1.**[Cassandra](https://www.geeksforgeeks.org/apache-cassandra-nosql-database/):** Cassandra is a high-performance and highly scalable distributed [NoSQL database](https://www.geeksforgeeks.org/introduction-to-nosql/) 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**](https://www.geeksforgeeks.org/rdbms-full-form/)**:** Relational Database Management System (RDBMS) is a Database management system or software that is designed for relational databases and uses [Structured Query Language (SQL)](https://www.geeksforgeeks.org/sql-tutorial/) 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.

**Difference between Cassandra and RDBMS:**

| S.No. | CASSANDRA | RDBMS |
| --- | --- | --- |
| 1. | Cassandra is a high performance and highly scalable distributed NoSQL database management system. | RDBMS is a Database management system or software which is designed for relational databases. |
| 2. | Cassandra is a NoSQL database. | RDBMS uses SQL for querying and maintaining the database. |
| 3. | It deals with unstructured data. | It deals with structured data. |
| 4. | It has a flexible schema. | It has fixed schema. |
| 5. | Cassandra has peer-to-peer architecture with no single point of failure. | RDBMS has master-slave core architecture means a single point of failure. |
| 6. | Cassandra handles high volume incoming data velocity. | RDBMS handles moderate incoming data velocity. |
| 7. | In RDBMS there is limited data source means data come from many locations. | In Cassandra there are various data source means data come from one/few location. |
| 8. | It supports simple transactions. | It supports complex and nested transactions. |
| 9. | In Cassandra the outermost container is Keyspace. | In RDBMS the outermost container is database. |
| 10. | Cassandra follows decentralized deployments. | RDBMS follows centralized deployments. |
| 11. | In Cassandra data written in many locations. | In RDBMS mainly data are written in one location. |
| 12. | In Cassandra row represents a unit of replication. | In RDBMS row represents a single record. |
| 13. | In Cassandra column represents a unit of storage. | In RDBMS column represents an attribute. |
| 14. | In Cassandra, relationships are represented using collections. | In RDBMS relationships are represented using keys and join etc. |

2. This chapter introduces the Cassandra query language shell and explains how to use its commands.

By default, Cassandra provides a prompt Cassandra query language shell **(cqlsh)** that allows users to communicate with it. Using this shell, you can execute **Cassandra Query Language (CQL)**.

Using cqlsh, you can

* define a schema,
* insert data, and
* execute a query.

3. Cassandra is based on distributed system architecture. In its simplest form, Cassandra can be installed on a single machine or in a docker container, and it works well for basic testing. A single Cassandra instance is called a **node**. Cassandra supports **horizontal scalability** achieved by adding more than one node as a part of a Cassandra **cluster**. The scalability works with linear performance improvement if the resources are configured optimally.

4. **📖 A Student Hub**

For high school and college students alike, this Notion template is intended for those seeking a rubric to keep track of class grades. Included in this template is also a template for daily to-dos and unique class notebooks. The example used is for an American college freshman.

5. Python is an object-oriented programming language. Everything is in Python treated as an object, including variable, function, list, tuple, dictionary, set, etc. Every object belongs to its class. For example - An integer variable belongs to integer class. An object is a real-life entity. An object is the collection of various data and functions that operate on those data. An object contains the following properties.

* **State -** The attributes of an object represents its state. It also reflects the properties of an object.
* **Behavior -** The method of an object represents its behavior.
* **Identity -** Each object must be uniquely identified and allow interacting with the other objects.

Let's understand the object in the aspect of classes.

The classes and objects are the essential key to the object-oriented programming. Classes are the blueprint of the object. Classes are used to bundling the data and functionality together. Each newly created class must have its object. Let's understand real-life example of class and object.

A human is a class which may have may attributes such as walking, sleeping, thinking, etc. Suppose we want to name and age of 100 humans, so we don't need to create a class for every person. We just need to instantiate the multiple objects of that perticular class.