Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра системотехніки

Дисципліна: «СПЕЦІАЛІЗОВАНІ БАЗИ ДАНИХ (NOSQL)»

**ПРАКТИЧНЕ ЗАНЯТТЯ № 3**

**«ВИВЧЕННЯ ОСНОВНИХ МОЖЛИВОСТЕЙ СУБД MONGODB»**

|  |  |  |
| --- | --- | --- |
| Виконав:  ст. гр. ІТКН-18-2  Гуляєв Д.С. |  | Прийняв:  Доц. каф. Хряпкін О.В. |

Харків 2020

1. **Мета роботи**

Розробити програмний засіб, що взаємодіє з СУБД MongoDB. Отримати практичні навички виконання основних запитів до БД. Дослідити наведені цифри щодо виконання різних запитів.

**2. Постановка задачі**

1. Розробити схему (схеми) документів у форматі BSON, які будуть відповідати за зберігання даних.

2. Розробити інтерфейс (програму) забезпечує взаємодію з СУБД MongoDB.

3. Програма повинна містити DAO, яка включає методи функції: додавання, перегляд, редагування, пошук по декількох ключах, ... документів.

4. Провести експерименти з вивчення продуктивності (на наборах даних в 100, 1000, 10000, 50000, 100000, 500000 записів). Заміряти час виконання операцій:

* Додавання і читання записів в БД (Mongodb і MySQL).
* Створити індекси (мінімум по двох полях) за якими проводяться запити в обох БД і повторити попередній пункт.

5. Побудувати графіки залежностей проведених експериментів.

6. Зробити висновки.

7. Оформити звіт, повинен містити схеми таблиць, документів у форматі BSON. Побудувати графіки залежностей для всіх операцій з використанням індексів і без.

**3. Схема документів**

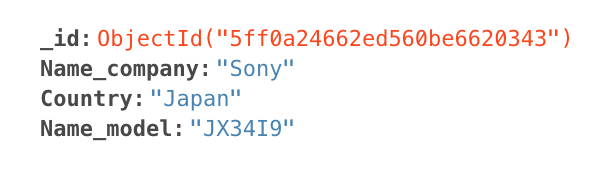


Рисунок 1 – Документ «Model»



Рисунок 2 – Документ «Type»

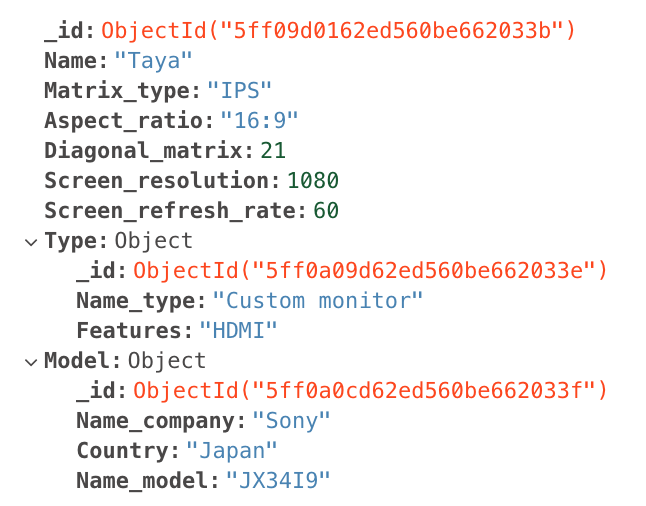
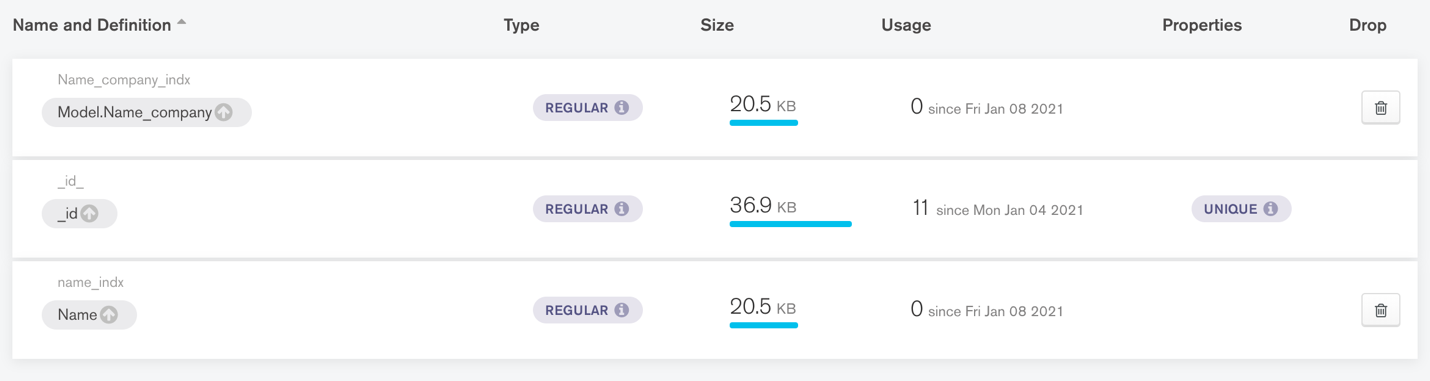


