Leveraging Key-Value NoSQL Databases for Enhanced Decision Support Systems: A Comparative Analysis

SUMMARY REPORT

Chethiya Galkaduwa (CS/2017/008), Praveen Bhawantha (CS/2017/017)
Faculty of Computing and Technology, University of Kelaniya

Abstract

The implementation of a big data mart using NoSQL technologies, in particular the key-value store, is investigated in this report. It examines the relationship between modern database management systems' benefits and traditional data warehousing methods. Based on actual experience using the Oracle NoSQL Database, the process of going from conceptual schema to logical models is detailed, and its benefits and drawbacks are noted.

Introduction

In today's data-driven economy, companies gather data from various channels to gain a competitive edge and achieve strategic goals. Traditional data warehousing approaches need to be revised in handling significant data volumes, leading to research into alternative database systems like NoSQL. This report aims to improve decision support systems by mapping conceptual models to logical models, using three transformation patterns and the TCP-H Benchmark to evaluate Read Request Latency.[2]

Research on NoSQL databases has shifted towards decision-making information systems and warehousing. Studies propose multidimensional data warehouse implementation using document-oriented databases and introduce two other models. A columnar NoSQL data warehousing benchmark is presented, but no multidimensional schema is provided. Three approaches use column-oriented NoSQL systems for effective data warehouse implementation.

A study proposes a transformation approach for implementing UML class diagrams in column-oriented NoSQL databases. Two transformation rules are proposed for instantiating data warehouses under column-oriented and document-oriented models. Other research contributions include formalizing logical data models and using graph-oriented NoSQL databases for big data warehouses. Key-value models are not widely studied in NoSQL data warehousing solutions.

Methodology

I. Formalization

Formalization of the concept and the models enables researchers and practitioners to establish clear and precise representations of data marts, data warehouses, and database management systems. Using NoSQL key-value stores can break down into two parts,

A. Multidimensional schema

This is considered the core of the data mart architecture, and this is widely used to build dimensional data marts and data warehouses. Formally the MS is defined by,

$Func^{MS}: F \rightarrow \{D_1, \ldots, Dm\}$, where:

$F \in \{F_1, \ldots, Fi\}$, a Fact instance $\{D_1, \ldots, D_m\} \subset \{D_1, \ldots, D_n\}$, $m \prec n$, a set of dimensions.

$Func^{MS}$, a function that links the fact with its associated dimensions.

A fact contains the data to be analyzed, known as measures, and maintains links to associated dimensions. Its defined as a triplet (N^F, M^F, L^F), where:

- N^F is the Fact name.
- M^F is a finite set of measures $\{m_1, \ldots, m_n\}$ associated with aggregation functions.
- L^F is a finite set of links to dimensions $\{l_1, \ldots, l_n\}$

In data analysis, dimensions hold the descriptors that provide measured context. They are made up of textual values and hierarchies and can be flat or snowflake-like.
This can be shown as (ND,PD, HD)
ND is the dimension name.
PD = {p1,......,pi} a set of attributes called parameters.
HD = {h1,....,hj} a set of hierarchies.

The example demonstrates that Line Order is a fact having dimensions such as Customer, Product, and Order Date and measurements such as Quantity, Tax, and Discount. When aggregating data, hierarchies are used to create a logical tree-like structure where each member has a single parent and several children.

B. Key-value store

The mathematical notation for a key-value NoSQL database model is introduced in this section. KV stands for a collection of ordered pairs (K V) made up of distinct keys and associated values.

$$KV \subseteq K \times V = \{(k,v) \mid k \in K \wedge v \in \wedge \ \forall\ (q,w) \in K \times V : k = q \Rightarrow v = w\}$$

The key in a key-value NoSQL database must be distinct, and the structure and data type of the value are typically unrestricted. Any type of content is acceptable, including text, JSON documents, and even embedded key-value pairs.
The notation formalization uses are as follows:

"{}" denotes a key-value pair and a nested record.
"⇒" symbol maps a key to its associated value.
"[]" denotes a value.

An illustration of its formalization is given by an example. The schema for a "Person" key-value pair is shown, along with a nested record called "Address" that contains information on the person's name, city, address, and country.

To create a huge data mart using NoSQL key-value stores, future talks and changes will be built upon this formalization of the key-value store.

II. Approach Review

The key-value NoSQL paradigm is used to suggest three methods for instantiating a huge data mart: FLA, HLA, and EHLA. These methods offer a table structure akin to SQL on top of key-value storage. FLA optimizes queries at the expense of storage size by storing fact and dimensions in a single table. Facts and dimensions are separated by HLA into different tables. By expanding dimension tables to indicate hierarchical levels, EHLA expands HLA. With Oracle NoSQL Database enabling joins inside the same hierarchy, NoSQL join support is dependent on the DBMS.

III. Transformation Rules

A data logical model can set up a big data warehouse using a NoSQL key-value database. The models are named as FLM, SHLM, and SnHLM, which aligns with full data denormalization, star and snow normalization in ROLAP techniques. To simplify the modeling process, the multidimensional schema is restricted to a fact table.

