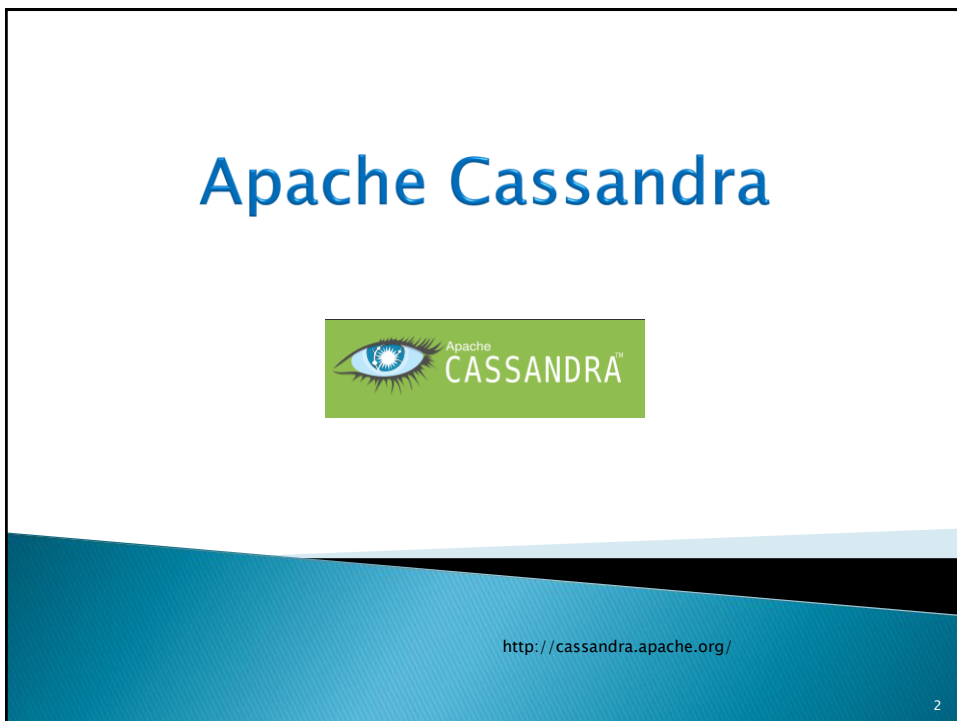


1



2

Objectives

- ▶ To introduce Apache Cassandra
- ▶ To describe Cassandra's architecture
- ▶ To discuss data model
- ▶ To introduce Cassandra Query Language (CQL)
- ▶ To perform CRUD operations
- ▶ To discuss transactions



3

3

Part 1



<http://cassandra.apache.org/>

4

4

Overview

- ▶ Apache Cassandra is an open source, distributed and decentralized 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.
- ▶ Cassandra is a column-oriented database.
- ▶ Its distribution design is based on Amazon's Dynamo and its data model on Google's Bigtable.

5

5

Cassandra Facts



- ▶ Official Online Resources
 - <http://cassandra.apache.org>
- ▶ Technologies and Language
 - implemented in Java.
- ▶ Access Methods
 - A command-line access to the store and Cqlsh interface (new); Column Family-CQL API, Table-Thrift API as well as an internal Java API.
 - Clients for multiple languages including Java, Python, Grails, PHP, .NET and Ruby are available.
 - Hadoop integration is also supported.
- ▶ Query Language:
 - Cassandra Query Language (CQL) in Cassandra query language shell (cqlsh)
- ▶ Open-Source License:
 - Apache License version 2.
- ▶ Who Uses It :
 - Facebook, Digg, Reddit, Twitter, and others.

