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 6

Тема. Часові ряди в MongoDB

Мета роботи: навчитися створювати колекції часових рядів та агрегаційні запити до них в MongoDB.

Порядок виконання роботи

1. Отримати індивідуальний варіант завдання.
2. Створити колекції часових рядів.
3. Створити агрегаційні запити до колекцій часових рядів.
4. Обґрунтувати створення колекцій часових рядів.
5. Обґрунтувати створення агрегаційних запитів до колекцій часових рядів.

Виконання роботи

1.Відповідно до розробленої бази даних у 2-й лабораторній роботі продовжуємо роботу з нею.

2.Створення колекцій часових рядів

```
/**
 * 1. Створення колекцій часових рядів
 * Завдання: Створити дві спеціалізовані колекції для аналізу даних
 */
async function createTimeSeriesCollections() {
    try {
        await client.connect();
        const db = client.db();
        // 1.1 Колекція user_activity_log
        // Призначення: Відстеження дій користувачів на сайті
        // Особливості:
        // - Гранулярність 'hours' (години) - достатньо для аналізу поведінки
        // - TTL 30 днів - оптимальний період для аналізу поведінки користувачів
        // - Дозволяє виявляти пікові періоди активності
        // - Корисно для оптимізації маркетингових кампаній
        await db.createCollection('user_activity_log', {
            timeseries: {
                timeField: 'timestamp',  // Поле з часовими мітками
                metaField: 'userId',     // Поле з ідентифікатором користувача
                granularity: 'hours'     // Рівень деталізації
            },
            expireAfterSeconds: 2592000 // 30 днів в секундах
        });
```

Рисунок 6.1–Код створення колекції user_activity_log

```
        // 1.2 Колекція sales_transactions
        // Призначення: Моніторинг продажів в реальному часі
        // Особливості:
        // - Гранулярність 'minutes' (хвилини) - для детального аналізу
        // - TTL 7 днів - достатньо для оперативного аналізу
        // - Дозволяє відстежувати динаміку продажів
        // - Корисно для управління запасами та логістикою
        await db.createCollection('sales_transactions', {
            timeseries: {
                timeField: 'timestamp',  // Поле з часовими мітками
                metaField: 'productId',  // Поле з ідентифікатором продукту
                granularity: 'minutes'  // Більш висока деталізація
            },
            expireAfterSeconds: 604800 // 7 днів в секундах
        });
        console.log('Колекцію sales_transactions створено');
    } catch (error) {
        console.error('Помилка при створенні колекцій:', error);
    } finally {
        await client.close();
    }
```

Рисунок 6.2–Код для створення колекції sales_transactions

```
/**
 * 2. Заповнення колекцій тестовими даними
 * Завдання: Заповнити колекції реалістичними даними для аналізу
 */
async function populateCollections() {
    try {
        await client.connect();
        const db = client.db();
        // 2.1 Очистка колекцій перед заповненням
        await db.collection('user_activity_log').deleteMany({});
        await db.collection('sales_transactions').deleteMany({});
        console.log('Колекції очищено');
        // 2.2 Заповнення колекції user_activity_log
        // Формат документів:
        // - timestamp: Час дії користувача
        // - action: Тип дії (login, view_product, add_to_cart, checkout)
        // - userId: Ідентифікатор користувача
        // - metadata: Додаткові дані (пристрій, локація, тощо)
        await db.collection('user_activity_log').insertMany([
            {
                timestamp: new Date('2023-05-01T08:00:00Z'),
                action: 'login',
                userId: 'user1',
                metadata: { device: 'mobile', location: 'Kyiv' }
```

Рисунок 6.3–Заповнення колекцій даними

Рисунок 6.4–Продовження коду для заповнення даними

Рисунок 6.5–Код заповнення колекції sales_transactions

```
            metadata: { category: 'electronics', region: 'west' }
        },
        {
            timestamp: new Date('2023-05-01T09:30:00Z'),
            amount: 2999,
            productId: 'prod2',
            metadata: { category: 'appliances', region: 'east' }
        },
        {
            timestamp: new Date('2023-05-01T10:00:00Z'),
            amount: 999,
            productId: 'prod3',
            metadata: { category: 'accessories', region: 'center' }
        },
        {
            timestamp: new Date('2023-05-01T10:30:00Z'),
            amount: 1499,
            productId: 'prod1',
            metadata: { category: 'electronics', region: 'west' }
        }
    ]);
    console.log('Колекцію sales_transactions заповнено');
} catch (error) {
```