Рисунок 3 – Документ «Monitor»

Створені індекси для документу монітор:

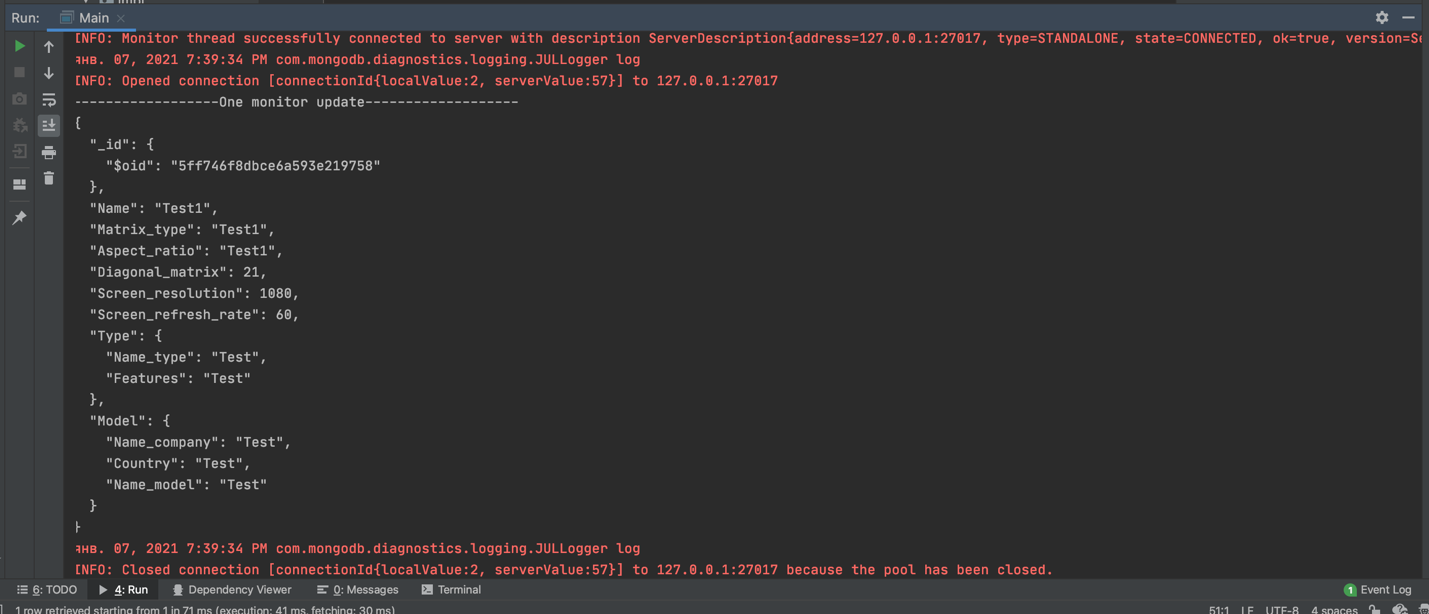


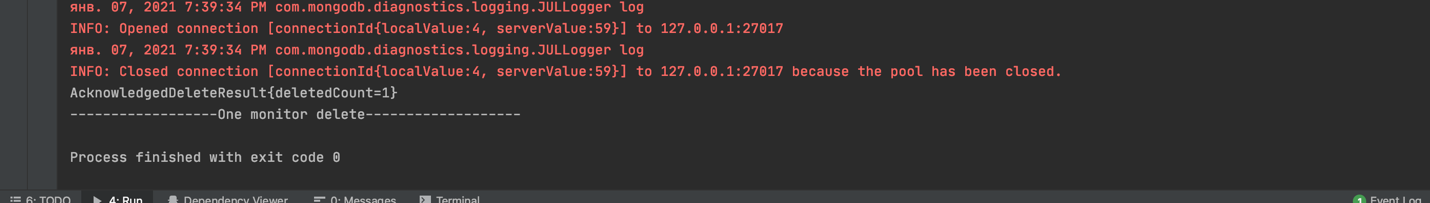
**4. Перевірка працездатності розробленого модуля DAO:**

**Код перевірки працездатності:**

**Main**

import Dao.ModelDao;  
import Dao.MonitorDao;  
import Dao.TypeDao;  
import Impl.ModelDaoImpl;  
import Impl.MonitorDaoImpl;  
import Impl.TypeDaoImpl;  
import Models.Model;  
import Models.Monitor;  
import Models.Type;  
import com.mongodb.client.MongoClient;  
import com.mongodb.client.MongoClients;  
import org.bson.Document;  
import org.bson.json.JsonWriterSettings;  
import org.bson.types.ObjectId;  
  
import java.util.ArrayList;  
import java.util.List;  
  
import static java.util.Arrays.*asList*;  
  
public class Main {  
 public static void main(String[] args) {  
  
 //insert model  
// Model model = new Model("Test","Test", "Test");  
// ModelDao modelDao = new ModelDaoImpl();  
// modelDao.createModel(model);  
  
 //insert type  
// Type type = new Type("Test", "Test");  
// TypeDao typeDao = new TypeDaoImpl();  
// typeDao.createType(type);  
  