6

6

History



- ▶ Cassandra was developed at Facebook for inbox search.
- ▶ It was named to reference a Greek myth – Cassandra was a cursed oracle, who would tell the truth, but nobody would believe her.
 - The developers thought their database design was better than Oracle for handling data at scale.
- ▶ It was open-sourced by Facebook in July 2008.
- ▶ Cassandra was accepted into Apache Incubator in March 2009.
- ▶ It was made an Apache Software Foundation (ASF) top-level project since February 2010.
 - ASF provides support for 300+ Apache Projects and their Communities, furthering its mission of providing Open Source software for the public good.
 - The Apache Incubator provides services to projects which want to enter the Apache Software Foundation (ASF).
 - Other ASF database projects include:
 - [Apache CouchDB](#)
 - [Apache Derby](#)
 - [Apache HBase](#)

<https://www.apache.org/>

7

7

Advantages



- ▶ **Fast writes:**
 - Cassandra was designed to run on cheap commodity hardware. It performs very fast writes and can store hundreds of terabytes of data, without sacrificing the read efficiency.
- ▶ **Elastic scalability:**
 - Cassandra is highly scalable; it allows to add more hardware to accommodate more customers and more data as per requirement.
 - Cassandra is linearly scalable, i.e., it increases your throughput as you increase the number of nodes in the cluster. Therefore it maintains a quick response time.
- ▶ **'Always on' architecture:**
 - Cassandra has no single point of failure and it is continuously available for business-critical applications that cannot afford a failure.
- ▶ **Flexible data storage:**
 - Cassandra accommodates all possible data formats including: structured, semi-structured, and unstructured. It can dynamically accommodate changes to your data structures according to your need.
 - Cassandra has SQL-like query language and support search through secondary indexes.
- ▶ **Easy data distribution:**
 - Cassandra provides the flexibility to distribute data where you need by replicating data across multiple data centres.
- ▶ **Transaction support:**
 - Cassandra supports properties like Atomicity, Isolation, and Durability (AID).

8

8

When to use Cassandra?

- ▶ If your application has the following characteristics then you should definitely use Cassandra :
 - The main focus is on the writes i.e the number of writes is exceptionally high as compared to the number of reads.
 - There are minimum updates in your application.
 - There is a requirement to integrate with Big Data, Hadoop, Hive, Spark etc.
 - There is a need for real-time data analytics and report generations.
 - The application is distributed.
 - There is no need for joins or aggregates.
 - You want your application to work globally.

9

9

Disadvantages



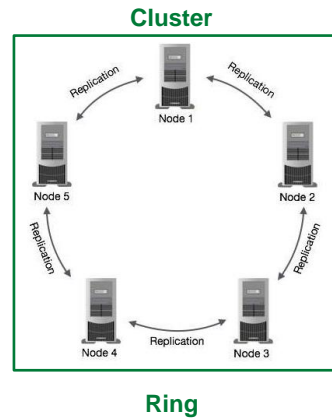
- ▶ No Support for full ACID Properties :
 - Cassandra does not provide full ACID and relational data properties.
- ▶ No support for Aggregates :
 - Cassandra does not support aggregates, if you need to do a lot of them, choose another database.
- ▶ Latency :
 - Making excessive requests and reading more data slows down the actual transaction, resulting in latency issues.
- ▶ Joins can be an Issue :
 - No join or subquery support. You may be able to find a workaround for this one, but that might affect the performance and increase the overhead.
- ▶ Data Duplication :
 - Here data is modelled around queries instead of its structure due to which same data is store multiple times.
- ▶ Slow Reads :
 - Reads are slower. Cassandra was optimized from the beginning for fast writes. Reads were not as much of a concern but that quickly changed as more use cases were considered.
- ▶ JVM Memory management can be an issue :
 - To store huge amount of data, JVM is required to manage the memory which itself is a language, and so garbage collection is not done by the application but by a language in Cassandra.

10

10

Architecture

- ▶ Cassandra has peer-to-peer distributed system across its nodes, and data is distributed among all the nodes in a **Cluster** - the outermost container.
- ▶ Cluster sometimes called the ring, because Cassandra assigns data to nodes in the cluster by arranging them in a ring
 - All the nodes in a cluster play the same role.
 - Each node
 - is a single machine running Cassandra,
 - is independent and at the same time interconnected to other nodes,
 - can accept read and write requests, regardless of where the data is actually located in the cluster.
 - When a node goes down, read/write requests can be served from other nodes in the network.



