cassandra

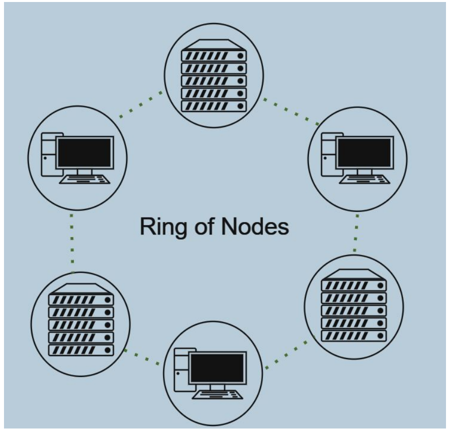
by Nick Valentino, Patrick Schaeffer, and Kayley Harnett

Database paradigm - Wide Column

- Effectively manage large semi-structured datasets.
- A wide-column database is a specialized type of NoSQL database designed to efficiently handle and manage substantial semi-structured or sparse data sets.

What makes the paradigm unique

- Cassandra's use of Nodes being linked together:
  - A node can hold ~2-4Tb of data
  - Nodes are linked together in a ring allowing for more protection
- Data is replicated between nodes
  - If one node goes down data will not be lost
  - Replication Factor is the number times data is replicated
- The nodes distribute workload
  - Better performance



A Node is a single instance of a computer or server running cassandra

Big name/types of users

- Netflix
- Activision
- Apple
- Discord
- IBM
- Instagram
- Spotify
- REDDIT

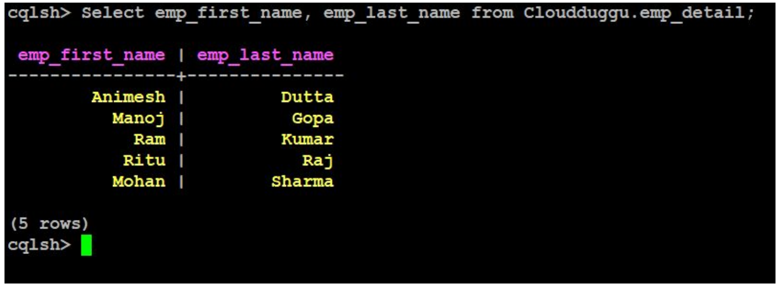
How does it support CRUD operations

Create

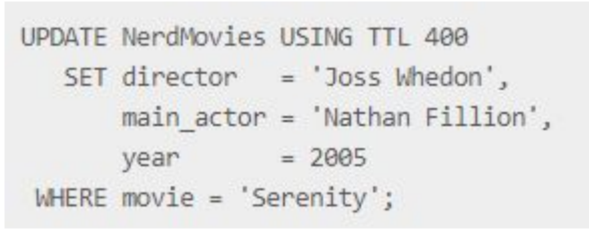
- uses CREATE KEYSPACE/CREATE TABLE
- returns an error if keyspace/table already exists

Read

- Uses SELECT statements



```
cqlsh> Select emp_first_name, emp_last_name from Cloudduggu.emp_detail;

 emp_first_name | emp_last_name
----------------+---------------
        Animesh |         Dutta
          Manoj |          Gopa
            Ram |         Kumar
           Ritu |           Raj
          Mohan |        Sharma

(5 rows)
cqlsh>
```



```
UPDATE NerdMovies USING TTL 400
   SET director   = 'Joss Whedon',
       main_actor = 'Nathan Fillion',
       year       = 2005
 WHERE movie = 'Serenity';
```

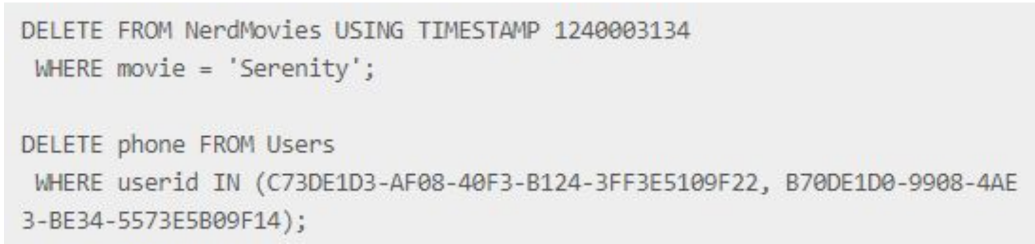
Update

- uses UPDATE statement
- does not check if the row exists - if it exists it is updated, if not it is created

Delete

- uses DELETE statement
- can specify a specific column to delete; removes whole row by default

