Module: EGT206 Data Management

Practical 9: Hands on Redis with Python

Activity1: Redis Installation

1. Go to XpanAI and launch a terminal.

2. Type the following commands to install redis on your machine:

   apt-get update

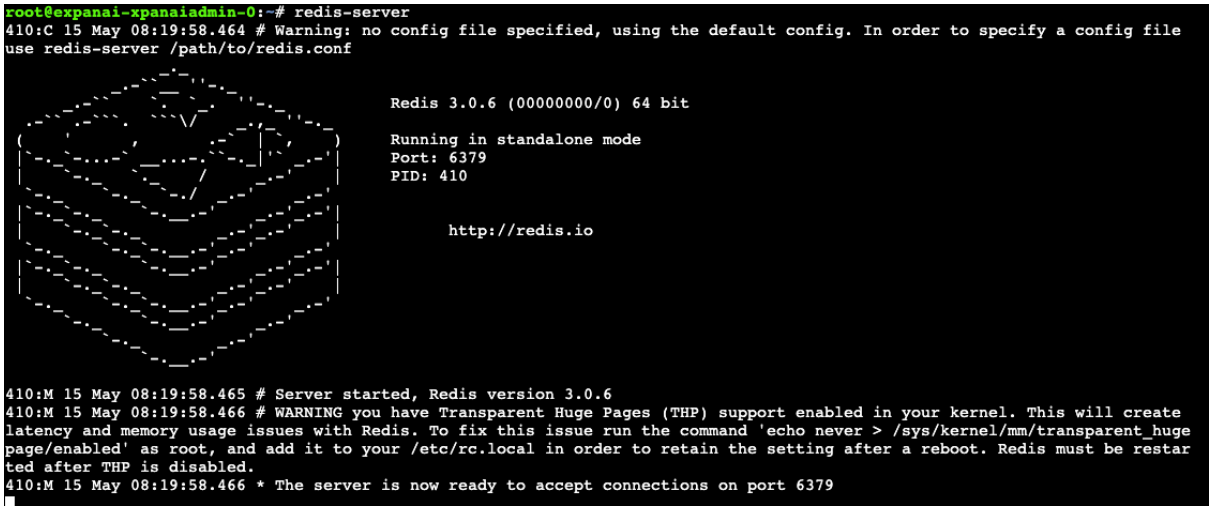
   apt-get install redis-server

```
The following NEW packages will be installed:
  libjemalloc1 redis-server redis-tools
0 upgraded, 3 newly installed, 0 to remove and 67 not upgraded.
Need to get 519 kB of archives.
After this operation, 1507 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

   [Type 'y' when prompted]

3. To start redis:

   redis-server



4. Start a new terminal and type following command:
   redis-cli
   This will open a redis prompt.

```
127.0.0.1:6379>
```

   In the above prompt, 127.0.0.1 is your machine's IP address and 6379 is the port on which Redis server is running.

5. Now type the following PING command.
   ping

```
127.0.0.1:6379> ping
PONG
```

   This shows that Redis is successfully installed on your machine.

Activity2: Redis python libraries installation

1. Download Prac9.ipynb from brightspace and upload to XpanAI workspace.

2. Run following command to install redis python libraries.

