

Лабораторна робота №2

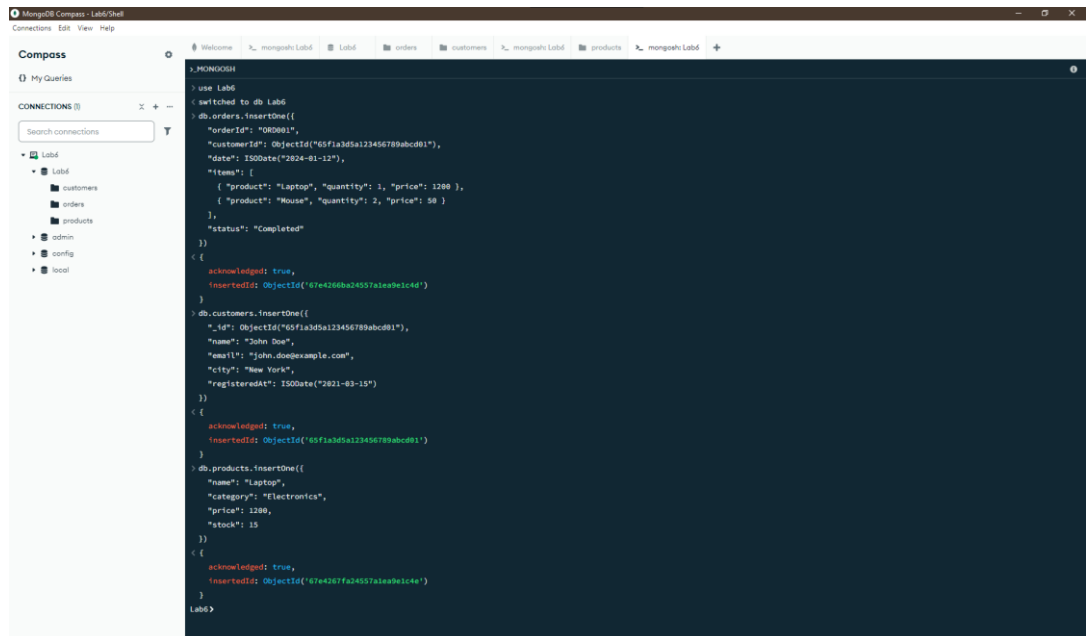
З дисципліни: Бази даних та інформаційні системи
Студента групи МІТ-31: Полюховича А.І

Тема: Практичне використання Aggregation Framework у MongoDB

Мета роботи: Закріпити знання про основні стадії Aggregation Framework. Навчитися будувати ефективні агрегаційні запити. Освоїти методи фільтрації, групування, сортування та обробки масивів у MongoDB. Практично працювати з `$match`, `$group`, `$sort`, `$unwind`, `$lookup`, `$project`. Аналізувати продуктивність агрегацій та оптимізувати запити.

Завдання:

1. Створити колекції `orders`, `customers`, та `products`



2. Частина 1. Базові агрегаційні операції

- Відфільтруйте замовлення за останні 3 місяці

```
> db.orders.aggregate([
  {
    $match: {
      date: { $gte: new Date(new Date().setMonth(new Date().getMonth() - 3)) }
    }
  }
])
```

- Групування замовлень за місяцем

```

> db.orders.aggregate([
  {
    $group: {
      _id: { $month: "$date" },
      totalOrders: { $sum: 1 }
    }
  }
])
< {
  _id: 1,
  totalOrders: 1
}

```

- Сортування за сумою замовлення

```

> db.orders.aggregate([
  {
    $addFields: {
      totalAmount: {
        $sum: {
          $map: {
            input: "$items",
            as: "item",
            in: { $multiply: ["$$item.quantity", "$$item.price"] }
          }
        }
      }
    }
  },
  { $sort: { totalAmount: -1 } }
])
< {
  _id: ObjectId('67e4266ba24557a1ea9e1c4d'),
  orderId: '000001',
  customerId: ObjectId('65f1a3d5a123456789abcd01'),
  date: 2024-01-12T00:00:00.000Z,
  items: [
    {
      product: 'Laptop',
      quantity: 1,
      price: 1200
    },
    {
      product: 'Mouse',
      quantity: 2,
      price: 50
    }
  ],
  status: 'Completed',
  totalAmount: 1300
}

```

3. Частина 2: Робота з масивами

- Розгорніть масив `items` у замовленнях

```
> db.orders.aggregate([
  { $unwind: "$items" }
])
< {
  _id: ObjectId('67e4266ba24557a1ea9e1c4d'),
  orderId: 'ORD001',
  customerId: ObjectId('65f1a3d5a123456789abcd01'),
  date: 2024-01-12T00:00:00.000Z,
  items: {
    product: 'Laptop',
    quantity: 1,
    price: 1200
  },
  status: 'Completed'
}
{
  _id: ObjectId('67e4266ba24557a1ea9e1c4d'),
  orderId: 'ORD001',
  customerId: ObjectId('65f1a3d5a123456789abcd01'),
  date: 2024-01-12T00:00:00.000Z,
  items: {
    product: 'Mouse',
    quantity: 2,
    price: 50
  },
  status: 'Completed'
}
```

- Підрахуйте кількість проданих одиниць товарів

```
> db.orders.aggregate([
  { $unwind: "$items" },
  {
    $group: {
      _id: "$items.product",
      totalSold: { $sum: "$items.quantity" }
    }
  }
])
< {
  _id: 'Laptop',
  totalSold: 1
}
{
  _id: 'Mouse',
  totalSold: 2
}
```