All updates and deletions within same partition key are applied atomically + in isolation



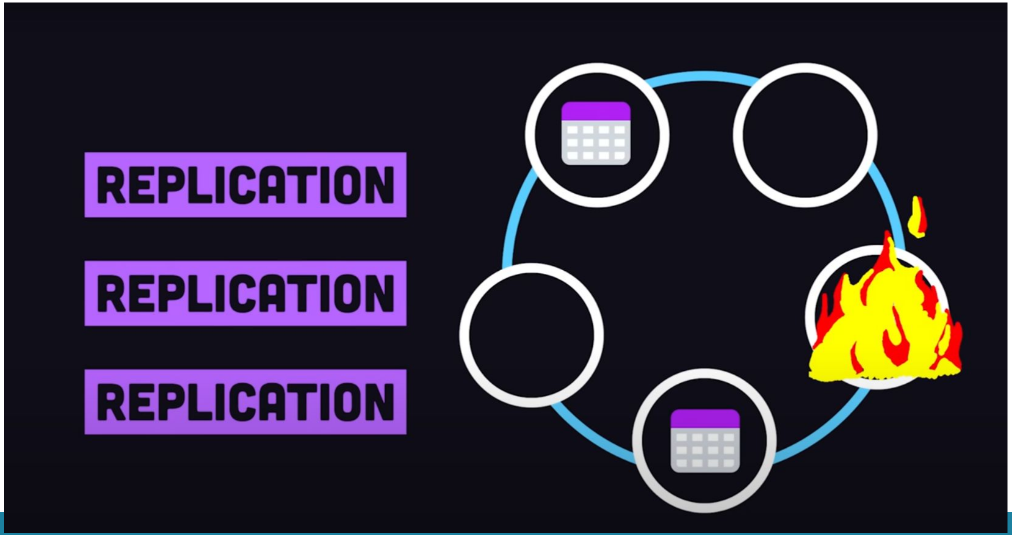
```
DELETE FROM NerdMovies USING TIMESTAMP 1240003134
 WHERE movie = 'Serenity';

DELETE phone FROM Users
 WHERE userid IN (C73DE1D3-AF08-40F3-B124-3FF3E5109F22, B70DE1D0-9908-4AE
3-BE34-5573E5B09F14);
```

Does it support ACID properties?

- The latest release of Cassandra (5.0) does support ACID compliant transactions
- With so many copies of the Database how does it keep them all in sync?
  - Uses ACCORD - a timestamp protocol to decide which transactions to do when
- Uses read repair to check data replicas against each other and scan for data inconsistencies and return most current information
  - blocking - info isn’t passed to user until the read repair is complete and information is up to date
- Does support consistent Hashing
  - Uses a partition key to distribute data among its nodes

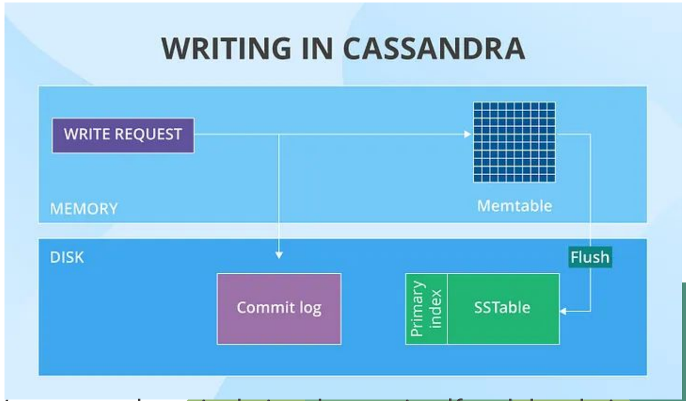
Does it support sharding/replications? How

Cassandra supports data replication - due to Cassandra's nodes being all linked together, if one node goes down, there is still data somewhere else that can be called on. If the data is just lost in a node it will be restored.



How does it perform?

- Cassandra's Performance is really fast this is due to each instance of cassandra being ran can do work to itself before being implemented to the rest of the nodes.
- Since every instance will have a copy of the data We can make reads/writes really fast since it’s taking information from that instance.



We can see the write being done to itself and then being stored into the memtable and then commit log

How does it scale?

Nodes allow Cassandra to scale horizontally - meaning that storage can be added by adding more computers or servers to the mix.

How does an application connect and communicate with the DB

- We can use SDK's to connect and use in our programming languages like python, C++, and Java
- CQL (Cassandra Query Language)
  - Very similar to SQL