# Настройка докер

name: mongo-sharding-repl

services:

configSrv:

image: mongo:latest

container\_name: configSrv

restart: always

ports:

- "27017:27017"

networks:

app-network:

ipv4\_address: 173.17.0.10

volumes:

- config-data:/data/db

command:

[

"--configsvr",

"--replSet", "config\_server",

"--bind\_ip\_all",

"--port", "27017"

]

healthcheck:

test: [ "CMD", "mongo", "--eval", "db.adminCommand('ping')" ]

interval: 5s

start\_period: 10s

shard1:

image: mongo:latest

container\_name: shard1

restart: always

ports:

- "27018:27018"

networks:

app-network:

ipv4\_address: 173.17.0.11

volumes:

- shard1-data:/data/db

command:

[

"--shardsvr",

"--replSet", "shard1",

"--bind\_ip\_all",

"--port", "27018"

]

healthcheck:

test: [ "CMD", "mongo", "--eval", "db.adminCommand('ping')" ]

interval: 5s

start\_period: 10s

shard2:

image: mongo:latest

container\_name: shard2

restart: always

ports:

- "27019:27019"

networks:

app-network:

ipv4\_address: 173.17.0.8

volumes:

- shard2-data:/data/db

command:

[

"--shardsvr",

"--replSet", "shard2",

"--bind\_ip\_all",

"--port", "27019"

]

healthcheck:

test: [ "CMD", "mongo", "--eval", "db.adminCommand('ping')" ]

interval: 5s

start\_period: 10s

mongos\_router:

image: mongo:latest

container\_name: mongos\_router

restart: always

ports:

- "27020:27020"

networks:

app-network:

ipv4\_address: 173.17.0.12

command:

[

"mongos",

"--configdb", "config\_server/configSrv:27017",

"--bind\_ip\_all",

"--port", "27020"

]

healthcheck:

test: [ "CMD", "mongo", "--eval", "db.adminCommand('ping')" ]

interval: 5s

start\_period: 10s

pymongo\_api:

container\_name: pymongo\_api

build:

context: api\_app

dockerfile: Dockerfile

image: kazhem/pymongo\_api1:1.0.0

depends\_on:

- mongos\_router

ports:

- 8080:8080

networks:

app-network:

ipv4\_address: 173.17.0.9

environment:

MONGODB\_URL: "mongodb://mongos\_router:27020" # Уточните порт для подключения

MONGODB\_DATABASE\_NAME: "somedb"

REDIS\_URL: "redis://redis\_service:6379"

shard1\_1:

image: mongo:latest

container\_name: shard1\_1

restart: always

ports:

- "27021:27021"

networks:

app-network:

ipv4\_address: 173.17.0.13

volumes:

- shard1\_1-data:/data/db

command:

[

"--shardsvr",

"--replSet", "shard1",

"--bind\_ip\_all",

"--port", "27021"

]

healthcheck:

test: [ "CMD", "mongo", "--eval", "db.adminCommand('ping')" ]

interval: 5s

start\_period: 10s

shard1\_2:

image: mongo:latest

container\_name: shard1\_2

restart: always

ports:

- "27022:27022"

networks:

app-network:

ipv4\_address: 173.17.0.14

volumes:

- shard1\_2-data:/data/db

command:

[

"--shardsvr",

"--replSet", "shard1",

"--bind\_ip\_all",

"--port", "27022"

]

healthcheck:

test: [ "CMD", "mongo", "--eval", "db.adminCommand('ping')" ]

interval: 5s

start\_period: 10s

shard1\_3:

image: mongo:latest

container\_name: shard1\_3

restart: always

ports:

- "27023:27023"

networks:

app-network:

ipv4\_address: 173.17.0.15

volumes:

- shard1\_3-data:/data/db

command:

[

"--shardsvr",

"--replSet", "shard1",

"--bind\_ip\_all",

"--port", "27023"

]

healthcheck:

test: [ "CMD", "mongo", "--eval", "db.adminCommand('ping')" ]

interval: 5s

start\_period: 10s

shard2\_1:

image: mongo:latest

container\_name: shard2\_1

restart: always

ports:

- "27024:27024"

networks:

app-network:

ipv4\_address: 173.17.0.16

volumes:

- shard2\_1-data:/data/db

command:

[

"--shardsvr",

"--replSet", "shard2",

"--bind\_ip\_all",

"--port", "27024"

]

healthcheck:

test: [ "CMD", "mongo", "--eval", "db.adminCommand('ping')" ]

interval: 5s

start\_period: 10s

shard2\_2:

image: mongo:latest

container\_name: shard2\_2

restart: always

ports:

- "27025:27025"

networks:

app-network:

ipv4\_address: 173.17.0.17

volumes:

- shard2\_2-data:/data/db

command:

[

"--shardsvr",

"--replSet", "shard2",

"--bind\_ip\_all",

"--port", "27025"

]

healthcheck:

test: [ "CMD", "mongo", "--eval", "db.adminCommand('ping')" ]

interval: 5s

start\_period: 10s

shard2\_3:

image: mongo:latest

container\_name: shard2\_3

restart: always

ports:

- "27026:27026"

networks:

app-network:

ipv4\_address: 173.17.0.18

volumes:

- shard2\_3-data:/data/db

command:

[

"--shardsvr",

"--replSet", "shard2",

"--bind\_ip\_all",

"--port", "27026"

]

healthcheck:

test: [ "CMD", "mongo", "--eval", "db.adminCommand('ping')" ]

interval: 5s

start\_period: 10s

redis:

image: "redis:latest"

container\_name: redis\_service

ports:

- "6379"

volumes:

- redis-data:/data

- ./redis/redis.conf:/usr/local/etc/redis/redis.conf

command: [ "redis-server", "/usr/local/etc/redis/redis.conf" ]

networks:

app-network:

ipv4\_address: 173.17.0.2

networks:

app-network:

driver: bridge

ipam:

driver: default

config:

- subnet: 173.17.0.0/16

volumes:

config-data:

shard1-data:

shard2-data:

shard1\_1-data:

shard1\_2-data:

shard1\_3-data:

shard2\_1-data:

shard2\_2-data:

shard2\_3-data:

redis-data:

# Скрипт

#!/bin/bash

set -x

# Инициализируем БД

echo "Initiate the config server"

docker exec -i configSrv mongosh --host 173.17.0.10 --port 27017 <<EOF

rs.initiate({

\_id: "config\_server",

configsvr: true,

members: [

{ \_id: 0, host: "configSrv:27017" }

]

});

exit;

EOF

if [ $? -ne 0 ]; then

echo "Ошибка инициализации config server"

fi

echo "Initiate shard1"

docker exec -i shard1 mongosh --host 173.17.0.11 --port 27018 <<EOF

rs.initiate({

\_id: "shard1",

members: [

{ \_id: 0, host: "shard1:27018" },

{ \_id: 1, host: "shard1\_1:27021" },

{ \_id: 2, host: "shard1\_2:27022" },

{ \_id: 3, host: "shard1\_3:27023" }

]

});

exit;

EOF

if [ $? -ne 0 ]; then

echo "Ошибка инициализации shard1"

fi

echo "Initiate shard2"

docker exec -i shard2 mongosh --host 173.17.0.8 --port 27019 <<EOF

rs.initiate({

\_id: "shard2",

members: [

{ \_id: 0, host: "shard2:27019" },

{ \_id: 1, host: "shard2\_1:27024" },

{ \_id: 2, host: "shard2\_2:27025" },

{ \_id: 3, host: "shard2\_3:27026" }

]

});

exit;

EOF

if [ $? -ne 0 ]; then

echo "Ошибка инициализации shard2"

fi

# Инициализация маршрутизатора

echo "Initiate the mongos router"

docker exec -i mongos\_router mongosh --host 173.17.0.12 --port 27020 <<EOF

sh.addShard("shard1/shard1:27018,shard1\_1:27021,shard1\_2:27022,shard1\_3:27023");

sh.addShard("shard2/shard2:27019,shard2\_1:27024,shard2\_2:27025,shard2\_3:27026");

sh.enableSharding("somedb");

sh.shardCollection("somedb.helloDoc",{"name":"hashed"});

use somedb;

for(var i = 0; i < 1000; i++) db.helloDoc.insertOne({age:i, name:"ly"+i});

db.helloDoc.countDocuments();

EOF

if [ $? -ne 0 ]; then

echo "Ошибка при инициализации маршрутизатора mongos"

fi