

LK2222134_ACCA7004_JointAssi gnment

by Liam Kelly

Submission date: 05-May-2023 02:23PM (UTC+0100)

Submission ID: 204932733

File name: 42197_Liam_Kelly_LK2222134_ACCA7004_JointAssignment_488469_431261952.docx (452.12K)

Word count: 1226

Character count: 7256

ACCA7004 - Web Development & Web Services Joint Assignment

Liam Kelly – 2222134 – 2222134@uwtsd.student.ac.uk

Table of Contents

Liam Kelly – 2222134 – 2222134@uwtsd.student.ac.uk.....	1
Introduction.....	3
The Principles of Big Data.....	3
The principles of NoSQL.....	4
Document-oriented databases.....	4
Conclusion	6
Works Cited	7

Video link:

<https://youtu.be/iaGWXrfqbh4>

Link to GitHub Repo:

https://github.com/Lez-Dev/RESTful_PPP

Introduction

Over the last two decades the volume of data generated by a variety of sources has increased at a rapid rate, this explosion of data, or 'Big Data', has become characteristic of technology of the 21st Century [1]. Traditional data management models such as relational-databases which serviced us in a world where data storage was expensive and data captured was uniform and structured are now often insufficient to handle the demands of Big Data [2] (Fig.1). Other technologies such as Not only Structured Query Language (NoSQL) and NewSQL are filling this unmet need [3]. In this report we will explore the principles of Big Data and NoSQL databases.

The Principles of Big Data

Big Data refers to large volume, varied data from different sources captured at a rapid rate and is defined under the (original) three V's; Volume, Velocity, Variety [1]. The rise of increased accessibility to personal devices and the advent of the Internet of Things (IoT) there is now vastly more data sources, with data volume estimate increases of 40% year on year [4]. This poses new challenges for web applications. Relational databases are rigid in their structure, with defined schema in advance often being a pre-requisite and only offering vertical scalability [4]. This lends to issues of affordability regarding an application as well as difficulty in being able to manage large swathes of potentially unrelated and unstructured data. In this context NoSQL databases are a valuable tool for the management of the data makeup of Web 2.0 and beyond, a world where data embodies the extreme aspects of the three V's [4].

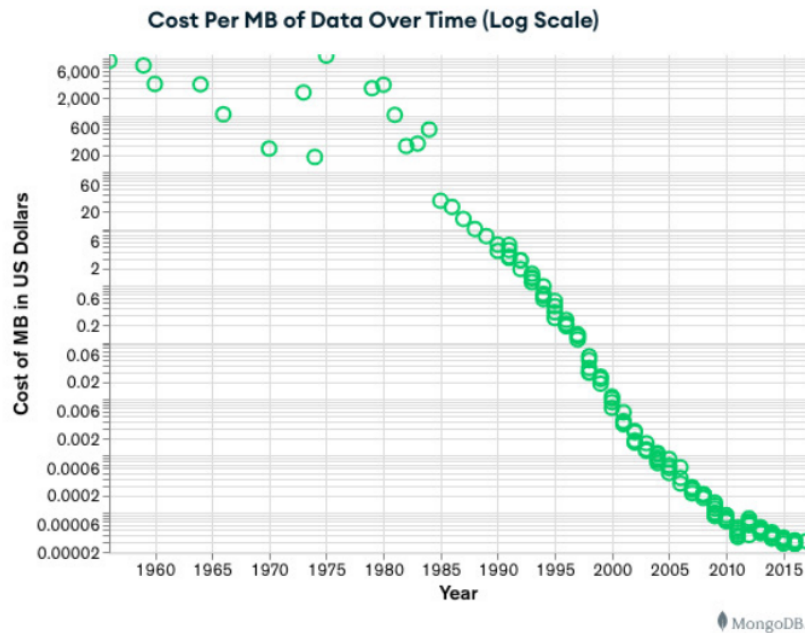


Figure 1: Graph depicting the changing costs in a Megabyte (MB) of storage since 1950. [2]

The principles of NoSQL

NoSQL databases are distributed by design, expressing sharding to organise clusters, lending to horizontal scaling, flexibility and fast processing [3]. NoSQL also hosts schema-less data storage permitting analysis and access to large collections of structured and unstructured data. With this, NoSQL provides the availability of four forms of database structure; document, key-value, column-oriented and graph [5]. Each type has its own use-case, benefits and shortcomings, but this report will focus explicitly on document databases.

