Assignment – 5

Submitted by Mudit Kumar g23ai2037

```
import csv
import re
from traceback import print_stack
from pyparsing import Regex
import redis
import pandas as pd
from redis.commands.search.field import TextField, NumericField, TagField
from redis.commands.search.indexDefinition import IndexDefinition
from redis.commands.search.query import Query
import sys



class Redis_Client:
    redis = None

    def _init_(self, host=None, port=None, db=None, username=None, password=None):
        self.host = host or "redis-15824-mudit.c264.us-east-1-1.ec2.redns.redis-cloud.com"
        self.port = port or 16682
        self.db = db or 0
        self.username = username or ''
        self.password = password or ''

        self.redis = redis.Redis(host=self.host, port=self.port, db=self.db, username=self.username,
password=self.password)

    def connect(self):
        self.redis.ping()
        print("Connected to Redis")
```

```
def load_users(self, file):
    """
    Load the users dataset into Redis DB.
    """
    pipeline = self.redis.pipeline()

    with open(file, "r", encoding="utf-8") as f:
        for line in f:
            data = line.strip().split(' ')
            user_id = data[0].replace('"', '')
            user_data = {k.strip('"'): v.strip('"') for k, v in zip(data[1::2], data[2::2])}
            pipeline.hset(user_id, mapping=user_data)
    pipeline.execute()
    print("User data loaded successfully.")

def load_scores(self, file):
    """
    Load the scores dataset into Redis DB.
    """
    pipeline = self.redis.pipeline()
    with open(file, "r") as f:
        reader = csv.reader(f)
        next(reader)  # Skip header
        for row in reader:
            user_id, score = row[0], float(row[1])
            pipeline.zadd("leaderboard:2", {user_id: score})
    pipeline.execute()
    print("Scores loaded successfully.")
```

```
def delete_all(self):
    """
    Erase everything in the DB.
    """
    self.redis.flushdb()

def query1(self, usr):
    """
    Return all the attributes of the user by usr.
    """
    print("Executing query 1.")
    usr = f"user:{usr}"

    result = self.redis.hgetall(usr)
    print(result)
    return result

def query2(self, usr):
    """
    Return the coordinate (longitude and latitude) of the user by the usr.
    """
    print("Executing query 2.")
    usr = f"user:{usr}"

    longitude = self.redis.hget(usr, "longitude")
    latitude = self.redis.hget(usr, "latitude")
    coordinates = {"longitude": longitude, "latitude": latitude}
    print(coordinates)
    return coordinates

def query3(self):
```

```
        """
        Get the keys and last names of the users whose ids do not start with an odd number.
        """
        print("Executing query 3.")
        keys = []
        last_names = []
        cursor = 1280

        while True:
            cursor, batch = self.redis.scan(cursor=cursor, match="user:*", count=100)
            for key in batch:
                user_id = key.decode().split(":")[-1]
                if user_id[0] not in ['1', '3', '5', '7', '9']:
                    last_name = self.redis.hget(key, "last_name")
                    keys.append(key.decode())
                    last_names.append(last_name.decode() if last_name else None)

            if cursor == 0:
                break

        print(f"Query3: Keys and last names of users whose IDs do not start with an odd number:
{list(zip(keys, last_names))}")


    def query4(self):
        """
        Return the female in China or Russia with the latitude between 40 and 46.
        """
        print("Executing query 4.")
