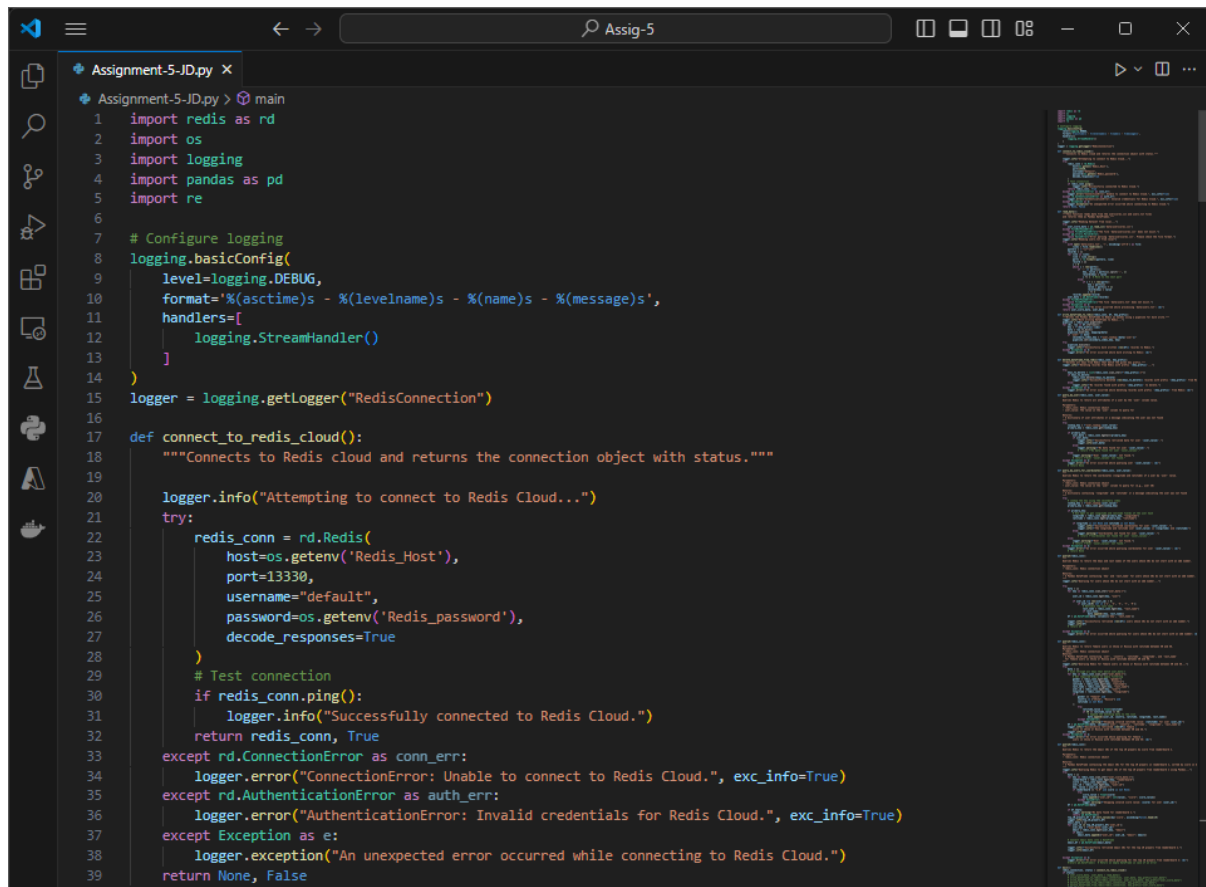




Big data Management
Assignment
Jeyadev L
G23AI2071

Connection to Redis



The image shows a code editor window titled "Assig-5" with a file named "Assignment-5-JD.py". The code is a Python script for connecting to Redis Cloud. It includes imports for redis, os, logging, pandas, and re. It configures logging with a StreamHandler. A function named "connect_to_redis_cloud()" is defined, which attempts to connect to Redis Cloud using the "rd" module. It sets host, port, username, password, and decode_responses. It tests the connection with "ping()" and logs the result. It includes exception handling for "rd.ConnectionError", "rd.AuthenticationError", and a general "Exception".

```
1 import redis as rd
2 import os
3 import logging
4 import pandas as pd
5 import re
6
7 # Configure logging
8 logging.basicConfig(
9     level=logging.DEBUG,
10     format='%(asctime)s - %(levelname)s - %(name)s - %(message)s',
11     handlers=[
12         logging.StreamHandler()
13     ]
14 )
15 logger = logging.getLogger("RedisConnection")
16
17 def connect_to_redis_cloud():
18     """Connects to Redis cloud and returns the connection object with status."""
19
20     logger.info("Attempting to connect to Redis Cloud...")
21     try:
22         redis_conn = rd.Redis(
23             host=os.getenv('Redis_Host'),
24             port=13330,
25             username="default",
26             password=os.getenv('Redis_password'),
27             decode_responses=True
28         )
29         # Test connection
30         if redis_conn.ping():
31             logger.info("Successfully connected to Redis Cloud.")
32             return redis_conn, True
33     except rd.ConnectionError as conn_err:
34         logger.error("ConnectionError: Unable to connect to Redis Cloud.", exc_info=True)
35     except rd.AuthenticationError as auth_err:
36         logger.error("AuthenticationError: Invalid credentials for Redis Cloud.", exc_info=True)
37     except Exception as e:
38         logger.exception("An unexpected error occurred while connecting to Redis Cloud.")
39     return None, False
```