https://commons.wikimedia.org/wiki/File:Data_replication.jpg

11

11

Data Model



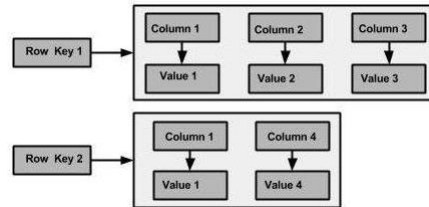
- ▶ The data model of Cassandra is significantly different from an RDBMS.
- ▶ It consists of **four main components**:
- ▶ **Cluster:**
 - The outermost structure in Cassandra
 - Made up of multiple nodes and keyspaces
 - Collection of related nodes can be combined into a Data centres - useful components when serving customers in different geographical areas
 - Cluster can contain one or more data centres
- ▶ **Keyspace:**
 - A keyspace is the container for data in Cassandra (like a database in RDBS)
 - The keyspace is used to group Column families together.
 - Each keyspace has at least one and often many column families.
- ▶ **Column Family:**
 - Is a container for an ordered collection of rows, each of which is itself an ordered collection of columns. (Multiple columns with the row key reference)
 - You can freely add any column to any column family at any time, depending on your needs
- ▶ **Column:**
 - Consisting of a column name, value, and timestamp

12

12

Column Family

- ▶ A column family is a container for an ordered collection of rows.
 - Each row, in turn, is an ordered collection of columns.



- ▶ Column
 - A column is the basic data structure of Cassandra with three values, namely key or column name, value, and a time stamp.
- ▶ SuperColumn
 - A super column is a special column, therefore, it is also a key-value pair. But a super column stores a map of sub-columns.

13

13

Column Family vs Relational Table

- ▶ Unlike relational tables, Cassandra does not force individual rows to have all the columns.

Relational Table	Cassandra column Family
A schema in a relational model is fixed. Once we define certain columns for a table, while inserting data, in every row all the columns must be filled at least with a null value.	In Cassandra, although the column families are defined, the columns are not. You can freely add any column to any column family at any time.
Relational tables define only columns and the user fills in the table with values.	In Cassandra, a table contains columns, or can be defined as a super column family.

14

14

Column and SuperColumn

▶ Column

- A column is the basic data structure of Cassandra with three values, namely key or column name, value, and a time stamp. Given below is the structure of a column.

▶ Structure of column

Column		
name	value	timestamp

▶ SuperColumn

- A super column is a special column, therefore, it is also a key-value pair. But a super column stores a map of sub-columns.
- Generally column families are stored on disk in individual files.
 - Therefore, to optimize performance, it is important to keep columns that you are likely to query together in the same column family, and a super column can be helpful here.

▶ Structure of Super Column

SuperColumn	
name	cols: map <byte[], column>

15

Quiz



1. Project of Cassandra was taken by which of the following developer?

- Amazon
- LinkedIn
- Google
- Apache

2. What is the difference between a Cluster and a Keyspace?

3. Which of the following API is used for Column Families and Tables in Cassandra?

- Column Family-CQL API ;Table-CQL API
- Column Family-CQL API ;Table-Thrift API
- Column Family-Thrift API ;Table-Thrift API
- Column Family-Thrift API ;Table-CQL API

