груп (1НФ).
- Немає часткових залежностей (2НФ).
- Немає транзитивних залежностей (3НФ).
- Дані не дублюються, аномалії оновлення/видалення виправлені.

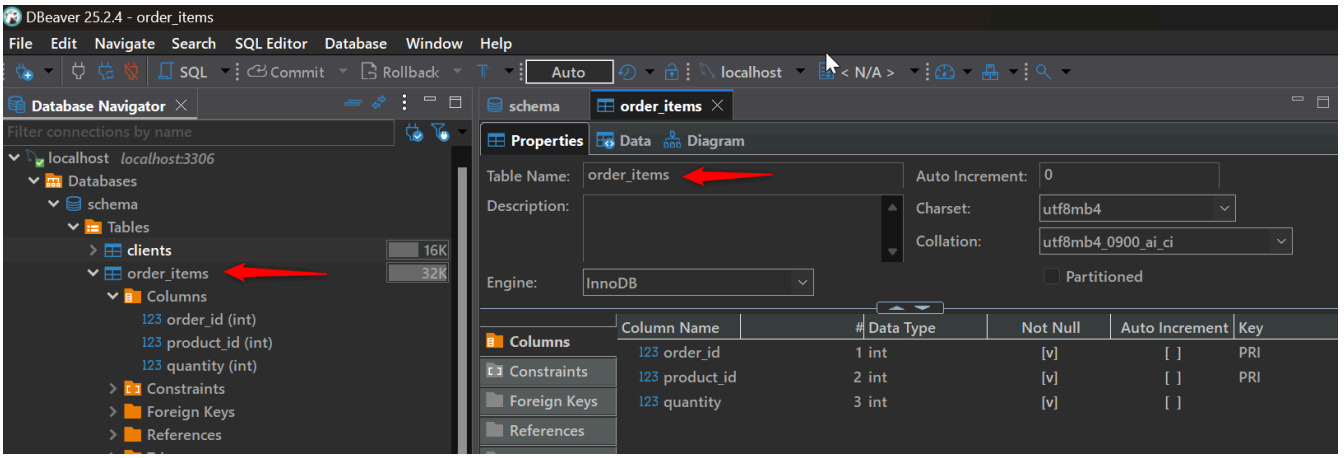
---

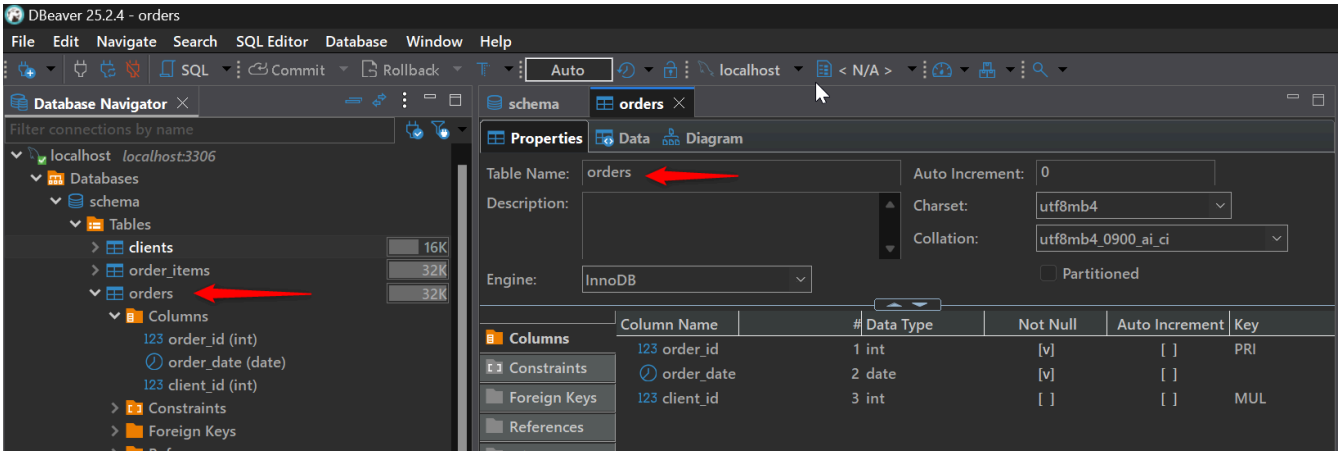
4. ER-діаграма отриманих таблиць

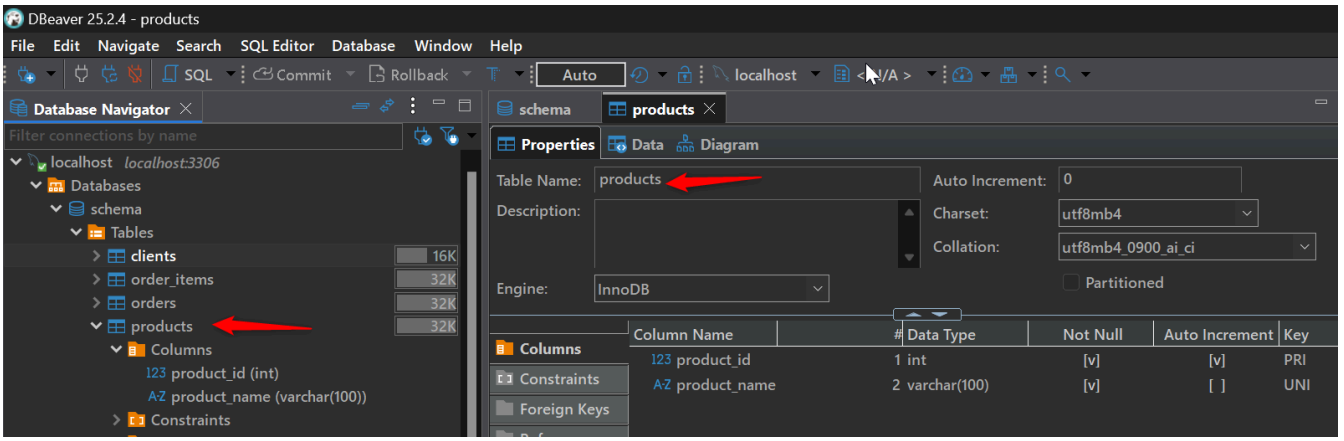
Тепер візуалізуємо зв'язки (кардинальність) між нашими 4 таблицями:

- **Clients** має зв'язок "один-до-багатьох" (1:N) з **Orders**.
  - *Один клієнт може мати багато замовлень, але одне замовлення належить одному клієнту.*
- **Orders** має зв'язок "один-до-багатьох" (1:N) з **Order_Items**.
  - *Одне замовлення може складатися з багатьох позицій (товарів).*
- **Products** має зв'язок "один-до-багатьох" (1:N) з **Order_Items**.
  - *Один продукт може бути в багатьох позиціях замовлень.*









---

5. Створення таблиць в базі даних (DDL-скрипти)

DDL (Data Definition Language) коду на основі 3НФ схеми та ER-діаграми.

SQL

```
-- Створюємо таблицю Клієнтів
CREATE TABLE Clients (
    client_id INT PRIMARY KEY AUTO_INCREMENT, -- AUTO_INCREMENT для автоматичної генерації ID
    client_name VARCHAR(100) NOT NULL,        -- NOT NULL, бо ім'я є обов'язковим
```

```
    address VARCHAR(255)
);

-- Створюємо таблицю Товарів
CREATE TABLE Products (
    product_id INT PRIMARY KEY AUTO_INCREMENT,