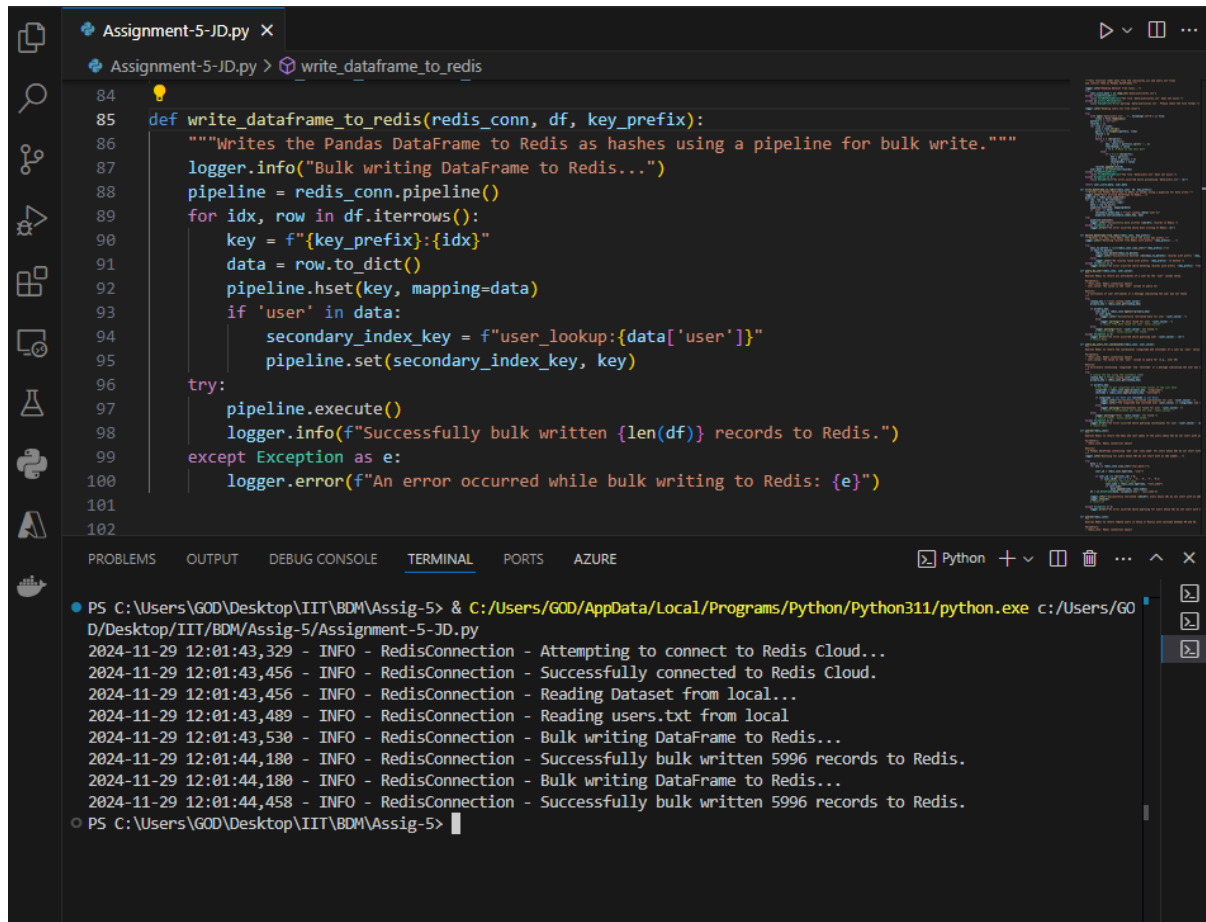
Reading Data from CSV and TXT files

```
def read_data():
    """This function reads data from the userscores.csv and users.txt files
    and returns them as Pandas DataFrames."""

    logger.info("Reading Dataset from local...")
    try:
        user_score_data = pd.read_csv('data/userscores.csv')
    except FileNotFoundError:
        raise FileNotFoundError("The file 'data/userscores.csv' does not exist.")
    except pd.errors.ParserError:
        raise ValueError("Error parsing 'data/userscores.csv'. Please check the file format.")
    logger.info("Reading users.txt from local")
    try:
        with open('data/users.txt', 'r', encoding='utf-8') as file:
            lines = file.readlines()
            pattern = r'"([^"]+)"'
            records = []
            for line in lines:
                line = line.strip()
                parts = re.findall(pattern, line)
                record = {}
                i = 0
                while i < len(parts):
                    if ':' in parts[i]:
                        key, value = parts[i].split(':', 1)
                        record[key] = value
                        i += 1 # Move to the next part
                    else:
                        if i + 1 < len(parts):
                            key = parts[i]
                            value = parts[i + 1]
                            record[key] = value
                            i += 2
                records.append(record)
            user_data = pd.DataFrame(records)
    except FileNotFoundError:
        raise FileNotFoundError("The file 'data/users.txt' does not exist.")
    except Exception as e:
        raise ValueError(f"An error occurred while processing 'data/users.txt': {e}")
    return user_score_data, user_data
```

Write Function to Redis.