16

16

Part 2

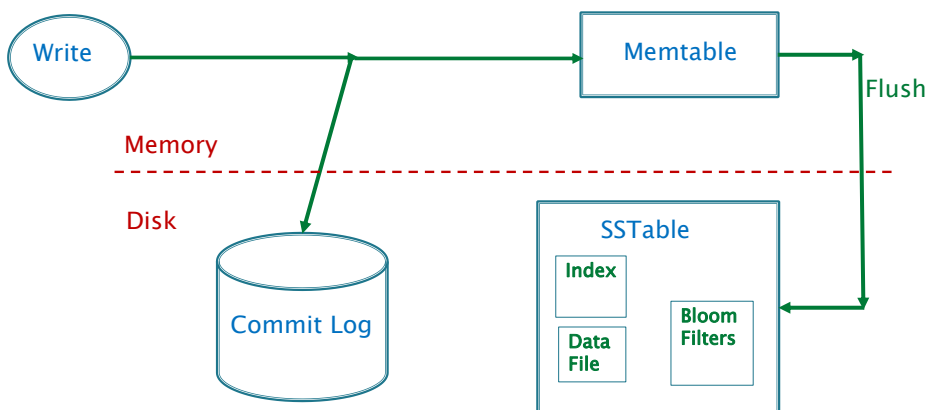


<http://cassandra.apache.org/>

17

17

Write operation

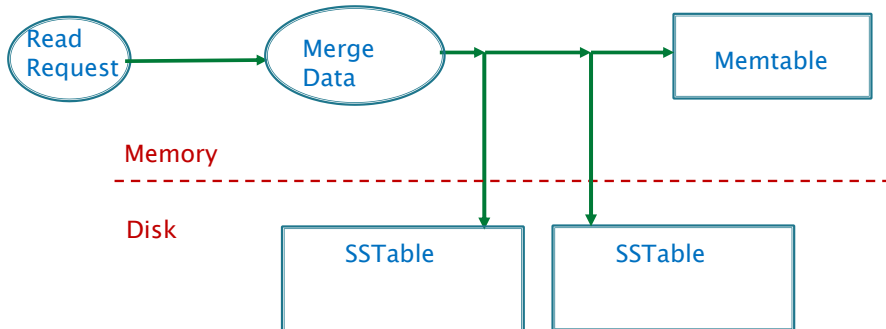


- All writes are automatically partitioned and replicated throughout the cluster.
- Cassandra periodically consolidates the SSTables, discarding unnecessary data.

18

18

Read operation



- For every read request Cassandra needs to read data from all applicable SSTables (all SSTables for a column family) and scan the memtable for applicable data fragments.

19

19

Components



- Commit log
 - is a crash-recovery mechanism in Cassandra. Every write operation is written to the commit log.
- Mem-table
 - is a memory-resident data structure. After commit log, the data will be written to the mem-table. Sometimes, for a single-column family, there will be multiple mem-tables.
- SSTable
 - It is a disk file to which the data is flushed from the mem-table when its contents reach a threshold value.
- Bloom filter
 - These are quick, nondeterministic, algorithms for testing whether an element is a member of a set. It is a special kind of cache. Bloom filters are accessed after every query.

20

20

Cassandra Query Language

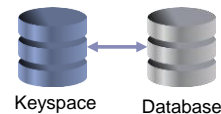
- ▶ By default, Cassandra provides a prompt Cassandra query language shell (**cqlsh**) that allows users to communicate with it.
- ▶ Using this shell, you can execute **Cassandra Query Language (CQL)**.
 - To create a table,
 - To insert data,
 - To execute a query.
- ▶ CQL treats the database (Keyspace) as a container of tables.
- ▶ To start cqlsh:



```
$ cqlsh
Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.11.5 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
cqlsh>
```

21

Keyspace



- ▶ A keyspace is the container for data in Cassandra.
- ▶ Like a relational database, a keyspace has a name and a set of attributes that define keyspace-wide behaviour.
- ▶ Basic attributes of a **Keyspace**:
 - **Keyspace Name**
 - **Replication factor**
 - It is the number of machines in the cluster that will receive copies of the same data.
 - **Replica placement strategy**- the strategy to place replicas in the ring:
 - **simple strategy** (replicas in a single data centre, in a manner that is not aware of their placement on a data centre rack),
 - **old network topology strategy** (rack-aware strategy)
 - **network topology strategy** (allows you to specify more evenly than the above strategy how replicas should be placed across data centres).
 - **Durable Writes** – optional, provides a means to instruct Cassandra whether to use commitlog for updates on the current keyspace or not (default value is TRUE)

```
cqlsh> CREATE KEYSPACE test
      WITH replication = {'class':'SimpleStrategy','replication_factor':3};
cqlsh> USE test;
cqlsh:test>
```

22

CQL Data Definition Commands

Command	Description
CREATE KEYSPACE	Creates a KeySpace in Cassandra.
USE	Connects to a created KeySpace.
ALTER KEYSPACE	Changes the properties of a KeySpace
DROP KEYSPACE	Removes a KeySpace
CREATE TABLE	Creates a table in a KeySpace.
ALTER TABLE	Modifies the column properties of a table.
DROP TABLE	Removes a table.
TRUNCATE	Removes all the data from a table.
CREATE INDEX	Defines a new index on a single column of a table.
DROP INDEX	Deletes a named index.

