Massive Amount of Data and Apache Cassandra

By: Bridget Brown and Sunanda Reddy

What is Apache Cassandra?

It is a NoSQL (Not Only SQL) database management system, created to handle mass amounts of data across multiple databases. This program is free to users and is widely used by many companies due to its easy design without compromising performance.

Apache Cassandra does not have a single point of failure due to the way it stores and distributes data. It is created as a Columnar Storage Architecture which distributes data across homogenous nodes. Each node performs all other operations therefore if one fails, it is redirected to the near most node.

Where did the Research Begin?

Avinash Lakshman and Prashant Malik invented Apache Cassandra exclusively for the inbox search feature for Facebook.

In 2008 they made Cassandra public, as an open source on Google Code.

In 2009 Apache saw how innovative this creation was and acquired it from facebook, it became an Apache Incubator Project.

In 2010 Cassandra usage started to skyrocket and has been a top performing software of Apache since.

Today Cassandra is used by such relevant and widely known companies such as Netflix, Twitter, Instagram, Uber, etc.

Quick Demo

Step 1: Create Keyspace in Cassandra and use it (same way we create database in SQL)

```
CREATE KEYSPACE cassandra_dev IF NOT EXISTS store WITH
REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor' : '1' };
```

```
USE cassandra_dev;
```

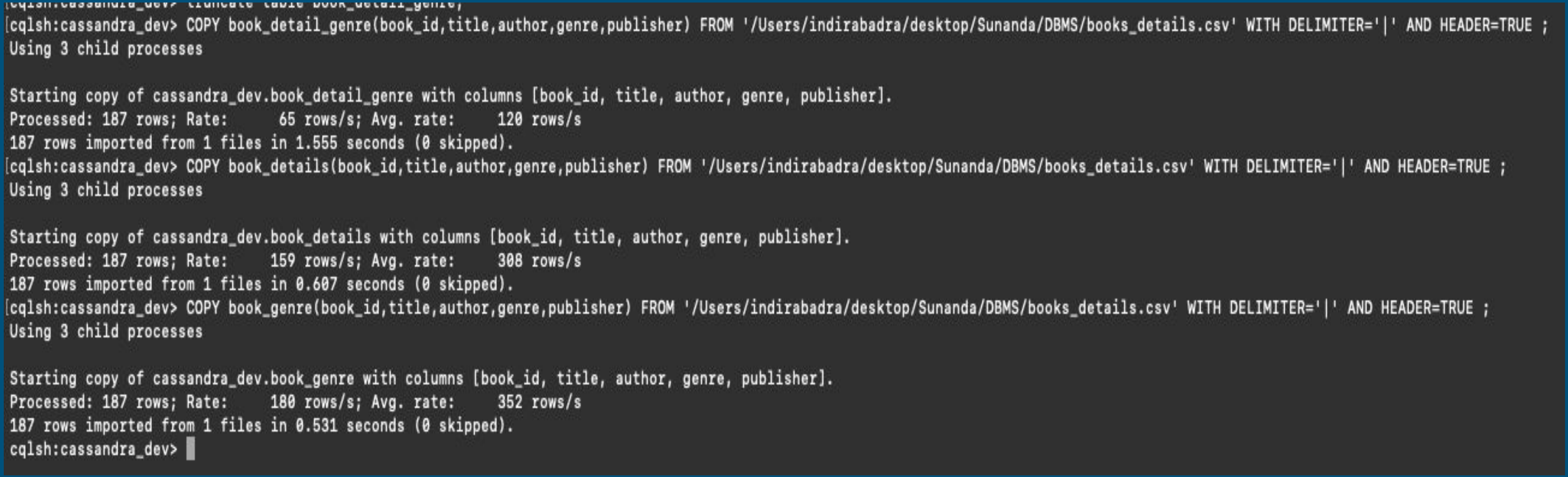
Quick Demo

Step 2: Create tables by having different combination of primary key.

