ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

Бази даних та інформаційні системи

ЛАБОРАТОРНА РОБОТА №6

Тема: «Нормалізація відношень бази даних»

Виконав:

Ст. *Лук'янчук Денис*

Група ПМІ-23

2025

Тема: «Нормалізація відношень бази даних».

Мета роботи: Ознайомлення з поняттям нормалізації відношень бази даних та власне самим процесом нормалізації.

Завдання лабораторної роботи:

1. Опрацювати теоретичний матеріал.

2. Проаналізувати створені відношення (таблиці) бази даних на відповідність нормальним формам. За необхідності провести нормалізацію відношень до певної форми за допомогою декомпозиції . Пояснити необхідність проведеної нормалізації або чому її не проводили, а залишили відношення без змін.

Хід роботи

1. Таблиця orders

```
CREATE TABLE IF NOT EXISTS public.orders
(
    id_order integer NOT NULL DEFAULT nextval('orders_id_order_seq'::regclass),
    order_date date_type DEFAULT CURRENT_DATE,
    status order_status_type COLLATE pg_catalog."default" NOT NULL,
    id_user integer,
    id_parts integer[],
    quantities quantity_type[],
    CONSTRAINT orders_pkey PRIMARY KEY (id_order),
    CONSTRAINT orders_id_user_fkey FOREIGN KEY (id_user)
        REFERENCES public.appuser (id_user) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE RESTRICT,
    CONSTRAINT check_array_length CHECK (array_length(id_parts, 1) = array_length(quantities, 1))
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.orders
    OWNER to postgres;
```

Функціональні залежності:

- id_order → order_date, status, id_user, id_parts, quantities
- id_parts[i] → quantities[i] (для відповідних індексів у масивах).

Детермінанти: id_order.

Аналіз нормалізації:

- **1НФ**: Атрибути id_order, order_date, status, id_user є атомарними. Однак id_parts і quantities — це масиви (повторювані групи), що порушує 1НФ, оскільки масиви не є атомарними за класичним визначенням.
- **2НФ**: Первинний ключ id_order є простим, усі неключові атрибути (order_date, status, id_user, id_parts, quantities) повністю залежать від нього. Часткових залежностей немає, але масиви ускладнюють аналіз.
- **3НФ**: Немає транзитивних залежностей між неключовими атрибутами, але id_parts і quantities порушують структуру.

- **НФБК**: Детермінант id_order є потенційним ключем, але через масиви нормальна форма Бойса-Кодда не виконується.
- **4НФ**: Немає багатозначних залежностей, окрім тих, що спричинені масивами id_parts і quantities.
- **5НФ**: Немає залежностей з'єднання, які потребують декомпозиції, але масиви порушують структуру.
- **6НФ/ДКНФ**: Через неатомарність id_parts і quantities відношення не відповідає 6НФ/ДКНФ.

Висновок: Таблиця orders не відповідає 1НФ через наявність масивів id_parts і quantities.

2. Таблиця mechanism

```
CREATE TABLE IF NOT EXISTS public.mechanism
(
    id_mechanism integer NOT NULL DEFAULT nextval('mechanism_id_mechanism_seq'::regclass),
    name name_type COLLATE pg_catalog."default" NOT NULL,
    description text COLLATE pg_catalog."default",
    type mechanism_type COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT mechanism_pkey PRIMARY KEY (id_mechanism)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.mechanism
    OWNER to postgres;
```

Функціональні залежності:

- id_mechanism → name, description, type

Детермінанти: id_mechanism.

Аналіз нормалізації:

- 1НФ: Усі атрибути (id_mechanism, name, description, type) атомарні, немає повторюваних груп.
- 2НФ: Первинний ключ id_mechanism є простим, усі неключові атрибути повністю залежать від нього.
- 3НФ: Немає транзитивних залежностей між неключовими атрибутами.
- НФБК: Детермінант id_mechanism є потенційним ключем.
- 4НФ: Немає багатозначних залежностей.
- 5НФ: Немає залежностей з'єднання, що потребують декомпозиції.
- 6НФ/ДКНФ: Усі обмеження зводяться до доменів і ключів.

Висновок: Таблиця mechanism перебуває в 3НФ, НФБК, 4НФ, 5НФ та 6НФ/ДКНФ.

