SOFTWARE TESTING METHODOLO HW 7 TinyDB Evaluation Test

In this assignment I will test tinyDB driver some operations.

case 1 search existing data by valid query
case 2 insert data by valid query
case 3 modify existing data by valid query
case 4 delete existing data by valid query
case 5 search Non-existing data by valid query
case 6 modify Non-existing data by valid query
case 7 Delete Non-existing data by valid query
case 8 can not insert exits data

Test Result:

```
Output
........
----------------------------------------
----------------------------
Ran 8 tests in 0.005s
OK
case 2 insert data by valid query
case 1 search existing data by valid
query
case 3 modify existing data by valid
query
case 4 delete existing data by valid
query
case 5 search Non-existing data by valid
query
case 6 modify Non-existing data by valid
query
case 7 Delete Non-existing data by valid
query
case 8 can not insert exits data
```

Codes:

```python
from tinydb import TinyDB, where
import unittest

def condb():
    db = TinyDB('db.json')
    return db

class Test_001_Insert_by_valid_query_Function(unittest.TestCase):

    def setUp(self):
        self.db = TinyDB('db.json')

    def tearDown(self):
        self.db.purge()
        self.db.all()

    def test_simple_insert_valid_exist(self):
        print("case 2 insert data by valid query")
        self.db.insert({'Name': 'cshu', 'Email': 'cshu1@kent.edu', 'int' : 1, 'char':1})
        result=self.db.search(where('Name') == 'cshu')
        self.assertEqual(result,[{'Name': 'cshu', 'Email': 'cshu1@kent.edu', 'int' : 1,

class Test_002_Search_existing_data_by_valid_query_Function(unittest.TestCase):
    def setUp(self):
        self.db = TinyDB('db.json')

    def tearDown(self):
        self.db.purge()
        self.db.all()

    def test_simple_search_valid_exist(self):
        print("case 1 search existing data by valid query")
        self.db.insert({'Name': 'cshu', 'Email': 'cshu1@kent.edu', 'int' : 1, 'char':1})
        result=self.db.search(where('Name') == 'cshu')

        self.assertEqual(result,[{'Name': 'cshu', 'Email': 'cshu1@kent.edu', 'int' : 1,

class Test_003_Modify_existing_data_by_valid_query_Function(unittest.TestCase):
    def setUp(self):
        self.db = TinyDB('db.json')

    def tearDown(self):
        self.db.purge()
        self.db.all()

    def test_simple_modify_valid_exist(self):
        print("case 3 modify existing data by valid query")
        self.db.insert({'Name': 'first', 'Email': 'first@kent.edu', 'int' : 1, 'char':1}
        self.db.update({'int': 10}, where('Name') == 'first')

        result=self.db.search(where('Name') == 'first')
```

```python
class Test_004_Delete_existing_data_by_valid_query_Function(unittest.TestCase):
    def setUp(self):
        self.db = TinyDB('db.json')

    def tearDown(self):
        self.db.purge()
        self.db.all()

    def test_simple_delete_valid_exist(self):
        print("case 4 delete existing data by valid query")
        self.db.insert({'Name': 'cshu', 'Email': 'cshu1@kent.edu', 'int' : 1, 'char':1
        self.db.remove(where('Name') == 'cshu')
        result=self.db.search(where('Name') == 'cshu')

        self.assertEqual(result,[])

class Test_005_Search_Not_existing_data_by_valid_query_Function(unittest.TestCase):
    def setUp(self):
        self.db = TinyDB('db.json')

    def tearDown(self):
        self.db.purge()
        self.db.all()

    def test_simple_search_not_exist(self):
        print("case 5 search Non-existing data by valid query")
        result=self.db.search(where('Name') == 'Ted')
        self.assertEqual(result,[])

class Test_006_Modify_Not_existing_data_by_valid_query_Function(unittest.TestCase):
    def setUp(self):
        self.db = TinyDB('db.json')

    def tearDown(self):
        self.db.purge()
        self.db.all()

    def test_simple_modify_not_exist(self):
        print("case 6 modify Non-existing data by valid query")
        result=self.db.update({'int': 10}, where('Name') == 'Ted')
        self.assertEqual(result,None)

class Test_007_Delete_Not_existing_data_by_valid_query_Function(unittest.TestCase):
    def setUp(self):
        self.db = TinyDB('db.json')

    def tearDown(self):
        self.db.purge()
        self.db.all()

    def test_simple_delete_not_exist(self):
        print("case 7 Delete Non-existing data by valid query")
```

```python

class Test_008_Insert_exits_data_Function(unittest.TestCase):
    def setUp(self):
        self.db = TinyDB('db.json')

    def tearDown(self):
        self.db.purge()
        self.db.all()

    def test_simple_insert_by_query(self):
        print("case 8 can not insert exits data")
        self.db.insert({'Name': 'Yingyu Wu', 'Email': 'ywu23@kent.edu', 'int' : 1, 'char'
        self.db.insert({'Name': 'Yingyu Wu', 'Email': 'ywu23@kent.edu', 'int' : 1, 'char'
        result_array = self.db.search(where('Name') == 'Yingyu Wu')
        num = len(result_array)
        # print (result_array)
        #print("search one key,get %d result" %(num))
        self.assertEqual(2,num)

if __name__ == "__main__":
    unittest.main()
```