МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО
КАФЕДРА АВТОМАТИЗАЦІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

ЗВІТ З ЛАБОРАТОРНИХ РОБІТ
З НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
«СУЧАСНІ ПАРАДИГМИ ЗБЕРЕЖЕННЯ ДАНИХ»

Виконав: студент групи КН-22-1(а)
Панцюк К.В.
Перевірив: асист. каф. Андреєв П.І.

КРЕМЕНЧУК 2025

Лабораторна робота 5

Тема. Операції агрегації

Мета роботи: навчитися створювати агрегаційні запити в MongoDB.

Порядок виконання роботи

1.Отримати індивідуальний варіант завдання.

2.Створити агрегацію з використанняму етапу агрегації $match.

3.Створити агрегацію з використанняму етапу агрегації $group.

4.Створити агрегацію з використанняму етапу агрегації $project.

5.Створити агрегацію з використанняму етапу агрегації $sort.

6.Створити агрегацію з використанняму етапу агрегації $lookup.

7.Створити агрегацію з використанняму етапу агрегації $unwind.

8.Обґрунтувати вибір структури агрегаційних запитів.

Виконання роботи

1.Продовжуємо роботу з базою даних якою створили під час виконання 2-ї лабораторної роботи.

2.Створити агрегацію з використанняму етапу агрегації $match.

```
// AggregationOperations.js
const { MongoClient } = require('mongodb');
const uri = 'mongodb://127.0.0.1:27017/electronicsStore';
const client = new MongoClient(uri);

async function runAggregations() {
    try {
        await client.connect();
        const db = client.db();

        // 1. $match: Знайти всі замовлення з промокодом
        const matchPipeline = [
            { $match: { promoCode: { $exists: true, $ne: null } } }
        ];
        console.log("1. Замовлення з промокодами:");
        const matchResult = await db.collection('orders').aggregate(matchPipeline).toArray();
        console.log(matchResult);
```

Рисунок 5.1–Код та умова з використанням етапу агрегації $match.

Рисунок 5.2–Результат виводу з використанням етапу агрегації $match.

Рисунок 5.3– Результат виводу з використанням етапу агрегації $match.

Продовдення

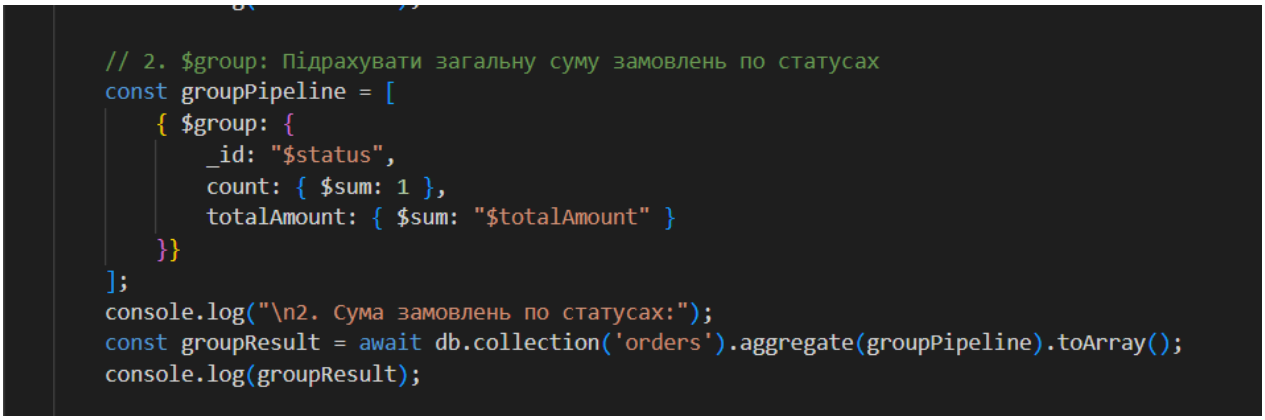

Рисунок 5.4–Код створити агрегації з використанням етапу агрегації $group.

```
2. Сума замовлень по статусах:
[
  { _id: 'delivered', count: 3, totalAmount: 70997 },
  { _id: 'processing', count: 1, totalAmount: 18997 },
  { _id: 'shipped', count: 1, totalAmount: 59998 },
  { _id: 'pending', count: 1, totalAmount: 79998 }
]
```

Рисунок 5.5–Вивід агрегації з використанням етапу агрегації $group.

```
// 3. $project: Вибрати тільки потрібні поля продуктів
const projectPipeline = [
    { $project: {
        name: 1,
        price: 1,
        category: "$categoryId",
        brand: 1,
        isAvailable: { $gt: ["$stock", 0] }
    }}
];
console.log("\n3. Спрощена інформація про продукти:");
const projectResult = await db.collection('products').aggregate(projectPipeline).limit(5).toArray();
console.log(projectResult);
```

Рисунок 5.6–Код агрегації з використанням етапу агрегації $project.

