Assignment: Dockerized Redis

Name: Fareed Hassan Khan

ERP ID: 25367

Course: Big Data Analytics

Course Code: 60490

Project Description

This project is done on docker container creating a containerized Redis environment. Showing all the commands that are required for any naïve user to understand the basics of Redis.

command 1 - 4 shows how to initialize the Redis environment in docker container along with uploading the user data in the database.

While the main commands section shows the basics of Redis. A total of 61 commands has been used in this assignment explaining several topics in redis.

Dataset Description

Multiple datasets are used in this assignment. Below is the link from where the user can download the dataset in order to make the commands work.

Dataset	Download Link
Users' dataset	redis-datasets/user-d ... s · GitHub
IPL dataset	IPL-Dataset/data ... GitHub
Baby names dataset	data-baby-names/b ... names · GitHub

Docker-compose file data for creating a Redis container

```yaml
version: "3.8"
services:
 redis:
  image: redis
  container_name: redis-assignment
  environment:
   - PUID=1000
   - PGID=1000
  volumes:
   - 'D:/redis/database:/data'
  ports:
   - 27017:27017
  restart: "unless-stopped"
```

---

1) docker-compose up -d

```
[+] Running 7/7
 - redis Pulled                                    40.8s
  - c229119241af Pull complete                     35.3s
  - bed41eb8190a Pull complete                     35.4s
  - 5e59eaa723f1 Pull complete                     35.5s
  - fd5ad7669819 Pull complete                     35.8s
  - 566c064eef6e Pull complete                     35.8s
  - 20c7cfac25de Pull complete                     35.9s
[+] Running 1/1
 - Container redis-assignment  Started
```

---

2) docker ps

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
f965945c0c67	redis	"docker-entrypoint.s..."	20 minutes ago	Up 20 minutes	6379/tcp	redis-assignment

---

3) docker exec -it redis-assignment su

```
root@f965945c0c67:/
```

---

4) root@a0f80020ef88:/# cd data/redis-data-master

Main Commands

User's Dataset Description

Users' dataset is collected from GitHub website. Each id contains the following 22 unique element information.

Element	Type	Description
First_name	String (unique)	First name of the user
Last_name	String (unique)	Last name of the user
Gender	String	Name of the movie
Email	String	Email of the user
...	...	...
Latitude	String	Latitude value of his home
Last_login	String	Last login of the website (timestamp)

---

1) Importing import-users.redis file in redis database

```
redis-cli < ./import_users.redis
```

```
(integer) 11
(integer) 11
...
...
(integer) 11
(integer) 11
"5996 Users Created"
```

---

2) Following operation retrieve the information showing total keys with extra information.

```
info keyspace
```

```
# Keyspace
db0: keys = 5996, expires = 0, avg_ttl = 0
```

3) Following operation retrieve all the information of specific key (i.e., user 258).

```
HGETALL user:258
```

```
1) "first_name"
2) "Anetta"
3) "last_name"
4) "Musla"
5) "email"
6) "amusla75@tamu.edu"
7) "gender"
8) "female"
...  ... ...
16) "Nong Bun Nak"
17) "longitude"
18) "102.368115"
19) "latitude"
20) "14.740841"
21) "last_login"
22) "1573946075"
```

---

4) Printing all keys.

```
Keys *
```

```
1) "user:3450"
2) "user:4496"
3) "user:3820"
4) "user:5722"
5) "user:4195"
...
5992) "user:4430"
5993) "user:3621"
5994) "user:4873"
5995) "user:712"
5996) "user:258"
```

---

5) Printing first_name of a single key (i.e., user 271).

```
HMGET user:271 first_name
```

```
"Lazare"
```

6) Printing last_name of a single key (i.e., user 271).

HMGET user:271 last_name

"Cuthbert"

---

7) Adding a new field to user:3130.

hmset user:3130 outlook notgiving

Ok

---

8) Finding the newly inserted field for user:3130.

hmget user:3130 outlook

“notgiving”

---

9) Deleting the newly inserted field for user:3130.

hdel user:3130 outlook

(integer) 1

---

10) Increment the last_login value by 10 of user:3130.