23

23

CQL Data Manipulation Commands

► DML commands

Command	Description
INSERT	Adds columns for a row in a table.
UPDATE	Updates a column of a row.
DELETE	Deletes data from a table.
BATCH	Executes multiple DML statements at once.

► CQL clauses

Clause	Description
SELECT	This clause reads data from a table
WHERE	The where clause is used along with select to read a specific data.
ORDERBY	The orderby clause is used along with select to read a specific data in a specific order.

24

24

Data Types



Simple data types:

Data Type	Description
blob	Represents arbitrary bytes
Boolean	Represents true or false
counter	Represents counter column
decimal	Represents variable-precision decimal
double	Represents 64-bit IEEE-754 floating point
float	Represents 32-bit IEEE-754 floating point
int	Represents 32-bit signed int
varchar	Represents UTF8 encoded string
varint	Represents arbitrary-precision integer

Collections:

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.

25

User-defined datatypes and functions

- ▶ **Cqlsh** provides users a facility of creating their own data types.
- ▶ The commands to use 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.
- ▶ Cassandra also allows users to create user-defined functions (UDFs) and user-defined aggregate functions (UDAs).
- ▶ Functions are used to manipulate stored data in queries. Retrieving results using standard aggregate functions are also available for queries.

26

26

Quiz



1. Which of the following is not part of the Cassandra architecture?
 - A. Mem-table
 - B. Big-table
 - C. Bloom filter
 - D. Commit Log

2. How can a user create a keyspace in Cassandra?
 - A. Keyspace.Create()
 - B. Create keyspace
 - C. Keyspace.Build

3. In which of the following data type order of the element entered is maintained?
 - A. Set
 - B. Map
 - C. List
 - D. None of these

27

27

Part 3



<http://cassandra.apache.org/>

28

28

Create Table

- ▶ You can create a table using the command **CREATE TABLE**.

- ▶ Syntax:

```
CREATE (TABLE | COLUMNFAMILY) <tablename>
('<column-definition>' , '<column-definition>')
(WITH <option> AND <option>)
```

- ▶ The primary key is a column that is used to uniquely identify a row.
 - Defining a primary key is mandatory while creating a table.
 - A primary key is made of one or more columns of a table.
 - Example:

```
cqlsh> USE rainforest;
cqlsh:rainforest> CREATE TABLE emp(
    emp_id int PRIMARY KEY,
    emp_name varchar,
    emp_city varchar,
    emp_sal decimal,
    emp_phone varchar
);

cqlsh:rainforest> select * from emp;

 emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+-----
(0 rows)
```

A column family and a table are the same thing.
 The name "column family" was used in the older Thrift API.
 The name "table" is used in the newer CQL API.
 More info on the APIs can be found here: <http://wiki.apache.org/cassandra/API>

29

Alter Table

- ▶ Using **ALTER** command, you can perform the following operations
 - Add a column
 - Drop a column

```
cqlsh:rainforest> ALTER TABLE emp
... ADD emp_email text;

cqlsh:rainforest> select * from emp;

 emp_id | emp_city | emp_email | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+-----+-----
(0 rows)

cqlsh:rainforest> ALTER TABLE emp DROP emp_email;

cqlsh:rainforest> select * from emp;

 emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+-----
(0 rows)
```

30

30

Drop Table & Truncate Table

- ▶ You can drop a table using the command **Drop Table**.

```
cqlsh:rainforest> DROP TABLE emp;
cqlsh:rainforest> DESCRIBE COLUMNFAMILIES;
```

- ▶ You can truncate a table using the TRUNCATE command.
 - When you truncate a table, all the rows of the table are deleted permanently.

```
cqlsh:rainforest> select * from student;

 s_id | s_aggregate | s_branch | s_name
-----+-----+-----+-----
    1 |           70 |      IT |   ram
    2 |           75 |      EEE |  rahman
(2 rows)

cqlsh:rainforest> TRUNCATE student;

cqlsh:rainforest> select * from student;

 s_id | s_aggregate | s_branch | s_name
-----+-----+-----+-----
(0 rows)
```