CREATE TABLE IF NOT EXISTS Book_details (Book_id INT, Title TEXT, Author TEXT, Genre TEXT, Publisher TEXT, PRIMARY KEY(Book_id));	CREATE TABLE IF NOT EXISTS Book_genre (Book_id INT, Title TEXT, Author TEXT, Genre TEXT, Publisher TEXT, PRIMARY KEY(genre));	CREATE TABLE IF NOT EXISTS Book_detail_genre (Book_id INT, Title TEXT, Author TEXT, Genre TEXT, Publisher TEXT, PRIMARY KEY(genre,book_id));

Quick Demo

Step 3: Load Data from CSV file using Copy command

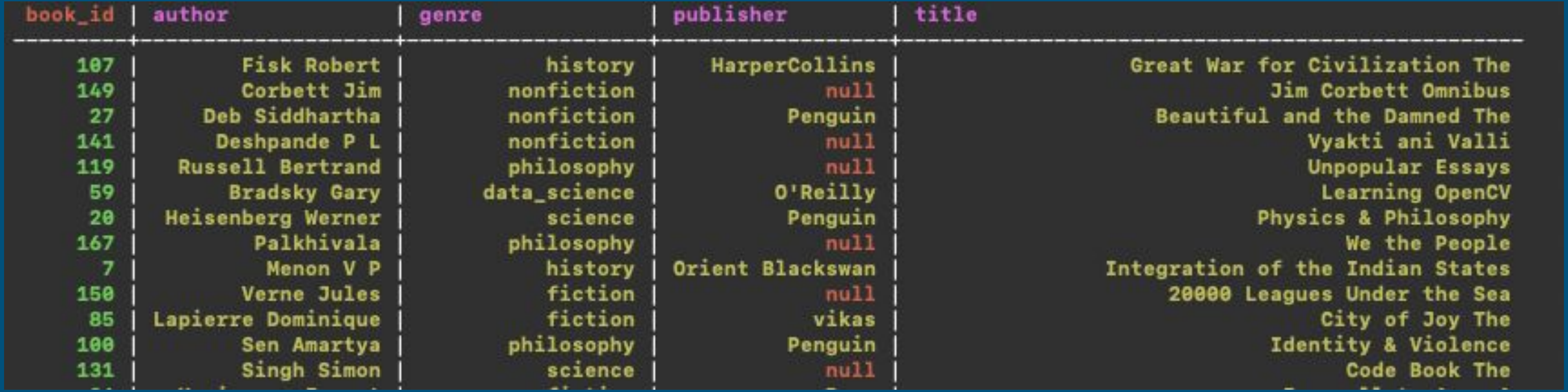


```
[cqlsh:cassandra_dev> truncate table book_detail_genre;
[cqlsh:cassandra_dev> COPY book_detail_genre(book_id,title,author,genre,publisher) FROM '/Users/indirabadra/desktop/Sunanda/DBMS/books_details.csv' WITH DELIMITER='|' AND HEADER=TRUE ;
Using 3 child processes

Starting copy of cassandra_dev.book_detail_genre with columns [book_id, title, author, genre, publisher].
Processed: 187 rows; Rate:      65 rows/s; Avg. rate:     120 rows/s
187 rows imported from 1 files in 1.555 seconds (0 skipped).
[cqlsh:cassandra_dev> COPY book_details(book_id,title,author,genre,publisher) FROM '/Users/indirabadra/desktop/Sunanda/DBMS/books_details.csv' WITH DELIMITER='|' AND HEADER=TRUE ;
Using 3 child processes

Starting copy of cassandra_dev.book_details with columns [book_id, title, author, genre, publisher].
Processed: 187 rows; Rate:     159 rows/s; Avg. rate:     308 rows/s
187 rows imported from 1 files in 0.607 seconds (0 skipped).
[cqlsh:cassandra_dev> COPY book_genre(book_id,title,author,genre,publisher) FROM '/Users/indirabadra/desktop/Sunanda/DBMS/books_details.csv' WITH DELIMITER='|' AND HEADER=TRUE ;
Using 3 child processes

Starting copy of cassandra_dev.book_genre with columns [book_id, title, author, genre, publisher].
Processed: 187 rows; Rate:     180 rows/s; Avg. rate:     352 rows/s
187 rows imported from 1 files in 0.531 seconds (0 skipped).
cqlsh:cassandra_dev> 
```

Quick Demo

Step 4: Query from tables to retrieve data based on partition keys provided as filter and check for differences.

