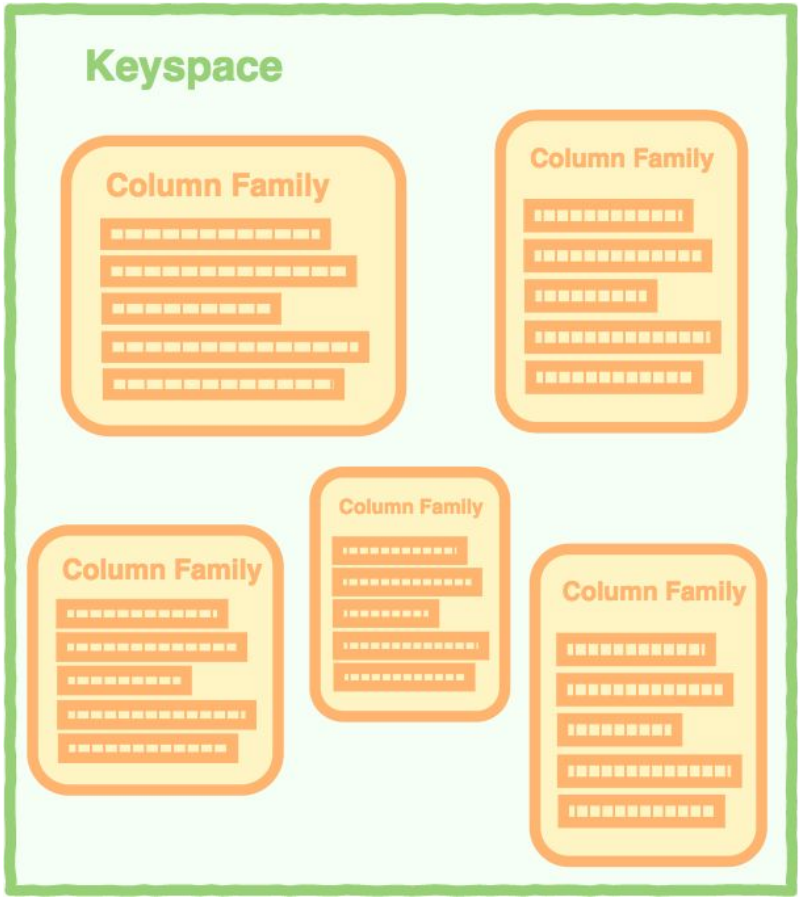
Unit - 3

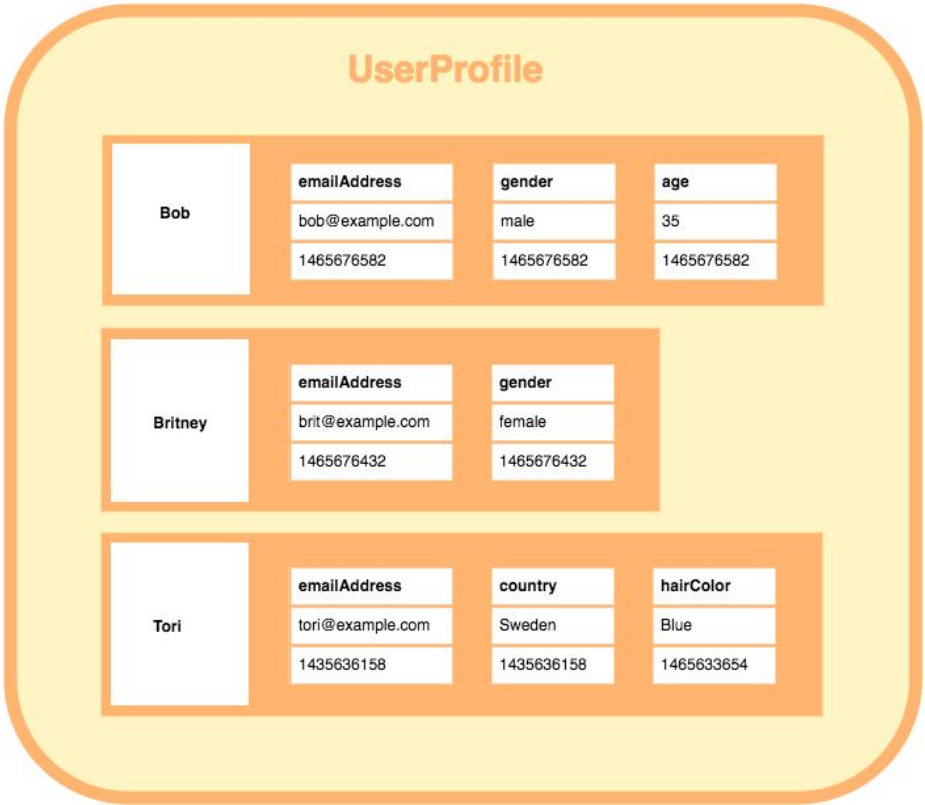
Column-Oriented Databases

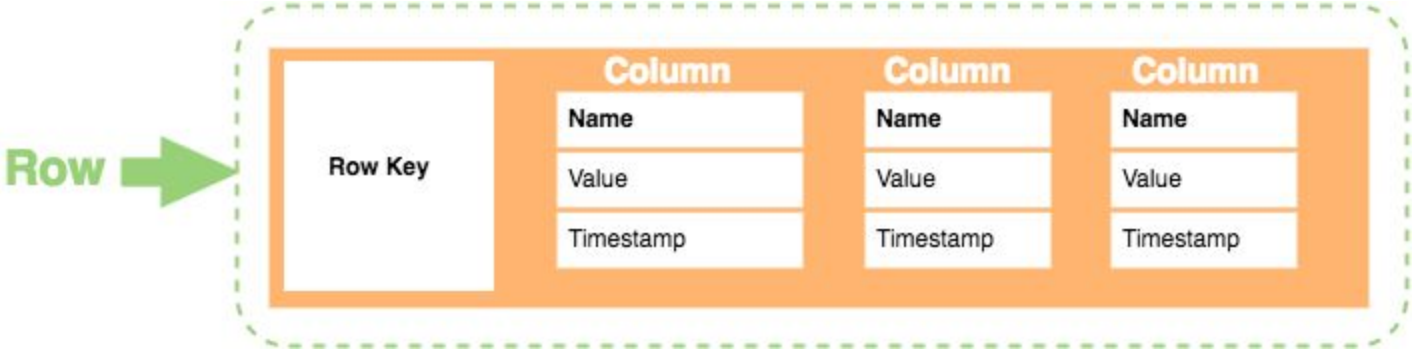
What Is a Column-Family Data Store

- A column store database is a type of database that stores data using a column oriented model.
- Columns store databases use a concept called a keyspace. A keyspace is kind of like a schema in the relational model. The keyspace contains all the column families (kind of like tables in the relational model), which contain rows, which contain columns.



Ex:


UserProfile
Bob
emailAddress
bob@example.com
1465676582
gender
male
1465676582
age
35
1465676582
Britney
emailAddress
brit@example.com
1465676432
gender
female
1465676432
Tori
emailAddress
tori@example.com
1435636158
country
Sweden
1435636158
hairColor
Blue
1465633654



Here’s a breakdown of each element in the row:

- Row Key. Each row has a unique key, which is a unique identifier for that row.
- Column. Each column contains a name, a value, and timestamp.
- Name. This is the name of the name/value pair.
- Value. This is the value of the name/value pair.
- Timestamp. This provides the date and time that the data was inserted. This can be used to determine the most recent version of data.

Benefits

- Compression. Column stores are very efficient at data compression and/or partitioning.
- Aggregation queries. Due to their structure, columnar databases perform particularly well with aggregation queries (such as SUM, COUNT, AVG, etc).
- Scalability. Columnar databases are very scalable. They are well suited to massively parallel processing (MPP), which involves having data spread across a large cluster of machines – often thousands of machines.
- Fast to load and query. Columnar stores can be loaded extremely fast. A billion row table could be loaded within a few seconds. You can start querying and analysing almost immediately.

Examples of Column Store DBMSs

- Bigtable
- Cassandra
- HBase
- Vertica
- Druid
- Accumulo
- Hypertable

- Apache Cassandra is an open source, distributed and decentralized/distributed storage system (database), for managing very large amounts of structured data spread out across the world. It provides highly available service with no single point of failure.
- Notable points
  - Apache Cassandra was initially designed at Facebook to implement a combination of Amazon's Dynamo distributed storage and replication techniques and Google's Bigtable data and storage engine model.
  - It was open-sourced by Facebook in July 2008.
  - Cassandra was accepted into Apache Incubator in March 2009.
  - It is a column-oriented NoSQL database
  - Scalable, fault-tolerant and consistent
  - Cassandra implements a Dynamo-style replication model with no single point of failure, but adds a more powerful "column family" data model.
  - Cassandra is being used by some of the biggest companies such as Facebook, Twitter, Cisco, Rackspace, ebay, Twitter, Netflix, and more.