LineOrder = { lineOrderKey ⇒ [quantity,status,discount] }
and one dimension table:
Supplier={lineOrderKey,supplierKey⇒
[name,address,nationName]}

The three techniques (FLM, SHLM, and SnHLM) and the accompanying transformation patterns based on the chosen formalization are described as follow,

A. FLM (Flat Logical Model)

Consists of a single two-dimensional array of data elements, key-value data table defined by $T(Id^T ,Att^T ,R^T)$.

- Id^T is the Fact key.
- Att^T is a foreign key referencing a dimension key or a measure.
- R^T Set of nested records.

B. SHLM (Star Hierarchical Logical Model)

The SHLM is created by connecting the elements of the multidimensional schema through parent/child relationships. The transformation process from the multidimensional schema to SHLM is as follow,

The fact is mapped to a key-value pair. A dimension is mapped to a child key-value pair.

C. SnHLM (Snow Hierarchical Logical Model)

A tree-like structure is included in the model, which is an extension of the star hierarchical logical model and connects several dimensions to one another. Every table in this model has a single parent, and in each parent fact table's dimension child tables are duplicated; therefore, data can be flexible and well-organized.

IV. Implementation

A. Experimental Overview

To evaluate query performance and storage consumption in a NoSQL key-value database, two experiments were carried out. The first experiment examined response time and read request latency for various dimensions. Second experiment to measure the storage space per logical model using scale factor. These tests revealed information about the system's query performance and storage effectiveness.

To evaluate the models three analytical queries are used. These queries gradually include dimensions in their computation.

B. Oracle NoSQL database

Oracle NoSQL Databases is a key-value database that supports JSON, SQL-like table and key-value data types, as well as parent-child join and aggregation operations. Basic, Enterprise, and Community editions are all accessible.

C. Environment setting

Data Generation:
To assess the performance of data modeling, data is generated using a TCP-H benchmark.[3] Data is generated using KoalaBench, which has been modified for the meta data model. Data is generated by DBGen for the project, which is accessible on GitHub. An import function mentioned in the README file is used to load JSON files into the Oracle NoSQL Database. [4]

Software Settings:
Experiments uses Oracle NoSQL database in Docker engine with two setups: single node and expanded 3x1 cluster on Docker Swarm. Host machine: Intel Xeon w3530, 8 GB

Experiments and Results

Experiments:

In Experiment 1, an Oracle NoSQL database and two scale factors (sf=1 and sf=10) are used to measure query execution time in a three-node cluster. The query execution language is HiveQL. Below are the results.

FLM performs better since aggregations don't require any joins, whereas SnHLM performs marginally worse than SHLM because it requires more joins, especially when scale factors are bigger.

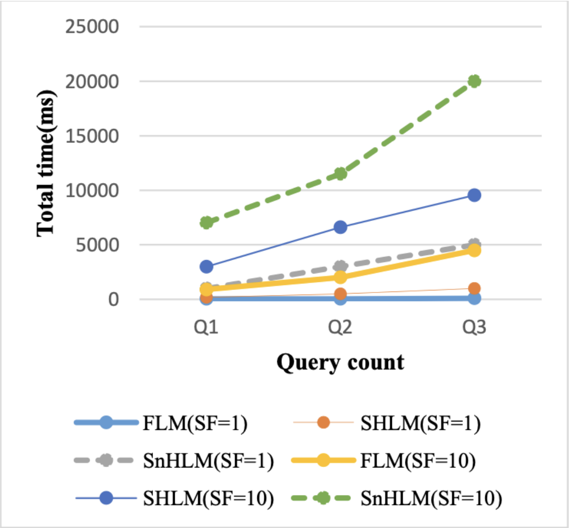

Fig.1. Query response time (milliseconds)

In Experiment 2, the Oracle NoSQL database's storage space allocation is examined. Due to data redundancy, SnHLM consumes three times as much disk space as SHLM and FLM. SnHLM requires 44.6 GB of storage space, SHLM 15.3 GB, and FLM 14.88 GB for sf=10 (15 million records). FLM provides effective space utilization, however its single table structure necessitates upkeep.

Conclusion

This report explores the integration of a big data mart with a NoSQL database. According to storage space and query performance, three logical models which are FLM, SHLM, and SnHLM were suggested and evaluated. FLM displayed exceptional performance. Comparative research and database conversions from relational to key-value are among the future challenges.

References

[1] R. Kimball, “Kimball Dimensional Modeling Techniques,” pp. 1–24, 2013, doi: 10.1016/B978-0- 12-411461-6.00009-5.

[2] TPC, “No Title,” Trans. Perform. Councli, “TPC Benchmarks” [Online], 2015.

[3] M. Chevalier, M. El Malki, A. Kopliku, O. Teste, and R. Tournier, “Un benchmark enrichi pour l’évaluation des entrepôts de données noSQL volumineuses et variables,” Eda, 2017.

[4] A. KHALIL and M. BELAISSAOUI, “Benchmark generator for big data trend,” [Online]. Available: https://github.com/khalilabdelhaq/BigData.git