

A Proposed NoSQL Solution For The Welsh Care Record Service

Stephen Protheroe-Hewitt (2112883)

1. Introduction

The Welsh Care Record Service (WCRS) collects electronic patient documents and stores them within a central data repository, supporting the storage and retrieval of patient documents and standardising documentation [1]

While there are advantages, there is minimal scalability and fixed schemas within SQL relational databases, reducing flexibility [2].

This study analyses and evaluates different document repository solutions to work in conjunction with SQL Server, highlighting the chosen option and the reasoning behind this, especially as NoSQL is becoming increasingly used.

2. Aim and Objectives

The aim is to analyse and evaluate document repository solutions and creating a model.

The objectives are:

- To critically analyse and evaluate current literature
- To identify, design and implement a solution
- To compare results between current and implemented solution

3. Literature Review

The literature review explored the different NoSQL databases available to understand the different NoSQL database types available and what to utilise for WCRS, concluding the use of MongoDB for the NoSQL database due to better overall performance, with Azure Blob Storage used due to existing infrastructure within Digital Health and Care Wales (DHCW).

The best solution would be a hybrid architecture combining MongoDB’s metadata, indexing and search functionalities for quick document retrieval and querying, and blob storage for efficient, secure and scalable storage of large document files, to work alongside the current SQL Server.

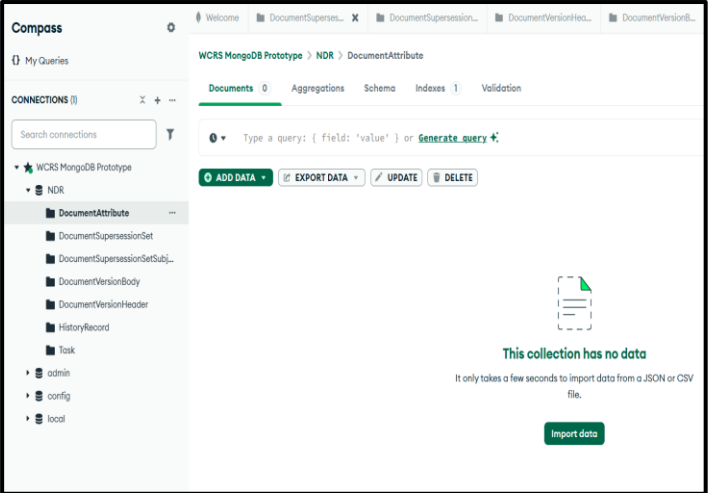
4. Design

Azure was used due to existing infrastructure within Digital Health and Care Wales (DHCW), with a MongoDB container and Azure Blob Storage deployed to create a MongoDB prototype database, allowing for seamless integration if developed further.

Limitations outside of project scope included security settings and system integration within DHCW, which could form part of a future development if this prototype were to progress into a production environment.

5. Implementation

The Azure Blob Storage was created to allow the import and export of data within MongoDB, while a WCRS prototype database was created on MongoDB Compass to mimic the current WCRS production version where possible, providing a framework for future development.



Showing all 14 items						
	Name	Last modified	Access tier	Blob type	Size	Linear state
☐	DocumentAttribute (Bowl - 10000).json	11/07/2025, 16:49:52	Not (Indexed)	Block blob	1.67 MB	Available
☐	DocumentAttribute (Bowl - 10000).xlsx	10/07/2025, 13:58:12	Not (Indexed)	Block blob	508.15 KB	Available
☐	DocumentSupersessionSet (Bowl).json	11/07/2025, 16:49:53	Not (Indexed)	Block blob	1.53 MB	Available
☐	DocumentSupersessionSet (Bowl).xlsx	10/07/2025, 13:58:12	Not (Indexed)	Block blob	465.8 KB	Available
☐	DocumentSupersessionSet Subject Identifier (Bowl).json	11/07/2025, 16:49:53	Not (Indexed)	Block blob	1.28 MB	Available
☐	DocumentSupersessionSet Subject Identifier (Bowl).xlsx	10/07/2025, 13:58:12	Not (Indexed)	Block blob	495.83 KB	Available
☐	DocumentVersionBody (Bowl - 10000).json	11/07/2025, 16:49:57	Not (Indexed)	Block blob	3.83 MB	Available
☐	DocumentVersionBody (Bowl - 10000).xlsx	11/07/2025, 13:28:42	Not (Indexed)	Block blob	816.87 KB	Available
☐	DocumentVersionHeader (Bowl).json	11/07/2025, 16:49:54	Not (Indexed)	Block blob	1.45 MB	Available

8. References

- [1] R. Watson, *WCRS Technical Specification*, Cardiff: Digital Health and Care Wales, 2017.
- [2] MongoDB, “Understanding SQL vs NoSQL Databases”, *MongoDB* [Online] Available: <https://www.mongodb.com/resources/basics/databases/nosql-explained/nosql-vs-sql> [Accessed Apr. 14, 2025].

6. Results

The results show that MongoDB outperforms SQL Server in the majority of read and write operations, proving that a NoSQL database could provide a viable alternative to using SQL Server only. However, despite MongoDB outperforming SQL Server for write operations, there was still some considerable times when these operations were executed,

Table	Retrieve all data		Specific Criteria		Sorts		Limit Results	
	SQL Server	MongoDB	SQL Server	MongoDB	SQL Server	MongoDB	SQL Server	MongoDB
Document Attribute	171	3	10734	3	76	9	67	1
Document Supersession Set	122	2	6	4	636	10	2	1
Document Supersession Set Subject Identifier	103	3	1	3	11	7	3	1
Document Version Body	1078	3	122	4	19473	10	984	1
Document Version Header	233	0	232	1	664	2	655	0
History Record	186	3	341	3	909	14	6	1
Task	137	3	32	4	89	9	5	1
Average Total	290	2	1638	3	3123	9	245	1

Table	Insertions		Updates		Deletions	
	SQL Server	MongoDB	SQL Server	MongoDB	SQL Server	MongoDB
Document Attribute	2000	538	37333	314	26333	2093
Document Supersession Set	833	490	800	198	26667	2079
Document Supersession Set Subject Identifier	833	500	4333	224	4333	8476
Document Version Body	5667	410	1300	324	138000	2135
Document Version Header	667	327	1600	182	231333	2092
History Record	1000	927	400	242	2667	2990
Task	833	1010	1667	232	2000	2108
Average Total	1690	600	6776	245	61619	3139

Table	Average Row Size	
	SQL Server	MongoDB
Document Attribute		174
Document Supersession Set		85
Document Supersession Set Subject Identifier		105
Document Version Body		693
Document Version Header		478
History Record		259
Task		309
Average Total		300

This study compared different read and write operations between MongoDB and SQL Server, with MongoDB providing better performance in most tests. While a basic prototype and specific to DHCW, this can be developed upon to work in conjunction with the current WCRS infrastructure to improve processes and data flows between systems and services in DHCW and NHS Wales. This includes the automation of data flows to and from MongoDB, converting the data as required to ensure both databases can read and map data items correctly with no issues.

Further investigation is needed into implementing security processes, frameworks and procedures into Azure Blob storage and MongoDB to ensure data is protected between databases and minimising the risk of attacks against MongoDB, mitigating any data breaches. Research could expand on the different database vulnerabilities highlighted in the literature review and create processes that protect both the databases and the data transfers between them.