4. Частина 3: З'єднання колекцій (\$lookup)

- Отримання інформації про клієнтів у замовленнях

```

> db.orders.aggregate([
  {
    $lookup: {
      from: "customers",
      localField: "customerId",
      foreignField: "_id",
      as: "customer_info"
    }
  }
])
< {
  _id: ObjectId('67e4266ba24557a1ea9e1c4d'),
  orderId: '0RD001',
  customerId: ObjectId('65f1a3d5a123456789abcd01'),
  date: 2024-01-12T00:00:00.000Z,
  items: [
    {
      product: 'Laptop',
      quantity: 1,
      price: 1200
    },
    {
      product: 'Mouse',
      quantity: 2,
      price: 50
    }
  ],
  status: 'Completed',
  customer_info: [
    {
      _id: ObjectId('65f1a3d5a123456789abcd01'),
      name: 'John Doe',
      email: 'john.doe@example.com',
      city: 'New York',
      registeredAt: 2021-03-15T00:00:00.000Z
    }
  ]
}

```

- Визначте найбільш активних клієнтів

```

> db.orders.aggregate([
  {
    $group: {
      _id: "$customerId",
      totalOrders: { $sum: 1 }
    }
  },
  { $sort: { totalOrders: -1 } },
  { $limit: 5 }
])
< {
  _id: ObjectId('65f1a3d5a123456789abcd01'),
  totalOrders: 1
}

```

5. Частина 4: Оптимізація запитів

- Перевірте продуктивність запиту

```

> db.orders.explain("executionStats").aggregate([
  { $match: { status: "Completed" } }
])
< {
  explainVersion: '1',
  queryPlanner: {
    namespace: 'Lab6.orders',
    parsedQuery: {
      status: {
        '$eq': 'Completed'
      }
    },
    indexFilterSet: false,
    queryHash: '5D6543D9',
    planCacheShapeHash: '5D6543D9',
    planCacheKey: '405CB45D',
    optimizationTimeMillis: 0,
    optimizedPipeline: true,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'COLLSCAN',
      filter: {
        status: {
          '$eq': 'Completed'
        }
      },
      direction: 'forward'
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 1,
    executionTimeMillis: 1,
    totalKeysExamined: 0,
    totalDocsExamined: 1,
    executionStages: {
      isCached: false,
      stage: 'COLLSCAN',
      filter: {
        status: {
          '$eq': 'Completed'
        }
      },
      nReturned: 1,
      executionTimeMillisEstimate: 0,
      works: 2,
      advanced: 1,
      needTime: 0,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      direction: 'forward',
      docsExamined: 1
    }
  },
  queryShapeHash: '1DB714845DB57A135C73C4BE447B26F5A67C0E2D0453FB7AB21F6496CA1ECE24',
  command: {
    aggregate: 'orders',
    pipeline: [
      {
        '$match': {
          status: 'Completed'
        }
      }
    ]
  },

```

```

serverInfo: {
  host: 'DESKTOP-F3ENC70',
  port: 27017,
  version: '8.0.5',
  gitVersion: 'cb9e2e5e552ee39dea1e39d7859336456d0c9820'
},
serverParameters: {
  internalQueryFacetBufferSizeBytes: 104857600,
  internalQueryFacetMaxOutputDocSizeBytes: 104857600,
  internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
  internalDocumentSourceGroupMaxMemoryBytes: 104857600,
  internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
  internalQueryProhibitBlockingMergeOnMongoS: 0,
  internalQueryMaxAddToSetBytes: 104857600,
  internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600,
  internalQueryFrameworkControl: 'trySbeRestricted',
  internalQueryPlannerIgnoreIndexWithCollationForRegex: 1
},
ok: 1
}

```

- Оптимізуйте агрегаційний запит
(Оптимізація через індексацію:)

```

> db.orders.createIndex({ date: 1 })
< date_1

```

6. Додаткові завдання

- Визначте категорії товарів із найбільшою кількістю продажів

```

> db.orders.aggregate([
  { $unwind: "$items" },
  {
    $lookup: {
      from: "products",
      localField: "items.product",
      foreignField: "name",
      as: "product_info"
    }
  },
  { $unwind: "$product_info" },
  {
    $group: {
      _id: "$product_info.category",
      totalSold: { $sum: "$items.quantity" }
    }
  },
  { $sort: { totalSold: -1 } }
])
< {
  _id: 'Electronics',
  totalSold: 1
}

```

- Розрахуйте середню ціну товарів у кожній категорії

```
> db.products.aggregate([
  {
    $group: {
      _id: "$category",
      avgPrice: { $avg: "$price" }
    }
  }
])
< {
  _id: 'Electronics',
  avgPrice: 1200
}
```

- Знайдіть користувачів, які зробили більше одного замовлення

```
> db.orders.aggregate([
  {
    $group: {
      _id: "$customerId",
      orderCount: { $sum: 1 }
    }
  },
  { $match: { orderCount: { $gt: 1 } } }
])
<
```

Висновки:

Під час лабораторної роботи було розглянуто основи роботи з MongoDB, зокрема вставку даних, агрегацію та аналіз продуктивності запитів. Виконаний аналіз показав, що запити без індексів використовують повне сканування колекції (COLLSCAN), що може бути неефективним для великих наборів даних. Для оптимізації рекомендовано створення індексів, що дозволить прискорити вибірку та зменшити навантаження на базу даних.