        try:
            self.redis.ft("idx").create_index([
                TextField("gender"),
```

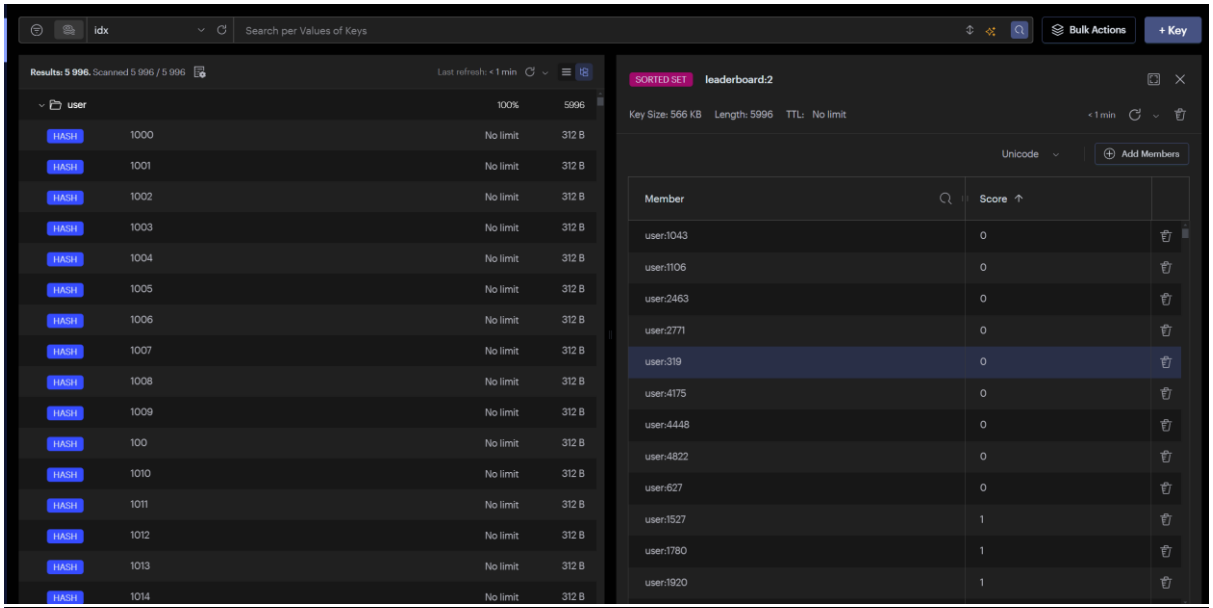
```
            TagField("country"),
            NumericField("latitude")
        ], definition=IndexDefinition(prefix=["user:"]))
        print("Index created successfully.")
    except Exception as e:
        print(f"Index creation error: {e}")

    query = Query('@gender:female @country:{China|Russia} @latitude:[40 46]')

    try:
        results = self.redis.ft("idx").search(query)
        if results.total > 0:
            print("Query4: Female users in China or Russia with latitude 40-46:")
            for doc in results.docs:
                print(f"ID: {doc.id}, Gender: {doc.gender}, Country: {doc.country}, Latitude:
{doc.latitude}")
        else:
            print("Query4: No matching users found.")
        return results
    except Exception as e:
        print(f"Search query error: {e}")
        return None

  def query5(self):
    """
    Get the email ids of the top 10 players (in terms of score) in leaderboard:2.
    """
    print("Executing query 5.")
    top_players = self.redis.zrevrange("leaderboard:2", 0, 9, withscores=True)

    emails_with_scores = []
```

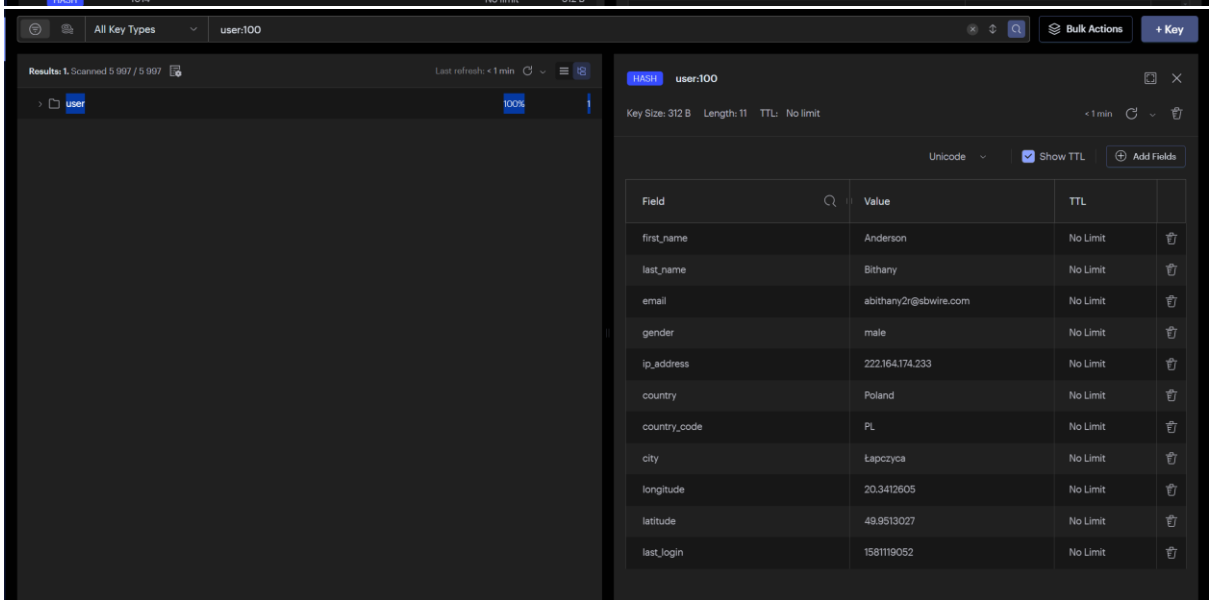
```
        for user_key, score in top_players:
            user_key = user_key.decode()
            email = self.redis.hget(user_key, "email")
            if email:
                emails_with_scores.append((email.decode(), score))
            else:
                emails_with_scores.append(("No email found", score))

        print("Top 10 players with their emails and scores:")
        for email, score in emails_with_scores:
            print(f"Email: {email}, Score: {score}")

        return emails_with_scores


rs = Redis_Client()
rs.connect()
rs.load_users("users.txt")
rs.load_scores("userscores.csv")
rs.query1(299)
rs.query2(2836)
rs.query3()
rs.query4()
rs.query5()
```

```
Connected to Redis
User data loaded successfully.
Scores loaded successfully.
Executing query 1.
{b'first_name': b'Susie', b'last_name': b'Jendrys', b'email': b'sjendrys8a@webmd.com', b'gender': b'female', b'ip_address': b'30.157.248.216', b'country': b'China', b'country_code'
', b'city': b'Xuetian', b'longitude': b'120.568856', b'latitude': b'29.00091', b'last_login': b'1571176806'}
Executing query 2.
