Міністерство освіти і науки, молоді та спорту України
Національний технічний університет України
“Київський політехнічний інститут”
Фізико-Технічний інститут

Лабораторна робота № 6
з семестрового курсу
“Проектування високонавантажених систем”

Виконала:
Студентка групи ФІ-03
Швець Катерина

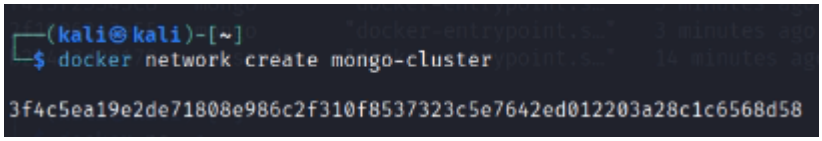
Київ, 2023

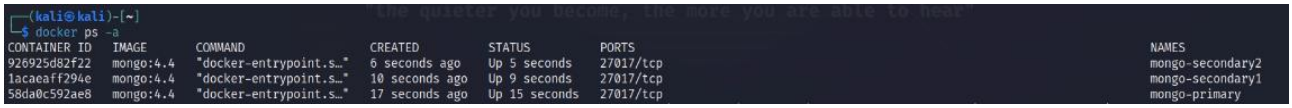
Task 6 - Налаштування реплікації та перевірка відмовостійкості MongoDB

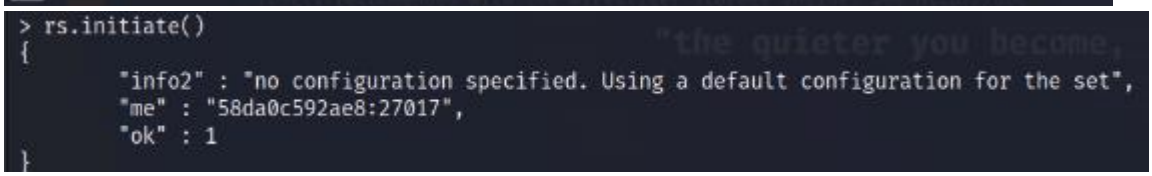
Ознайомтесь з реплікацію даних в MongoDB
http://docs.mongodb.org/manual/core/replication-introduction/

Завдання:

1) Налаштувати реплікацію в конфігурації: Primary with Two Secondary Members (всі ноди можуть бути запущені як окремі процеси або у Docker контейнерах) - http://docs.mongodb.org/manual/core/replica-set-architecture-three-members/
    - Deploy a Replica Set for Testing and Development- http://docs.mongodb.org/manual/tutorial/deploy-replica-set-for-testing/
    - http://www.tugberkugurlu.com/archive/setting-up-a-mongodb-replica-set-with-docker-and-connecting-to-it-with-a--net-core-app







```
rs0:SECONDARY> rs.add("mongo-secondary1:27017")
{
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1705750890, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1705750890, 1)
}
rs0:PRIMARY> rs.add("mongo-secondary2:27017")
{
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1705750892, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1705750892, 1)
}
```

```
rs0:PRIMARY> rs.status()
```
Вивід:


```
{
        "set" : "rs0",
        "date" : ISODate("2024-01-20T11:41:50.082Z"),
        "myState" : 1,
        "term" : NumberLong(1),
        "syncSourceHost" : "",
        "syncSourceId" : -1,
        "heartbeatIntervalMillis" : NumberLong(2000),
        "majorityVoteCount" : 2,
        "writeMajorityCount" : 2,
        "votingMembersCount" : 3,
        "writableVotingMembersCount" : 3,
        "optimes" : {
                "lastCommittedOpTime" : {
                        "ts" : Timestamp(1705750906, 1),
                        "t" : NumberLong(1)
                },
                "lastCommittedWallTime" : ISODate("2024-01-20T11:41:46.076Z"),
                "readConcernMajorityOpTime" : {
                        "ts" : Timestamp(1705750906, 1),
                        "t" : NumberLong(1)
                },
                "readConcernMajorityWallTime" : ISODate("2024-01-20T11:41:46.076Z"),
                "appliedOpTime" : {
                        "ts" : Timestamp(1705750906, 1),
                        "t" : NumberLong(1)
                },
                "durableOpTime" : {
                        "ts" : Timestamp(1705750906, 1),
                        "t" : NumberLong(1)
                },
                "lastAppliedWallTime" : ISODate("2024-01-20T11:41:46.076Z"),
                "lastDurableWallTime" : ISODate("2024-01-20T11:41:46.076Z")
```

```
},
"lastStableRecoveryTimestamp" : Timestamp(1705750876, 8),
"electionCandidateMetrics" : {
        "lastElectionReason" : "electionTimeout",
        "lastElectionDate" : ISODate("2024-01-20T11:41:16.018Z"),
        "electionTerm" : NumberLong(1),
        "lastCommittedOpTimeAtElection" : {
                "ts" : Timestamp(0, 0),
                "t" : NumberLong(-1)
        },
        "lastSeenOpTimeAtElection" : {
                "ts" : Timestamp(1705750875, 1),
                "t" : NumberLong(-1)
        },
        "numVotesNeeded" : 1,
        "priorityAtElection" : 1,
        "electionTimeoutMillis" : NumberLong(10000),
        "newTermStartDate" : ISODate("2024-01-20T11:41:16.055Z"),
        "wMajorityWriteAvailabilityDate" : ISODate("2024-01-20T11:41:16.104Z")
},
"members" : [
        {
                "_id" : 0,
                "name" : "58da0c592ae8:27017",
                "health" : 1,
                "state" : 1,
                "stateStr" : "PRIMARY",
                "uptime" : 3070,
                "optime" : {
                        "ts" : Timestamp(1705750906, 1),
                        "t" : NumberLong(1)
                },
                "optimeDate" : ISODate("2024-01-20T11:41:46Z"),
                "lastAppliedWallTime" : ISODate("2024-01-20T11:41:46.076Z"),
                "lastDurableWallTime" : ISODate("2024-01-20T11:41:46.076Z"),
                "syncSourceHost" : "",
                "syncSourceId" : -1,
                "infoMessage" : "Could not find member to sync from",
                "electionTime" : Timestamp(1705750876, 1),
                "electionDate" : ISODate("2024-01-20T11:41:16Z"),
                "configVersion" : 3,
                "configTerm" : 1,
                "self" : true,
                "lastHeartbeatMessage" : ""
        },
        {
                "_id" : 1,
                "name" : "mongo-secondary1:27017",
                "health" : 1,
                "state" : 2,
                "stateStr" : "SECONDARY",
                "uptime" : 19,
                "optime" : {
                        "ts" : Timestamp(1705750906, 1),
```

```
                "t" : NumberLong(1)
            },
            "optimeDurable" : {
                "ts" : Timestamp(1705750906, 1),
                "t" : NumberLong(1)
            },
            "optimeDate" : ISODate("2024-01-20T11:41:46Z"),
            "optimeDurableDate" : ISODate("2024-01-20T11:41:46Z"),
            "lastAppliedWallTime" : ISODate("2024-01-20T11:41:46.076Z"),
            "lastDurableWallTime" : ISODate("2024-01-20T11:41:46.076Z"),
            "lastHeartbeat" : ISODate("2024-01-20T11:41:48.327Z"),
            "lastHeartbeatRecv" : ISODate("2024-01-20T11:41:49.334Z"),
            "pingMs" : NumberLong(0),
            "lastHeartbeatMessage" : "",
            "syncSourceHost" : "58da0c592ae8:27017",
            "syncSourceId" : 0,
            "infoMessage" : "",
            "configVersion" : 3,
            "configTerm" : 1
        },
        {
            "_id" : 2,
            "name" : "mongo-secondary2:27017",
            "health" : 1,
            "state" : 2,
            "stateStr" : "SECONDARY",
            "uptime" : 17,
            "optime" : {
                "ts" : Timestamp(1705750906, 1),
                "t" : NumberLong(1)
            },
            "optimeDurable" : {
                "ts" : Timestamp(1705750906, 1),
                "t" : NumberLong(1)
            },
            "optimeDate" : ISODate("2024-01-20T11:41:46Z"),
            "optimeDurableDate" : ISODate("2024-01-20T11:41:46Z"),
            "lastAppliedWallTime" : ISODate("2024-01-20T11:41:46.076Z"),
            "lastDurableWallTime" : ISODate("2024-01-20T11:41:46.076Z"),
            "lastHeartbeat" : ISODate("2024-01-20T11:41:48.326Z"),
            "lastHeartbeatRecv" : ISODate("2024-01-20T11:41:48.720Z"),
            "pingMs" : NumberLong(0),
            "lastHeartbeatMessage" : "",
            "syncSourceHost" : "mongo-secondary1:27017",
            "syncSourceId" : 1,
            "infoMessage" : "",
            "configVersion" : 3,
            "configTerm" : 1
        }
    ],
    "ok" : 1,
    "$clusterTime" : {
        "clusterTime" : Timestamp(1705750906, 1),
        "signature" : {
```

```
                "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                "keyId" : NumberLong(0)
            }
        },
        "operationTime" : Timestamp(1705750906, 1)
}
```