I use pandas dataframe to write into Redis which eliminates the need to specify any data type or schema



The screenshot displays a Visual Studio Code editor window with a Python script named `Assignment-5-JD.py`. The script defines a function `write_dataframe_to_redis` that takes a Redis connection, a pandas DataFrame, and a key prefix as arguments. It uses a Redis pipeline to bulk write the DataFrame to Redis, with a secondary index for the 'user' column. The terminal output shows the script's execution, including successful connections to Redis Cloud and the bulk writing of 5996 records.

```
84
85 def write_dataframe_to_redis(redis_conn, df, key_prefix):
86     """Writes the Pandas DataFrame to Redis as hashes using a pipeline for bulk write."""
87     logger.info("Bulk writing DataFrame to Redis...")
88     pipeline = redis_conn.pipeline()
89     for idx, row in df.iterrows():
90         key = f"{key_prefix}:{idx}"
91         data = row.to_dict()
92         pipeline.hset(key, mapping=data)
93         if 'user' in data:
94             secondary_index_key = f"user_lookup:{data['user']}"
95             pipeline.set(secondary_index_key, key)
96     try:
97         pipeline.execute()
98         logger.info(f"Successfully bulk written {len(df)} records to Redis.")
99     except Exception as e:
100         logger.error(f"An error occurred while bulk writing to Redis: {e}")
101
102
```

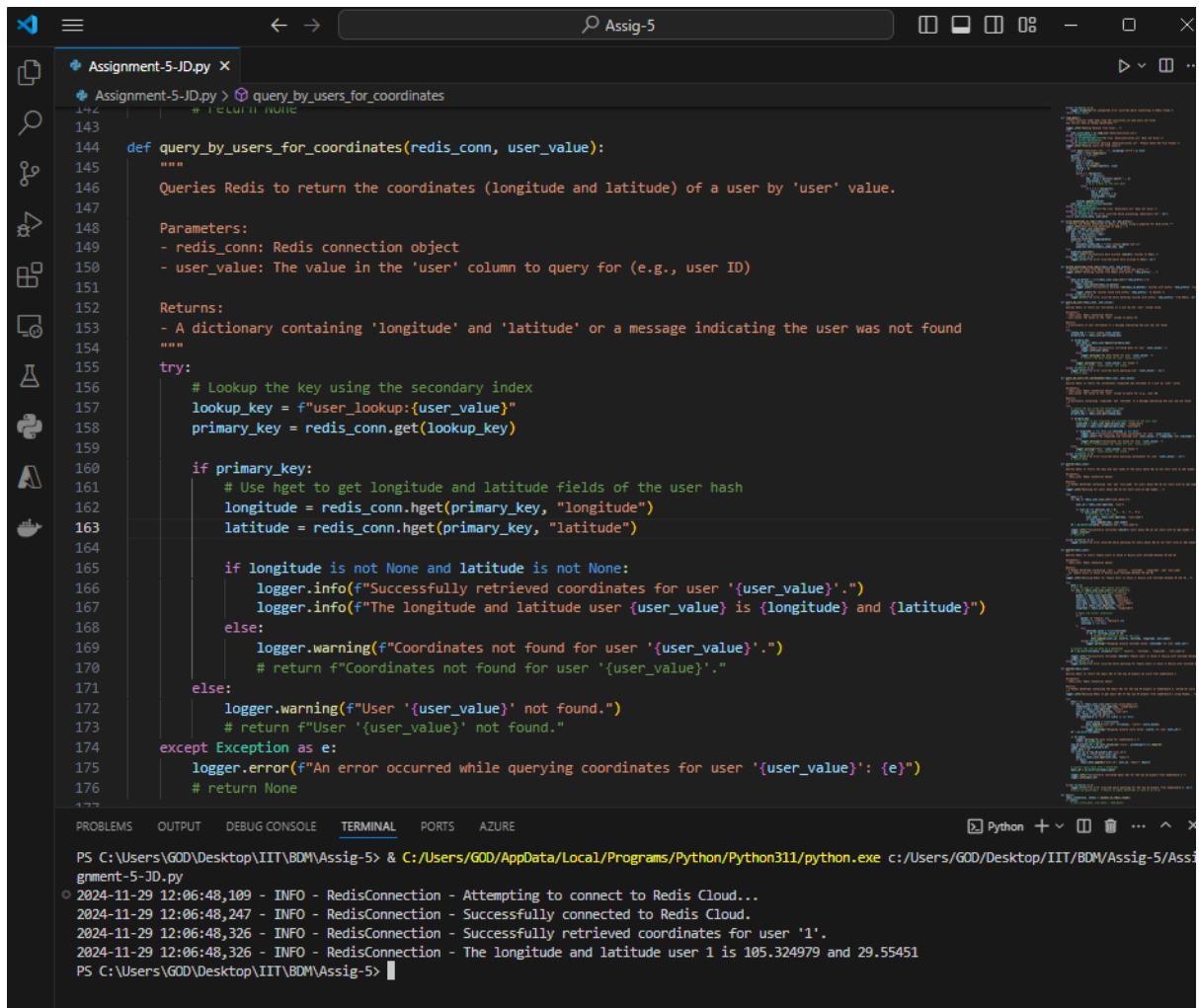
Terminal Output:

```
PS C:\Users\GOD\Desktop\IIT\BDM\Assig-5> & C:/Users/GOD/AppData/Local/Programs/Python/Python311/python.exe c:/Users/GOD/Desktop/IIT/BDM/Assig-5/Assignment-5-JD.py
2024-11-29 12:01:43,329 - INFO - RedisConnection - Attempting to connect to Redis Cloud...
2024-11-29 12:01:43,456 - INFO - RedisConnection - Successfully connected to Redis Cloud.
2024-11-29 12:01:43,456 - INFO - RedisConnection - Reading Dataset from local...
2024-11-29 12:01:43,489 - INFO - RedisConnection - Reading users.txt from local
2024-11-29 12:01:43,530 - INFO - RedisConnection - Bulk writing DataFrame to Redis...
2024-11-29 12:01:44,180 - INFO - RedisConnection - Successfully bulk written 5996 records to Redis.
2024-11-29 12:01:44,180 - INFO - RedisConnection - Bulk writing DataFrame to Redis...
2024-11-29 12:01:44,458 - INFO - RedisConnection - Successfully bulk written 5996 records to Redis.
PS C:\Users\GOD\Desktop\IIT\BDM\Assig-5>
```

