Контрольне завдання
Створення бази даних інтернет-магазина

Рішення цього завдання представлене на прикладі інтернет-магазина, який торгує ноутбуками.

Завдання 1. Обґрунтування вибору бази даних.

Вибір типу бази даних для сайту залежить від кількох ключових факторів. Ось основні аспекти, які варто враховувати:

1. *Тип і структура даних*

- Реляційна БД (SQL) – підходить, якщо дані мають чітку структуру і зв’язки між таблицями (наприклад, інтернет-магазин, CRM-система).
- Документоорієнтована БД (NoSQL) – якщо працюєте з неструктурованими або напівструктурованими даними (наприклад, блоги, каталоги, динамічні веб-додатки).
- Ключ-значення (Redis, DynamoDB) – добре підходить для кешування та швидких операцій читання-запису.
- Графова БД (Neo4j, ArangoDB) – якщо потрібно зберігати складні зв’язки між об’єктами (наприклад, соціальні мережі, рекомендаційні системи).

2. *Масштабованість та навантаження*

- Горизонтальне масштабування – NoSQL-БД (MongoDB, Cassandra) легше масштабуються.
- Вертикальне масштабування – SQL-БД (MySQL, PostgreSQL) потребують потужніших серверів при зростанні навантаження.
- Частота записів і читання – якщо сайт має багато запитів на читання, кешування (Redis, Memcached) може допомогти.

3. *Продуктивність і швидкодія*

- SQL-БД добре підходять для транзакцій із високою цілісністю даних.
- NoSQL (особливо документоорієнтовані) швидше працюють з великими обсягами нереляційних даних.
- Гібридні рішення (наприклад, використання SQL-БД разом із Redis для кешування) можуть покращити швидкодію.

4. *Гнучкість та розширюваність*

- Чи буде змінюватися структура даних? Якщо часто змінюється – краще NoSQL (MongoDB).
- Чи є складні запити? Якщо потрібні складні JOIN-запити, SQL-БД кращий вибір.

5. *Безпека і цілісність даних*

- SQL-БД (PostgreSQL, MySQL) забезпечують ACID-транзакції, важливі для фінансових операцій.
- NoSQL часто жертвує частиною безпеки заради швидкодії, але є механізми для її підвищення.

6. *Вартість обслуговування*

- Чи потрібно платити за ліцензію? (PostgreSQL – безкоштовний, а Oracle DB – комерційний).
- Вартість серверів і масштабування (NoSQL часто дешевший при великому навантаженні).

Для класичного веб-сайту, блогу або інтернет-магазину найкраще підійде – MSSQL, MySQL, PostgreSQL.

Для сайтів із динамічним контентом та складними зв’язками – MongoDB або інший NoSQL.

Для високонавантажених сайтів – комбінація SQL + NoSQL (наприклад, PostgreSQL + Redis).

Завдання 2. Скласти схему РБД відповідно до вашого завдання.

Схема БД магазину, що торгує ноутбуками надано на рис. 1. У цій базі даних представлено 5 таблиць:

Магазини (shops)

- id - ідентифікатор магазину
- address - адреса магазину

Ноутбуки (notebooks)

- id – ідентифікатор ноутбука
- код - код ноутбука
- name - назва ноутбука
- img - посилання на картинку ноутбука

- specs - json , що зберігає характеристики ноутбука ({характеристика_1: значение_1, характеристика_2: значение_2})
- in_stock - наявність ноутбука
- shop_id - зовнішній ключ на таблицю `shops` по полю `id`
- price - ціна ноутбука