31

31

DML – INSERT

- ▶ You can insert data into the columns of a row in a table using the command **INSERT**

```
cqlsh:rainforest> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal)
VALUES(1,'ram', 'Hyderabad', 9848022338, 50000);

cqlsh:rainforest> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal)
VALUES(2,'robin', 'Hyderabad', 9848022339, 40000);

cqlsh:rainforest> INSERT INTO emp (emp_id, emp_name, emp_city, emp_phone, emp_sal)
VALUES(3,'rahman', 'Chennai', 9848022330, 45000);

cqlsh:rainforest> SELECT * FROM emp;

 emp_id | emp_city | emp_name | emp_phone | emp_sal
-----+-----+-----+-----+-----
    1 | Hyderabad |      ram | 9848022338 | 50000
    2 | Hyderabad |     robin | 9848022339 | 40000
    3 |  Chennai |    rahman | 9848022330 | 45000
(3 rows)
```

32

32

DML – UPDATE

- ▶ **UPDATE** is the command used to update data in a table. The following keywords are used while updating data in a table –
 - **Where** – This clause is used to select the row to be updated.
 - **Set** – Set the value using this keyword.
 - **Must** – Includes all the columns composing the primary key.

```
cqlsh:rainforest> UPDATE emp SET emp_city='Delhi',emp_sal=50000 WHERE emp_id=2;
cqlsh:rainforest> select * from emp;
```

emp_id	emp_city	emp_name	emp_phone	emp_sal
1	Hyderabad	ram	9848022338	50000
2	Delhi	robin	9848022339	50000
3	Chennai	rahman	9848022330	45000

(3 rows)

33

33

DML – DELETE

- ▶ You can delete data from a table using the **DELETE** command.
 - You can delete one column data:

```
cqlsh:rainforest> DELETE emp_sal FROM emp WHERE emp_id=3;
cqlsh:rainforest> select * from emp;
```

emp_id	emp_city	emp_name	emp_phone	emp_sal
1	Hyderabad	ram	9848022338	50000
2	Delhi	robin	9848022339	50000
3	Chennai	rahman	9848022330	null

(3 rows)

- Or an entire row from a table:

```
cqlsh:rainforest> DELETE FROM emp WHERE emp_id=3;
cqlsh:rainforest> select * from emp;
```

emp_id	emp_city	emp_name	emp_phone	emp_sal
1	Hyderabad	ram	9848022338	50000
2	Delhi	robin	9848022339	50000

(2 rows)

34

34

BATCH

- Using **BATCH**, you can execute multiple modification statements (insert, update, delete) simultaneously.

```
cqlsh:rainforest> select * from emp;
```

emp_id	emp_city	emp_name	emp_phone	emp_sal
1	Hyderabad	ram	9848022338	50000
2	Delhi	robin	9848022339	50000
3	Chennai	rahman	9848022330	45000

(3 rows)

```
cqlsh:rainforest> BEGIN BATCH
... INSERT INTO emp (emp_id, emp_city, emp_name, emp_phone, emp_sal)
  values ( 4, 'Pune', 'rajeev', 9848022331, 30000);
... UPDATE emp SET emp_sal = 50000 WHERE emp_id = 3;
... DELETE emp_city FROM emp WHERE emp_id = 2;
... APPLY BATCH;
cqlsh:rainforest> select * from emp;
```

emp_id	emp_city	emp_name	emp_phone	emp_sal
1	Hyderabad	ram	9848022338	50000
2	null	robin	9848022339	50000
3	Chennai	rahman	9848022330	50000
4	Pune	rajeev	9848022331	30000

(4 rows)