Query 1

```
Assignment-5-JD.py > query_by_user
113
114 def query_by_user(redis_conn, user_value):
115     """
116     Queries Redis to return all attributes of a user by the 'user' column value.
117
118     Parameters:
119     - redis_conn: Redis connection object
120     - user_value: The value in the 'user' column to query for
121
122     Returns:
123     - A dictionary of user attributes or a message indicating the user was not found
124     """
125     try:
126         lookup_key = f"user_lookup:{user_value}"
127         primary_key = redis_conn.get(lookup_key)
128
129         if primary_key:
130             user_data = redis_conn.hgetall(primary_key)
131             if user_data:
132                 logger.info(f"Successfully retrieved data for user '{user_value}'.")
133                 logger.info(user_data)
134             else:
135                 logger.warning(f"No data found for user '{user_value}'.")
136                 # return f"No data found for user '{user_value}'."
137         else:
138             logger.warning(f"User '{user_value}' not found.")
139             # return f"User '{user_value}' not found."
140     except Exception as e:
141         logger.error(f"An error occurred while querying user '{user_value}': {e}")
142         # return None
```

Query 2



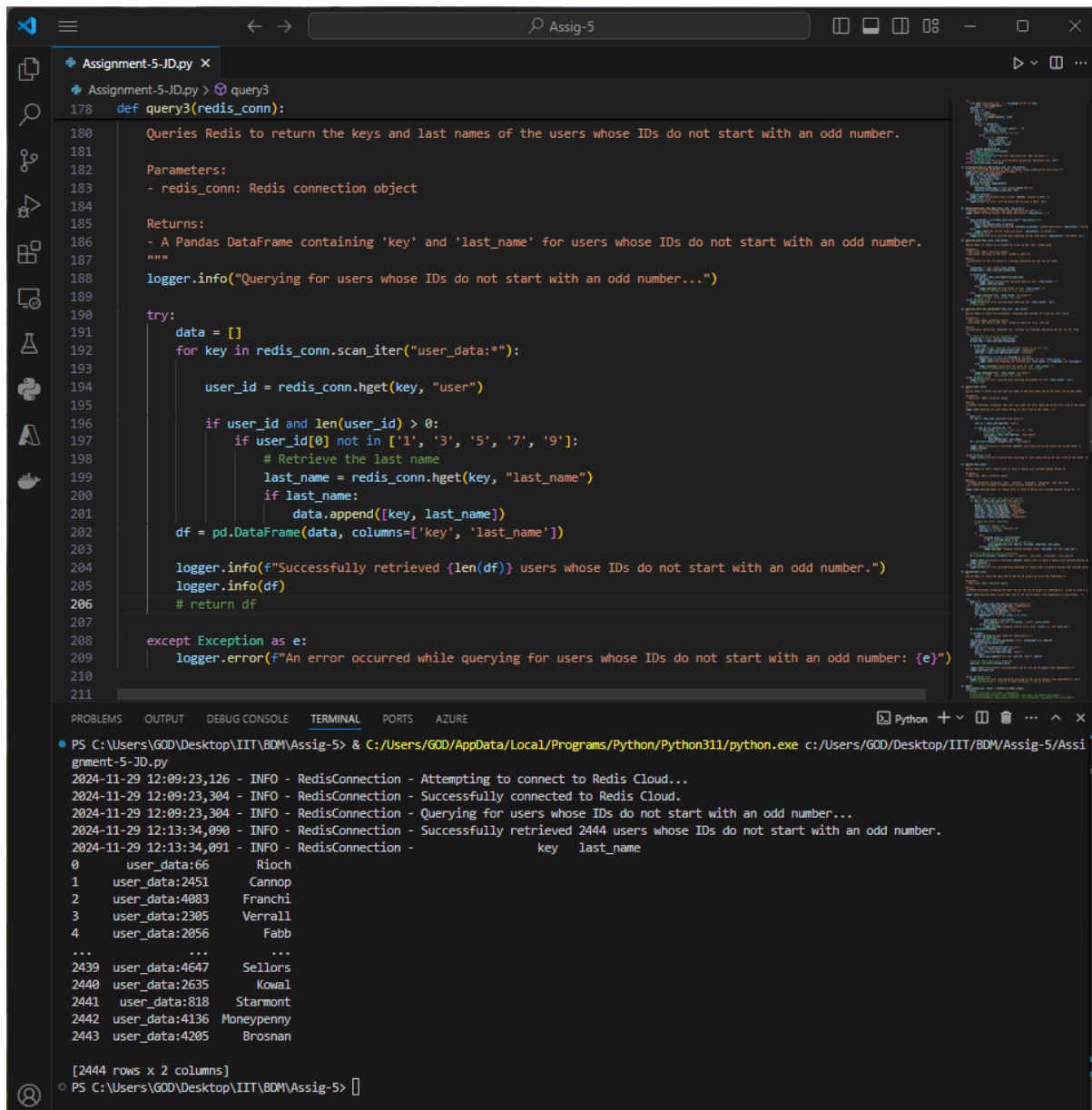
The screenshot shows a Visual Studio Code editor window with a file named "Assignment-5-JD.py". The editor is displaying a Python function `query_by_users_for_coordinates` that interacts with a Redis database. The function takes a Redis connection object and a user value as input and returns a dictionary with longitude and latitude coordinates or a message indicating the user was not found. The function uses `redis_conn.hget` to retrieve the coordinates for a given user value. It also includes logging statements to track the success or failure of the query.

```
142 # return None
143
144 def query_by_users_for_coordinates(redis_conn, user_value):
145     """
146     Queries Redis to return the coordinates (longitude and latitude) of a user by 'user' value.
147
148     Parameters:
149     - redis_conn: Redis connection object
150     - user_value: The value in the 'user' column to query for (e.g., user ID)
151
152     Returns:
153     - A dictionary containing 'longitude' and 'latitude' or a message indicating the user was not found
154     """
155     try:
156         # Lookup the key using the secondary index
157         lookup_key = f"user_lookup:{user_value}"
158         primary_key = redis_conn.get(lookup_key)
159
160         if primary_key:
161             # Use hget to get longitude and latitude fields of the user hash
162             longitude = redis_conn.hget(primary_key, "longitude")
163             latitude = redis_conn.hget(primary_key, "latitude")
164
165             if longitude is not None and latitude is not None:
166                 logger.info(f"Successfully retrieved coordinates for user '{user_value}'.")
167                 logger.info(f"The longitude and latitude user {user_value} is {longitude} and {latitude}")
168             else:
169                 logger.warning(f"Coordinates not found for user '{user_value}'.")
170                 # return f"Coordinates not found for user '{user_value}'."
171         else:
172             logger.warning(f"User '{user_value}' not found.")
173             # return f"User '{user_value}' not found."
174     except Exception as e:
175         logger.error(f"An error occurred while querying coordinates for user '{user_value}': {e}")
176         # return None
177
```

The terminal output shows the execution of the script. It starts with a command prompt prompt `PS C:\Users\GOD\Desktop\IIT\BDM\Assig-5>` and then runs the command `C:/Users/GOD/AppData/Local/Programs/Python/Python311/python.exe c:/Users/GOD/Desktop/IIT/BDM/Assig-5/Assigment-5-JD.py`. The output shows the following log messages:

```
2024-11-29 12:06:48,189 - INFO - RedisConnection - Attempting to connect to Redis Cloud...
2024-11-29 12:06:48,247 - INFO - RedisConnection - Successfully connected to Redis Cloud.
2024-11-29 12:06:48,326 - INFO - RedisConnection - Successfully retrieved coordinates for user '1'.
2024-11-29 12:06:48,326 - INFO - RedisConnection - The longitude and latitude user 1 is 105.324979 and 29.55451
PS C:\Users\GOD\Desktop\IIT\BDM\Assig-5>
```

Query 3



The image shows a Visual Studio Code editor window with a file named "Assignment-5-JD.py". The script defines a function `query3(redis_conn)` that queries a Redis database for user data. The function's docstring describes its parameters and returns. The code uses `redis_conn.scan_iter` to iterate over keys, `redis_conn.hget` to retrieve user IDs and last names, and `pd.DataFrame` to store the results. It includes logging for query execution and error handling.

```
178 def query3(redis_conn):
179     """
180     Queries Redis to return the keys and last names of the users whose IDs do not start with an odd number.
181
182     Parameters:
183     - redis_conn: Redis connection object
184
185     Returns:
186     - A Pandas DataFrame containing 'key' and 'last_name' for users whose IDs do not start with an odd number.
187     """
188     logger.info("Querying for users whose IDs do not start with an odd number...")
189
190     try:
191         data = []
192         for key in redis_conn.scan_iter("user_data:*"):
193             user_id = redis_conn.hget(key, "user")
194
195             if user_id and len(user_id) > 0:
196                 if user_id[0] not in ['1', '3', '5', '7', '9']:
197                     # Retrieve the last name
198                     last_name = redis_conn.hget(key, "last_name")
199                     if last_name:
200                         data.append([key, last_name])
201         df = pd.DataFrame(data, columns=['key', 'last_name'])
202
203         logger.info(f"Successfully retrieved {len(df)} users whose IDs do not start with an odd number.")
204         logger.info(df)
205         # return df
206
207     except Exception as e:
208         logger.error(f"An error occurred while querying for users whose IDs do not start with an odd number: {e}")
209
210
211
```

The terminal output shows the execution of the script. It logs the connection to Redis Cloud, the query execution, and the successful retrieval of 2444 users. The output is displayed as a table with columns 'key' and 'last_name'.

```
PS C:\Users\GOD\Desktop\IIT\BDM\Assig-5> & C:/Users/GOD/AppData/Local/Programs/Python/Python311/python.exe c:/Users/GOD/Desktop/IIT/BDM/Assig-5/Assigment-5-JD.py
2024-11-29 12:09:23,126 - INFO - RedisConnection - Attempting to connect to Redis Cloud...
2024-11-29 12:09:23,304 - INFO - RedisConnection - Successfully connected to Redis Cloud.
2024-11-29 12:09:23,304 - INFO - RedisConnection - Querying for users whose IDs do not start with an odd number...
2024-11-29 12:13:34,090 - INFO - RedisConnection - Successfully retrieved 2444 users whose IDs do not start with an odd number.
2024-11-29 12:13:34,091 - INFO - RedisConnection -
      key  last_name
0  user_data:66      Rioch
1  user_data:2451    Cannop
2  user_data:4083   Franchi
3  user_data:2305   Verrall
4  user_data:2056    Fabb
...
2439 user_data:4647  Sellors
2440 user_data:2635   Kowal
2441 user_data:818   Starmont
2442 user_data:4136  Money Penny
2443 user_data:4205  Brosnan

[2444 rows x 2 columns]
PS C:\Users\GOD\Desktop\IIT\BDM\Assig-5>
```

Query 4

```
Assignment-5-JD.py X
Assignment-5-JD.py > main
212 def query4(redis_conn):
213     """
214     Queries Redis to return female users in China or Russia with latitude between 40 and 46.
215     Parameters:
216     - redis_conn: Redis connection object
217     Returns:
218     - A Pandas DataFrame containing 'user', 'country', 'latitude', 'longitude', and 'last_name'
219     for female users in China or Russia with latitude between 40 and 46.
220     """
221     logger.info("Querying Redis for female users in China or Russia with latitude between 40 and 46...")
222     try:
223         data = []
224         # Scan through all keys that match user_data:*
225         for key in redis_conn.scan_iter("user_data:*"):
226             # Get required fields to apply filtering
227             gender = redis_conn.hget(key, "gender")
228             country = redis_conn.hget(key, "country")
229             latitude = redis_conn.hget(key, "latitude")
230             last_name = redis_conn.hget(key, "last_name")
231             user_id = redis_conn.hget(key, "user")
232             longitude = redis_conn.hget(key, "longitude")
233             if (
234                 gender == "female" and
235                 country in ["China", "Russia"] and
236                 latitude is not None
237             ):
238                 try:
239                     latitude_value = float(latitude)
240                     if 40 <= latitude_value <= 46:
241                         # Append the matching data to the list
242                         data.append([user_id, country, latitude, longitude, last_name])
243                     except ValueError:
244                         logger.warning(f"Skipping invalid latitude value: {latitude} for user {user_id}")
245             df = pd.DataFrame(data, columns=['user', 'country', 'latitude', 'longitude', 'last_name'])
246             logger.info(f"Successfully retrieved {len(df)} female \
247 users in China or Russia with latitude between 40 and 46.")
248             logger.info(df)
249         except Exception as e:
250             logger.error(f"An error occurred while querying for female \
251 users in China or Russia with latitude between 40 and 46: {e}")
252
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE
2024-11-29 12:17:49,001 - INFO - RedisConnection - Querying Redis for female users in China or Russia with latitude between 40 and 46...
2024-11-29 12:31:20,479 - INFO - RedisConnection - Successfully retrieved 97 female users in China or Russia with latitude between 40 and 46.
2024-11-29 12:31:20,479 - INFO - RedisConnection - user country latitude longitude last_name
0 3568 China 41.985186 122.836723 MacAless
1 5920 Russia 43.8507498 131.8648449 Dollen
1 5920 Russia 43.8507498 131.8648449 Dollen
1 5920 Russia 43.8507498 131.8648449 Dollen
2 5306 Russia 43.2374865 45.0503473 Maddin
3 658 China 42.629452 120.639361 Allanby
4 1143 China 43.817071 125.323544 Ryde
1 5920 Russia 43.8507498 131.8648449 Dollen
2 5306 Russia 43.2374865 45.0503473 Maddin
3 658 China 42.629452 120.639361 Allanby
1 5920 Russia 43.8507498 131.8648449 Dollen
2 5306 Russia 43.2374865 45.0503473 Maddin
1 5920 Russia 43.8507498 131.8648449 Dollen
```



```

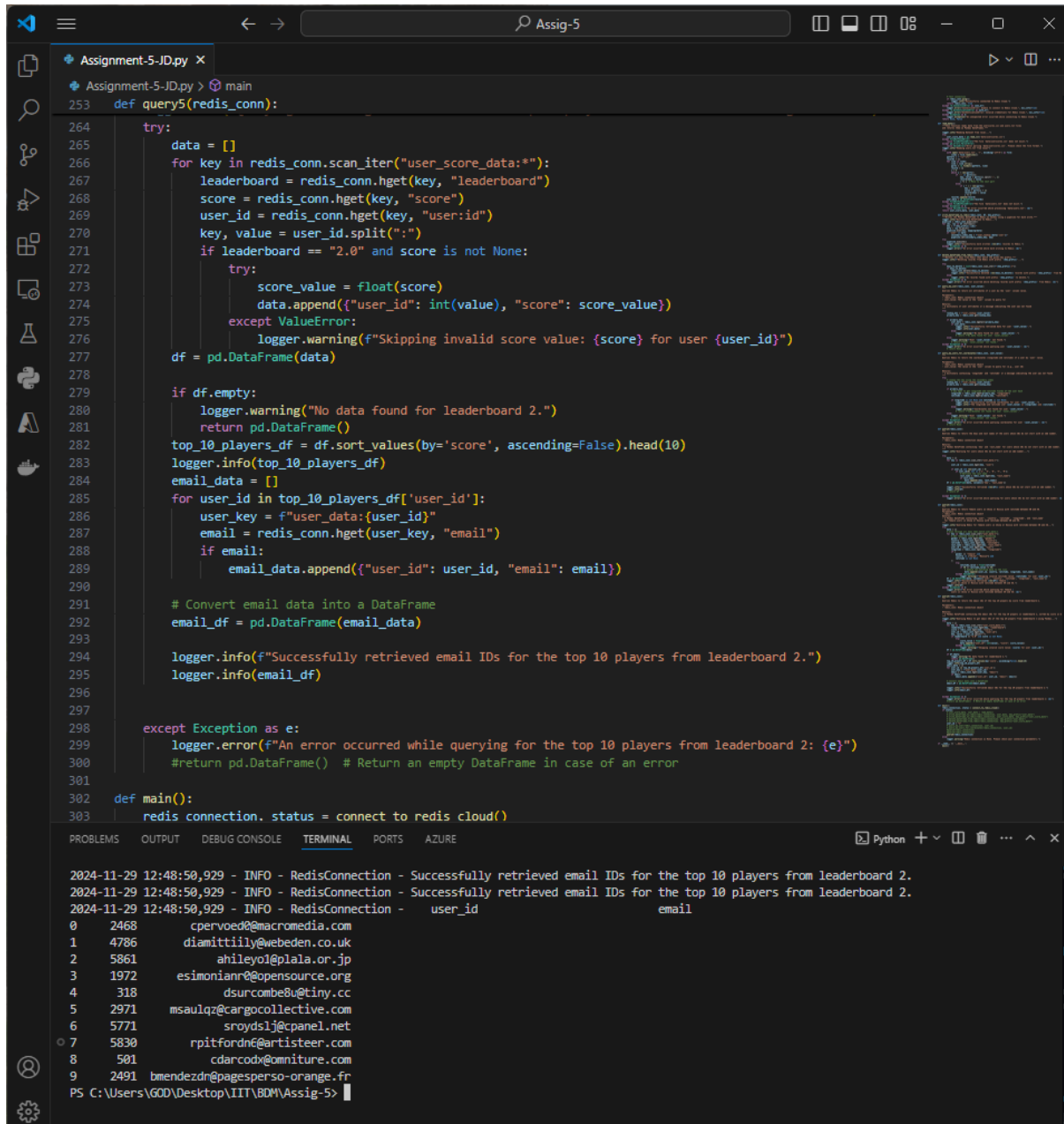
2024-11-29 12:31:20,479 - INFO - RedisConnection - user country lati
0 3568 China 41.985186 122.836723 MacAless
1 5920 Russia 43.8507498 131.8648449 Dollen
1 5920 Russia 43.8507498 131.8648449 Dollen
1 5920 Russia 43.8507498 131.8648449 Dollen
2 5306 Russia 43.2374865 45.0503473 Maddin
3 658 China 42.629452 120.639361 Allanby
4 1143 China 43.817071 125.323544 Ryde
1 5920 Russia 43.8507498 131.8648449 Dollen
2 5306 Russia 43.2374865 45.0503473 Maddin
3 658 China 42.629452 120.639361 Allanby
1 5920 Russia 43.8507498 131.8648449 Dollen
2 5306 Russia 43.2374865 45.0503473 Maddin
1 5920 Russia 43.8507498 131.8648449 Dollen
1 5920 Russia 43.8507498 131.8648449 Dollen
1 5920 Russia 43.8507498 131.8648449 Dollen
1 5920 Russia 43.8507498 131.8648449 Dollen
1 5920 Russia 43.8507498 131.8648449 Dollen
1 5920 Russia 43.8507498 131.8648449 Dollen
2 5306 Russia 43.2374865 45.0503473 Maddin
2 5306 Russia 43.2374865 45.0503473 Maddin
3 658 China 42.629452 120.639361 Allanby
4 1143 China 43.817071 125.323544 Ryde
.. ... .. ... ..
.. ... .. ... ..
92 1576 Russia 45.8697083 43.3478599 Relfe
93 5162 China 42.016401 121.668489 Rolfini
93 5162 China 42.016401 121.668489 Rolfini
94 4372 China 44.439044 125.1797741 Clawley
94 4372 China 44.439044 125.1797741 Clawley
95 4364 Russia 43.47139 43.84694 Adamek
96 2568 China 44.766541 129.688614 Landre

```

[97 rows x 5 columns]

PS C:\Users\GOD\Desktop\IIT\BDM\Assig-5> █

Query 5



The image shows a VS Code editor window with a Python script named `Assignment-5-JD.py`. The script defines a function `query5(redis_conn)` that queries a Redis database for user scores and emails. The script is executed from the command line, and the output is displayed in the terminal window at the bottom.

```
Assignment-5-JD.py X
Assignment-5-JD.py > main
253 def query5(redis_conn):
254     try:
255         data = []
256         for key in redis_conn.scan_iter("user_score_data:*"):
257             leaderboard = redis_conn.hget(key, "leaderboard")
258             score = redis_conn.hget(key, "score")
259             user_id = redis_conn.hget(key, "user:id")
260             key, value = user_id.split(":")
261             if leaderboard == "2.0" and score is not None:
262                 try:
263                     score_value = float(score)
264                     data.append({"user_id": int(value), "score": score_value})
265                 except ValueError:
266                     logger.warning(f"Skipping invalid score value: {score} for user {user_id}")
267         df = pd.DataFrame(data)
268
269         if df.empty:
270             logger.warning("No data found for leaderboard 2.")
271             return pd.DataFrame()
272         top_10_players_df = df.sort_values(by='score', ascending=False).head(10)
273         logger.info(top_10_players_df)
274         email_data = []
275         for user_id in top_10_players_df['user_id']:
276             user_key = f"user_data:{user_id}"
277             email = redis_conn.hget(user_key, "email")
278             if email:
279                 email_data.append({"user_id": user_id, "email": email})
280
281         # Convert email data into a DataFrame
282         email_df = pd.DataFrame(email_data)
283
284         logger.info(f"Successfully retrieved email IDs for the top 10 players from leaderboard 2.")
285         logger.info(email_df)
286
287     except Exception as e:
288         logger.error(f"An error occurred while querying for the top 10 players from leaderboard 2: {e}")
289         #return pd.DataFrame() # Return an empty DataFrame in case of an error
290
291 def main():
292     redis_connection.status = connect_to_redis_cloud()
293
294 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE
295 2024-11-29 12:48:50,929 - INFO - RedisConnection - Successfully retrieved email IDs for the top 10 players from leaderboard 2.
296 2024-11-29 12:48:50,929 - INFO - RedisConnection - Successfully retrieved email IDs for the top 10 players from leaderboard 2.
297 2024-11-29 12:48:50,929 - INFO - RedisConnection - user_id email
298 0 2468 cpervoed0@macromedia.com
299 1 4786 diamittitilly@webeden.co.uk
300 2 5861 ahiley01@plala.or.jp
301 3 1972 esimonianr0@opensource.org
302 4 318 dsurcombe8u@tiny.cc
303 5 2971 msaulqz@cargocollective.com
304 6 5771 sroydslj@cpanel.net
305 7 5830 rpitfordn0@artisteer.com
306 8 501 cdarcodx@omniture.com
307 9 2491 bmendezdn@pagesperso-orange.fr
308 PS C:\Users\GOD\Desktop\IIIT\BDM\Assig-5>
```