Рисунок 6.6– Продовження коду заповнення колекції sales_transactions

```
Ця програма демонструє використання колекцій часових рядів для аналізу даних електронного магазину

Колекцію user_activity_log створено
Колекцію sales_transactions створено
Колекції очищено
Колекцію user_activity_log заповнено
Колекцію sales_transactions заповнено
```

Рисунок 6.7–Результат виконання коду наведеного вище

3. Створити агрегаційні запити до колекцій часових рядів.

```
/**
 * 3. Агрегаційні запити до колекцій
 * Завдання: Виконати 5 аналітичних запитів для бізнес-аналізу
 */
async function runAnalytics() {
    try {
        await client.connect();
        const db = client.db();
        // 3.1 Активність користувачів по годинах
        // Призначення: Виявлення пікових періодів активності
        // Користь:
        // - Допомагає планувати серверні ресурси
        // - Корисно для оптимізації маркетингових кампаній
        const activityByHour = [
            {
                $group: {
                    _id: {
                        userId: "$userId",
                        hour: { $dateToString: { format: "%Y-%m-%d %H:00", date: "$timestamp" } }
                    },
                    actions: { $push: "$action" },
                    count: { $sum: 1 }
                }
            },
            { $sort: { "_id.hour": 1 } }
        ];
        console.log("\n3.1 Активність користувачів по годинах:");
        const activityResult = await db.collection('user_activity_log').aggregate(activityByHour).toArray();
        console.log(activityResult);
```

Рисунок 6.8–Код для створення агрегаційних запитів активності користувачів по годинах

```
    // 3.2 Продажі по годинах
    // Призначення: Аналіз динаміки продажів
    // Користь:
    // - Допомагає в плануванні маркетингових акцій
    // - Корисно для управління персоналом
    const salesByHour = [
        {
            $group: {
                _id: {
                    $dateToString: { format: "%Y-%m-%d %H:00", date: "$timestamp" }
                },
                totalSales: { $sum: "$amount" },
                transactions: { $sum: 1 }
            }
        },
        { $sort: { "_id": 1 } }
    ];
    console.log("\n3.2 Продажі по годинах:");
    const salesResult = await db.collection('sales_transactions').aggregate(salesByHour).toArray();
    console.log(salesResult);
```