 //insert monitor  
// Model model = new Model("Test","Test", "Test");  
// Type type = new Type("Test", "Test");  
// Monitor monitor = new Monitor("Test", "Test", "Test", 21, 1080, 60, type, model);  
// MonitorDao monitorDao = new MonitorDaoImpl();  
// monitorDao.createMonitor(monitor);  
  
 //find by name model  
// ModelDao modelDao = new ModelDaoImpl();  
// modelDao.getByModel("Test");  
  
 //find by type  
// TypeDao typeDao = new TypeDaoImpl();  
// typeDao.getByNameType("Custom monitor");  
  
 //find by name monitor  
// MonitorDao monitorDao = new MonitorDaoImpl();  
// monitorDao.getByName("Taya");  
  
 //find all models  
// ModelDao modelDao = new ModelDaoImpl();  
// modelDao.getAllModels();  
  
 //find all types  
// TypeDao typeDao = new TypeDaoImpl();  
// typeDao.getAllTypes();  
  
 //find all monitors  
// MonitorDao monitorDao = new MonitorDaoImpl();  
// monitorDao.getAllMonitors();  
  
 //update model  
// ObjectId id = new ObjectId("5ff1f52b7f11f303c4b7ceaa");  
// Model model = new Model(id,"Test1", "Test1", "Tes1");  
// ModelDao modelDao = new ModelDaoImpl();  
// modelDao.updateModel(model);  
  