5

35

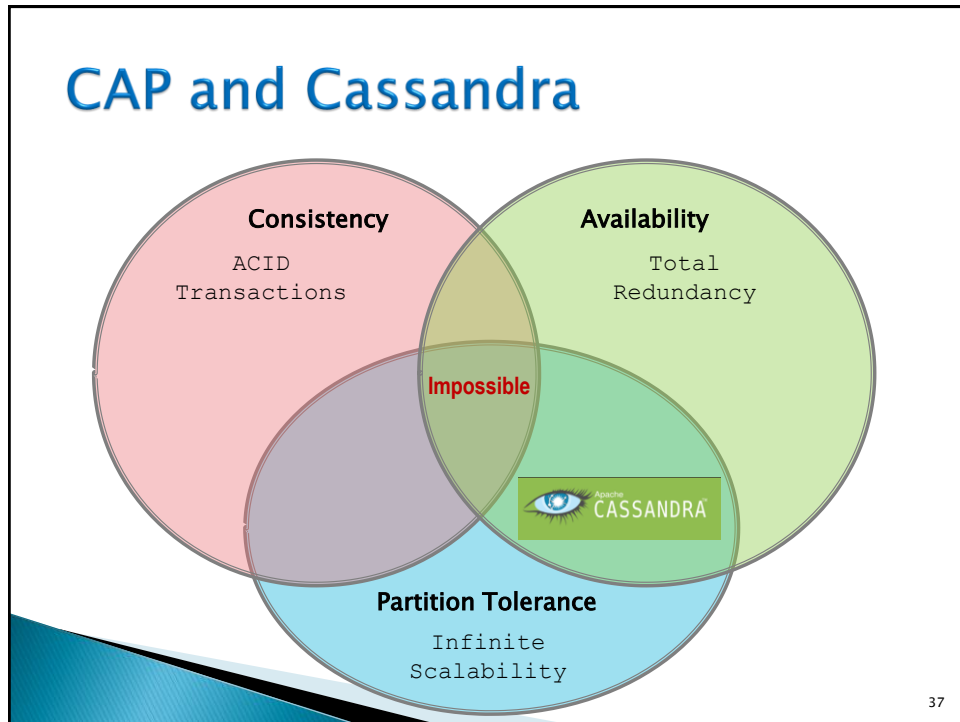
Transactions



- While Cassandra does not support ACID (transaction) properties but it gives you the 'AID' among it .
 - That is Writes to Cassandra are atomic, isolated, and durable in nature.
 - The "C" of ACID—consistency—does not apply to Cassandra, as there is no concept of referential integrity or foreign keys.
- Apache Cassandra operations follow the BASE paradigm which means that they are basically Available, Soft-state Eventually-consistent.
- Each such BASE operation can have a consistency level.
 - Cassandra offers you to tune your **Consistency Level** as per your needs .
 - You can either have partial or full consistency that is you might want a particular request to complete if just one node responds, or you might want to wait until all nodes respond.
 - Using lower consistency levels yield higher availability and better latency at the price of weaker consistency.


36

36



37

Quiz



- CRUD stands for _____.
 - Create Update Read Delete
 - Change Upload Return Download
 - Create Update Read Drop
- Which of the following attribute can be used to execute a statement repeatedly?
 - Copy Paste
 - Repeat
 - Batch
 - Index
- What command is used to delete all the rows in a table?
 - Clear
 - Truncate
 - Delete
 - Drop

38

38

Further Reading

- ▶ Books:
 - Carpenter J., Cassandra : the definitive guide, O'Reilly Media, 2016
 - Bradberry R., Lubow E., Practical Cassandra: A Developer's Approach, Addison-Wesley Professional; 2013
- ▶ Articles:
 - Introduction to Apache Cassandra's Architecture:
 - <https://dzone.com/articles/introduction-apache-cassandras>
 - Apache Cassandra: The Truth Behind Tunable Consistency, Lightweight Transactions & Secondary Indexes.
 - <https://blog.yugabyte.com/apache-cassandra-lightweight-transactions-secondary-indexes-tunable-consistency/>
- ▶ Documentation:
 - The Cassandra Query Language (CQL):
 - <http://cassandra.apache.org/doc/latest/cql/index.html>
- ▶ Videos:
 - What is Apache Cassandra? Apache Cassandra Tutorial:
 - <https://www.youtube.com/watch?v=iDhIjrJ7hG0>



39

39

Essentials

- ✓ Introduced Apache Cassandra
- ✓ Described Cassandra's architecture
- ✓ Discussed data model
- ✓ Introduced Cassandra Query Language (CQL)
- ✓ Performed CRUD operations
- ✓ Discussed transactions



40

40