Перевірила стан реплікаційного сету, все підключено.

2) Продемонструвати Read Preference Modes: читання з *primary* і *secondary* node (http://docs.mongodb.org/manual/core/read-preference/)

```
rs0:PRIMARY> db.collection.find().readPref('secondary')
rs0:PRIMARY> db.collection.insertOne({data:"text"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("65abb34342859db387744cb5")
}
rs0:PRIMARY> db.collection.find().readPref('secondary')
{ "_id" : ObjectId("65abb34342859db387744cb5"), "data" : "text" }
```

3) Спробувати зробити запис з однією відключеною нодою та *write concern* рівнім 3 та нескінченім таймаутом. Спробувати під час таймаута включити відключену ноду

З відключеною нодою не віходить, тому що команда чекає підтвердження від усіх трьох нод

```
┌──(kali㉿kali)-[~]
└─$ docker pause mongo-secondary1

mongo-secondary1

┌──(kali㉿kali)-[~]
└─$ docker exec -it mongo-primary mongo

MongoDB shell version v4.4.28
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("f63839b2-2cf4-4cba-af61-f1df9f1076a2") }
MongoDB server version: 4.4.28
---
The server generated these startup warnings when booting:
        2024-01-20T10:50:40.317+00:00: Using the XFS filesystem is strongly recommended with t
        2024-01-20T10:50:40.480+00:00: Access control is not enabled for the database. Read an
        2024-01-20T10:50:40.481+00:00: /sys/kernel/mm/transparent_hugepage/enabled is 'always'
---
rs0:PRIMARY> db.collection.insert({ item: "test" }, { writeConcern: { w: 3, wtimeout: 0 } })
```

підключимо нашу ноду та знову виконаємо запит

```
┌──(kali㉿kali)-[~]
└─$ docker unpause mongo-secondary1

mongo-secondary1

┌──(kali㉿kali)-[~]
└─$ docker exec -it mongo-primary mongo

MongoDB shell version v4.4.28
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("000dc8dd-6d7c-4243-9913-57925adf75db") }
MongoDB server version: 4.4.28
---
The server generated these startup warnings when booting:
        2024-01-20T10:50:40.317+00:00: Using the XFS filesystem is strongly recommended with the
        2024-01-20T10:50:40.480+00:00: Access control is not enabled for the database. Read and
        2024-01-20T10:50:40.481+00:00: /sys/kernel/mm/transparent_hugepage/enabled is 'always'.
---
rs0:PRIMARY> db.collection.insert({ item: "test" }, { writeConcern: { w: 3, wtimeout: 0 } })
WriteResult({ "nInserted" : 1 })
```