Document-oriented databases

Document database storage is formatted in collections using JavaScript Object Notation (JSON)-like documents [6]. This document structure allows for a generalised and flexible data structure store along with the ability to query the data unlike other NoSQL solutions (*Fig.2*). This is facilitated by the indexing of documents as object entities, within which, can be other nested objects – creating a rich and detailed storage solution with a hierarchical structure [6].

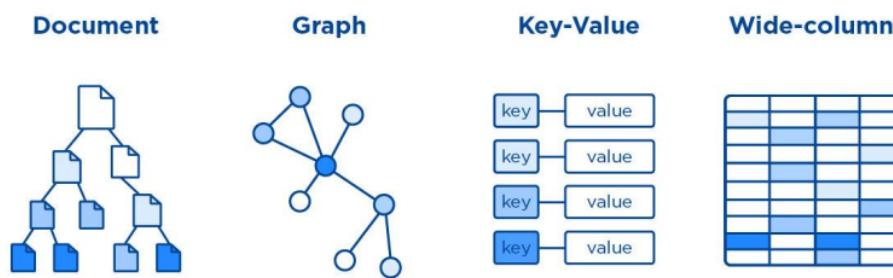


Figure 2: Graphical representation of the four types of NoSQL database solutions [7].

Servers now are required to process large numbers of concurrent users compared to those of the past [3]. As a result, NoSQL and in particular document stores lend to this constant influx of varied data particularly in web applications [8]. NoSQL and document stores specifically address the issue of unstructured data readily. From a commercial perspective it would also allow an application to adapt and change in response to the business needs of the period. Rather than having to explicitly define relational database structure in preparation, document stores can allow for a more readily available storage application, as well as a more dynamically responsive and future scalable solution should the needs of the application change [9]. Take the scenario of the application of this project. If operated at a full capacity it could be required to process and store over 50 million individual registrations [10], the transaction, and customer data associated with them (*Fig.3*). Both SQL and NoSQL databasing would be capable of this of course but the technical overhead for SQL would be much greater compared to that of NoSQL [11]. This information can be managed through document stores with nested data all the while maintaining a query-enabled and indexed store [2].

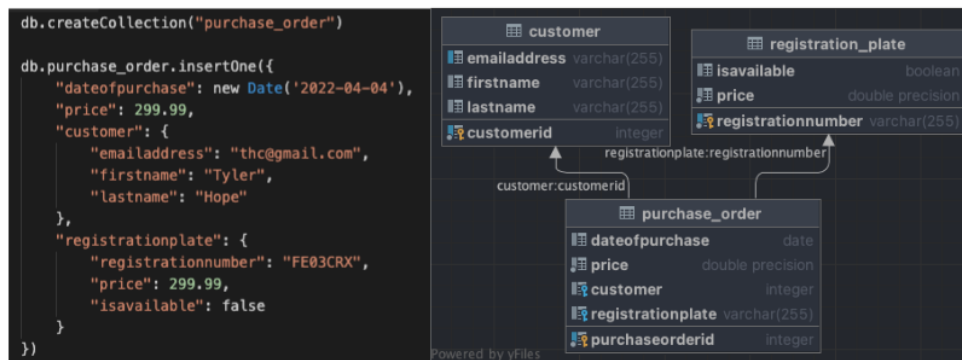


Figure 3: Demonstration of a document store (left) where purchase order, customer information and product information can all be stored in the same document object. This in comparison to the relational schema required for the same information in a relational-database (right).

Not only this, should the application expand in the future or the legal requirements of numberplates change a document store would be better poised to adapt its data structure to meet those requirements. Another scenario of application expansion could be in real-time analytics. Real-time analytics are used in e-commerce applications to analyse customer behaviour, purchasing trends and the provision of personalised recommendations. These rapid response times are possible due to the performance of NoSQL databases like document stores [11].

