Chapter 11: Python Pickle and Unpickle

Python pickle module is used for serializing and de-serializing a Python object structure. Any object in Python can be pickled so that it can be saved on disk. What pickle does is that it “serializes” the object first before writing it to file. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. The idea is that this character stream contains all the information necessary to reconstruct the object in another python script.

Example 1: Creating a pickle

```
import pickle
mylist = ['a', 'b', 'c', 'd']
with open('datafile.txt', 'wb') as fh:
   pickle.dump(mylist, fh)
```

Example 2: Unpickling

```
import pickle
pickle_off = open ("datafile.txt", "rb")
data = pickle.load(pickle_off)
print(data)
```

Example 3: Creating a pickle of dictionary

```
import pickle
EmpID = {1:"Zack",2:"53050",3:"IT",4:"38",5:"Flipkart"}
pickling_on = open("EmpID.pickle","wb")
pickle.dump(EmpID, pickling_on)
pickling_on.close()
```

Example 4: Unpickling a dictionary

```
import pickle
pickle_off = open("EmpID.pickle", 'rb')
EmpID = pickle.load(pickle_off)
print(EmpID)
```

Example 5: Pickling and Unpickling

```
import pickle
def storeData():
    # initializing data to be stored in db
    sachin = {'name' : 'Sachin', 'fullname' : 'Sachin Ramesh Tendulkar',
    'age' : 50, 'pay' : 4000000}
    ravi  = {'name' : 'Ravi', 'name' : 'Ravi Nallan',
    'age' : 45, 'pay' : 500000}
  # database
    db = {}
    db['sachin'] = sachin
    db['ravi'] = ravi

    # Its important to use binary mode
    dbfile = open('examplePickle', 'ab')

    # source, destination
    pickle.dump(db, dbfile)
    dbfile.close()

def loadData():
    # for reading also binary mode is important
    dbfile = open('examplePickle', 'rb')
    db = pickle.load(dbfile)
    print(db)
    for keys in db:
        print(keys, '=>', db[keys])
    dbfile.close()

if __name__ == '__main__':
    storeData()
    loadData()
```