St. Petersburg State University
Mathematics and Mechanics Faculty
Department of Analytical Information Systems

Experimental Comparison of DBMS Performance

Kochkar Roman, 341
Scientific Adviser: Boris Novikov

St. Petersburg — 2015

Abstract

Proponents of nosql database systems claim significant performance advantages of these systems. However, these claims are rarely, if ever, supported by experimental or theoretical studies.

We present an experimental comparison of alternative implementations on top of traditional SQL and nosql systems. We demonstrate that lack of high-level querying capabilities of nosql systems may cause significant penalties in both development and run-time.

This paper contains 4 pages.

1 Introduction

New classes of database systems are gaining increasing pupularity during last decade. Several completely different approaches and architectures are united under *nosql movement* umprella.The use of word 'movement' itself suggests that the main reasons are political and economical, rether than technical.

Although the proponets of the nosql system claim significant performance benefits, these claims are rarely, if ever, supported by deep analysis or experimental studies and actually many technical aspects seem to be controversial. An emotionally rich keynote talk of C. Mohan [6] demonstrate several deficiencies of noslq systems.

Some studies show that expectations for performance of nosql systems are not satisfied. For example, the chart at [**?**] demonstrates that relational system may outperform nosql system even for an application in target domain of the specific nosql system.

This research provides an experimental comparative performance study of an OLTP-style application implemented on top of relational and nosql systems. The basic assumption is that complexity of the application does not depend on the underlying system. Consequently, a simplicity (or lack) of querying facilities in nosql systems results in more complex application code. Specifically, a complex SQL query that provides all data needed for a certain business function must be replaced with application code containing several nosql databse invocations. In the circumstances described above, a fair comparison must be done at the business function level, rather than just a databse level. Moreover, the implimentation of such business function should be done in the best possible way for the specific system. For example, replacement of a complex and highly efficient SQL query with a series of simple queries equivalet to nosql databsae invocation is considered unfair.

The rest of the paper is organized as follows.

The precise description of the problem addressed in the paper is presented in section 2. Section 3 describes experiments and provides the analysis of findings. Section 4 describes related work. Conclusion provides a summary of results and outlines the future work.

2 The Problem and Approach

The main problem stated in this paper is to compare one representative of SQL family (PostgreSQL [2]) with one representative of NoSQL family (Cassandra [1]) with the intention to make this comparison more fair.

Fair comparison means measuring time from the first query request to the last data response from DBMS. In such a comparison we measure the time when the whole amount of work for retrieving data is done. Also fair comparison would include more than one query for data model, common in real world. Note, that if the query requires several requests, measured time would include time of in-application data processing between API calls. The reason for that is that all functions that are not implemented in DBMS are escalated to the application level, if the data model requires them.

For comparison it was decided to choose a data model of an internet book store. This data model is organized next way. There are book and author entities. One book can have multiple authors. A book with publishing house form publishing, that also has cost. There is a client entity that consists of first and last name. Client orders publishings. One client order can have multiple publishings. Order also has two dates: date of receiving is when order was first received from the client and date of fulfilling is when the last publishing was sent to the client. Publications, after they are ordered, then packaged and sent to the client. One package can contain multiple publications and even publications from different orders. Package also has date of sending.

3 Experiments

3.1 Environment

All experiments are made on one server on virtual machine. Guest OS is Ubuntu 14.04. There are 4GB of RAM and 2 processors dedicated for VM. Version of PostgreSQL is 9.3, version of Cassandra is 3.0.1.

There are about 40000 books, 40000 clients, 600000 orders and 2.5 million books ordered currently in generated dataset. All books were retrieved from website `http://www.goodreads.com/` from section 'Best Books Ever'. Orders are generated next way. Client has random number or books from 2 to 10 in one order with normal distribution. Also each client has random number of orders from 4 to 35 with normal distribution. All received dates of orders are randomly spread across 900 days after 01.01.2011 for each client.

All the measurements are made on next queries:

- Find all packages, containing publications from last 2 orders of client with name 'Damian Singh'
- Find all publications in last 4 orders of client with id 25312
- Find all authors of publications ordered by client with id 35900

The main motivation for using these queries is that they are applied to multiple entities and they also require some additional work besides retrieving objects from collections.

3.2 Results

All experiments are currently being made so this work is still in progress.

4 Related Work

Currently, there are a lot of NoSQL systems and most of them promise to solve the problem of horizontal scalability, that was faced with era of Web 2.0. Rick Cattell in his article [3] describes most common types of NoSQL with examples and also notes that nowadays there are relational systems that provide scalability comparable to NoSQL. In contrast to NoSQL these systems provide an SQL interface as well as ACID transactions but simply penalize a client for complex queries, that span many nodes.

Most of NoSQL systems differ in approaches they use to solve problems that emerge in distributed computations and partitioning. Indrawan-Santiago [4] provides a brief history of data stores and describes core characteristics of NoSQL systems that can be used as base for comparison of such systems. She also compares several NoSQL databases using these characteristics.

Mostly, NoSQL systems are quite young and were started from scratch. So they lack some core functionality that is provided by mature systems. Mohan [6] analyses several common weaknesses of NoSQL systems basing on his own experience with systems like System R, Lotus Notes and so on. He also mentions that all the functionality that is not implemented inside database system lays on sholders of application developers and becomes their responsibility.

Yishan Li and Manoharan S. [5] compare SQL Server with several representatives of NoSQL family basing on time of next operations: instantiate collection, read, write, delete. Their experiments shows that SQL Server, used as key-value store, outperforms some NoSQL databases.

5 Future Work

For the current moment all data is already generated and imported into PostgreSQL database. Cassandra is installed on VM.

To finish this work next steps are required:

1. Finish experiments in PostgreSQL
2. Create schema for Cassandra
3. Export all data from PostgreSQL and import it into Cassandra

4. Do experiments in Cassandra

5. Summarize results

References

[1] Cassandra home page. `http://cassandra.apache.org/`.

[2] Postgresql home page. `http://www.postgresql.org/`.

[3] Rick Cattell. Scalable sql and nosql data stores. *SIGMOD Rec.*, 39(4):12–27, May 2011.

[4] Maria Indrawan-Santiago. Database research: Are we at a crossroad? reflection on nosql. In *15th International Conference on Network-Based Information Systems, NBiS 2012, Melbourne, Australia, September 26-28, 2012*, pages 45–51, 2012.

[5] Yishan Li and S Manoharan. A performance comparison of sql and nosql databases. In *2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, pages 15–19. IEEE, 2013.

[6] C. Mohan. History repeats itself: sensible and nonsensql aspects of the nosql hoopla. In Giovanna Guerrini and Norman W. Paton, editors, *Joint 2013 EDBT/ICDT Conferences, EDBT '13 Proceedings, Genoa, Italy, March 18-22, 2013*, pages 11–16. ACM, 2013.