    product_name VARCHAR(100) NOT NULL UNIQUE -- UNIQUE, щоб не було двох товарів з
однаковою назвою
    -- сюди можна додати ціну: price DECIMAL(10, 2) NOT NULL
);

-- Створюємо таблицю Замовлень
CREATE TABLE Orders (
    order_id INT PRIMARY KEY,              -- Використовуємо початкові ID (101, 102...)
    order_date DATE NOT NULL,
    client_id INT,                         -- Це буде зовнішній ключ

    FOREIGN KEY (client_id) REFERENCES Clients(client_id)
        ON DELETE SET NULL  -- Якщо клієнта видалять, замовлення залишиться, але без зв'язку
        ON UPDATE CASCADE   -- Якщо ID клієнта зміниться (рідко), він оновиться тут
);

-- Створюємо асоціативну таблицю Позицій Замовлення
CREATE TABLE Order_Items (
    order_id INT,                          -- Частина складеного PK, і також FK
    product_id INT,                        -- Частина складеного PK, і також FK
    quantity INT NOT NULL CHECK (quantity > 0), -- Перевірка, що кількість > 0

    PRIMARY KEY (order_id, product_id), -- Складений первинний ключ

    FOREIGN KEY (order_id) REFERENCES Orders(order_id)
        ON DELETE CASCADE, -- Якщо замовлення видаляється, всі його позиції видаляються
    FOREIGN KEY (product_id) REFERENCES Products(product_id)
        ON DELETE RESTRICT -- Не дозволяємо видалити продукт, якщо на нього є замовлення
);
```