Лабораторна робота №9

З дисципліни: Бази даних та інформаційні системи
Студента групи МІТ-31: Добровольського Дмитра

Тема: Розширені можливості Redis

Мета роботи: Закріпити знання про роботу з Redis та ознайомитися з розширеним функціоналом — транзакціями, скриптами на Lua, публікацією/підпискою (Pub/Sub) та потоками Redis Streams
Завдання:

1. Транзакції у Redis

Ознайомтесь з командами MULTI, EXEC, DISCARD, WATCH.

WATCH balance
MULTI
DECR balance
INCR purchases
EXEC

Створіть просту транзакцію, що додає кілька ключів.

Приклад базової транкзакції:

```
127.0.0.1:6379> MULTI
OK
127.0.0.1:6379(TX)> SET key1 "Hello"
QUEUED
127.0.0.1:6379(TX)> SET key2 "World"
QUEUED
127.0.0.1:6379(TX)> EXEC
1) OK
2) OK
```

Спробуйте використати WATCH для імітації конкурентного доступу.

Після EXEC обидві команди виконуються одночасно.

Імітація конкурентного доступу з WATCH
У CLI 1:

```
127.0.0.1:6379> SET balance 100
OK
127.0.0.1:6379> SET purchases 0
OK
127.0.0.1:6379> WATCH balance
OK
127.0.0.1:6379> MULTI
OK
127.0.0.1:6379(TX)> DECR balance
QUEUED
127.0.0.1:6379(TX)> INCR purchases
QUEUED
```

У CLI 2 (нове вікно):

```
127.0.0.1:6379> SET balance 999
OK
```

В CLI 1 вводимо і ось що бачимо:

```
127.0.0.1:6379(TX)> EXEC
(nil)
```

Redis відповіла (nil) — транзакція не виконалась, бо balance змінився під час WATCH.

2. Lua-скрипти в Redis

Напишіть простий Lua-скрипт, який перевіряє наявність ключа і створює його, якщо не існує.

Наприклад:

```
EVAL "if redis.call('exists', KEYS[1]) == 0 then
  return redis.call('set', KEYS[1], ARGV[1])
else
  return 'exists'
end"
```

1 testkey "value"

```
127.0.0.1:6379> EVAL "if redis.call('exists', KEYS[1]) == 0 then return redis.call('set', KEYS[1], ARGV[1]) else return 'exists' end" 1 testkey "value"
"exists"
```

Якщо запусти скрипт перший раз → створить testkey,якщо запусти ще раз → Redis скаже 'exists'.(на скриншоті 2 варіант)

3. Механізм Pub/Sub

В одному терміналі запустіть підписку на канал:

SUBSCRIBE news

```
127.0.0.1:6379> SUBSCRIBE news
1) "subscribe"
2) "news"
3) (integer) 1
eading messages... (press Ctrl-C to quit or any key to type command)
```

В іншому терміналі виконайте публікацію повідомлення:

PUBLISH news "Redis is awesome!"

```
127.0.0.1:6379> PUBLISH news "Redis is awesome!"
(integer) 1
127.0.0.1:6379>
```

Перше вікно отримало повідомлення:

```
127.0.0.1:6379> SUBSCRIBE news
1) "subscribe"
2) "news"
3) (integer) 1
1) "message"
2) "news"
3) "Redis is awesome!"
```

Вивчіть можливість використання Pub/Sub для чатів або нотифікацій.

4. Redis Streams (потоки даних)

Додайте події до потоку:

XADD mystream * sensor-id 1234 temperature 19.8

```
127.0.0.1:6379> XADD mystream * sensor-id 1234 temperature 19.8
"1744286269113-0"
127.0.0.1:6379> XADD mystream * sensor-id 1235 temperature 20.1
"1744286269116-0"
```

Прочитайте останні події:

XRANGE mystream - +

```
127.0.0.1:6379> XRANGE mystream - +
1) 1) "1744286269113-0"
   2) 1) "sensor-id"
      2) "1234"
      3) "temperature"
      4) "19.8"
2) 1) "1744286269116-0"
   2) 1) "sensor-id"
      2) "1235"
      3) "temperature"
      4) "20.1"
```

Створіть споживача потоку та зчитайте нові повідомлення:

XREAD COUNT 2 STREAMS mystream 0

```
127.0.0.1:6379> XREAD COUNT 2 STREAMS mystream 0
1) 1) "mystream"
   2) 1) 1) "1744286269113-0"
         2) 1) "sensor-id"
            2) "1234"
            3) "temperature"
            4) "19.8"
      2) 1) "1744286269116-0"
         2) 1) "sensor-id"
            2) "1235"
            3) "temperature"
            4) "20.1"
```

Після першого разу можна використовувати ID останнього повідомлення, щоб зчитувати тільки нові:

```
127.0.0.1:6379> XREAD STREAMS mystream 1712593331424-0
1) 1) "mystream"
   2) 1) 1) "1744286269113-0"
         2) 1) "sensor-id"
            2) "1234"
            3) "temperature"
            4) "19.8"
      2) 1) "1744286269116-0"
         2) 1) "sensor-id"
            2) "1235"
            3) "temperature"
            4) "20.1"
```

Висновок:

Під час лабораторної роботи було закріплено знання про роботу з Redis та ознайомився з розширеним функціоналом — транзакціями, скриптами на Lua, публікацією/підпискою (Pub/Sub) та потоками Redis Streams.