HINCRBY user:3130 last_login 10

(integer) 1584821549

---

11) Finding the newly incremented field for user:3130

hmget user:3130 last_login

“1584821549”

12) Giving the key user:99 a new value (i.e., new_99)

```
set user:99 new_99
```

OK

```
get user:99
```

“new_99”

---

13) Removing all keys

```
flushall
```

OK

---

14) Checking if field exist or not? (Also indicates that redis queries are case sensitive).

```
hexists user:100 first_name
```

(integer) 1

```
hexists user:100 First_name
```

(integer) 0

---

15) Setting a field already exists (No change will take place).

```
hsetnx user:100 first_name fareed
```

(integer) 0

16) Returning only the fields, not their values.

```
hkeys user:100
```

```
1) "first_name"
2) "last_name"
3) "email"
4) "gender"
...
7) "country_code"
8) "city"
9) "longitude"
10) "latitude"
11) "last_login"
```

---

17) Returning values only, not the field names.

```
hvals user:100
```

```
1) "Anderson"
2) "Bithany"
3) "abithany2r@sbwire.com"
4) "male"
5) "222.164.174.233"
6) "Poland"
7) "PL"
8) "\xc5\x81apczyca"
9) "20.3412605"
10) "49.9513027"
11) "1581119052"
```

---

18) Returning total number of fields.

```
hlen user:100
(integer) 11
```

19) Returning total number of fields.

```
hlen user:100
(integer) 11
```

IPL Dataset Description

IPL dataset is collected from GitHub website. Following are the three columns, that have been used for explaining the set operation in Redis.

Element	Type	Description
Team1	String	Name of the team1
Team2	String	Name of the team2
Winner	String	Name of team (Either team1 or team2)

---

20) Copying the csv in docker redis container.

```
Docker cp ipl.csv redis-assignment:/data
```

---

21) adding team1 as a set in redis-cli using awk command.

```
cat ipl.csv | awk -F',' '{print " SADD \"team1\" \""$7"\" \n" }' | redis-cli --pipe

All data transferred. Waiting for the last reply...
Last reply received from server.
errors: 0, replies: 817
```

---

22) adding winners as a set in redis-cli using awk command.

```
cat ipl.csv | awk -F',' '{print " SADD \"winners\" \""$11"\" \n" }' | redis-cli --pipe

All data transferred. Waiting for the last reply...
Last reply received from server.
errors: 0, replies: 817
```

23) Printing the output of both team1 and winners set.

```
smembers team1

1) "Rajasthan Royals"
2) "Mumbai Indians"
3) "Kochi Tuskers Kerala"
4) "Sunrisers Hyderabad"
•••
14) "Deccan Chargers"
15) "Rising Pune Supergiants"
16) "Delhi Daredevils"
```

```
smembers winners

1) "Rajasthan Royals"
2) "Mumbai Indians"
5) "field"
•••
16) "Deccan Chargers"
17) "Kolkata Knight Riders"
18) "Rising Pune Supergiants"
19) "Delhi Daredevils"
```

---

24) Set difference between team1 and winners set.

```
sdiff team1 winners

1) "team1"
2) "0"
```

```
sdiff winners team1

1) "field"
2) "winner"
3) "bat"
4) "NA"
```

---

25) Intersection of team1 and winners.

(Output will be team1, since each team has won the match in their journey)

```
sinter team1 winners

1) "Rajasthan Royals"
2) "Mumbai Indians"
3) "Kochi Tuskers Kerala"
4) "Sunrisers Hyderabad"
•••
14) "Deccan Chargers"
15) "Rising Pune Supergiants"
16) "Delhi Daredevils"
```

26) Confirming that Kashmir has not been the part of IPL teams yet.

```
sismember team1 Kashmir
```

```
(integer) 0
```

---

27) Randomly choosing winner of IPL 2022.

```
srandmember team1
```

```
"Delhi Daredevils"
```

---

28) Scanning the winners set and searching for the first team's name start with S?

```
sscan winners 0 match S*
```

```
1) "6"
2) 1) "Sunrisers Hyderabad"
```

---

29) Converting the normal set to a sorted set using zinterstore command

```
ZINTERSTORE zset-copy 1 team1
```

```
1) "0"
2) "Chennai Super Kings"
3) "Deccan Chargers"
4) "Delhi Capitals"
...
15) "Royal Challengers Bangalore"
16) "Sunrisers Hyderabad"
17) "team1"
```

30) Printing first four items of sorted set.

```
zrange zset-copy 0 3


1) "0"
2) "Chennai Super Kings"
3) "Deccan Chargers"
4) "Delhi Capitals"
```

---

31) Printing first four items of sorted set with scores.

```
zrange zset-copy 0 3 withscore


1) "0"
2) "1"
3) "Chennai Super Kings"
...
8) "1"
9) "Delhi Daredevils"
10) "1"
```

---

32) Using zincrby for including a new team with rest of the teams.

```
zincrby zset-copy 2 Kashmir
"2"
```

---

33) Confirming the newly inserted value (As you can see Kashmir is at 18).

```
zincrby zset-copy 2 Kashmir


1) "0"
...
17) "team1"
18) "Kashmir"
```

Names Dataset Description

Names dataset is collected from GitHub website. Following is the column, that have been used for explaining the list operation in Redis.