Select * from book_details; select * from book_genre; select * from book_detail_genre



```
 book_id | author              | genre               | publisher          | title
---------+---------------------+---------------------+--------------------+-------------------------------------------
     107 |        Fisk Robert  |            history  |     HarperCollins  |     Great War for Civilization The
     149 |        Corbett Jim  |         nonfiction  |              null  |            Jim Corbett Omnibus
      27 |    Deb Siddhartha   |         nonfiction  |           Penguin  |      Beautiful and the Damned The
     141 |     Deshpande P L   |         nonfiction  |              null  |               Vyakti ani Valli
     119 |  Russell Bertrand   |         philosophy  |              null  |               Unpopular Essays
      59 |       Bradsky Gary  |       data_science  |          O'Reilly  |                Learning OpenCV
      20 | Heisenberg Werner   |            science  |           Penguin  |            Physics & Philosophy
     167 |        Palkhivala   |         philosophy  |              null  |                  We the People
       7 |          Menon V P  |            history  | Orient Blackswan   |   Integration of the Indian States
     150 |        Verne Jules  |            fiction  |              null  |        20000 Leagues Under the Sea
      85 | Lapierre Dominique  |            fiction  |             vikas  |                 City of Joy The
     100 |       Sen Amartya   |         philosophy  |           Penguin  |             Identity & Violence
     131 |       Singh Simon   |            science  |              null  |                  Code Book The
```

Some Features

- **Query Language:** CQL(Cassandra Query Language) an alternative to SQL so its very efficient and easy for many people to use.
- **Scalability:** It is designed to read and write throughout. You are able to add new data centers when and nodes when needed, fast and without trouble.
- **Fault- Tolerance:** Due to the homogenous nodes, the program cannot fail. Makes it extremely business friendly to people who cannot afford mistakes/ data lossage.
- **Flexible Data Storage:** Accepts all kinds of storage formats for example, structured, semi-structured, and unstructured
- **Easy Data Distribution:** Replicates data across multiple centers making it quick and easy. Even when the system is under high stress the data will be evenly distributed across nodes. Cassandra can also run on multiple machines while still uniform.

Importance

Network Costs are High

- With Cassandra you are able to keep the costs down because there is no master node that needs to keep receiving data.

Our Favorite Favorite Applications use it

- For example Netflix uses it for their geographical capabilities, its flexibility, and strong performance.

Does not have a single failure point

Makes People's lives easier

- It is easy to use and it is free, therefore businesses are easily able to use it and not have issues with extra costs/ learning a new complicated system.
- Most companies do not have room for error, Cassandra eliminates that risk.

Cassandra VS SQL

- Cassandra is able to hold more data than SQL
- Facilitates automatic distribution, therefore fast data transfer while SQL is manual
- Cassandra handles everything SQL does plus videos, images, sounds, etc. SQL can only handle text, characters and numbers.
- The writing performance is higher in Cassandra
- It is of high scalability, while SQL has many limitations

Work Citation

A. Chebotko, A. Kashlev and S. Lu, "A Big Data Modeling Methodology for Apache Cassandra," 2015 IEEE International Congress on Big Data, 2015, pp. 238-245, doi: 10.1109/BigDataCongress.2015.41.

“Netflix Heads into the Clouds.” Rik Farrow, Feb. 2012.

R. Burtica, E. M. Mocanu, M. I. Andreica and N. Ţăpuş, "Practical application and evaluation of no-SQL databases in Cloud Computing," 2012 IEEE International Systems Conference SysCon 2012, 2012, pp. 1-6, doi: 10.1109/SysCon.2012.6189510.

<Question and Answers>

Thank You