Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра системотехніки

Дисципліна: «СПЕЦІАЛІЗОВАНІ БАЗИ ДАНИХ (NOSQL)»

**ПРАКТИЧНЕ ЗАНЯТТЯ № 5**

**«СТВОРЕННЯ ТА ДОСЛІДЖЕННЯ РЕПЛІКАЦІЇ В СУБД MONGODB»**

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

Харків 2020

1. **Мета роботи**

Створити репліку для спроектованої БД, що складається з декількох серверів. Провести експериментальні дослідження роботи репліки в декількох режимах. Отримати практичні навички організації репліки і створення клієнтського підключення.

**2. Постановка задачі**

1. Створити репліку з декількох серверів (мінімум 3).

2. Модифікувати програму (попереднє практичне завдання) для підключення до створеної репліці.

3. Врахувати можливість помилки запису - укласти код записи в БД в блок try / catch. Якщо виникає помилка - поставити затримку в 1 с. І повторити запис цього ж документа. Якщо кількість спроб перевищить 3 - то перервати запис документів в БД.

4. Провести експерименти по запису та читання (пошуку документів). Записати 10000 документів, перевірити цілісність записи (прочитати ці ж 10000 документів).

5. Повторити експеримент. В процесі запису завершити один з процесів сервера MongoDB (slave). Оцінити результати.

6. Повторити попередній експеримент. В процесі запису завершити процес сервера MongoDB (майстер). Оцінити результати.

7. Повторити експерименти (4, 5) в режимах запису (WriteConcern): W1, UNACKNOWLEDGED, JOURNALED, MAJORITY.

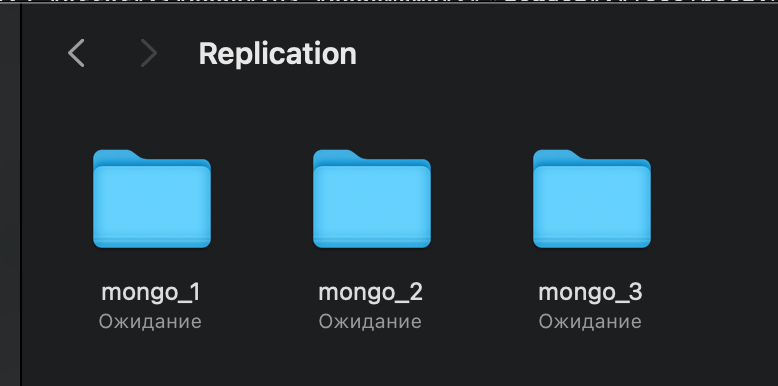
**3. Створення реплікації**

Для творення реплікації в MongoDB, потрібно підняти хоча б три бази даних. Створюємо три бази даних – один primary, й два secondary сервера.

Всі операції запису йдуть виключно до primary, а потім дублюються до secondary. Що ж до читання даних, то клієнт може звертатись до любого інтанцу БД.

**3.1 Створення репліки:**

Спочатку потрібно створити репліку, щоб подалі підключитись до неї. Для створення репліки використаємо командну строку та наступний код (запускаємо кожний сервер):



mongod --replSet myapp --dbpath /Users/apple/Desktop/My\_prog/3\_curse/NoSql/Replication/mongo\_1 --port 27011

mongod --replSet myapp --dbpath /Users/apple/Desktop/My\_prog/3\_curse/NoSql/Replication/mongo\_2 --port 27012

mongod --replSet myapp --dbpath /Users/apple/Desktop/My\_prog/3\_curse/NoSql/Replication/mongo\_3 --port 27013

Рисунок 1 – Результат створення репліки

**3.1 Підключення до створеної репліки:**

Для підключення до репліки вставим наступний код:

mongodb://localhost:27011,localhost:27012,localhost:27013/?replicaSet=myapp&retryWrites=true

**3.2 Код перевірки працездатності створеної репліки:**

public class Replication {  
 MonitorDao monitorDao = new MonitorDaoImpl();  
 public void replication() throws InterruptedException {  
 monitorDao.deleteAllMonitors();  
 long startTime = System.*nanoTime*();  
 Model model = new Model("Test","Test", "Test");  
 Type type = new Type("Test", "Test");  
 for(int i = 0; i < 500; i++){  
 insert(new Monitor("Test", "Test", "Test", 21, 1080, 60, type, model));  
 }  
 long endTime = System.*nanoTime*();  
 System.*out*.println("Insert: " + (double) (endTime - startTime) / 1000000000);  
 startTime = System.*nanoTime*();  
 monitorDao.getAllMonitors();  
 endTime = System.*nanoTime*();  
 System.*out*.println("Select: " + (double) (endTime - startTime) / 1000000000);  
 }

**Main**

// Replication  
 new Replication.Replication().replication();

Врахувати можливість помилки запису - укласти код записи в БД в блок try / catch. Якщо виникає помилка - поставити затримку в 1 с. І повторити запис цього ж документа. Якщо кількість спроб перевищить 3 - то перервати запис документів в БД.

public boolean insert(Monitor monitor) throws InterruptedException {  
 int retries = 3;  
 while (retries > 0) {  
 try {  
 if(monitorDao.createMonitor(monitor) != false){  
 return true;  
 }  
 } catch (Exception e) {  
 retries--;  
 Thread.*sleep*(1000);  
 }  
 }  
 System.*out*.println("Error to insert document");  
 return false;  
}

****

****

**3.3 Проведення експериментів з різним типом WriteConcern**

Для дослідження роботи репліки в декількох режимах використовували наступні значення WriteConcern – W1 (значення за замовчанням), JOURNALED, MAJORITY.

Таблиця 1 – Результат дослідження.

|  |  |  |  |
| --- | --- | --- | --- |
|  | W1 | JOURNALED | MAJORITY |
| Без виключення серверу | 0,658 / 0,0028 | 1,944 / 0,0023 | 3, 2 / 0,0029 |
| З виключенням primary | 0,855 / 0,0026 | 1,47/0,0021 | 2.93/0,003 |
| З виключенням  secondary | 0,808 / 0,0035 | 1,53 / 0,0021 | 3,08 / 0,0025 |

**4. Висновки**

В ході робота була створена репліка для спроектованої БД, що складається з декількох серверів(один primary, й два secondary сервера). Провели експериментальні дослідження роботи репліки в декількох режимах. Отримали практичні навички організації репліки і створення клієнтського підключення. На практиці перевірили швидкість роботи репліки в різних режимах, найшвидші ті, що записують лише у primary сервер(W1), найповільніші, потребує підтвердження успішного запису більшості secondary серверів(MAJORITY).