4) Аналогічно попередньому пункту, але задати скінченний таймаут та дочекатись його закінчення. Перевірити чи данні записались і чи доступні на читання з рівнем *readConcern: “majority”*

Знову відключимо ноду на момент роботи запиту

```
┌──(kali㉿kali)-[~]
└─$ docker pause mongo-secondary1
```

```
rs0:PRIMARY> db.collection.insert({ item: "test" }, { writeConcern: { w: 3, wtimeout: 5000 } })
WriteResult({
        "nInserted" : 1,
        "writeConcernError" : {
                "code" : 64,
                "codeName" : "WriteConcernFailed",
                "errmsg" : "waiting for replication timed out",
                "errInfo" : {
                        "wtimeout" : true,
                        "writeConcern" : {
                                "w" : 3,
                                "wtimeout" : 5000,
                                "provenance" : "clientSupplied"
                        }
                }
        }
})
```

```
┌──(kali㉿kali)-[~]
└─$ docker unpause mongo-secondary1
```

```
rs0:PRIMARY> db.collection.find({ item: "test" }).readConcern("majority")
{ "_id" : ObjectId("65abb49c411f7dc11fac4d50"), "item" : "test" }
{ "_id" : ObjectId("65abb60076aecbb058a449eb"), "item" : "test" }
{ "_id" : ObjectId("65abb67a4e6de8eb319e25cf"), "item" : "test" }
```

5) Продемонстрував перевибори primary node в відключивши поточний primary (Replica Set Elections) - http://docs.mongodb.org/manual/core/replica-set-elections/
   - і що після відновлення роботи старої primary на неї реплікуються нові дані, які з'явилися під час її простою

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it mongo-secondary1 mongo
```

```
rs0:SECONDARY> rs.status()
```

```
{
        "set" : "rs0",
```

```
"date" : ISODate("2024-01-20T12:12:40.137Z"),
"myState" : 2,
"term" : NumberLong(1),
"syncSourceHost" : "58da0c592ae8:27017",
"syncSourceId" : 0,
"heartbeatIntervalMillis" : NumberLong(2000),
"majorityVoteCount" : 2,
"writeMajorityCount" : 2,
"votingMembersCount" : 3,
"writableVotingMembersCount" : 3,
"optimes" : {
        "lastCommittedOpTime" : {
                "ts" : Timestamp(1705752756, 1),
                "t" : NumberLong(1)
        },
        "lastCommittedWallTime" : ISODate("2024-01-20T12:12:36.445Z"),
        "readConcernMajorityOpTime" : {
                "ts" : Timestamp(1705752756, 1),
                "t" : NumberLong(1)
        },
        "readConcernMajorityWallTime" : ISODate("2024-01-20T12:12:36.445Z"),
        "appliedOpTime" : {
                "ts" : Timestamp(1705752756, 1),
                "t" : NumberLong(1)
        },
        "durableOpTime" : {
                "ts" : Timestamp(1705752756, 1),
                "t" : NumberLong(1)
        },
        "lastAppliedWallTime" : ISODate("2024-01-20T12:12:36.445Z"),
        "lastDurableWallTime" : ISODate("2024-01-20T12:12:36.445Z")
},
"lastStableRecoveryTimestamp" : Timestamp(1705752726, 1),
"members" : [
        {
                "_id" : 0,
                "name" : "58da0c592ae8:27017",
                "health" : 1,
                "state" : 1,
                "stateStr" : "PRIMARY",
                "uptime" : 1869,
                "optime" : {
                        "ts" : Timestamp(1705752756, 1),
                        "t" : NumberLong(1)
                },
                "optimeDurable" : {
                        "ts" : Timestamp(1705752756, 1),
                        "t" : NumberLong(1)
                },
                "optimeDate" : ISODate("2024-01-20T12:12:36Z"),
                "optimeDurableDate" : ISODate("2024-01-20T12:12:36Z"),
                "lastAppliedWallTime" : ISODate("2024-01-20T12:12:36.445Z"),
                "lastDurableWallTime" : ISODate("2024-01-20T12:12:36.445Z"),
                "lastHeartbeat" : ISODate("2024-01-20T12:12:38.279Z"),
```