A	Atomicity – changes made are completed in full an ‘all or nothing’ concept. It also gives the ability for rollbacks and committing transactions.
C	Consistency – returns the database to its consistent state after processing the inconsistency of a transaction.
I	Isolation – transactions are isolated, restricted, and guided by a set of pre-determined rules of how data interacts with other data. This is achieved for example, by locking data and row versioning.
D	Durability – transactions are committed the data log before they are written to the data file. This prevents transactions reverting or changing state unexpectedly.

Figure 4: Author-generated graphic of the ACID principles [6].

Where document stores fall short however, is with in regard to transactions. Relational-databases are ACID compliant (Fig.4) [6]. This guarantees data consistency which bears particular importance in data stores which process financial transactions or for example, an application that manages the individual ownership and purchasing of vehicle registrations. In response, document stores do support BASE transactions [6] (Fig.5) and some providers such as MongoDB now claim that their software supports ACID transactions [12].

B	Basically Available – a guarantee to the availability of data
A	
S	Soft state – ability for the system to change over time
E	Eventual consistency – the system eventually is consistent when it stops receiving input

Figure 5: Author-generated graphic of the BASE principles [6].

This pivot, even if momentarily, away from ACID transactions is to address the processing power required to manage Big Data which historically has promoted performance and data availability over consistency [3].

Conclusion

In conclusion, the rise of Big Data in web applications poses new challenges for both SQL and NoSQL, each of which have their own use cases for application data. This report has endeavoured to briefly highlight where the principles of NoSQL and Big Data might play a part in an application such as the one developed in the other arm of this project.

Works Cited

- [1] Oracle, "What is Big Data?," Oracle, 2022. [Online]. Available: <https://www.oracle.com/uk/big-data/what-is-big-data/#:~:text=The%20definition%20of%20big%20data,especially%20from%20new%20data%20sources..> [Accessed 26 April 2023].
- [2] MongoDB, "What is NoSQL?," MongoDB, 2023. [Online]. Available: <https://www.mongodb.com/nosql-explained>. [Accessed 19 April 2023].
- [3] A. Pavlo and M. Aslett, "What's Really New with NewSQL?," *SIGMOD Record*, vol. 45, no. 2, pp. 45-55, 2016.
- [4] J. Bhogal and I. Choksi, "Handling Big Data using NoSQL," in *International Conference on Advanced Information Networking and Applications Workshops*, Washington D.C., 2015.
- [5] MongoDB, "Understanding the Different Types of NoSQL Databases," MongoDB, 2023. [Online]. Available: <https://www.mongodb.com/scale/types-of-nosql-databases>. [Accessed 20 April 2023].
- [6] A. Kelly and D. McGreary, *Making Sense of NoSQL*, Shelter Island, NY: Manning Publications, 2014.
- [7] abcloudz, "NoSQL Technologies," abcloudz, 2022. [Online]. Available: <https://abcloudz.com/services/nosql-databases/>. [Accessed 27 April 2023].
- [8] MongoDB, "What is a Document Database?," MongoDB, 2023. [Online]. Available: <https://www.mongodb.com/document-databases>. [Accessed 26 April 2023].
- [9] A. Ali, J. Qadir and R. Rasool, "Big data for development: applications and techniques," *Big Data Analytics*, vol. 1, no. 2, 2016.
- [10] Driving & Vehicle licensing Agency, "About DVLA Personalised Registrations," Driving & Vehicle licensing Agency, 2023. [Online]. Available: <https://dvlaregistrations.dvla.gov.uk/about/>. [Accessed 27 April 2023].
- [11] R. Čerešňák and M. Kvet, "Comparison of query performance in relational and non-relation databases," in *13th International Scientific Conference on Sustainable, Modern and Safe Transport*, High Tatras, Novy Smokovec, 2019.
- [12] MongoDB, "MongoDB Multi-Document ACID Transactions," MongoDB, 2020.

FINAL GRADE

72/100

GENERAL COMMENTS

Instructor

Implementation

REST (30): 20

AJAX (20): 18

Code/Doc (10): 6

Features (20): 14

Report (20): 14

Feedback:

=====

Should have described the 4Vs in more detail. Wanted a little more detail about a specific NoSQL database and scenarios for using it. In general, this is a good overview.

The application demonstrates the use of AJAX and react in a very nice way. I would have liked to have seen you expand upon the concepts such as using a new library. But there are a number of nice features in the UI.