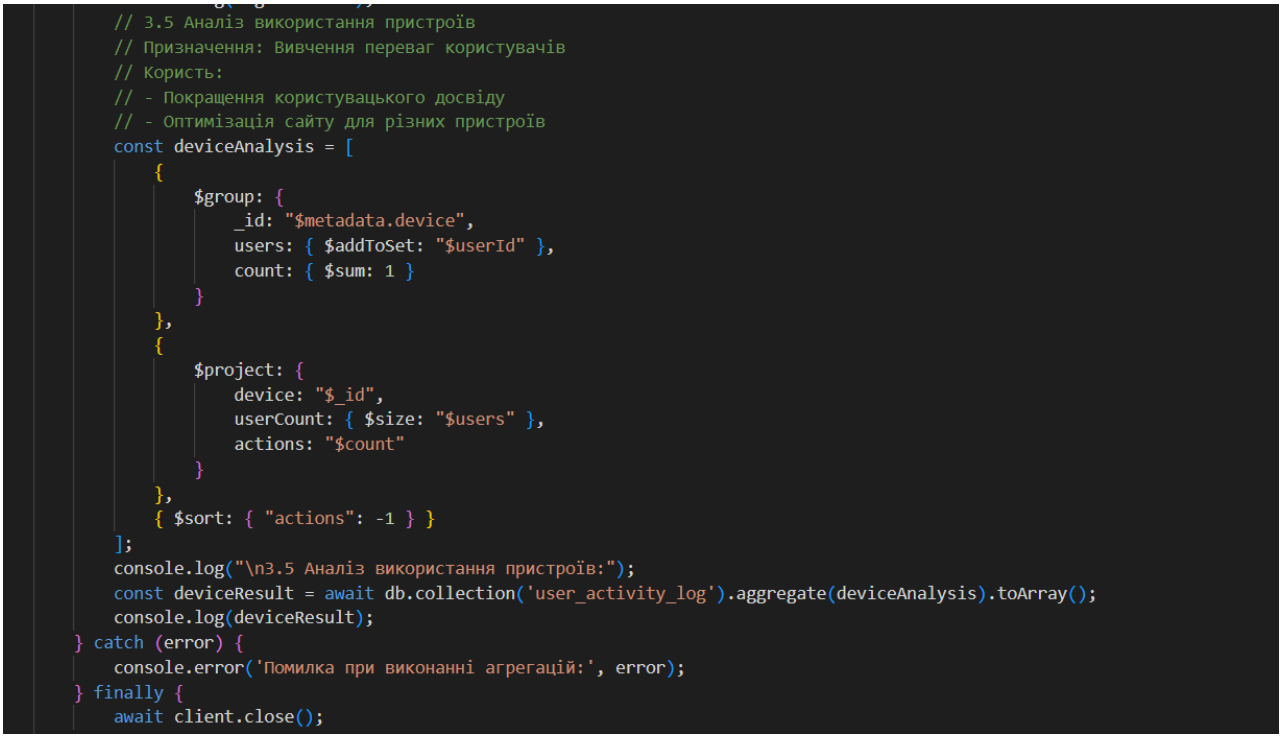
Рисунок 6.9–Код для створення агрегаційного запиту продажу по годинам

```
console.log(salesResult);
// 3.3 Найпопулярніші продукти
// Призначення: Визначення лідируючих товарів
// Користь:
// - Оптимізація асортименту
// - Управління запасами
const popularProducts = [
    {
        $group: {
            _id: "$productId",
            totalSales: { $sum: "$amount" },
            transactions: { $sum: 1 }
        }
    },
    { $sort: { "totalSales": -1 } },
    { $limit: 3 }
];
console.log("\n3.3 Найпопулярніші продукти:");
const productsResult = await db.collection('sales_transactions').aggregate(popularProducts).toArray();
console.log(productsResult);
```

Рисунок 6.10–Код для створення агрегаційного запиту найпопулярніших товарів

```
// 3.4 Продажі по регіонах
// Призначення: Географічний аналіз продажів
// Користь:
// - Оптимізація логістики
// - Регіональні маркетингові стратегії
const salesByRegion = [
    {
        $group: {
            _id: "$metadata.region",
            totalAmount: { $sum: "$amount" },
            productsSold: { $sum: 1 }
        }
    },
    { $sort: { "totalAmount": -1 } }
];
console.log("\n3.4 Продажі по регіонах:");
const regionResult = await db.collection('sales_transactions').aggregate(salesByRegion).toArray();
console.log(regionResult);
```

Рисунок 6.11–Код для створення агрегаційного запиту продажу по регіонах



```
    // 3.5 Аналіз використання пристроїв
    // Призначення: Вивчення переваг користувачів
    // Користь:
    // - Покращення користувацького досвіду
    // - Оптимізація сайту для різних пристроїв
    const deviceAnalysis = [
        {
            $group: {
                _id: "$metadata.device",
                users: { $addToSet: "$userId" },
                count: { $sum: 1 }
            }
        },
        {
            $project: {
                device: "$_id",
                userCount: { $size: "$users" },
                actions: "$count"
            }
        },
        { $sort: { "actions": -1 } }
    ];
    console.log("\n3.5 Аналіз використання пристроїв:");
    const deviceResult = await db.collection('user_activity_log').aggregate(deviceAnalysis).toArray();
    console.log(deviceResult);
} catch (error) {
    console.error('Помилка при виконанні агрегацій:', error);
} finally {
    await client.close();
```

Рисунок 6.12 – Код для створення агрегаційного запиту використання пристроїв

```
3.1 Активність користувачів по годинах:
[
  {
    _id: { userId: 'user1', hour: '2023-05-01 08:00' },
    actions: [ 'login' ],
    count: 1
  },
  {
    _id: { userId: 'user2', hour: '2023-05-01 08:00' },
    actions: [ 'login' ],
    count: 1
  },
  {
    _id: { userId: 'user1', hour: '2023-05-01 09:00' },
    actions: [ 'view_product' ],
    count: 1
  },
  {
    _id: { userId: 'user1', hour: '2023-05-01 10:00' },
    actions: [ 'add_to_cart' ],
    count: 1
  },
  {
    _id: { userId: 'user2', hour: '2023-05-01 11:00' },
    actions: [ 'checkout' ],
    count: 1
  }
]
```