{'longitude': b'1.9638715', 'latitude': b'47.9035673'}
```

Executing query 3.
Query3: Keys and last names of users whose IDs do not start with an odd number: [('user:4510', 'Hearley'), ('user:4590', 'Brownstein'), ('user:2492', 'Mendez'), ('user:872', 'Silverman'), ('user:2535', 'Pavlata'), ('user:2132', 'Quincey'), ('user:47', 'Iacomo'), ('user:2253', 'Sibery'), ('user:894', 'Bartens'), ('user:60', 'Davidde'), ('user:608', 'Sibbons'), ('user:2079', 'Wanjek'), ('user:4647', 'Conquest'), ('user:4180', 'Wallsam'), ('user:2305', 'Todhunter'), ('user:4218', 'Reason'), ('user:4756', 'Dudny'), ('user:2093', 'Huxster'), ('user:4479', 'De'), ('user:4487', 'Kroll'), ('user:613', 'New'), ('user:2273', 'Blees'), ('user:2863', 'Samson'), ('user:4685', 'Matula'), ('user:4243', 'Welham'), ('user:816', 'McMahon'), ('user:4779', 'Hughman'), ('user:4473', 'Simkins'), ('user:2557', 'Kenney'), ('user:4620', 'Hinkens'), ('user:4899', 'Calendar'), ('user:4252', 'Cottis'), ('user:2708', 'Roughan'), ('user:2756', 'Petrus'), ('user:857', 'Kelwick'), ('user:2458', 'Buye'), ('user:2832', 'Kelleher'), ('user:2130', 'Clausen-Thue'), ('user:4272', 'Szymaniak'), ('user:2902', 'Klimkowski'), ('user:2243', 'Smieton'), ('user:2591', 'Draycott'), ('user:2014', 'Pady'), ('user:2937', 'Van'), ('user:2485', 'Kimblin'), ('user:2161', 'Keizman'), ('user:2981', 'Bichard'), ('user:293', 'Spurriar'), ('user:2473', 'Bowmen'), ('user:4084', 'Franchi'), ('user:2136', 'Madrell'), ('user:4566', 'Raeburn'), ('user:4774', 'Gaveltone'), ('user:4941', 'Bebbell'), ('user:234', 'Maudlin'), ('user:870', "L'oiseau"), ('user:2212', 'Sumpner'), ('user:4097', 'Thomasen'), ('user:4683', 'Huchot'), ('user:2376', 'Gabbott'), ('user:2850', 'Yaakov'), ('user:4402', 'Lantaph'), ('user:4628', 'St.'), ('user:2550', 'Lanyon'), ('user:4251', 'McCorley'), ('user:4648', 'Sellors'), ('user:2842', 'Ganter'), ('user:4594', 'Neagle'), ('user:4455', 'Lukacs'), ('user:4258', 'Kuzma'), ('user:2719', 'Buckingham'), ('user:2662', 'Kearton'), ('user:2694', 'Rodrigues'), ('user:4953', 'Jaffa'), ('user:873', 'Ickov'), ('user:2502', 'Beningfield'), ('user:656', 'Klishin'), ('user:2565', 'Betham'), ('user:4210', 'Duce'), ('user:499', 'Glasner'), ('user:4513', 'Weatherley'), ('user:4804', 'Sutheran'), ('user:4671', 'Drogan'), ('user:4744', 'Bain'), ('user:2687', 'Vickress'), ('user:4504', 'Riccetti'), ('user:2083', 