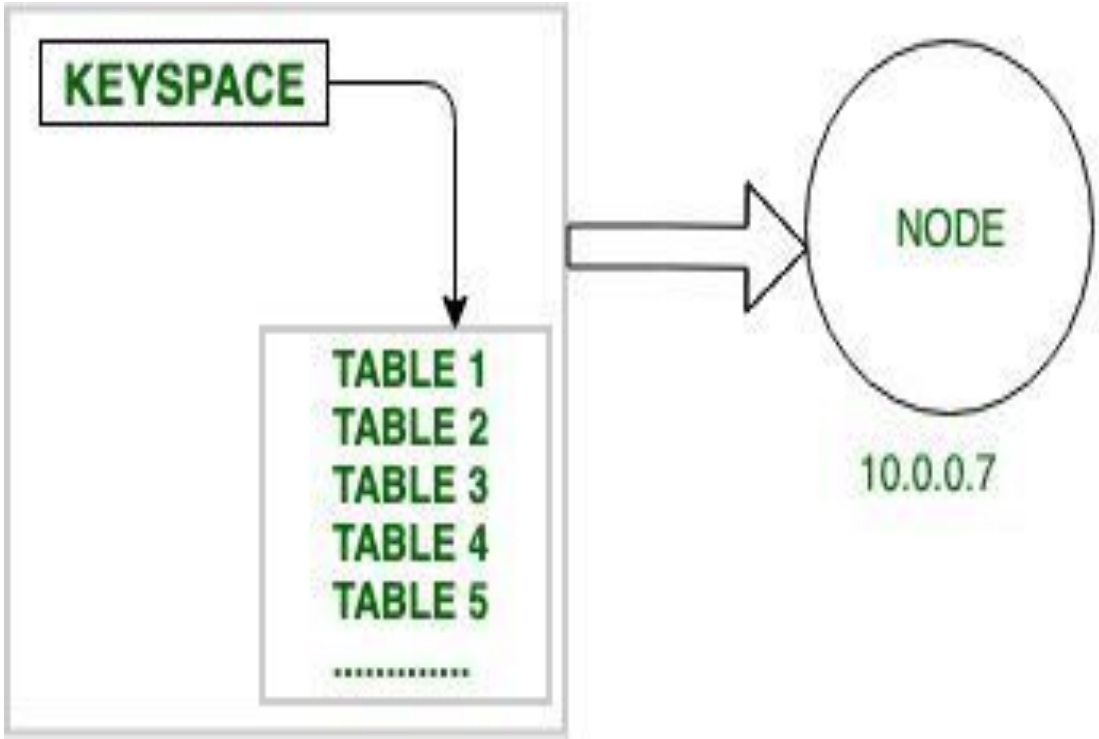
Cassandra Architecture

- Basic Terminology:
  - Node
  - Data center
  - Cluster
- Operations:
  - Read Operation
  - Write Operation
- Storage Engine:
  - CommitLog
  - Memtables
  - SSTables
- Data Replication Strategies

Node

Node:

Node is the basic component in Apache Cassandra. It is the place where actually data is stored. For Example:As shown in diagram node which has IP address 10.0.0.7 contain data (keyspace which contain one or more tables).



DataCenter

Data Center is a collection of nodes.

For example:

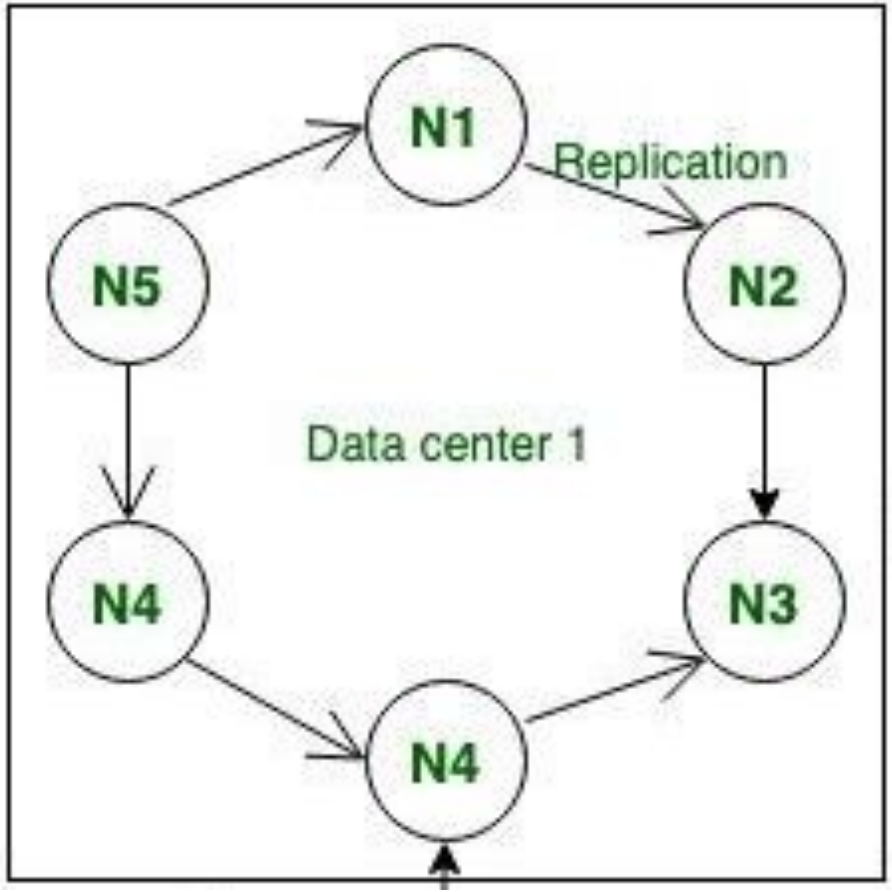
DC – N1 + N2 + N3 ….

DC: Data Center

N1: Node 1

N2: Node 2

N3: Node 3



Cluster

It is the collection of many data centers.

For example:

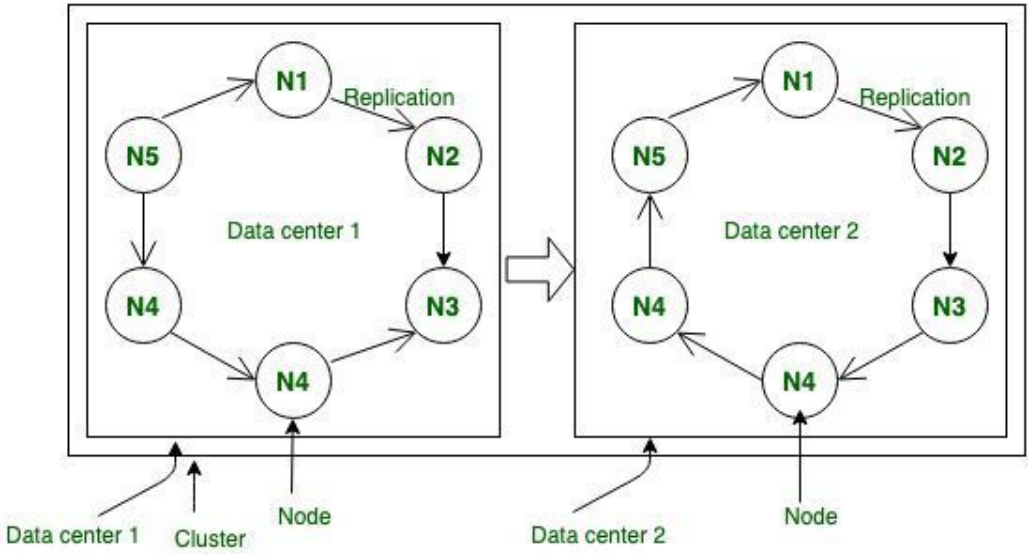
C = DC1 + DC2 + DC3….