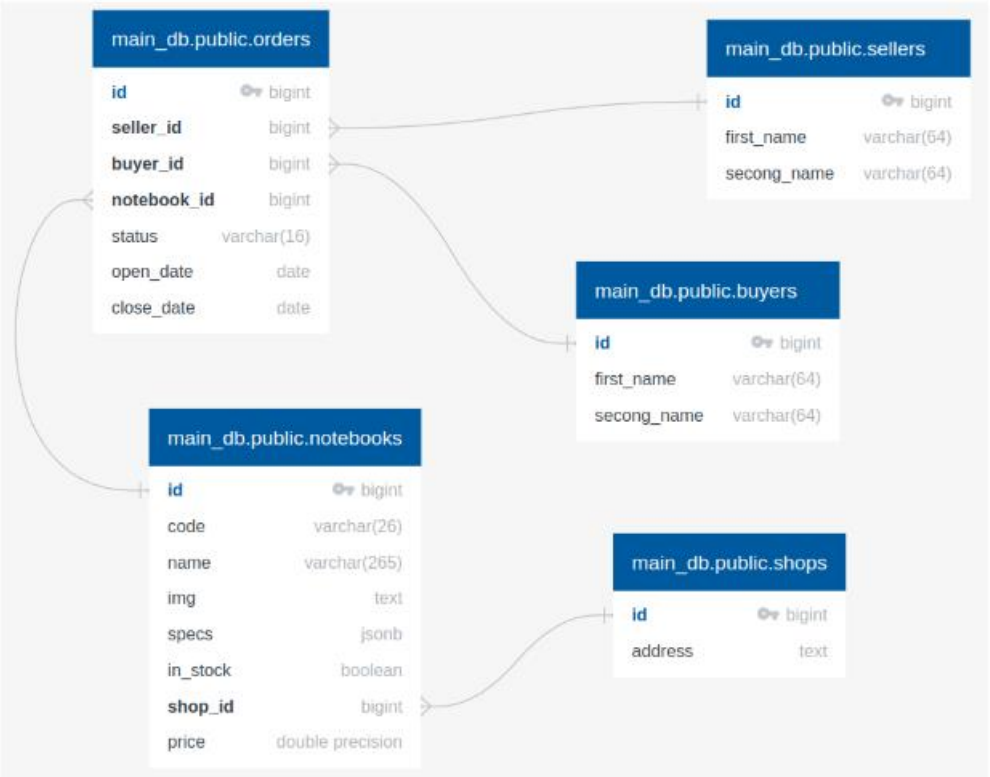

Рис. 1. **Реляційна база даних інтернет-магазина**

Продавці (sellers)

- id - ідентифікатор продавця
- first_name - ім'я продавця
- second_name - прізвище продавця

Покупці (buyers)

- id - ідентифікатор покупця
- first_name - ім'я покупця
- second_name - прізвище покупця

Замовлення (orders)

- id - ідентифікатор замовлення

- seller_id - зовнішній ключ на таблицю ` sellers` по полю ` id`
- buyer_id - зовнішній ключ на таблицю ` buyers` по полю ` id`
- notebook_id - зовнішній ключ на таблицю ` notebooks` по полю ` id`
- status - статус замовлення (наприклад, обробляється, доставляється, закритий, скасований і т.д.)
- open_date - дата відкриття замовлення
- close_date - дата закриття замовлення

Завдання 3. Створити БД за створеною вами схемою у будь-якій СУБД.

```
CREATE TABLE "main_db"."public"."notebooks"(
" id " bigint NOT NULL,
" code " varchar (26) NULL,
" name " varchar (265) NOT NULL,
" img " text NULL,
" specs " jsonb NULL,
" in_stock " boolean NOT NULL,
" shop_id " bigint NOT NULL,
" price " double precision NOT NULL,
CONSTRAINT " pk_notebooks " PRIMARY KEY (
" id "
)
);

CREATE TABLE "main_db"."public"."shops"(
" id " bigint NOT NULL,
" address " text NOT NULL,
CONSTRAINT " pk_shops " PRIMARY KEY (
" id "
)
);

CREATE TABLE "main_db"."public"."sellers"(
" id " bigint NOT NULL,
```

```
    " first_name " varchar (64) NOT NULL,
    " secong_name " varchar (64) NOT NULL,
    CONSTRAINT " pk_sellers " PRIMARY KEY (
    " id "
    )
    );

    CREATE TABLE "main_db"."public"."buyers"(
    " id " bigint NOT NULL,
    " first_name " varchar (64) NOT NULL,
    " secong_name " varchar (64) NOT NULL,
    CONSTRAINT " pk_buyers " PRIMARY KEY (
    " id "
    )
    );

    CREATE TABLE "main_db"."public"."orders"(
    " id " bigint NOT NULL,
    " seller_id " bigint NOT NULL,
    " buyer_id " bigint NOT NULL,
    " notebook_id " bigint NOT NULL,
    " status " varchar (16) NOT NULL,
    " open_date " date NOT NULL,
    " close_date " date NULL,
    CONSTRAINT " pk_orders " PRIMARY KEY (
    " id "
    )
    );

    ALTER TABLE "main_db"."public"."notebooks" ADD
CONSTRAINT " fk_notebooks_shop_id " FOREIGN
KEY("shop_id")
    REFERENCES "main_db"."public"."shops" ("id");

    ALTER TABLE "main_db"."public"."orders" ADD
CONSTRAINT " fk_orders_seller_id " FOREIGN
KEY("seller_id")
    REFERENCES "main_db"."public"."sellers" ("id");
```

```
    ALTER TABLE "main_db"."public"."orders" ADD
CONSTRAINT " fk_orders_buyer_id " FOREIGN KEY("
buyer_id ")
    REFERENCES "main_ db ". " public ". " buyers " ("
id ");

    ALTER TABLE "main_db"."public"."orders" ADD
CONSTRAINT " fk_orders_notebook_id " FOREIGN
KEY("notebook_id ")
    REFERENCES "main_db"."public"."notebooks" ("id");
```