Рисунок 6.13–Результат виводу першої агрегації

```
3.2 Продажі по годинах:
[
  { _id: '2023-05-01 09:00', totalSales: 4998, transactions: 2 },
  { _id: '2023-05-01 10:00', totalSales: 2498, transactions: 2 }
]

3.3 Найпопулярніші продукти:
[
  { _id: 'prod1', totalSales: 3498, transactions: 2 },
  { _id: 'prod2', totalSales: 2999, transactions: 1 },
  { _id: 'prod3', totalSales: 999, transactions: 1 }
]

3.4 Продажі по регіонах:
[
  { _id: 'west', totalAmount: 3498, productsSold: 2 },
  { _id: 'east', totalAmount: 2999, productsSold: 1 },
  { _id: 'center', totalAmount: 999, productsSold: 1 }
]

3.5 Аналіз використання пристроїв:
[
  { _id: 'mobile', device: 'mobile', userCount: 1, actions: 3 },
  { _id: 'desktop', device: 'desktop', userCount: 1, actions: 2 }
]
```

Рисунок 6.14–Результат виводу агрегації з 2-ї по 5-ту

4. Обґрунтувати створення колекцій часових рядів.

Колекція user_activity_log

Призначення: Відстеження дій користувачів на сайті електронного магазину з метою аналізу поведінки та оптимізації користувацького досвіду.

Обґрунтування вибору параметрів:

Гранулярність 'hours':

Дозволяє отримати детальну картину активності користувачів протягом дня

Достатньо для виявлення пікових періодів активності (ранок, день, вечір)

Не створює надмірного навантаження на систему зберігання

Оптимальна для аналізу щоденних трендів поведінки користувачів

TTL 30 днів:

Період достатній для аналізу довгострокових трендів поведінки

Дозволяє виявляти сезонні коливання активності

Економно використовує дисковий простір завдяки автоматичному видаленню застарілих даних

Відповідає типовому циклу аналізу поведінки користувачів в e-commerce

Поля timeField, metaField і granularity:

timestamp (timeField): Точний час дії користувача для аналізу часових трендів

userId (metaField): Ідентифікатор користувача для сегментації та персоналізації

Гранулярність 'hours': Оптимальний баланс між деталізацією та обсягом даних

Переваги використання:

Автоматичне індексування поля timestamp для швидких запитів

Компактне зберігання даних порівняно з регулярними колекціями

Ефективне виконання часових запитів та агрегацій

Автоматичне видалення застарілих даних без додаткового коду

Бізнес-цілінність:

Допомагає оптимізувати маркетингові кампанії

Дозволяє виявляти проблеми в користувацькому досвіді

Корисно для планування серверних ресурсів

Дозволяє аналізувати шлях користувача від входу до покупки

Колекція sales_transactions

Призначення: Моніторинг продажів в реальному часі для оперативного прийняття бізнес-рішень.

Обґрунтування вибору параметрів:

Гранулярність 'minutes':

Необхідна для детального аналізу динаміки продажів

Дозволяє відстежувати миттєві зміни в продажах

Критично важлива для оперативного реагування на зміни ринку

Дозволяє аналізувати вплив маркетингових акцій в реальному часі

TTL 7 днів:

Період достатній для аналізу тижневих трендів продажів

Забезпечує актуальність даних для швидкого прийняття рішень

Економно використовує ресурси завдяки автоматичному очищенню

Відповідає циклу оперативного управління запасами

Поля timeField, metaField і granularity:

timestamp (timeField): Точний час транзакції для аналізу динаміки

productId (metaField): Ідентифікатор продукту для аналізу асортименту

Гранулярність 'minutes': Висока деталізація для точного аналізу

Переваги використання:

Оптимізоване зберігання великих обсягів транзакційних даних

Швидке виконання запитів за часом та продуктом

Автоматичне управління життєвим циклом даних

Можливість інтеграції з системами оперативного моніторингу

Бізнес-цілінність:

Дозволяє оперативно реагувати на зміни попиту

Корисно для управління запасами та логістикою

Допомагає в плануванні маркетингових акцій

Дозволяє аналізувати ефективність цінової політики

5. Обґрунтування створення агрегаційних запитів

1. Активність користувачів по годинах

Призначення: Виявлення пікових періодів активності користувачів на сайті.