C: Cluster

DC1: Data Center 1

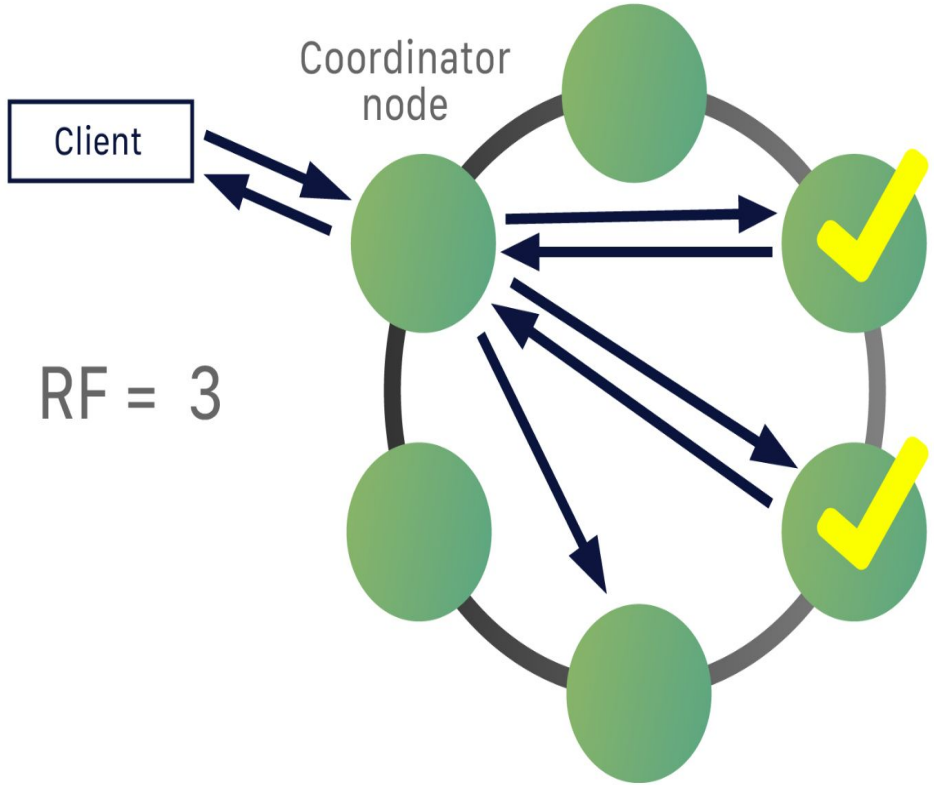
DC2: Data Center 2

DC3: Data Center 3



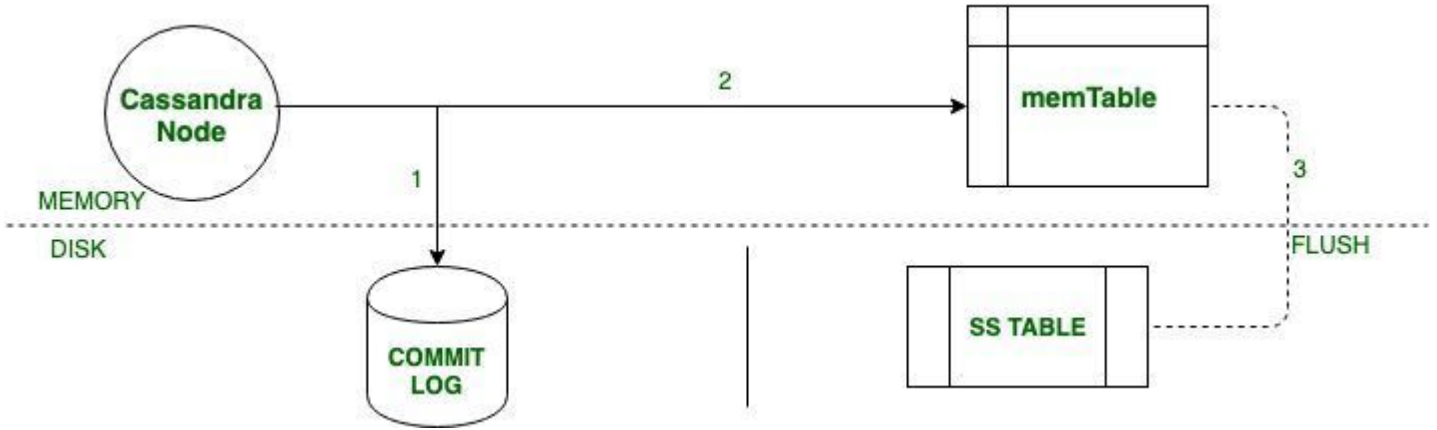
Read Operation

- Direct Request
  - Send request to one of the nodes
- Digest Request
  - Send request to N nodes as specified in the CONSISTENCY parameter
- Read - Repair Request
  - Will be triggered in case of INCONSISTENT nodes to make them consistent



Write Operation

- Step 1 : Write Operation as soon as we receives request then it is first dumped into commit log to make sure that data is saved.
- Step-2: Insertion of data into table that is also written in MemTable that holds the data till it’s get full.
- Step-3: If MemTable reaches its threshold then data is flushed to SS Table.