Після виконання завдання в базі даних повинно з'явитися 5 таблиць.

Завдання 4. Заповнити базу даних довільними даними .

Завдання 5. Уявити створену на попередньому кроці БД у нереляційному вигляді, записану в `JSON`.

Для подання створеної на попередньому кроці БД зробимо наступне :

1. Створимо нову БД *notebooks_shop*
2. Створимо 5 колекцій, що зберігають відповідні документи:
    - shops
    - sellers
    - buyers
    - notebooks
    - orders
3. Додамо документи до кожної колекцію відповідно до записів (даними з таблиць) з попереднього кроку.
4. Зробимо кілька тестових запитів

Запит для створення нової БД:

```
use notebooks_shop ;
```

Запит для створення колекцій :

```
db.createCollection (" shops ");
db.createCollection (" sellers ");
db.createCollection (" buyers ");
db.createCollection (" notebooks ");
db.createCollection (" orders ");
```

Запит на додавання документів для кожної колекції :

```
db.shops.insertMany ([
{
_ id : ObjectId (1),
      address : ' вул . Пушкіна , д. Кукушкіно 534 '
}
]);

db.sellers.insertMany ([
{
_ id : ObjectId (1),
      first_name : 'Андрій',
      second_name : ' Полейчук '
},
{
_ id : ObjectId (2),
      first_name : ' Петро ',
      second_name : ' Памужак '
}
]);

db.buyers.insertMany ([
{
_ id : ObjectId (1),
      first_name : 'Андрій',
      second_name : 'Чуйко'
},
{
_ id: ObjectId (2),
```

```
      first_name: 'Владислав',
      second_name: 'Юрченко'
},
{
_ id : ObjectId (3),
      first_name: ' Дмитро ',
      second_name: 'Цуркан'
},
{
_ id : ObjectId (4),
      first_name: 'Степан',
      second_name: ' Квітків '
}
]);

db.notebooks.insertMany ([
{
_ id : ObjectId (1),
      code : 'bdk12kd',
      name : 'Lenovo Legion Y7',
      img : null
      specs : null
      in_stock : true ,
      shop_id : ObjectId (1),
      price : 16000.0,
},
{
_ id : ObjectId (2),
      code : 'bdk123vzxkd',
      name : 'Lenovo Legion Y5',
      img : null
      specs : null
      in_stock : true ,
      shop_id : ObjectId (1),
      price : 8500.0,
}
]);
```

```
db.orders.insertMany ([
{
_ id : ObjectId (1),
      seller_id : ObjectId (1),
      buyer_id : ObjectId (1),
      notebook_id : ObjectId (1),
      status : ' доставляється ',
      open_date : '2023-02-10',
      close_date : null
}
]);
```

Тестові запити:

Отримати список усіх ноутбуків у магазині

```
db.notebooks.find ();
```

Отримати список ноутбуків , які купив `Андрій Чуйко` (незважаючи на те, що у цього покупця при створенні документа в колекції *orders* ми вводили *buyer_id: ObjectId (1)*, MongoDB після виконання запиту на додавання хешує отримане числове значення в *ObjectId('000000014e5c877da6db5375')`)*

```
db.orders.find({buyer_id:ObjectId('000000014e5c877da6db
5375')});
```

Отримати кількість замовлень

```
  db.orders.countDocuments ();
```