```
        "lastHeartbeatRecv" : ISODate("2024-01-20T12:12:38.279Z"),
        "pingMs" : NumberLong(0),
        "lastHeartbeatMessage" : "",
        "syncSourceHost" : "",
        "syncSourceId" : -1,
        "infoMessage" : "",
        "electionTime" : Timestamp(1705750876, 1),
        "electionDate" : ISODate("2024-01-20T11:41:16Z"),
        "configVersion" : 3,
        "configTerm" : 1
    },
    {
        "_id" : 1,
        "name" : "mongo-secondary1:27017",
        "health" : 1,
        "state" : 2,
        "stateStr" : "SECONDARY",
        "uptime" : 4914,
        "optime" : {
            "ts" : Timestamp(1705752756, 1),
            "t" : NumberLong(1)
        },
        "optimeDate" : ISODate("2024-01-20T12:12:36Z"),
        "lastAppliedWallTime" : ISODate("2024-01-20T12:12:36.445Z"),
        "lastDurableWallTime" : ISODate("2024-01-20T12:12:36.445Z"),
        "syncSourceHost" : "58da0c592ae8:27017",
        "syncSourceId" : 0,
        "infoMessage" : "",
        "configVersion" : 3,
        "configTerm" : 1,
        "self" : true,
        "lastHeartbeatMessage" : ""
    },
    {
        "_id" : 2,
        "name" : "mongo-secondary2:27017",
        "health" : 1,
        "state" : 2,
        "stateStr" : "SECONDARY",
        "uptime" : 1867,
        "optime" : {
            "ts" : Timestamp(1705752756, 1),
            "t" : NumberLong(1)
        },
        "optimeDurable" : {
            "ts" : Timestamp(1705752756, 1),
            "t" : NumberLong(1)
        },
        "optimeDate" : ISODate("2024-01-20T12:12:36Z"),
        "optimeDurableDate" : ISODate("2024-01-20T12:12:36Z"),
        "lastAppliedWallTime" : ISODate("2024-01-20T12:12:36.445Z"),
        "lastDurableWallTime" : ISODate("2024-01-20T12:12:36.445Z"),
        "lastHeartbeat" : ISODate("2024-01-20T12:12:38.279Z"),
        "lastHeartbeatRecv" : ISODate("2024-01-20T12:12:38.278Z"),
```

```
                "pingMs" : NumberLong(0),
                "lastHeartbeatMessage" : "",
                "syncSourceHost" : "58da0c592ae8:27017",
                "syncSourceId" : 0,
                "infoMessage" : "",
                "configVersion" : 3,
                "configTerm" : 1
        }
    ],
    "ok" : 1,
    "$clusterTime" : {
        "clusterTime" : Timestamp(1705752756, 1),
        "signature" : {
            "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
            "keyId" : NumberLong(0)
        }
    },
    "operationTime" : Timestamp(1705752756, 1)
}
```


Бачимо що секондарі1 став новим праймері

```
┌──(kali㉿kali)-[~]
└─$ docker start mongo-primary

mongo-primary
```

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it mongo-secondary1 mongo
```

Додамо дані під час простою

```
rs0:PRIMARY> db.collection.insert({ item: "new data after election" })
WriteResult({ "nInserted" : 1 })
```