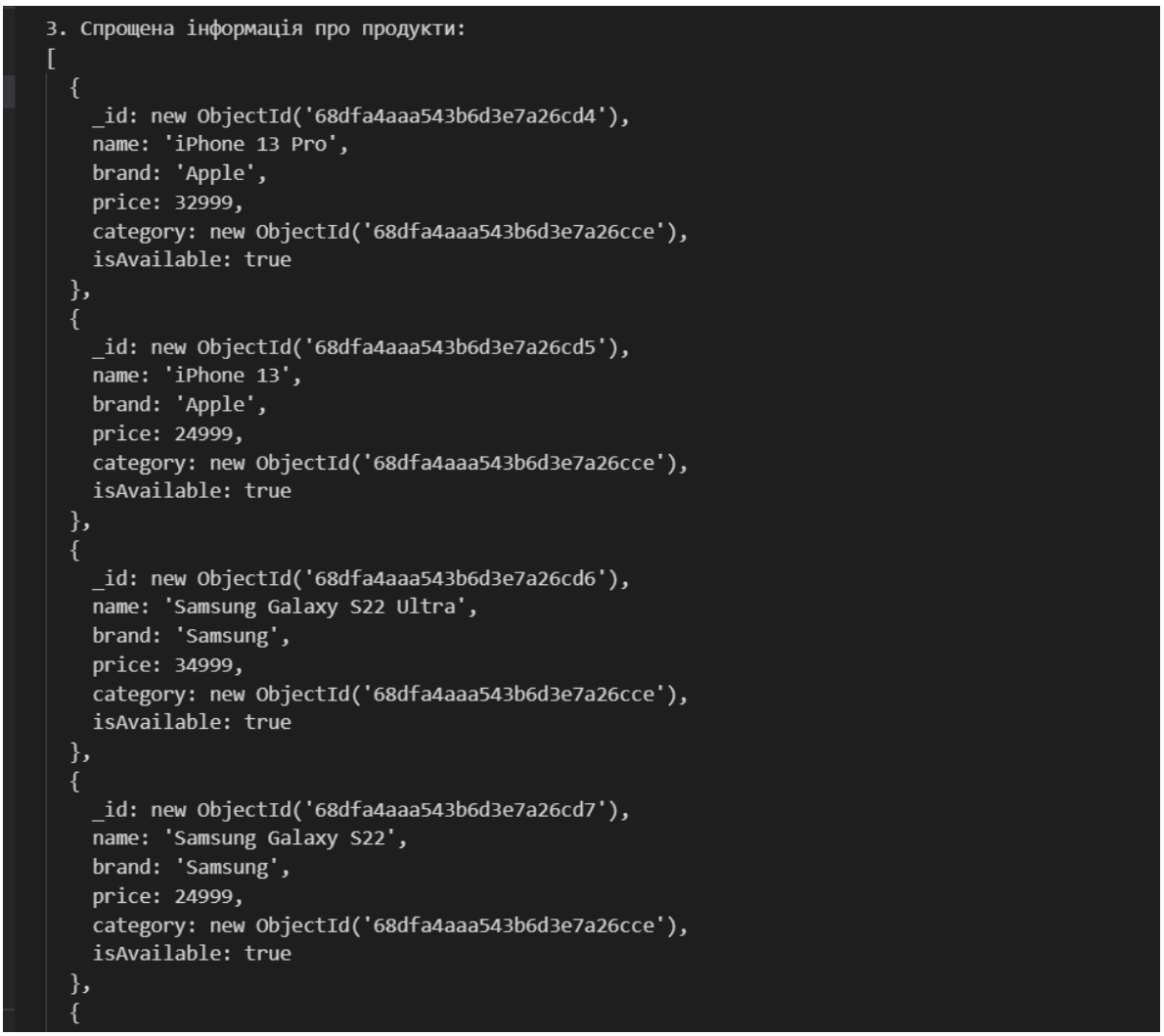

Рисунок 5.7–Результат виводу агрегації з використанням етапу агрегації $project.

