
Database Report

Tudor Haba
2420645

Table of Contents

<i>Abstract</i>.....	3
<i>SQL vs NoSQL Database</i>	3
<i>SQL</i>	3
<i>NoSQL</i>.....	4
<i>Indexes</i>.....	5
<i>Transaction</i>	5
<i>References</i>	7
<i>Appendix</i>	8

Abstract

In this report I discussed what an SQL database, what a NoSQL database is and I bring forward some of their similarities and differences and what the best use case scenarios are. I also discuss what indexing is and I identify some secondary indexes that could be used by Eco Bike Hub to improve the performance of their queries. Finally, I talk about my implementation of the transactions I wrote for Eco Bike Hub showcasing my understanding of what they meant and how I solved them.

SQL vs NoSQL Database

SQL

SQL is a language used to manipulate and access data that is stored on a database system [1]. An SQL database system is made up of tables which are used to store the data. Each table has a unique primary key to identity each record stored within the database [2]. Before using an SQL database, we need to define a database schema that will underline the definitions of the tables along with the relations all the tables have between them. Along with the schema we also need to perform normalization of the database to ensure that no duplicated data is found across the database to comply with the ACID properties [2]. This in turn takes time as we need to ensure that all of the tables are independent from each other, but they can still communicate with each other by correctly identifying the linking data, foreign keys, between them. Depending on the level of normalization required for the database it can be very time consuming to define a schema.

A SQL database has to follow some principles in order for the data to stay reliable and consistent throughout the system. These principles are called ACID. ACID stands for atomicity , consistency, isolation and durability [3]. Starting with atomicity, it ensures that either all the rules in a transaction pass in one go or fails completely. This makes sure the database is left in a state with no errors and can continue operation normally. A transaction is made up of multiple read and write commands to access and manipulate the data [3]. Consistency refers to leaving the database in a valid state from another valid state after a transaction is completed. Isolation prevents data conflicts to happen by separating each transaction from one another. Finally, durability refers to leaving the data permanently committed even after a system failure [3].

SQL databases offer many benefits and one of those benefits comes from the structured element of the database. Due to its rigidity and schema it's very good at handling complex queries on relational data [4]. SQL databases are vertically scalable, meaning you can easily add more resources to a server to meet the growing demand of the database, such as more RAM, better CPU [5].

However, SQL databases are somewhat less scalable when taking in account the cost and working with high volume of data can make it expensive as a powerful server is needed for all the querying necessary [1]. They have a very rigid schema we have to stick to which in turn makes them less flexible to storing a wide variety of data if it's not been defined from the beginning [1][4]. While SQL databases have a very strong data consistency throughout it gets slower as the database grows and turns into a large scale database [4].

NoSQL

NoSQL means 'not only SQL' and is a database management system that does not rely on the traditional model of relational database model system (RDBMS) unlike a SQL database [2][1]. NoSQL is used when large amounts of data need to be stored. The NoSQL database is better at handling structured and semi structured data compared to the traditional model, meaning that the data can be of various types [4][6]. The best example of this being social media generated data. Depending on the database type the data is stored in key values pairs, graphs, documents or columns. NoSQL does not have a schema that needs to be thought of before unlike a SQL database and this is what allows the storage of various data types as it can be dynamic [4][2]. NoSQL does not satisfy all of the ACID properties [2]. To allow this level of flexibility we need a way to code it and because of this we use different languages for the documents to be stored in a NoSQL database. They could vary from JSON, XML, BSON, YAML [2].

NoSQL priorities flexibility and scalability of data such as storing big data for example[4]. A NoSQL database is able to scale without having to add more physical components to the server as opposed to a traditional SQL database approach [5]. Due to the fact that no schemas are used NoSQL allows for a greater flexibility and a potential to have a dynamic schema in order to store multiple types of data in the same database [4]. Since NoSQL scales better horizontally, by fragmenting the data or adding more servers it is better suited for instances where large amounts of data needs to be stored as mention above an example of that being big data[5].

However, the majority of developers are not experienced using a NoSQL database [5]. The consistency of the data is lost at the hand of the high availability of the data and the performance gains as a NoSQL database will not support all the acid properties [6]. As there is no one language used for all of NoSQL databases switching between them might pose problems for people as a new language might have to be learnt to query the database thus taking time to adapt to the new database [6]. The lack of encryption of the files stored in a NoSQL database poses a security risk that's higher than a normal SQL database [6].

Indexes

As more and more data gets added to a database, its performance gets affected. This also affects the response time of complex queries that are performed as it has to manipulate and query a larger set of data than at the beginning[7][8]. Indexes are a mean to optimize the database to speed up the response time of those complex queries [7]. A secondary index is a way of accessing a record within a database by creating an index on a non-primary key in turn improving the performance of the database [9]. Instead of looking at the whole table if we are looking for a specific record we can retrieve data much faster by scanning the table for relevant data [10].

Looking at the current database of Eco Bikes Hub one potential secondary index could be the customer ID in the bicycle hire table. I believe there would be a need to query the data over and over again for all the bookings a customer has on the database, either by the staff needing the data or by customers wanting to know any of their future bookings. By having the customer ID as a secondary index, we can then directly use that for any lookups on the database related to a customer.