```
In [1]: !pip install redis

WARNING: The directory '/home/jovyan/.cache/pip/http' or its parent directory is not owned by
the current user and the cache has been disabled. Please check the permissions and owner of t
hat directory. If executing pip with sudo, you may want sudo's -H flag.
WARNING: The directory '/home/jovyan/.cache/pip' or its parent directory is not owned by the
current user and caching wheels has been disabled. check the permissions and owner of that di
rectory. If executing pip with sudo, you may want sudo's -H flag.
Collecting redis
  Downloading https://files.pythonhosted.org/packages/e0/7f/21a09f8c9f5db6f5e24432f7a0f916ca3
86025d74e4da6d0f5164aa9a78a/redis-4.5.5-py3-none-any.whl (240kB)
     |████████████████████████████████| 245kB 22.4MB/s eta 0:00:01
Requirement already satisfied: typing-extensions; python_version < "3.8" in /opt/conda/lib/py
thon3.7/site-packages (from redis) (3.10.0.2)
Requirement already satisfied: importlib-metadata>=1.0; python_version < "3.8" in /opt/conda/
lib/python3.7/site-packages (from redis) (4.10.1)
Collecting async-timeout>=4.0.2; python_full_version <= "3.11.2" (from redis)
  Downloading https://files.pythonhosted.org/packages/d6/c1/8991e7c5385b897b8c020cdaad718c5b0
87a6626d1d11a23e1ea87e325a7/async_timeout-4.0.2-py3-none-any.whl
Requirement already satisfied: zipp>=0.5 in /opt/conda/lib/python3.7/site-packages (from impo
rtlib-metadata>=1.0; python_version < "3.8"->redis) (3.7.0)
ERROR: aiohttp 3.7.3 has requirement async-timeout<4.0,>=3.0, but you'll have async-timeout
4.0.2 which is incompatible.
Installing collected packages: async-timeout, redis
  Found existing installation: async-timeout 3.0.1
    Uninstalling async-timeout-3.0.1:
      Successfully uninstalled async-timeout-3.0.1
Successfully installed async-timeout-4.0.2 redis-4.5.5
```

3. Run following commands to create connection to redis.

```
import redis
config = {
        "host": "127.0.0.1",
        "port": 6379,
}
r = redis.StrictRedis(**config)
r.info()
print ("Connected to DB-%d@%s:%d" % (r.connection_pool.connection_kwargs['db'],
r.connection_pool.connection_kwargs['host'], r.connection_pool.connection_kwargs['port']))
```

You should see following output.

```
import redis
# example connection parameters
config = {
    "host": "127.0.0.1",
    "port": 6379,
}

# actual connection parameters are provided by the Notebook environment
r = redis.StrictRedis(**config)
r.info()
print ("Connected to DB-%d@%s:%d" % (r.connection_pool.connection_kwargs['db'], r.connection_po

Connected to DB-0@127.0.0.1:6379
```

Activity3: Strings

Please complete the following activities using python codes(refer to Lecture Slides in Week 12).

Exercise

1. Assign following key-value pair.

Key	Value
Module1	English
Module2	Physics
Module3	Programming
Module5	Maths

```
r.mset({"Module1":"English", "Module2":"Physics", "Module3":"Programming", "Module5":"Maths"})
```

2. Check if key “Module4” exists or not in the database.
```
r.exists("Module4")
```
3. Check the value type stored at key “Module2”.
```
r.type("Module2")
```
4. Rename the key “Module5” to “Module4”.
```
r.rename("Module5", "Module4")
```
5. Get the first 5 character of key “Module3”.
```
r.getrange("Module3",0,2)
```

Activity4: Key expiration

Exercise

1. Set a key with string.

Key	Value	Expire Time
Module1	English	10 seconds

```
r.setex("Module1",10, "English")
```
2. Check the remaining time to live for the key.
(The command returns -2 if the key does not exist.
The command returns -1 if the key exists but has no associated expire.)
```
r.ttl("Module1")
```

Activity5: Lists

Exercise

1. Create the following key-value pair.

Key	Value
queue	1,3,5,6,7,8

```
r.rpush("queue",1,3,5,6,7,8)
```

2. Remove values 1,3 and 8 from the list, so that the list now looks like this.

Key	Value
queue	5,6,7

```
r.lpop("queue")
```
```
b'1'
```
```
r.lpop("queue")
```
```
b'3'
```
```
r.rpop("queue")
```
```
b'8'
```

3. Add values to the list, so that the list now looks like this.

Key	Value
queue	3,4,5,6,7,8,9

```
r.rpush("queue",8,9)
```
```
5
```
```
r.lpush("queue",4,3)
```
```
7
```

4. Check the length of the list.
```
r.llen("queue")
```
5. Get the third element from the list.
```
r.lindex("queue",2)
```
6. Trims the list to contain only 2nd to 5th element.
```
r.ltrim("queue",1,4)
```

Activity6: Sets

Exercise

1. Create the following key-value pair.

Key	Value
colorset1	blue orange green
colorset2	red purple yellow

