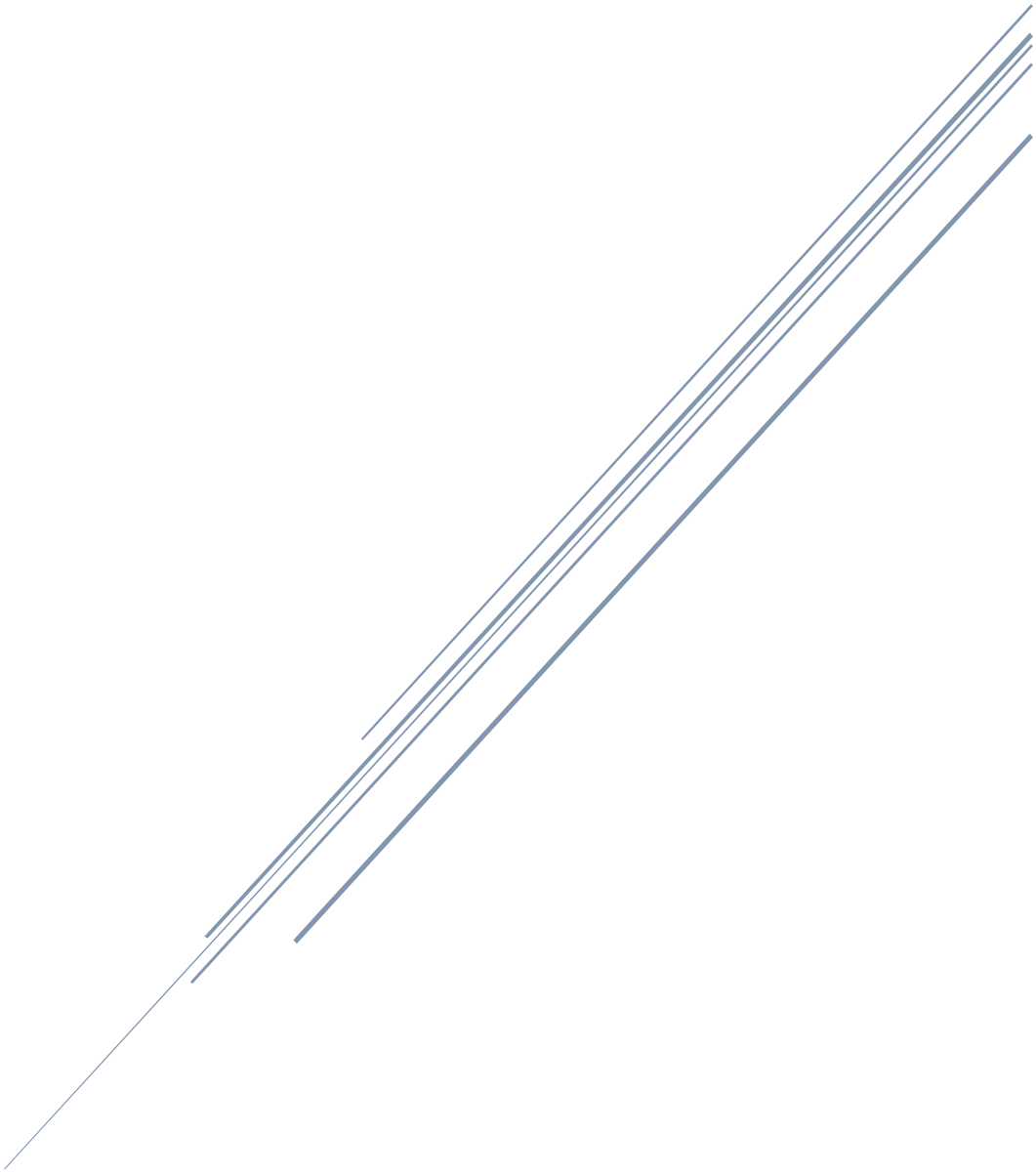
SQL/NOSQL REPORT

KS489



CO7517

Contents

Introduction

We need to do a thorough analysis to select the most suitable data store type for this blogging application along with the data store technology to be used. We figure out if the data store chosen will be suitable for current and future development and maintenance, tooling support and performance.

SQL vs NoSQL

Due to the nature of the basic project requirements, we would need a data source that could –

- Record large amounts of log data very frequently
- Have a data store that has good write performance over read performance
- Be able to increase data persistence over time (since we are recording lots and lots of data over time)

SQL was ruled out for the following reasons:

- Does not scale horizontally well enough and efficiently compared to other NoSQL databases
- Write performance isn't as good as other NoSQL database

We have opted to use a Column Wide or better known as Column Family datastore. It is part of the NoSQL aggregate database. It was selected since it addressed all the basic requirements of the project and it is suited for Content Management Systems and Blogging Systems. The Column Family will also allow for future project developments listed under – "Possible Future Project Requirements" below which lists extra information recording etc.…

Project Benefits

After doing an initial database feasibility assessment we have chosen Cassandra as our Column Family datastore to handle our blogging information. The reasons for choosing Cassandra are as follows –

- It is an open source project which keeps to our technology stack theme of being open source.
- The majority of the BlogPost project was made using Java and Cassandra was also made using java. It keeps to our JVM technology stack i.e. Gradle, Groovy, xtext, xtend, Java etc…
- The project is backed from Apache which steers and guides the project development nicely. The project is also backed by 192 contributors and the project gets regular updates.
- It is listed as the most popular Column Wide database and the 7th most popular database according to DB-Engines - http://db-engines.com/en/ranking
- There is a very good community and documentation. There is also lots of support on Stackoverflow there are plenty of forums to help developers.
- There is fault tolerant facilities including automatically replicating nodes or support for failed nodes.
- It scales very well with known figures of over 75 000 nodes with over 10 PetaBytes of data.
- Supports cloud facilities that we can use and take advantage of the continues delivery. We can host our data store on the cloud which comes with analytics.
- Can record extra information in different ways that won't affect existing data and functionality.
- It can be interfaced with many programming and scripting languages.

Possible Future Project Requirements that Cassandra will be able to adopt –

- Since we have recorded large amounts of log data we might want to use that data for some kind of analytics
- Record additional event logging information such as application state or to log errors etc…
- Record extra blogging information such as –
    - Blog / Article categories
    - Blog post versioning
    - Meta data tagging on blog posts i.e. HashTags used in Twitter
    - Comments per blog post
- Expiring Data – Delete or remove old data that is not relevant to the project.

Conclusion

There were other aggregate type data stores out there that could potentially be used with this application but we opted to use Cassandra as this suited our project requirements and future project requirements more. Cassandra is a mature project lead by a globally recognised company with lots of tooling and support. It boosted the best performance in a recent study conducted by apache themselves.