# Przetwarzanie rozproszone w MongoDB

Celem zajęć jest zapoznanie się z własnościami [MongoDB](https://www.mongodb.com/) umożliwiającymi przetwarzanie rozproszone.

## Przygotowanie środowiska

1. Zaloguj się do maszyny wirtualnej jako użytkownik postgres używając hasła RSBD#7102.
2. Uruchom sesję użytkownika root podając polecenie su -, podaj hasło RSBD#7102.
3. Dodaj repozytorium MongoDB do menadżera pakietów:

ln -s /etc/yum.repos.d/mongodb-org-3.4.repo mongo.repo

echo '[mongodb-org-3.4]'> mongo.repo

echo 'name=MongoDB Repository'>> mongo.repo

echo 'baseurl=https://repo.mongodb.org/yum/redhat/$releasever/mongodb-org/3.4/x86\_64/'>> \ mongo.repo

echo 'gpgcheck=1'>> mongo.repo

echo 'enabled=1'>> mongo.repo

echo 'gpgkey=https://www.mongodb.org/static/pgp/server-3.4.asc' >> mongo.repo

1. Zainstaluj oprogramowanie MongoDB:

yum install -y mongodb-org

1. Wyłącz obługę mechnizmu THP:

echo 'never' > /sys/kernel/mm/transparent\_hugepage/enabled

echo 'never' > /sys/kernel/mm/transparent\_hugepage/defrag

echo 0 > /sys/kernel/mm/transparent\_hugepage/khugepaged/defrag

echo 0 > /sys/kernel/mm/transparent\_hugepage/khugepaged/defrag

1. Dodaj adres lokalny maszyny do pliku /etc/hosts

echo ' 127.0.0.1 RSBD' >> /etc/hosts

1. Zamknij terminal użytkownika *root*.

## Konfiguracja replikacji danych

MongoDB udostępnia asynchroniczną replikację danych typu single master z automatycznym przełączaniem awaryjnym. Niniejsze ćwiczenie jest poświęcone konfiguracji i testowaniu tego mechanizmu.

1. Otwórz okno terminala pomocniczego.
2. Przygotuj konfigurację dla repliki bazy danych RSBD1.

mkdir -p ~/mongo/RSBD1/data  
echo 'systemLog:'>> RSBD.conf

echo ' destination: file'>> RSBD.conf

echo ' logAppend: true'>> RSBD.conf

echo ' path: /var/lib/pgsql/mongo/RSBD1/mongod.log'>> RSBD.conf

echo 'storage:'>> RSBD.conf

echo ' dbPath: /var/lib/pgsql/mongo/RSBD1/data'>> RSBD.conf

echo ' mmapv1:'>> RSBD.conf

echo ' smallFiles: true'>> RSBD.conf

echo ' journal:'>> RSBD.conf

echo ' enabled: true'>> RSBD.conf

echo 'processManagement:'>> RSBD.conf

echo ' fork: true'>> RSBD.conf

echo ' pidFilePath: /var/lib/pgsql/mongo/RSBD1/mongod.pid'>> RSBD.conf

echo 'net:'>> RSBD.conf

echo ' port: 27011'>> RSBD.conf

echo ' bindIp: 127.0.0.1'>> RSBD.conf

echo 'replication:'>> RSBD.conf

echo ' oplogSizeMB: 128'>> RSBD.conf

echo ' replSetName: RsbdRepSet'>> RSBD.conf

mv RSBD.conf RSBD1.conf

1. Uruchom replikę RSBD1 bazy danych MongoDB:

mongod --config RSBD1.conf

1. Otwórz terminal i nazwij go RSBD1 (Terminal->Set title).
2. W terminalu RSBD1 przyłącz się powłokę Mongo do repliki RSBD1. Uwaga: skonfigurowane repliki działają na lokalnym adresie i nie udostępniają danych publicznie. W związku z tym brak konfiguracji autoryzacji dostępu do danych nie jest krytyczny. Jednakże, w środowisku produkcyjnym należy taką [autoryzację skonfigurować](https://docs.mongodb.com/master/tutorial/enable-authentication/). Zignoruj ostrzeżenia o braku brak konfiguracji autoryzacji dostępu do danych.

mongo --port 27011

1. Powtórz punkty od 2 do 5 dla replik RSBD2 i RSBD3 działających na portach 27012 i 27013.
2. W powłoce Mongo repliki RSBD1 zainicjuj zbiór replikacji *RsbdRepSet*:

rs.initiate()

1. W powłoce Mongo repliki RSBD1 dodaj repliki RSBD2 i RSBD2 do zbióru replikacji *RsbdRepSet*:

rs.add("RSBD:27012")

rs.add("RSBD:27013")

1. W powłoce Mongo repliki RSBD1 sprawdź konfigurację replikacji:

rs.conf()

## Testowanie mechanizmu replikacji danych

1. W powłoce Mongo repliki RSBD1 utwórz przykładowy dokument w kolekcji *pracownicy*.

db.pracownicy.insertOne({nazwisko: "Kowalski", etat: "Stazysta", placa: 2222})

1. W powłoce Mongo repliki RSBD2 spróbuj odczytać zawartość kolekcji *pracownicy*. [Raport]

db.pracownicy.find()