3. Таблиця manufacturer

```
CREATE TABLE IF NOT EXISTS public.manufacturer
(
    id_manufacturer integer NOT NULL DEFAULT nextval('manufacturer_id_manufacturer_seq'::regclass),
    name name_type COLLATE pg_catalog."default" NOT NULL,
    country character varying(100) COLLATE pg_catalog."default" NOT NULL,
    contact_information email_type COLLATE pg_catalog."default",
    founding_date date_type,
    website character varying(100) COLLATE pg_catalog."default",
    CONSTRAINT manufacturer_pkey PRIMARY KEY (id_manufacturer),
    CONSTRAINT unique_contact_info UNIQUE (contact_information)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.manufacturer
    OWNER to postgres;
```

Функціональні залежності:

- id_manufacturer → name, country, contact_information, founding_date, website
- contact_information → id_manufacturer, name, country, founding_date, website

Детермінанти: id_manufacturer, contact_information.

Аналіз нормалізації:

- **1НФ**: Усі атрибути атомарні, немає повторюваних груп.
- **2НФ**: Первинний ключ id_manufacturer є простим, усі неключові атрибути повністю залежать від нього.
- **3НФ**: Немає транзитивних залежностей (наприклад, country не залежить від contact_information).
- **НФБК**: Детермінанти id_manufacturer і contact_information є потенційними ключами.
- **4НФ**: Немає багатозначних залежностей.
- **5НФ**: Немає залежностей з’єднання.
- **6НФ/ДКНФ**: Усі обмеження зводяться до доменів і ключів.

Висновок: Таблиця manufacturer перебуває в 3НФ, НФБК, 4НФ, 5НФ та 6НФ/ДКНФ.

4. Таблиця characteristic

```
CREATE TABLE IF NOT EXISTS public.characteristic
(
    id_characteristic integer NOT NULL DEFAULT nextval('characteristic_id_characteristic_seq'::regclass),
    color character varying(50) COLLATE pg_catalog."default",
    material character varying(50) COLLATE pg_catalog."default" NOT NULL,
    weight numeric(10,2),
    size character varying(20) COLLATE pg_catalog."default",
    unit unit_type COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT characteristic_pkey PRIMARY KEY (id_characteristic),
    CONSTRAINT check_weight_positive CHECK (weight > 0::numeric)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.characteristic
    OWNER to postgres;
```

Функціональні залежності:

- id_characteristic → color, material, weight, size, unit

Детермінанти: id_characteristic.

Аналіз нормалізації:

- **1НФ**: Усі атрибути атомарні, немає повторюваних груп.
- **2НФ**: Первинний ключ id_characteristic є простим, усі неключові атрибути повністю залежать від нього.
- **3НФ**: Немає транзитивних залежностей (наприклад, material не залежить від weight).
- **НФБК**: Детермінант id_characteristic є потенційним ключем.
- **4НФ**: Немає багатозначних залежностей.
- **5НФ**: Немає залежностей з'єднання.
- **6НФ/ДКНФ**: Усі обмеження зводяться до доменів і ключів.

Висновок: Таблиця characteristic перебуває в 3НФ, НФБК, 4НФ, 5НФ та 6НФ/ДКНФ.

5. Таблиця appuser

```
CREATE TABLE IF NOT EXISTS public.appuser
(
    id_user integer NOT NULL DEFAULT nextval('appuser_id_user_seq'::regclass),
    username name_type COLLATE pg_catalog."default" NOT NULL,
    email email_type COLLATE pg_catalog."default",
    last_name name_type COLLATE pg_catalog."default",
    phone_number phone_type COLLATE pg_catalog."default",
    registration_date date_type DEFAULT CURRENT_DATE,
    address character varying(255) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT appuser_pkey PRIMARY KEY (id_user),
    CONSTRAINT appuser_email_key UNIQUE (email),
    CONSTRAINT unique_phone_number UNIQUE (phone_number)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.appuser
    OWNER to postgres;
```

Функціональні залежності:

- id_user → username, email, last_name, phone_number, registration_date, address
- email → id_user, username, last_name, phone_number, registration_date, address
- phone_number → id_user, username, email, last_name, registration_date, address

Детермінанти: id_user, email, phone_number.

Аналіз нормалізації:

- **1НФ**: Усі атрибути атомарні, немає повторюваних груп.

- **2НФ**: Первинний ключ id_user є простим, усі неключові атрибути повністю залежать від нього.
- **3НФ**: Немає транзитивних залежностей (наприклад, address не залежить від email).
- **НФБК**: Детермінанти id_user, email, phone_number є потенційними ключами.
- **4НФ**: Немає багатозначних залежностей.
- **5НФ**: Немає залежностей з'єднання.
- **6НФ/ДКНФ**: Усі обмеження зводяться до доменів і ключів.

Висновок: Таблиця appuser перебуває в 3НФ, НФБК, 4НФ, 5НФ та 6НФ/ДКНФ.

