Big Data Warehouse:
Building Columnar NoSQL OLAP Cubes

Khaled Dehdouh, Computer Engineering Department of Cherchell Military Academy, TIPAZA Algeria

Omar Boussaid, ERIC Laboratory/ University of Lyon 2, Bron, France

Fadila Bentayeb, ERIC Laboratory/ University of Lyon, Lyon 2, Bron, France

ABSTRACT

In the Big Data warehouse context, a column-oriented NoSQL database system is considered as the storage model which is highly adapted to data warehouses and online analysis. Indeed, the use of NoSQL models allows data scalability easily and the columnar store is suitable for storing and managing massive data, especially for decisional queries. However, the column-oriented NoSQL DBMS do not offer online analysis operators (OLAP). To build OLAP cubes corresponding to the analysis contexts, the most common way is to integrate other software such as HIVE or Kylin which has a CUBE operator to build data cubes. By using that, the cube is built according to the row-oriented approach and does not allow to fully obtain the benefits of a column-oriented approach. In this article, the focus is to define a cube operator called MC-CUBE (MapReduce Columnar CUBE), which allows building columnar NoSQL cubes according to the columnar approach by taking into account the non-relational and distributed aspects when data warehouses are stored.

KEYWORDS

Big Data, Columnar, Data Warehouses, NoSQL Model

INTRODUCTION

The data warehouse is a database for online analytical processing (OLAP) to aid decision-making. It is designed according to a dimensional modelling which has for objective to observe facts through measures, also called indicators, according to the dimensions that represent the analysis axes (Inmon, 1992). It is often implemented in the relational database management system (RDBMS) (Chaudhuri & Dayal, 1997) Thanks to the OLAP (On-Line Analytical Processing), the users can create multidimensional representations related to the particular analysis contexts in compliance with the specific needs, according to the criteria which they define, called hypercubes or OLAP cubes (Chaudhuri & Dayal, 1997). Cube computation produces aggregations that are beyond the limits of the Group by (Gray et al., 1997). For example, in the case of calculation of the sum, it computes in a multidimensional way and returns sub-totals and totals for all possible combinations. This involves performance of all aggregations according to all levels of hierarchies of all dimensions. For a cube with three dimensions A, B and C, the performed aggregations relate to the following combinations:

DOI: 10.4018/IJDSST.2020010101

(A, B, C), (A, B, ALL), (A, ALL, C), (ALL, B, C), (A, ALL, ALL), (ALL, B, ALL), (ALL, ALL, C), (ALL, ALL, ALL). The (A, B, C) combination corresponds as the lowest (least) aggregate level of the cube, and the rest are considered as the high aggregate levels. The advent of the big data has created new opportunities for researchers to achieve high relevance and impact amid changes and transformations in how we study several science phenomena. Companies like Google and Microsoft are analyzing large volumes of data for business analysis and decisions, which impact the existing and the future technologies (Gandomi & Haider, 2015).

However, unusual volumes of data become an issue when faced with the limited capacities of traditional systems, especially when data storage is in a distributed environment which requires the use of parallel treatment as MapReduce paradigm (Dear & Ghemawat, 2004). To solve a part of this issue, other models have appeared such as the column-oriented NoSQL (Not Only SQL) which gives a data structure more adequate to the massive data warehouses (Bhogal & Choski, 2015). In the big data warehouses context, a column-oriented NoSQL database system is considered as the storage model which is highly adapted to data warehouses and online analysis (Rabuzin & Modruan, 2014). Indeed, the storage of data column by column allows values belonging to the same column to be shared in the same disk space which improves the column access time enormously when the aggregate operations are performed. Furthermore, the non-relational aspect that characterizes the NoSQL model when data are stored allows to deploy data easily in a distributed environment (Jerzy, 2012).

To build OLAP cubes corresponding to the analysis contexts, the most common way is to integrate other softwares such as HIVE which has a CUBE operator to build data cubes. By using that, the cube is built according to the row-oriented approach and does not allow to fully obtain the benefits of a column-oriented approach. To solve this problem, we propose an aggregation operator, called MC-CUBE (MapReduce Columnar CUBE) which allows OLAP cubes to be computed according to the columnar approach from big data warehouses implemented by using column-oriented NoSQL model. MC-CUBE implements the invisible join, used by the columnar RDBMS (Abadi et al., 2008), in order to compute aggregation from several tables and extend it to take into account all possible aggregations at different levels of granularity of the cube. To deal with very large data, MC-CUBE uses the MapReduce paradigm when handling data stored in a distributed environment.

We have evaluated the performance of MC-CUBE operator on star schema benchmark (SSB) (O'Neil et al., 2007), implemented within the column-oriented NoSQL DBMS HBase1[1] using Hadoop2[2]. The HBase DBMS and the Hadoop platform were chosen because of their distributed context which was necessary for storing and analyzing big data.

The rest of this paper is organized as follows. Section 2 gives a related work. Section 3 introduces basic concepts about columnar NoSQL Data warehouse. Section 4 explains the columnar approach that we propose for building a data cube. Section 5 introduces the MC-CUBE operator and shows the execution phases through an example. Section 6 shows performance results and exemplifies of MC-CUBE operator when OLAP cubes are performed. Finally, Section 7 concludes the paper and suggests some possible directions for future research work.

RELATED WORK

Big data have led data warehouses towards to distributed environments to store and to analyze the large amount of data. Since the column storage has outperformed the row storage, several research projects based on the columnar relational model have been commercialized such as InfoBright (Iezak & eastwood, 2009), Brighthouse (Iezak et al., 2008), Vectorwise (Zukowski et al., 2012), MonetDB (Idreos et al., 2012), SAP HANA (Farber et al., 2012), Blink (Barber et al., 2012), and Vertica (Lamb et al., 2012). These systems have led legacy RDBMS vendors to add columnar storage options to their existing engines (Larson et al., 2012). However, the relational model that is often used for storing data warehouses has shown its limits. Indeed, the use of distributed solutions based on the relational model is as costly as the implementation of the referential integrity constraints