Обґрунтування:

Дозволяє оптимізувати серверні ресурси відповідно до навантаження

Корисно для планування технічного обслуговування в періоди мінімальної активності

Допомагає в оптимізації маркетингових кампаній (розсилки, push-повідомлення)

Дозволяє виявляти проблеми з продуктивністю в пікові періоди

Корисно для планування робочого навантаження служби підтримки

Агрегаційний запит:

```
 {
  $group: {
   _id: {
    userId: "$userId",
    hour: { $dateToString: { format: "%Y-%m-%d %H:00", date: "$timestamp" } }
   },
   actions: { $push: "$action" },
   count: { $sum: 1 }
  }
 },
 { $sort: { "_id.hour": 1 } }
]
```

2. Продажі по годинах

Призначення: Аналіз динаміки продажів протягом дня.

Обґрунтування:

Дозволяє виявляти пікові періоди продажів

Корисно для планування маркетингових акцій

Допомагає в оптимізації роботи служби обробки замовлень

Дозволяє аналізувати вплив часових факторів на покупки

Корисно для управління персоналом та логістикою

Агрегаційний запит:

```
 {
  $group: {
```

```
    _id: {
      $dateToString: { format: "%Y-%m-%d %H:00", date: "$timestamp" }
    },
    totalSales: { $sum: "$amount" },
    transactions: { $sum: 1 }
   }
 },
 { $sort: { "_id": 1 } }
]
```

3. Найпопулярніші продукти

Призначення: Визначення найпопулярніших товарів за обсягом продажів.

Обґрунтування:

Допомагає в оптимізації асортименту та запасів

Корисно для планування закупівель та виробництва

Дозволяє виявляти тренди в попиті на продукти

Допомагає в розробці стратегій крос-селу та апселу

Корисно для управління ціновою політикою

Агрегаційний запит:

```
 {
  $group: {
    _id: "$productId",
    totalSales: { $sum: "$amount" },
    transactions: { $sum: 1 }
   }
 },
 { $sort: { "totalSales": -1 } },
 { $limit: 3 }
]
```

4. Продажі по регіонах

Призначення: Географічний аналіз розподілу продажів.

Обґрунтування:

Дозволяє оптимізувати логістику та доставку

Корисно для регіональних маркетингових стратегій

Допомагає виявляти регіони з високим та низьким попитом

Дозволяє адаптувати асортимент до регіональних особливостей

Корисно для планування розширення бізнесу

Агрегаційний запит:

```
 {
  $group: {
   _id: "$metadata.region",
   totalAmount: { $sum: "$amount" },
   productsSold: { $sum: 1 }
  }
 },
 { $sort: { "totalAmount": -1 } }
```

5. Аналіз використання пристроїв

Призначення: Вивчення переваг користувачів щодо пристроїв доступу.

Обґрунтування:

Допомагає в оптимізації сайту для різних пристроїв

Корисно для покращення користувацького досвіду

Дозволяє виявляти проблеми з сумісністю

Допомагає в плануванні розробки мобільних додатків

Корисно для адаптації маркетингових матеріалів

Агрегаційний запит:

```
 {
  $group: {
   _id: "$metadata.device",
   users: { $addToSet: "$userId" },
   count: { $sum: 1 }
```

```
    }
  },
  {
   $project: {
    device: "$_id",
    userCount: { $size: "$users" },
    actions: "$count"
   }
  },
  { $sort: { "actions": -1 } }
]
```

Загальні переваги агрегаційних запитів

Гнучкість аналізу:

Дозволяють отримувати різноманітні зрізи даних з однієї колекції

Можуть комбінуватися для отримання складних аналітичних звітів

Ефективність:

Використовують оптимізоване зберігання часових рядів

Виконуються швидко завдяки спеціалізованому індексуванню

Бізнес-цілінність:

Надають дані для прийняття обґрунтованих рішень

Дозволяють виявляти тренди та аномалії в даних

Корисні для оперативного та стратегічного планування

Автоматизація:

Можуть запускатися за розкладом для регулярного моніторингу

Дозволяють інтегруватися з системами візуалізації даних

Оперативність:

Надають актуальну інформацію для швидкого реагування

Дозволяють аналізувати великі обсяги даних в реальному часі

Кожен агрегаційний запит має чітке бізнес-обґрунтування і спрямований на вирішення конкретних завдань електронного магазину, від оптимізації

операційних процесів до покращення користувацького досвіду та збільшення продажів. Використання часових рядів з агрегаційними запитами створює потужний інструмент для аналізу даних, який допомогає приймати обґрунтовані бізнес-рішення на основі реальних даних про поведінку користувачів та динаміку продажів.

Відповіді на контрольні питання

1. Що таке колекція часових рядів у MongoDB і які поля обов'язково потрібно вказувати при її створенні?

Колекція часових рядів у MongoDB – це спеціалізований тип колекції, оптимізований для зберігання та обробки даних, що змінюються з часом (наприклад, показники датчиків, логи подій, фінансові транзакції).

При створенні колекції часових рядів обов'язково потрібно вказувати:

timeField: Поле, яке містить часові мітки для документів

metaField: Поле, яке містить метадані (наприклад, ідентифікатор датчика або користувача)

granularity: Рівень деталізації зберігання даних (може бути "seconds", "minutes" або "hours")

2. Яку роль виконують параметри timeField, metaField і granularity при створенні колекції типу time series?

timeField: Визначає поле в документах, яке містить часові мітки. Це поле автоматично індексується для швидких запитів за часом.

metaField: Визначає поле, яке містить метадані. Дозволяє групувати та аналізувати дані за додатковими критеріями.

granularity: Визначає рівень деталізації зберігання даних:

seconds: Найвища деталізація

minutes: Середня деталізація

hours: Нижча деталізація

3. Які особливості індексації і як MongoDB оптимізує зберігання даних у часових колекціях?

Особливості індексації:

Поле timeField автоматично індексується при створенні колекції

Індекс оптимізований для запитів за часові діапазони

Оптимізація зберігання:

Дані зберігаються в компактному форматі, економлячи дисковий простір

MongoDB використовує внутрішні механізми стиснення для часових даних

Документи з однаковою метадатою та близькими часові мітки зберігаються разом

Оптимізовано для запитів, що включають часові мітки та метадані

4. Як за допомогою агрегаційного пайплайну можна обчислити середнє значення метрики по годинах для певного датчика?

```
{
 $match: {
  sensorId: "sensor123" // Фільтрація за певним датчиком
 }
},
{
 $group: {
  _id: {
   $dateToString: {
    format: "%Y-%m-%d %H:00",
    date: "$timestamp"
   }
  },
  avgValue: { $avg: "$value" } // Обчислення середнього
 }
},
```

```
  {
    $sort: { "_id": 1 } // Сортування за часом
  }
]
```

5. Які переваги використання TTL (часу життя документів) у часових колекціях і як це налаштувати?

Переваги TTL:

Автоматичне видалення застарілих даних без додаткового коду

Економія дискового простору та ресурсів

Підтримання актуальності даних в колекції

Налаштування TTL:

```
 Copydb.createCollection("salesData", {
  timeseries: {
    timeField: "timestamp",
    metaField: "productId",
    granularity: "hours"
  },
  expireAfterSeconds: 604800 // 7 днів
});
```

6. Як налаштувати автоматичне видалення застарілих даних у часовій колекції за допомогою TTL (expireAfterSeconds) і які обмеження варто враховувати при цьому?

Налаштування TTL:

```
 Copydb.createCollection("logs", {
  timeseries: {...},
  expireAfterSeconds: 86400 // 1 день
});
```

Обмеження, які варто враховувати:

TTL повинен бути позитивним числом

Велика кількість видалень може тимчасово вплинути на продуктивність

TTL не можна застосовувати до звичайних колекцій

Зміна параметра TTL не впливає на документи, що вже існують

TTL не гарантує точного часу видалення

Видалення через TTL не генерує операції видалення в логах (oplog)