Another potential query that might get executed many times and it would only take longer as the database grows would be seeing all the repairs a mechanic has done. This could be happening because there might be a problem with one of the mechanics if customers turn around and their bicycles are not repaired correctly and we want to see exactly the repairs all the mechanics were assigned. Having a secondary index on the mechanic ID from repair booking table would benefit Eco Bikes Hub as they could look up every job a mechanic has ever done.

A complex query that might get performed by Eco Bikes Hub might be to check what bikes are booked for every store. To do this they would have to join multiple tables but if we were to create a secondary index on store ID from the bicycles table this would speed up the lookup of what bikes are in what store. Therefore by only looking up relevant bikes instead of the whole table of bikes we drastically reduce the time it takes to join the tables [10].

Transaction

I solved the issue of return dates not being registered on the system in case customer ask for them by creating a serializable transaction, so only one person at a time can access the record [11], that will set the date to the current date once the state of repair has changed. This means that when a member of staff will update the state of the repair from 'in repair' to 'repaired' a date will always be assigned to that record. Now if a customer was to ask for a date there would be no problem issuing them a return date.

The issues of customer being able to book bicycle times that are unavailable meant that customers were able to book a bicycle that was already booked prior to when the new

booking was created. I solved this problem by making a transaction that calls a procedure where I check that the requested date for the requested bicycle is available by looking at all the records for that bicycle and then if there are any I look at the dates for when that bicycle has been booked. I then check to see if the requested date falls between the dates the bicycle is already in use and if not then I allow the record to be created. This is also a serializable transaction that only let's one person make a booking for the bicycle at a time.

References

- [1] T. N. Khasawneh, M. H. AL-Sahlee, and A. A. Safia, 'SQL, NewSQL, and NOSQL Databases: A Comparative Survey', in *2020 11th International Conference on Information and Communication Systems (ICICS)*, Apr. 2020, pp. 013–021. doi: 10.1109/ICICS49469.2020.239513.
- [2] S. Abbas Fadhel and E. Ali Jameel, 'A COMPARISON BETWEEN NOSQL AND RDBMS: STORAGE AND RETRIEVAL', *MINAR Int. J. Appl. Sci. Technol.*, vol. 04, no. 03, pp. 172–184, Sep. 2022, doi: 10.47832/2717-8234.12.18.
- [3] 'ACID Properties In DBMS Explained', MongoDB. Accessed: Jan. 12, 2025. [Online]. Available: <https://www.mongodb.com/resources/basics/databases/acid-transactions>
- [4] B. Ana-Gabriela, 'CHOOSING THE APPROPRIATE DATABASE TYPE: A COMPARISON OF SQL, NOSQL, AND NEWSQL', no. 6.
- [5] T. Capris, P. Melo, N. M. Garcia, I. M. Pires, and E. Zdravevski, 'Comparison of SQL and NoSQL databases with different workloads: MongoDB vs MySQL evaluation', in *2022 International Conference on Data Analytics for Business and Industry (ICDABI)*, Oct. 2022, pp. 214–218. doi: 10.1109/ICDABI56818.2022.10041513.
- [6] M. A. Hassan, 'Relational and NoSQL Databases: The Appropriate Database Model Choice', in *2021 22nd International Arab Conference on Information Technology (ACIT)*, Dec. 2021, pp. 1–6. doi: 10.1109/ACIT53391.2021.9677042.
- [7] A. MEHEDINTU, C. PÎRVU, and C. ETEGAN, 'INDEXING STRATEGIES FOR OPTIMIZING QUERIES ON MYSQL.', *Ann. Univ. Petrosani Econ.*, vol. 10, no. 4, pp. 201–214, Sep. 2010.
- [8] A. Sar *et al.*, 'A Comparative Study and Analysis of Different Optimized Indexing Algorithms in Database Management Systems', in *2024 OPJU International Technology Conference (OTCON) on Smart Computing for Innovation and Advancement in Industry 4.0*, Jun. 2024, pp. 1–7. doi: 10.1109/OTCON60325.2024.10688336.
- [9] 'Secondary indexes'. Accessed: Jan. 15, 2025. [Online]. Available: https://docs.oracle.com/cd/E17275_01/html/programmer_reference/am_second.html
- [10] Atlassian, 'Indexing Essentials in SQL', Atlassian. Accessed: Jan. 14, 2025. [Online]. Available: <https://www.atlassian.com/data/sql/how-indexing-works>
- [11] 'SET TRANSACTION', PostgreSQL Documentation. Accessed: Jan. 11, 2025. [Online]. Available: <https://www.postgresql.org/docs/17/sql-set-transaction.html>

Appendix

Additional materials I used to do the transactions and to generate mock data for the database:

- <https://www.debart.com/dbforge/postgresql/studio/compare-dates.html#:~:text=You%20can%20use%20the%20BETWEEN,'31%2DJan%2D2020'%3B>
- <https://www.postgresql.org/docs/current/sql-createprocedure.html>
- <https://www.postgresql.org/docs/current/transaction-iso.html>
- <https://stackoverflow.com/questions/72179431/create-a-generic-procedure-which-inserts-data-into-any-table>
- <https://www.youtube.com/watch?v=WixlblbvZlo>
- <https://neon.tech/postgresql/postgresql-tutorial/postgresql-transaction>
- I used ChatGPT to generate the fake names of people and bicycles

ERD diagram of the database:

