Coverage for **test_function.py**: 100%

116 statements | 116 run | 0 missing | 0 excluded

```
1  def task2_1(z):
2      import pandas as pd
3      import psycopg2
4      #tweak the database parameters to match your specific postgres database
5      conn=psycopg2.connect(host='localhost',
6                            port='5432',
7                            user='postgres',
8                            password='mysecretpassword',
9                            database='postgres',
10                           #You may add the following line if you have schemas
11                           options="-c search_path=nfl"
12                          )
13     cur=conn.cursor()
14     cur.execute('SELECT short_name FROM "FIFA".players_20 order by ((skill_dribbling + skill_curve + skill_fk_accuracy + sk
15     result=[]
16     for row in cur:
17         result.append(row)
18
19     #dataframe=pd.DataFrame(result[0:z], columns=['shortname'])
20
21     conn.commit()
22     cur.close()
23     conn.close()
24     return result[0:z]
25
26 def task2_2(z):
27     import pandas as pd
28     import psycopg2
29     #tweak the database parameters to match your specific postgres database
30     conn=psycopg2.connect(host='localhost',
31                           port='5432',
32                           user='postgres',
33                           password='mysecretpassword',
34                           database='postgres',
35                           #You may add the following line if you have schemas
36                           options="-c search_path=nfl"
37                          )
38     cur=conn.cursor()
39     cur.execute('SELECT count (sofifa_id),club FROM "FIFA".players_20 WHERE contract_valid_until = 2021 GROUP BY club order
40     result=[]
41     for row in cur:
42         result.append(row)
43
44     #dataframe=pd.DataFrame(result[0:z], columns=['count','club'])
45
46     conn.commit()
47     cur.close()
48     conn.close()
49     return result[0:z]
50
51 def task2_3(z):
52     import pandas as pd
53     import psycopg2
54     #tweak the database parameters to match your specific postgres database
55     conn=psycopg2.connect(host='localhost',
56                           port='5432',
57                           user='postgres',
58                           password='mysecretpassword',
59                           database='postgres',
60                           #You may add the following line if you have schemas
61                           options="-c search_path=nfl"
62                          )
63     cur=conn.cursor()
64     cur.execute('SELECT count(sofifa_id),club FROM "FIFA".players_20 GROUP BY club order by count(sofifa_id) DESC;')
65     result=[]
66     for row in cur:
67         result.append(row)
68
69     #dataframe=pd.DataFrame(result[0:z], columns=['Number of Players','Club Name'])
70
71     conn.commit()
```

```
    cur.close()
    conn.close()
    return result[0:z]

def task2_4_1():
    import pandas as pd
    import psycopg2
    #tweak the database parameters to match your specific postgres database
    conn=psycopg2.connect(host='localhost',
                          port='5432',
                          user='postgres',
                          password='mysecretpassword',
                          database='postgres',
                          #You may add the following line if you have schemas
                          options="-c search_path=nfl"
                         )
    cur=conn.cursor()
    cur.execute('SELECT count(sofifa_id),nation_position FROM "FIFA".players_20 GROUP BY nation_position order by count(sof
    result=[]
    for row in cur:
        result.append(row)
        break

    #dataframe=pd.DataFrame(result, columns=['count','nation position'])

    conn.commit()
    cur.close()
    conn.close()
    return result

def task2_4_2():
    import pandas as pd
    import psycopg2
    #tweak the database parameters to match your specific postgres database
    conn=psycopg2.connect(host='localhost',
                          port='5432',
                          user='postgres',
                          password='mysecretpassword',
                          database='postgres',
                          #You may add the following line if you have schemas
                          options="-c search_path=nfl"
                         )
    cur=conn.cursor()
    cur.execute('SELECT count(sofifa_id),team_position FROM "FIFA".players_20 GROUP BY team_position order by count(sofifa_
    result=[]
    for row in cur:
        result.append(row)
        break

    #dataframe=pd.DataFrame(result, columns=['count','nation position'])

    conn.commit()
    cur.close()
    conn.close()
    return result

def task2_5():
    import pandas as pd
    import psycopg2
    #tweak the database parameters to match your specific postgres database
    conn=psycopg2.connect(host='localhost',
                          port='5432',
                          user='postgres',
                          password='mysecretpassword',
                          database='postgres',
                          #You may add the following line if you have schemas
                          options="-c search_path=nfl"
                         )
    cur=conn.cursor()
    cur.execute('SELECT count(sofifa_id),nationality FROM "FIFA".players_20 GROUP BY nationality order by count(sofifa_id) D
    result=[]
    for row in cur:
        result.append(row)
        break

    #dataframe=pd.DataFrame(result, columns=['count','nationality'])

```

```
149     conn.commit()
150     cur.close()
151     conn.close()
152     return result
153 
154 
155 def test_happy_path():
156     #Both happy and sad paths
157     task2_1(2)
158     task2_2(3)
159     task2_3(4)
160     task2_4_1()
161     task2_4_2()
162     task2_5()
163 
164     assert len(task2_1(2)) == 2, "Returned should be length 6"
165     assert len(task2_2(3)) == 3, "Returned should be length 6"
166     assert len(task2_3(4)) == 4, "Returned should be length 5"
167     assert len(task2_4_1()) == 1, "Returned should be length 1"
168     assert len(task2_4_2()) == 1, "Returned should be length 1"
169     assert len(task2_5()) == 1, "Returned should be length 1"
170 
171     assert task2_1(2) == [('T. Haye',), ('Àlex Corredera',)], "Returned value"
172     assert task2_2(3) == [(18, 'FC Ingolstadt 04'),(18, '1. FC Kaiserslautern'),(17, 'FC Girondins de Bordeaux')], "Returnec
173     assert task2_3(4) == [(33, 'VfL Wolfsburg'), (33, 'Norwich City'), (33, 'AS Monaco'), (33, 'Crystal Palace')], "Returnec
174     assert task2_4_1() == [(17152, None)], "Returned value"
175     assert task2_4_2() == [(7820, 'SUB')], "Returned value"
176     assert task2_5() == [(1667, 'England')], "Returned value"
177 
178 def test_sad_path():
179     task2_1(2)
180     task2_2(3)
181     task2_3(4)
182     task2_4_1()
183     task2_4_2()
184     task2_5()
185     assert task2_1(2) is not None, "Returned should not be None"
186     assert task2_2(3) is not None, "Returned should not be None"
187     assert task2_3(4) is not None, "Returned should not be None"
188     assert task2_4_1() is not None, "Returned should not be None"
189     assert task2_4_2() is not None, "Returned should not be None"
190     assert task2_5() is not None, "Returned should not be None"
191 
192     assert task2_1(2) is not int, "Returned should be integer"
193     assert task2_2(3) is not int, "Returned should be integer"
194     assert task2_3(4) is not int, "Returned should be integer"
```