'Clotworthy'), ('user:4057', 'Pavolillo'), ('user:2823', 'Hellewell'), ('user:2223', 'Buxsy'), ('user:286', 'Bagehot'), ('user:2416', 'Guilloux'), ('user:2826', 'Tootal'), ('user:2225', 'Margerrison'), ('user:2482', 'Cartmell'), ('user:2927', 'Gavan'), ('user:2830', 'Baggarley'), ('user:2356', 'De'), ('user:4090', 'Geyton'), ('user:2287', 'Batchellor'), ('user:4018', 'Anglim'), ('user:4285', 'Vasishchev'), ('user:4557', 'Tayloe'), ('user:2697', 'Stoffler'), ('user:891', 'Whyke'), ('user:697', 'Boardman'), ('user:4817', 'Dodshun'), ('user:4368', 'Erasmus'), ('user:2057', 'Fabb'), ('user:4993', 'Gough'), ('user:4440', 'Beane'), ('user:2775', 'Adamek'), ('user:2396', 'Dorian'), ('user:4301', 'Durber'), ('user:4861', 'Ostler'), ('user:4394', 'Ashmole'), ('user:416', 'Locks'), ('user:2629', 'Runcie'), ('user:252', 'Plues'), ('user:4967', 'Cridland'), ('user:4239', 'Sindall'), ('user:4704', 'Jerromes'), ('user:2770', 'Jaffra'), ('user:2437', 'Whittam'), ('user:469', 'Rames'), ('user:4555', 'Meckiff'), ('user:2996', 'MacRedmond'), ('user:275', 'Dun'), ('user:2023', 'Gotts'), ('user:4975', 'Franseco'), ('user:4908', 'McAndie'), ('user:4484', 'Keir'), ('user:2586', 'Gippes'), ('user:2340', 'Clausius'), ('user:2856', 'Napthine'), ('user:4164', 'Cassey'), ('user:2045', 'Lehon'), ('user:2081', 'Dainton'), ('user:4588', 'Gallandre'), ('user:4933', 'Colbertson'), ('user:2848', 'Sommerscales'), ('user:4771', 'Cogzell'), ('user:2303', 'Thornham'), ('user:4401', 'Gosforth'), ('user:691', 'Gorelli'), ('user:2914', 'Skuce'), ('user:802', 'Newell'), ('user:4190', 'Biaggelli'), ('user:2990', 'Tyers'), ('user:4982', 'Gyer'), ('user:4236', 'Shimony'), ('user:2642', 'Astill'), ('user:4151', 'Roskrug'), ('user:2417', 'Veldman'), ('user:4850', 'Tarbet'), ('user:418', 'Fibbit'), ('user:2339', 'Hartlebury'), ('user:2928', 'Roscoe'), ('user:4674', 'Gillebert'), ('user:2644', 'Thomasset'), ('user:2375', 'Pantin'), ('user:2589', "D'Cruze"), ('user:2604', 'Baynes'), ('user:2558', 'Applegate'), ('user:2044', 'Idiens'), ('user:2177', 'Gambell'), ('user:2907', 'Foden'), ('user:4184', 'Helian'), ('user:2274', 'Sorby'), ('user:2443', 'Luter'), ('user:2947', 'Shanklin'), ('user:4430', 'Dowzell'), ('user:4597', 'Bathurst'), ('user:472', 'Creber'), ('user:2367', 'Campagne'), ('user:66', 'Farady'), ('user:2053', 'Barrott'), ('user:2457', 'Glaisner'), ('

Executing query 4.
Index creation error: Index already exists
Query4: Female users in China or Russia with latitude 40-46:
ID: user:86, Gender: female, Country: Russia, Latitude: 43.2608797
ID: user:116, Gender: female, Country: Russia, Latitude: 43.4321902
ID: user:492, Gender: female, Country: Russia, Latitude: 43.446712
ID: user:509, Gender: female, Country: Russia, Latitude: 44.574779
ID: user:1163, Gender: female, Country: Russia, Latitude: 43.5621247
ID: user:1375, Gender: female, Country: Russia, Latitude: 41.5335463
ID: user:1576, Gender: female, Country: Russia, Latitude: 45.8697083
ID: user:1662, Gender: female, Country: Russia, Latitude: 44.6611202
ID: user:1708, Gender: female, Country: Russia, Latitude: 45.1242818
ID: user:1952, Gender: female, Country: Russia, Latitude: 45.4885295
Executing query 5.
Top 10 players with their emails and scores:
Email: lwandrackj5@jigsy.com, Score: 499.0
Email: jlabellow@list-manage.com, Score: 499.0
Email: hkarlolako0@arizona.edu, Score: 499.0
Email: mkincadeew@phoca.cz, Score: 499.0
Email: lharnett51@ning.com, Score: 499.0
Email: rpalleskelx@wikia.com, Score: 499.0
Email: cdyet9h@blogs.com, Score: 499.0
Email: adumbelton8i@wordpress.org, Score: 499.0
Email: itew2y@europa.eu, Score: 499.0
Email: dpriddlecz@wp.com, Score: 499.0


idx
Search per Values of Keys
Bulk Actions
+ Key
Results: 5 996. Scanned 5 996 / 5 996
Last refresh: < 1 min
user
100%
5996
HASH 1000 No limit 312 B
HASH 1001 No limit 312 B
HASH 1002 No limit 312 B
HASH 1003 No limit 312 B
HASH 1004 No limit 312 B
HASH 1005 No limit 312 B
HASH 1006 No limit 312 B
HASH 1007 No limit 312 B
HASH 1008 No limit 312 B
HASH 1009 No limit 312 B
HASH 100 No limit 312 B
HASH 1010 No limit 312 B
HASH 1011 No limit 312 B
HASH 1012 No limit 312 B
HASH 1013 No limit 312 B
HASH 1014 No limit 312 B
SORTED SET leaderboard:2
Key Size: 566 KB Length: 5996 TTL: No limit
< 1 min
Unicode
Add Members
Member
Score
user:1043 0
user:1106 0
user:2463 0
user:2771 0
user:319 0
user:4175 0
user:4448 0
user:4822 0
user:627 0
user:1527 1
user:1780 1
user:1920 1


All Key Types
user:100
Bulk Actions
+ Key
Results: 1. Scanned 5 997 / 5 997
Last refresh: < 1 min
user
100%
1
HASH user:100
Key Size: 312 B Length: 11 TTL: No limit
< 1 min
Unicode
Show TTL
Add Fields
Field
Value
TTL
first_name Anderson No Limit
last_name Bithany No Limit
email abithany2r@sbwire.com No Limit
gender male No Limit
ip_address 222.164.174.233 No Limit
country Poland No Limit
country_code PL No Limit
city Łapczyca No Limit
longitude 20.3412605 No Limit
latitude 49.9513027 No Limit
last_login 1581119052 No Limit