```
r.sadd("colorset1","blue","orange","green")
r.sadd("colorset2","red","purple","yellow")
```

2. Test if “red” value belongs to set “colorset1”.
```
r.sismember("colorset1","red")
```

3. Check the size of the set “colorset2”.
```
r.scard("colorset2")
```
4. Remove value “orange” from the set “colorset1”.
```
r.srem("colorset1","orange")
```
5. Combine “colorset2” and “colorset1” together.
```
r.sunion("colorset1","colorset2")
```

Activity7: Hashes

Exercise

1. Create the following key-value pair.

Key	Value
carspecs	brand: Toyota model: Alphard colour: silver year: 2020

```
r.hmset("carspecs",{"brand":"Toyota","model":"Alphard","colour":"silver","year":2020})
```
2. Delete hash field “year:2020”.
```
r.hdel("carspecs","year")
```
3. Determine whether hash field “horsepower” exists or not.
```
r.hexists("carspecs","horsepower")
```

Activity8: Sorted Sets

Exercise

1. Assign following key-value pair.

Key	Value	Score
leaderboard	user1	60
	user2	80
	user3	55
	user4	90
	user5	20

```
r.zadd("leaderboard",{"user1":60, "user2":80,"user3":55,"user4":90,"user5":20})
```

2. Get the members in sorted set “leaderboard”, ordered by scores in descending order.
```
r.zrevrangebyscore("leaderboard",100,0,withscores=True)
```
3. How many members in sorted set “leaderboard’ have score between 60 and 90?
```
r.zcount("leaderboard",60,90)
```

4. Increment the score of user3 by 20.
```
r.zincrby("leaderboard",20,"user3")
```
5. Remove user5 from the sorted set.
```
r.zrem("leaderboard","user5")
```
6. Remove users with score lower than 60 from the sorted set.
```
r.zremrangebyscore("leaderboard",0,60)
```
7. Get the score of user4 from the sorted set “leaderboard”.
```
r.zscore("leaderboard","user4")
```

Activity9: Write a Python application (optional)

Imagine you have decided to start a website to sell hats. You will use Redis to handle some of the product and inventory tracking for the website.

Following is the three hats models that you have.

Key	Value
HatModelA	color: black price: 49.99 style: fitted quantity: 1000 npurchased: 0
HatModelB	color: maroon price: 39.99 style: hipster quantity: 500 npurchased: 0
HatModelC	color: green price: 19.99 style: baseball quantity: 200 npurchased: 0

Follow the steps below to create a python code to achieve the above.

```python
# step 1: import the redis-py client package
import redis

# step 2: define our connection information for Redis
config = {
    "host": "127.0.0.1",
    "port": 6379,
}

# step 3: create the Redis Connection object
r = redis.StrictRedis(**config)

def create_database():

    try:

        # step 4: Use a suitable data type to contain the product details.
        hatmodela = {"color": "Black","price":49.99, "style":"fitted", "quantity":1000, "npurchased": 0}
        r.hmset("HatModelA", hatmodela)
        hatmodelb = {"color": "maroon","price":39.99, "style":"hipster", "quantity":500, "npurchased": 0}
        r.hmset("HatModelB", hatmodelb)
        hatmodelc = {"color": "green","price":19.99, "style":"baseball", "quantity":200, "npurchased": 0}
        r.hmset("HatModelC", hatmodelc)

        # step 5: Check to make sure "HatModelC"-value"color" exists
        msg = r.hexists("HatModelC","color")
        print("Does HatModelC exist?", msg)

    except Exception as e:
        print(e)

def click_purchase(hat_model):
#Simulates when user clicks Purchase
    try:

        # step 6: Check if the item is in stock.
        # If yes, increase its npurchased by 1 and decrease its quantity by 1.
        if r.hget(hat_model,"quantity") > b"0":
            r.hincrby(hat_model, "npurchased", 1)
            r.hincrby(hat_model, "quantity", -1)
        else:
            print("Sorry,", hat_model , "is out of stock!")

        #step 7: Get updated info for that item.
        msg = r.hgetall(hat_model)
        print(msg)

    except Exception as e:
        print(e)

if __name__ == '__main__':
    create_database()
    click_purchase("HatModelC")
```