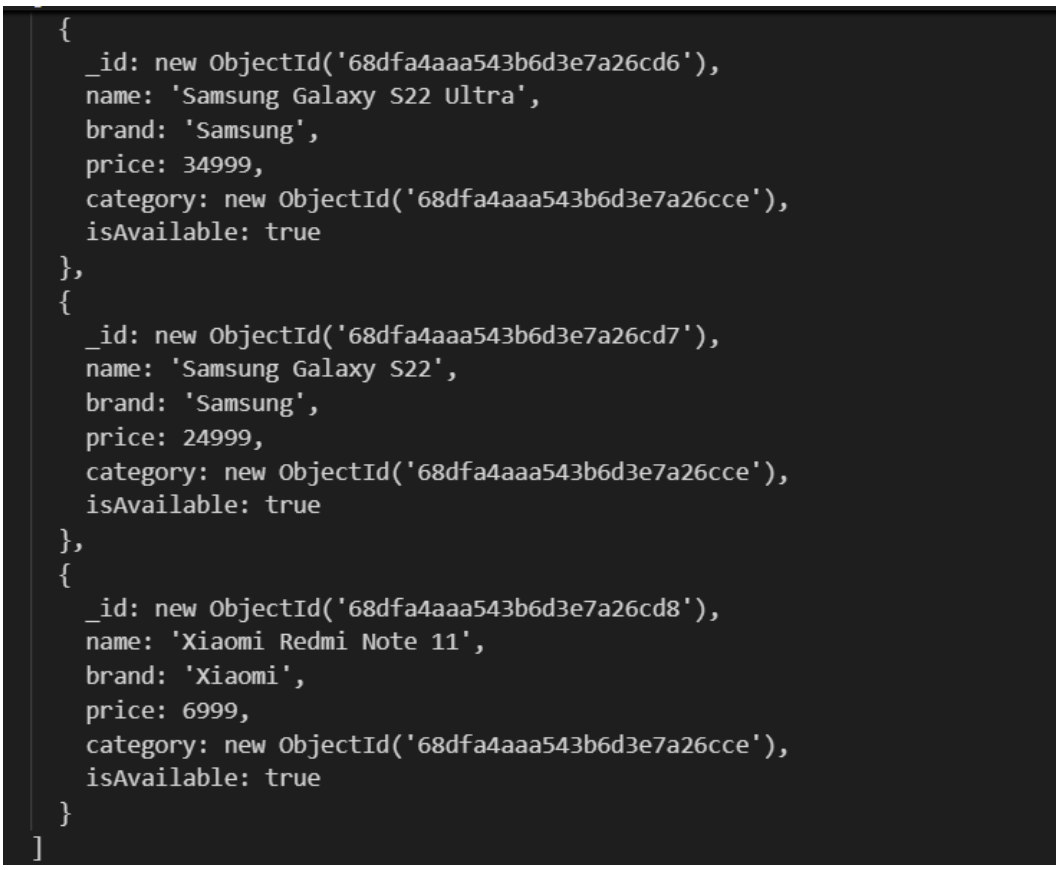

Рисунок 5.8 – Продовження виводу

```
// 4. $sort: Сортувати продукти за ціною (спадінням)
const sortPipeline = [
    { $sort: { price: -1 } },
    { $limit: 5 }
];
console.log("\n4. Найдорожчі продукти:");
const sortResult = await db.collection('products').aggregate(sortPipeline).toArray();
console.log(sortResult);
```

Рисунок 5.9 – Код для створення агрегації з використанням етапу агрегації $sort.

Рисунок 5.10 – Результат виконання агрегації з використанням етапу агрегації $sort.