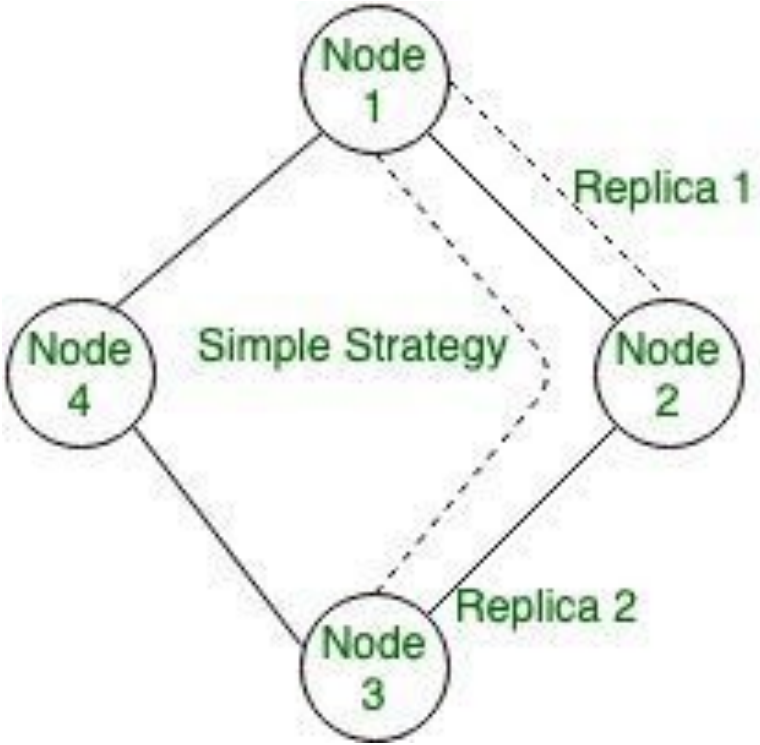


Replication Strategies

- It is used to ensure that there are no point of failures, Each data item is replicated at N hosts, where N is the replication factor configured \per-instance
- There are 2 types of replication strategies
  - Simple Strategy
  - Network Topology STrategy

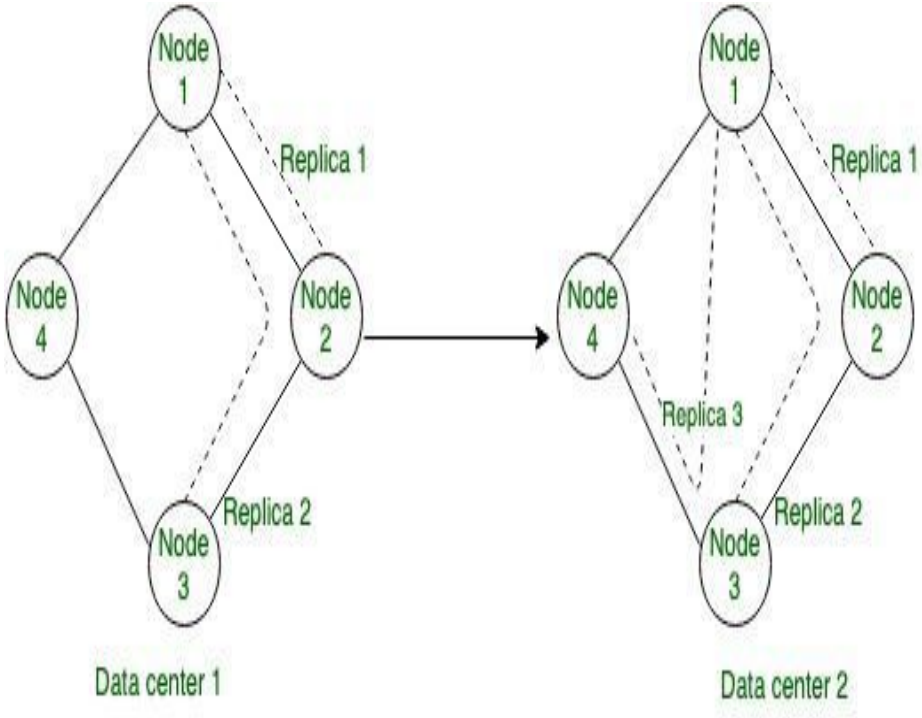
Simple Strategy

In this Strategy it allows a single integer RF (replication_factor) to be defined. It determines the number of nodes that should contain a copy of each row. For example, if replication_factor is 2, then two different nodes should store a copy of each row.