Element	Type	Description
Name	String	Contains boy and girl name

---

34) Copying the csv in docker redis container.

```
Docker cp names.csv redis-assignment:/data
```

---

35) Adding penames as a list in redis-cli using awk command.

```
cat names.csv | awk -F',' '{print " RPUSH \"penames\" \""$1"\" \n" }' | redis-cli --pipe
```

```
All data transferred. Waiting for the last reply...
Last reply received from server.
errors: 0, replies: 258001
```

---

36) Printing the output of list penames

```
lrange penames 0 -1
```

```
"name"
"John"
...
"Sloane"
"Elianna"
```

37) Including my own name in the list

```
lpush penames fareed
```

258002

---

38) Couting total number of elements in penames list

```
llen penames
```

258002

---

39) Checking whether fareed element exist in penames list or not?

```
LPOS penames fareed
```

(integer) 0 (i.e., the position of the element)

---

40) Popping the first element of the list.

```
LPOP penames
```

"fareed"

---

41) Deleting the list.

```
del penames
```

(integer) 1

Essay Dataset Description

Cricket Essay is collected from a blogging site for explaining the string operation in Redis.

---

42) Printing the first word of the string.

```
Getrange mystring 0 6
```

"Cricket"

---

43) Printing the entire string.

```
Getrange mystring 0 -1
```

"Cricket is a sport that requires the use of a bat is easily ••• different set of rules and duration."

---

44) Printing multiple keys at once.

```
Mget key1 mystring
```

1) "first_key"

2) "Cricket is a sport that requires the use of a bat is easily ... different set of rules and duration"

---

45) Replacing Cricket with Hockey.

```
setrange mystring 0 "Hockey "
```

(integer) 4588

46) Printing the newly updated string.

```
Getrange mystring 0 -1
```

"Hockey is a sport that requires the use of a bat is easily ••• different set of rules and duration."

---

47) Appending a new item at the end of the string.

```
append mystring “the end value”
```

"Hockey is a sport that requires the is easily ••• different set of rules and duration. The end value"

---

48) Finding the length of the string.

```
STRLEN mystring
```

(integer) 4591

Redis Streams

Redis Streams can help us to do a lot more than what was possible with Pub/Sub, Lists, and Sorted Sets. Among the many benefits, Redis Streams enables to collect large volumes of data arriving in high velocity.

49) XADD is used in Redis that can add data to a stream.

```
XADD mystreamdata * first_name Fareed last_name Khan
```

"1650026051575-0"

(It returns a unique ID for each item that is being inserted in the streaming data)

50) XRANGE is used to print the stream the data

```
XRANGE mystream - +
```

```
1) 1) "1650026051575-0"
   2) 1) "first_name"
      2) "Fareed"
      3) "last_name"
      4) "Khan"
```

---

51) Printing the specific range id's stream data.

```
XRANGE mystreamdata 1650026051575 1650026051580
```

```
1) 1) "1650026051575-0"
   2) 1) "first_name"
      2) "Fareed"
      3) "last_name"
      4) "Khan"
```

---

52) Printing the stream data based on specific id's range.

```
XRANGE mystreamdata 1650026051575 1650026051580
```

```
1) 1) "1650026051575-0"
   2) 1) "first_name"
      2) "Fareed"
      3) "last_name"
      4) "Khan"
```

53) Adding more stream data.

```
XADD mystreamdata * game cricket player_age 25
```

```
"1650035632393-0"
```

---

54) Printing all stream data.

```
XRANGE mystreamdata 1650026051575 1650026051580
```

```
1)  1) "1650035585488-0"
    2) 1) "first_name"
       2) "Osama"
       3) "last_name"
       4) "nayal"
2)  1) "1650035632393-0"
    2) 1) "game"
       2) "cricket"
       3) "player_age"
       4) "25"
```

---

55) Fetching the latest entry in stream data.

```
XREVRANGE mystreamdata 1650035585488 1650035122222
```

```
1)  1) "1650035632393-0"
    2) 1) "game"
       2) "cricket"
       3) "player_age"
       4) "25"
```

56) Trimming our stream data with XTRIM.

```
xtrim mystreamdata maxlen 1
```

```
integer 1
```

This will trim our stream data and maxlen “1” will decide the length of stream data is of 1. The most recent id’s will be the part of streaming data from now on.

---

57) Printing the updated streaming data.

```
XRANGE mystreamdata - +
```

```
1) 1) "1650035632393-0"
2) 1) "game"
   2) "cricket"
   3) "player_age"
   4) "25"
```