1. W powłoce Mongo repliki RSBD2 zezwól na odczyt danych z repliki typu *slave*:

rs.secondaryOk()

1. Powtórz krok 2 .
2. Powtórz kroki 3 i 4 w powłoce Mongo repliki RSBD3.
3. W powłoce Mongo repliki RSBD2 spróbuj dodać nowy dokument do kolekcji *pracownicy*. [Raport]

## Testowanie mechanizmu przełączenia awaryjnego

1. W terminalu pomocniczym zasymuluj awarię repliki RSBD2 typu *slave*:

kill -9 `cat ~/mongo/RSBD2/mongod.pid`

1. W powłoce Mongo repliki RSBD2 spróbuj odczytać zawartość kolekcji *pracownicy*. [Raport]
2. W powłoce Mongo repliki RSBD1 utwórz przykładowy dokument w kolekcji *pracownicy*.
3. W powłoce Mongo repliki RSBD3 spróbuj odczytać zawartość kolekcji *pracownicy*.
4. W terminalu pomocniczym zasymuluj awarię repliki RSBD3 typu *slave*:

kill -9 `cat ~/mongo/RSBD3/mongod.pid`

1. W powłoce Mongo repliki RSBD1 spróbuj utworzyć przykładowy dokument w kolekcji *pracownicy*. [Raport]
2. W powłoce Mongo repliki RSBD1 sprawdź status replikacji. [Raport]

rs.status()

1. W terminalu pomocniczym uruchom replikę RSBD2.

mongod --config RSBD2.conf

1. W powłoce Mongo repliki RSBD1 sprawdź status replikacji. [Raport]
2. W powłoce Mongo repliki RSBD2 odczytaj zawartość kolekcji *pracownicy*. [Raport]
3. W terminalu pomocniczym uruchom replikę RSBD3.

## Dodanie arbitra do zbioru replikacji

Zadaniem arbitra jest zwiększenie liczby węzłów zapewniających kworum w głosowaniu przy ustalaniu statusu replikacji. Arbiter nie replikuje danych w związku z tym nie jest mocno obciążony i może być skonfigurowany na maszynie, która pełni inne funkcje np. wspiera serwer aplikacji.

1. Utwórz replikę RSBD4 działającą na porcie 27014 korzystając z kroków 2 i 3 rozdziału 2.
2. W powłoce Mongo repliki typu *master* (Primary), zapewne będzie to replika RSDB1, dodaj arbitra do zbioru replikacji:

rs.addArb("RSBD:27014")

1. W celu zademonstrowania działania arbitra należy ustalić nieparzystą liczbę replik w zbiorze replikacji (licząc również arbitra/arbitrów), w naszym scenariuszu można by dodać dodatkową replikę, natomiast łatwiej będzie jedną z nich usunąć. W powłoce Mongo repliki typu *master* (Primary), zapewne będzie to replika RSDB1, usuń replikę RSBD3 ze zbioru replikacji:

rs.remove("RSBD:27013")

1. W powłoce Mongo repliki RSBD1 sprawdź konfigurację replikacji:

rs.conf()

1. W powłoce Mongo repliki RSBD1 sprawdź status replikacji.

rs.status()

1. Otwórz terminal i nazwij go RSBD4 (Terminal->Set title).
2. W terminalu RSBD4 przyłącz się powłokę Mongo do repliki RSBD4.

mongo --port 27014

1. W powłoce Mongo repliki RSBD4 spróbuj odczytać zawartość kolekcji *pracownicy*. [Raport]
2. W terminalu pomocniczym zasymuluj awarię repliki typu *slave*, załóżmy, że jest to replika RSBD2:

kill -9 `cat ~/mongo/RSBD2/mongod.pid`

1. W powłoce Mongo repliki RSBD1 sprawdź status replikacji.

rs.status()

1. W powłoce Mongo repliki typu *master* dodaj nowy dokument do kolekcji *pracownicy.* [Raport]

## Rekonfiguracja replikacji po trwałym utraceniu kworum

W przypadku trwałego utracenia repliki należy ją usunąć z konfiguracji replikacji.

1. Zasymuluj awarię arbitra:

kill -9 `cat ~/mongo/RSBD4/mongod.pid`

1. Zasymuluj trwałe uszkodzenie 3 replik danych:

rm -rf ~/mongo/RSBD2 ~/mongo/RSBD3 ~/mongo/RSBD4

1. W powłoce Mongo repliki RSBD1 sprawdź status replikacji.

rs.status()

1. W powłoce Mongo repliki RSBD1 spróbuj usunąć uszkodzone repliki z konfiguracji replikacji.

rs.remove("RSBD:27012")

rs.remove("RSBD:27014")

1. W powłoce Mongo repliki RSBD1 wymuś usunięcie uszkodzonych repliki z konfiguracji replikacji.

cfg = rs.conf()

cfg.members.splice(1,2)

rs.reconfig(cfg)

rs.reconfig(cfg,{force:1})

1. W powłoce Mongo repliki RSBD1 sprawdź status replikacji. [Raport]

rs.status()

1. W powłoce Mongo repliki typu *master* dodaj nowy dokument do kolekcji *pracownicy.* [Raport]