 //update type  
// ObjectId id = new ObjectId("5ff71eb0ed8d8b767e7e27f5");  
// Type type = new Type(id,"Test1","Test1");  
// TypeDao typeDao = new TypeDaoImpl();  
// typeDao.updateType(type);  
  
 //update monitor  
 ObjectId id = new ObjectId("5ff746f8dbce6a593e219758");  
 Model model1 = new Model("Test","Test", "Test");  
 Type type1 = new Type("Test", "Test");  
 Monitor monitor1 = new Monitor(id,"Test1", "Test1", "Test1", 21, 1080, 60, type1, model1);  
 MonitorDao monitorDao1 = new MonitorDaoImpl();  
 monitorDao1.updateMonitor(monitor1);  
  
 //delete model  
// ObjectId id = new ObjectId("5ff1f52b7f11f303c4b7ceaa");  
// Model model = new Model(id);  
// ModelDao modelDao = new ModelDaoImpl();  
// modelDao.deleteModel(model);  
  
 //delete type  
// ObjectId id = new ObjectId("5ff71eb0ed8d8b767e7e27f5");  
// Type type = new Type(id);  
// TypeDao typeDao = new TypeDaoImpl();  
// typeDao.deleteType(type);  
  
 //delete monitor  
 ObjectId id2 = new ObjectId("5ff746f8dbce6a593e219758");  
 Monitor monitor2 = new Monitor(id2);  
 MonitorDao monitorDao2 = new MonitorDaoImpl();  
 monitorDao2.deleteMonitor(monitor2);  
 }  
}

****

****

**5. Експерименти з вивчення продуктивності(на наборах даних в 100, 1000, 10000, 50000, 100000, 500000 записів:**

**5.1. Додавання й читання записів без індексів.**

Таблиця 1.- час додавання й читання записів без індексів

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Select | | Insert | |
| MySQL | Mongo | MySQL | Mongo |
| 100 | 1,73 | 1,41 | 1,134 | 0,174 |
| 1000 | 4,89 | 4,84 | 5,97 | 0,854 |
| 10000 | 32,39 | 23,08 | 198,24 | 11,29 |
| 50000 | 141,322 | 115,85 | 1095,99 | 49,42 |
| 100000 | 251,535 | 176,9 | 1898,22 | 77,41 |
| 500000 | 1057,111 | 674,333 | 9090,11 | 380,78 |

Використовуючи таблицю 1 були побудовані графіки залежності між часом та кількістю записів.

Рисунок 5 – Графік залежності для операції вибору даних

Рисунок 6 – Графік залежності між часом та кількістю записів для операції запису

**5.2. Додавання й читання записів з індексами**

Таблиця 2.- час додавання й читання записів з індексами

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Select | | Insert | |
| MySQL | Mongo | MySQL | Mongo |
| 100 | 0,053 | 0,064 | 0,177 | 0,0496 |
| 10000 | 0,091 | 0,058 | 8,315 | 0,821 |
| 100000 | 0,031 | 0,036 | 89,82 | 5,211 |
| 500000 | 0,071 | 0,033 | 343,17 | 17,397 |

Використовуючи таблицю 2 були побудовані графіки залежності між часом та кількістю записів.

Рисунок 7 – Графік залежності після створення індексу для операції створення записів

Рисунок 8 – Графік залежності після створення індексу для операції вибору даних

**6. Висновки**

В ході лабораторної роботи було спроектовано і розроблено нереляційну базу даних з відповідною структурою (таблиці, зв'язку) відповідно до обраної предметної області використовуючи MongoDB. Для створення модуля взаємодії програми та обраної СУБД замість DAO, використовувався Repository.

Переглянувши графіки дослідження швидкості виконання запитів в різних СУБД, можна зробити висновок , що значна різниця в швидкостях з’являється при великій кількості запитів на вставку записів , при цьому MySQL працює повільніше ніж Mongo майже в 30 раз. Створення індексів сприяло пришвидшенню виконання запитів в декілька разів.