Python Advance Assignment 4

1. Explain the differences between Cassandra and typical databases.

   Solution:- **Cassandra**-Cassandra is a high-performance and highly scalable distributed NoSQL database management system. Cassandra deals with unstructured data and handles a high volume of incoming data velocity. In Cassandra data is written in many locations also data come from many locations this row represents a unit of replication and the column represents a unit of storage.

   RDBMS:- Relational Database Management System (RDBMS) is a Database management system or software that is designed for relational databases and uses Structured Query Language (SQL) for querying and maintaining the database. It deals with structured data and handles moderate incoming data velocity. In RDBMS mainly data is written in one location also data come from one/few locations and a row represents a single record column that represents an attribute.

2. What exactly is CQLSH?

   Solution:- CQLSH is a command-line interface for interacting with Cassandra using CQL (the Cassandra Query Language). It is shipped with every Cassandra package, and can be found in the bin/ directory alongside the cassandra executable. `CQLSH` is implemented with the Python native protocol driver, and connects to the single specified node.

3. Explain the Cassandra cluster idea.

Solution:- A Cassandra cluster is **a collection of nodes, or Cassandra instances, visualized as a ring**. Cassandra clusters can be defined as “rack aware” or “datacenter aware” so that data replicas could be distributed in a way that could even survive physical outages of underlying infrastructure.

4. Give an example to demonstrate the class notion.

Solution:- A class is a user-defined blueprint or prototype from which objects are created. Classes provide a means of bundling data and functionality together. Creating a new class creates a new type of object, allowing new instances of that type to be made. Each class instance can have attributes attached to it for maintaining its state. Class instances can also have methods (defined by their class) for modifying their state.

Example:-

```
class Dog:


    attr1 = "mammal"
    attr2 = "dog"

    def fun(self):
        print("I'm a", self.attr1)
        print("I'm a", self.attr2)


Rodger = Dog()

print(Rodger.attr1)
```

Output:-

```
mammal
I'm a mammal
I'm a dog
```

5.

Solution:- **EXAMPLE:-**Person(Human) can be treated as a class which has properties such as name, age,gender etc. Every individual can be treated as an object of the class human or Person. Each individual will have different values of the properties of class Person.Everyone will have different names, age and gender.