Рисунок 5.11 – Результат виконання агрегації з використанням етапу агрегації $sort. Продовження

```
    updatedAt: 2023-01-15T00:00:00.000Z
  },
  {
    _id: new ObjectId('68dfa4aaa543b6d3e7a26cdd'),
    name: 'LG OLED TV 55"',
    description: 'Телевізор з OLED екраном та 4K роздільною здатністю',
    sku: 'LGOLED55',
    categoryId: new ObjectId('68dfa4aaa543b6d3e7a26cd0'),
    brand: 'LG',
    price: 29999,
    oldPrice: 31999,
    stock: 4,
    tags: [ 'tv', 'oled', '4k' ],
    isActive: true,
    createdAt: 2023-01-03T00:00:00.000Z,
    updatedAt: 2023-01-08T00:00:00.000Z
  }
]
```

Рисунок 5.12 – Результат виконання агрегації з використанням етапу агрегації $sort. Продовження

```
// 5. $lookup: Об'єднати замовлення з інформацією про користувачів
const lookupPipeline = [
    { $lookup: {
        from: "users",
        localField: "userId",
        foreignField: "_id",
        as: "user"
    }},
    { $unwind: "$user" },
    { $project: {
        orderNumber: 1,
        totalAmount: 1,
        userEmail: "$user.email",
        userName: { $concat: ["$user.firstName", " ", "$user.lastName"] }
    }}
];
console.log("\n5. Замовлення з інформацією про користувачів:");
const lookupResult = await db.collection('orders').aggregate(lookupPipeline).limit(3).toArray();
console.log(lookupResult);
```

Рисунок 5.13–Код для створення агрегації з використанням етапу агрегації $lookup

```
5. Замовлення з інформацією про користувачів:
[
  {
    _id: new ObjectId('68dfa4aaa543b6d3e7a26ce5'),
    orderNumber: 'ORD001',
    totalAmount: 32999,
    userEmail: 'user1@example.com',
    userName: 'Іван Петренко'
  },
  {
    _id: new ObjectId('68dfa4aaa543b6d3e7a26ce6'),
    orderNumber: 'ORD005',
    totalAmount: 7999,
    userEmail: 'user1@example.com',
    userName: 'Іван Петренко'
  },
  {
    _id: new ObjectId('68dfa4aaa543b6d3e7a26ce7'),
    orderNumber: 'ORD002',
    totalAmount: 59998,
    userEmail: 'user2@example.com',
    userName: 'Марія Сидоренко'
  }
]
```

Рисунок 5.14–Результат створення агрегації з використанням етапу агрегації $lookup

```
// 6. $unwind: Розгорнути масив товарів у замовленнях
const unwindPipeline = [
    { $unwind: "$items" },
    { $project: {
        orderNumber: 1,
        productName: "$items.name",
        productPrice: "$items.price",
        quantity: "$items.quantity"
    }}
];
console.log("\n6. Розгорнуті товари в замовленнях:");
const unwindResult = await db.collection('orders').aggregate(unwindPipeline).limit(5).toArray();
console.log(unwindResult);
```

Рисунок 5.15–Код для створення агрегацію з використанняму етапу агрегації $unwind.

```
6. Розгорнуті товари в замовленнях:
[
  {
    _id: new ObjectId('68dfa4aaa543b6d3e7a26ce5'),
    orderNumber: 'ORD001',
    productName: 'iPhone 13 Pro',
    productPrice: 32999,
    quantity: 1
  },
  {
    _id: new ObjectId('68dfa4aaa543b6d3e7a26ce6'),
    orderNumber: 'ORD005',
    productName: 'AirPods Pro',
    productPrice: 7999,
    quantity: 1
  },
  {
    _id: new ObjectId('68dfa4aaa543b6d3e7a26ce7'),
    orderNumber: 'ORD002',
    productName: 'iPhone 13',
    productPrice: 24999,
    quantity: 1
  },
  {
    _id: new ObjectId('68dfa4aaa543b6d3e7a26ce7'),
    orderNumber: 'ORD002',
    productName: 'Samsung Galaxy S22 Ultra',
    productPrice: 34999,
    quantity: 1
  },
  {
    _id: new ObjectId('68dfa4aaa543b6d3e7a26ce8'),
    orderNumber: 'ORD003',
    productName: 'Xiaomi Redmi Note 11',
    productPrice: 6999,
    quantity: 2
  }
```

Рисунок 5.16–Результат попереднього коду

Висновок

6. Використання $match для фільтрації замовлень з промокодами

Обґрунтування:

- Етап $match використовується на початковому етапі агрегації для фільтрації документів, що відповідають певним критеріям.
- У цьому випадку ми відбираємо лише ті замовлення, які мають промокоди (13romocode).
- **Переваги:** аналізувати ефективність використання промокодів

2. Використання $group для аналізу замовлень по статусах

Обґрунтування:

- Етап $group дозволяє об'єднати документи в групи за певним полем (у цьому випадку status)
- Обчислюються агрегатні значення: кількість замовлень (count) та сумарна вартість (totalAmount) для кожного статусу
- **Переваги:**
  - Надає узагальнену інформацію про розподіл замовлень
  - Допомагає в аналізі бізнес-процесів (наприклад, скільки замовлень знаходиться на кожному етапі обробки)
  - Дозволяє виявляти «вузькі місця» в обробці замовлень

3. Використання $project для спрощення структури продуктів

Обґрунтування:

- Етап $project використовується для вибору тільки необхідних полів і створення нових обчислюваних полів
- У цьому випадку ми:
  - Вибираємо тільки основні поля продукту (назва, ціна, категорія, бренд)
  - Додаємо обчислюване поле isAvailable, яке показує наявність товару на складі

- **Переваги:**
  - Зменшує обсяг передаваних даних
  - Покращує читабельність вихідних даних
  - Дозволяє додавати корисну бізнес-логіку (наприклад, перевірку наявності товару)

4. Використання $sort для знаходження найдорожчих продуктів

Обґрунтування:

- Сортування за ціною в порядку спадання з обмеженням на 5 результатів
- **Переваги:**
  - Дозволяє швидко ідентифікувати найдорожчі товари
  - Корисно для аналізу асортименту та цінової політики
  - Обмеження результатів ($limit) покращує продуктивність

5. Використання $lookup для об'єднання замовлень з даними користувачів

Обґрунтування:

- Етап $lookup дозволяє об’єднати дані з різних колекцій
- У цьому випадку ми:
  - З’єднуємо замовлення з даними користувачів
  - Використовуємо $unwind для розгортання масиву користувачів
  - Створюємо зручне для читання ім'я користувача за допомогою $concat
- **Переваги:**
  - Надає більш повну інформацію про замовлення
  - Дозволяє аналізувати поведінку клієнтів
  - Покращує користувацький досвід при перегляді даних

6. Використання $unwind для аналізу товарів у замовленнях

Обґрунтування:

- Розгортання масиву товарів (items) у замовленнях

- **Переваги:**
  - Кожен товар стає окремим документом для подальшого аналізу
  - Дозволяє легше обробляти та аналізувати окремі товари в замовленнях
  - Корисно для аналізу популярності товарів та поведінки покупців

Відповіді на контрольні питання

1. Що робить стадія $match у агрегаційній операції MongoDB і як її правильно використовувати?

Стадія $match використовується для фільтрації документів. Вона дозволяє відібрати документи, які відповідають певним умовам, і передати їх на наступні етапи агрегації. Правильно використовувати $match на ранніх етапах агрегації, щоб зменшити обсяг даних, які потрібно обробити.

2. Як за допомогою $group обчислити сумарне або середнє значення деякого поля в групах документів?

За допомогою $group можна обчислити сумарне значення за допомогою оператора $sum і середнє значення за допомогою оператора $avg.

Наприклад:

```
{ $group: {
   _id: "$category",
   total: { $sum: "$price" },
   average: { $avg: "$price" }
}}
```

3. Для чого використовується стадія $project у пайплайні агрегації, і як нею можна змінювати структуру вихідних документів?

Стадія $project використовується для зміни структури вихідних документів. Вона дозволяє включити або виключити певні поля, а також обчислити нові поля на основі існуючих.

4. Яка роль оператора $lookup в агрегації MongoDB, і як він дозволяє об’єднувати дані між колекціями?

Оператор $lookup використовується для об'єднання даних з інших колекцій. Він дозволяє отримати більш повну інформацію, об'єднуючи документи з різних колекцій.

5. Що робить стадія $unwind у агрегації і в яких випадках її застосовують?

Стадія $unwind використовується для розгортання масивів. Вона застосовується, коли потрібно обробити кожен елемент масиву окремо.

6. Як поєднувати $sort зі стадіями $match і $group у конвеєрі агрегації для формування звітів?

Для формування звітів можна поєднувати $sort з $match і $group наступним чином:

1. Використати $match для фільтрації документів.
2. Використати $group для групування і обчислення агрегатних значень.
3. Використати $sort для сортування результатів.

Наприклад:

```
const pipeline = [
   { $match: { status: 'delivered' } },
   { $group: {
      _id: "$userId",
      totalAmount: { $sum: "$totalAmount" }
   }},
   { $sort: { totalAmount: -1 } }
];
```