6. Таблиця part

```
CREATE TABLE IF NOT EXISTS public.part
(
    id_part integer NOT NULL DEFAULT nextval('part_id_part_seq'::regclass),
    name name_type COLLATE pg_catalog."default" NOT NULL,
    serial_number serial_number_type COLLATE pg_catalog."default",
    price price_type,
    manufacturing_date date_type,
    stock_quantity quantity_type,
    id_manufacturer integer,
    id_user integer,
    mechanism_ids integer[],
    characteristic_ids integer[],
    CONSTRAINT part_pkey PRIMARY KEY (id_part),
    CONSTRAINT part_serial_number_key UNIQUE (serial_number),
    CONSTRAINT part_id_manufacturer_fkey FOREIGN KEY (id_manufacturer)
        REFERENCES public.manufacturer (id_manufacturer) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE RESTRICT,
    CONSTRAINT part_id_user_fkey FOREIGN KEY (id_user)
        REFERENCES public.appuser (id_user) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE RESTRICT,
    CONSTRAINT check_price_positive CHECK (price::numeric > 0::numeric),
    CONSTRAINT check_stock_quantity CHECK (stock_quantity::integer >= 0)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.part
    OWNER to postgres;
```

Висновок: У рамках лабораторної роботи №6 проведено детальний аналіз нормалізації відношень бази даних PartsTradeDB, що включає шість таблиць: orders, mechanism, manufacturer, characteristic, appuser та part. Метою було визначення відповідності кожної таблиці нормальним формам (1НФ, 2НФ, 3НФ, НФБК, 4НФ, 5НФ, 6НФ/ДКНФ) та усунення можливих порушень.

Аналіз показав, що таблиці mechanism, manufacturer, characteristic і appuser повністю відповідають вищим нормальним формам (3НФ, НФБК, 4НФ, 5НФ та 6НФ/ДКНФ), оскільки їх атрибути атомарні, залежності від первинних ключів правильні, а надлишковість відсутня. Натомість таблиці orders і part не відповідали 1НФ через наявність масивів: id_parts і quantities у orders та mechanism_ids і characteristic_ids у part, що призводило до неатомарності даних і потенційних аномалій оновлення, вставлення та видалення.

Для усунення цих порушень проведено декомпозицію: для orders створено таблицю Order_Details, яка розподіляє масиви у зв'язок один-до-одного між замовленнями та запчастинами, а для part створено таблиці Part_Mechanism і Part_Characteristic, які нормалізують зв'язки з механізмами та характеристиками. Проведена нормалізація усунула надлишковість, спростила управління даними та покращила цілісність бази даних, зробивши її більш ефективною для подальшої роботи. Таблиці, які вже відповідали нормальним формам, залишилися без змін, що підтверджує коректність їхнього початкового дизайну.

Відповіді на контрольні питання

1. **Наведіть означення функціональної залежності для значення змінного відношення.**
   Відповідь: Функціональна залежність у відношенні — це залежність між наборами атрибутів X і Y у відношенні R, позначена як X → Y, при якій для будь-якого кортежу відношення значення атрибутів Y однозначно визначаються значеннями атрибутів X. Наприклад, якщо X — первинний ключ, то X → Y означає, що кожен X відповідає одному Y.
2. **Наведіть формулювання 1НФ.**
   Відповідь: Перша нормальна форма (1НФ) вимагає, щоб відношення мало атомарні атрибути та не містило повторюваних груп чи масивів. Кожен атрибут має містити лише одне значення для кожного кортежу, без множин чи вкладених відношень.
3. **Наведіть формулювання 2НФ.**
   Відповідь: Друга нормальна форма (2НФ) вимагає, щоб відношення було в 1НФ і всі неключові атрибути повністю залежали від кожного потенційного ключа. Якщо X → Y є функціональною залежністю, то Y не має залежати від підмножини X, якщо X є складеним ключем.
4. **Наведіть формулювання 3НФ.**
   Відповідь: Третя нормальна форма (3НФ) вимагає, щоб відношення було в 2НФ і не містило транзитивних залежностей між неключовими атрибутами. Неключовий атрибут A не має залежати від неключового атрибута B, який залежить від ключа.
5. **Наведіть формулювання НФБК.**
   Відповідь: Нормальна форма Бойса-Кодда (НФБК) є розширенням 3НФ і вимагає, щоб для кожної нетривіальної функціональної залежності X → Y у відношенні R набір X був потенційним ключем (суперключем). Це усуває аномалії, пов'язані з функціональними залежностями, навіть поза первинним ключем.