- Network Topology STrategy

In this strategy it allows a replication factor to be specified for each datacenter in the cluster. Even if your cluster only uses a single datacenter. This Strategy should be preferred over SimpleStrategy to make it easier to add new physical or virtual datacenters to the cluster later.



Cassandra Data types

Data Type	Constants	Description
ascii	strings	Represents ASCII character string
bigint	bigint	Represents 64-bit signed long
blob	blobs	Represents arbitrary bytes
Boolean	booleans	Represents true or false
counter	integers	Represents counter column
decimal	integers, floats	Represents variable-precision decimal
double	integers	Represents 64-bit IEEE-754 floating point
float	integers, floats	Represents 32-bit IEEE-754 floating point
inet	strings	Represents an IP address, IPv4 or IPv6
int	integers	Represents 32-bit signed int
text	strings	Represents UTF8 encoded string
timestamp	integers, strings	Represents a timestamp
timeuuid	uuids	Represents type 1 UUID
uuid	uuids	Represents type 1 or type 4
		UUID
varchar	strings	Represents uTF8 encoded string
varint	integers	Represents arbitrary-precision integer

Collection Types

Cassandra Query Language also provides a collection data types. The following table provides a list of Collections available in CQL.

Collection	Description
list	A list is a collection of one or more ordered elements.
map	A map is a collection of key-value pairs.
set	A set is a collection of one or more elements.

User-defined datatypes

Cqlsh provides users a facility of creating their own data types. Given below are the commands used while dealing with user defined datatypes.

- **CREATE TYPE** − Creates a user-defined datatype.
- **ALTER TYPE** − Modifies a user-defined datatype.
- **DROP TYPE** − Drops a user-defined datatype.
- **DESCRIBE TYPE** − Describes a user-defined datatype.
- **DESCRIBE TYPES** − Describes user-defined datatypes.

Create KeySpace

```
CREATE KEYSPACE Demo
WITH replication = {'class':'SimpleStrategy',
'replication_factor' : 2};
```

Alter KeySpace

```
ALTER KEYSPACE Demo

WITH replication = {'class':'SimpleStrategy',
'replication_factor' : 3};
```

Drop KeySpace

```
DROP KEYSPACE Demo;
```

CRUD operations on tables

Create:

Create table personal details(sid int, sname text, spg boolean, emails set<text>);

CRUD operations on tables

Read

```
Select * from personal;
```

CRUD operations on tables

Update

Update personal set spg=false where sid=123

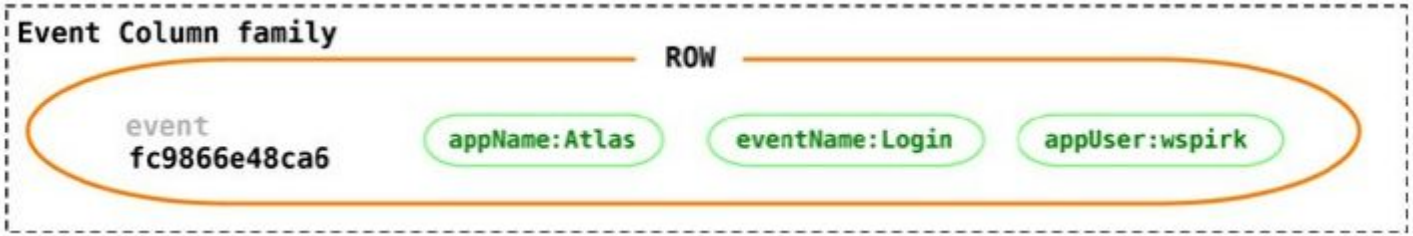
CRUD operations on tables

Delete

```
Delete from personal where sid=143;
```

Suitable use cases

- Event Logging



- Content Management Systems, Blogging Platforms
- Counters
- Expiring usage

When not to use

- Databases which requires ACID properties for reads and writes
- Early prototypes which requires query change, as the cost of query change is more when compared to schema change.