І підключаємося до старого праймері нод

```
┌──(kali㉿kali)-[~]
└─$ docker exec -it mongo-primary mongo
```

```
rs0:SECONDARY> db.getMongo().setReadPref('secondaryPreferred');
rs0:SECONDARY> db.collection.find({ item: "new data after election" })
{ "_id" : ObjectId("65abe4b30982d2834d8e61b3"), "item" : "new data after election" }
```

6) Привести кластер до неконсистентного стану користуючись моментом часу коли *primary node* не відразу помічає відсутність *secondary node*
    - відключивши дві *secondary node* протягом 5 сек. на мастер записати значення (з w:1) і перевірити, що воно записалось

```
┌──(kali㉿kali)-[~]
└─$ docker pause mongo-secondary1
mongo-secondary1

┌──(kali㉿kali)-[~]
└─$ docker pause mongo-secondary2
mongo-secondary2
```

```
rs0:PRIMARY> db.collection.insert({ item: "inconsistent item" }, { writeConcern: { w: 1 }
WriteResult({ "nInserted" : 1 })
```

    - спробувати зчитати це значення з різними рівнями *read concern* - `readConcern: {level: <"majority"|"local"| "linearizable">}`

```
rs0:PRIMARY> db.collection.find({ item: "inconsistent item" }).readConcern("majority")
{ "_id" : ObjectId("65abef021e4e02ecf494eb28"), "item" : "inconsistent item" }
rs0:PRIMARY> db.collection.find({ item: "inconsistent item" }).readConcern("local")
{ "_id" : ObjectId("65abef021e4e02ecf494eb28"), "item" : "inconsistent item" }
rs0:PRIMARY> db.collection.find({ item: "inconsistent item" }).readConcern("linearizable")
{ "_id" : ObjectId("65abef021e4e02ecf494eb28"), "item" : "inconsistent item" }
```

- включити дві інші ноди таким чином, щоб вони не бачили попереднього мастера (його можна відключити) і дочекатись поки вони оберуть нового мастера
- підключити (включити) попередню primary-ноду до кластеру і подивитись, що сталось зі значенням яке було на неї записано

7) Земулювати eventual consistency за допомогою установки затримки реплікації для репліки https://docs.mongodb.com/manual/tutorial/configure-a-delayed-replica-set-member/

```
rs0:PRIMARY> cfg.members[1].priority = 0
0
rs0:PRIMARY> cfg.members[1].slaveDelay = 120
120
rs0:PRIMARY> rs.reconfig(cfg)
{
        "operationTime" : Timestamp(1705766922, 1),
        "ok" : 0,
        "errmsg" : "This node, mongo-secondary1:27017, with _id MemberId(1) is not electable under the new configuration with {version: 4, term: 6} for replica set rs0"
        "code" : 104,
        "codeName" : "NodeNotElectable",
        "$clusterTime" : {
                "clusterTime" : Timestamp(1705766922, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
```

8) Лишити *primary* та *secondary* для якої налаштована затримка реплікації. Записати декілька значень. Спробувати прочитати значення з `readConcern: {level: "linearizable"}`
   Має бути затримка поки значення не реплікуються на більшість нод

```
rs0:PRIMARY> db.collection.insert({ item: "1" })
WriteResult({ "nInserted" : 1 })
rs0:PRIMARY> db.collection.insert({ item: "2" })
WriteResult({ "nInserted" : 1 })
rs0:PRIMARY> db.collection.insert({ item: "3" })
WriteResult({ "nInserted" : 1 })
rs0:PRIMARY> db.collection.find({ item: "1" }).readConcern("linearizable")
{ "_id" : ObjectId("65abf1de1e4e02ecf494eb29"), "item" : "1" }
```

Опис додаткових команд Replication Reference - http://docs.mongodb.org/manual/reference/replication/

Вимогу до оформлення протоколу:

Завдання здається особисто без протоколу, або надсилається протокол який має містити:

- команди та результати їх виконання