Project Synopsis

on

"NoSQL to RDBMS Converter"

Submitted as a part of course curriculum for

Bachelor of Technology

in

Computer Science

Submitted by

Saumya Singh(2000290120138)

Vageesha Rai(2000290120181)

Vikas Kumar Verma(2000290120189)

Under the Supervision of

Prof. Abhishek Goyal

KIET Group of Institutions, Ghaziabad

Department of Computer Science

Dr. A.P.J. Abdul Kalam Technical University 2023-2024

DECLARATION

We hereby declare that this submission is our work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text.

Signature of Students
Name: Saumya Singh, Vageesha Rai, Vikas Kumar Verma
Roll No.: 2000290120138, 2000290120181, 2000290120189
Date: 11-03-2024

CERTIFICATE

This is to certify that Project Report entitled “**NoSQL to RDBMS Converter**” which is submitted by **Saumya Singh, Vageesha Rai, Vikas Kumar Verma** in partial fulfillment of the requirement for the award of degree B. Tech. in Department of Computer Science of Dr A.P.J. Abdul Kalam Technical University, Lucknow is a record of the candidates own work carried out by them under my supervision. The matter embodied in this report is original and has not been submitted for the award of any other degree.

Date: 11-03-2024

Supervisor Signature
Abhishek Goyal
(Assistant Professor- Department -Computer Science)

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the synopsis of the B.Tech Major Project undertaken during B.Tech. Fourth Year. We owe a special debt of gratitude to ABHISHEK GOLYAL, ASSISTANT PROFESSOR, Department of Computer Science, KIET Group of Institutions, Delhi- NCR, Ghaziabad, for his constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen the light of the day.

We also take the opportunity to acknowledge the contribution of Dr. Ajay Kumar Shrivastava, Head of the Department of Computer Science, KIET Group of Institutions, Delhi- NCR, Ghaziabad, for his full support and assistance during the development of the project. We also do not like to miss the opportunity to acknowledge the contribution of all the faculty members of the department for their kind assistance and cooperation during the development of our project.

Last but not the least, we acknowledge our friends for their contribution to the completion of the project.

Signature:
Date : 11-03-2024
Name : Saumya Singh, Vageesha Rai, Vikas Kumar Verma
Roll No: 2000290120138, 2000290120181, 2000290120189

ABSTRACT

NoSQL databases are becoming more and more popular for managing unstructured data for applications because they provide scalability and performance guarantees. However, many organizations choose to migrate data from an operational NoSQL database to a relational database based on SQL in order to utilize the current tools for business intelligence, analytics, decision-making, and reporting. The existing methods for transforming NoSQL databases into relational databases require manual schema mapping, which is laborious and needs subject expertise. An efficient and automatic process is needed to transform an unstructured NoSQL database into a structured database. We proposed and evaluated a practical method in this study for automatically transforming a NoSQL database into a relational database. In our experimental evaluation, we used a variety of NoSQL files, such as ".CSV," ".JSON," and ".XML" files, together with MySQL as a relational database to perform transformation operations for different dataset sizes. As transferring data from a NoSQL database to a relational database, we observed exceptional performance as compared to the state-of-the-art approaches used at the time. Our proposed system can be best described as an ETL (Extraction, Transformation, Loading) tool that basically loads the data from NoSQL files to relational databases. It contains two major elements, Schema analyzer and dynamic data mapper. The schema analyzer analyzes the input file schema thoroughly and then decides and defines the schema of it in the relational database. After this process is complete, the next step is data mapping where the data from the file is mapped with the table in the relational database. By mapping the data from these NoSQL files into relational databases, the user can query the data and extract essential insights from it with Structured Query language. Using relational databases, we can run SQL queries over the data which is mapped from these files. So, our model provides better access and more usability to these files and the crucial information stored in them.

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

NoSQL databases are becoming more and more popular for managing unstructured data for applications because they provide scalability and performance guarantees. However, many organizations choose to migrate data from an operational NoSQL database to a relational database based on SQL in order to utilize the current tools for business intelligence, analytics, decision-making, and reporting. The existing methods for transforming NoSQL databases into relational databases require manual schema mapping, which is laborious and needs subject expertise. An efficient and automatic process is needed to transform an unstructured NoSQL database into a structured database. We proposed and evaluated a practical method in this study for automatically transforming a NoSQL database into a relational database. In our experimental evaluation, we used a variety of NoSQL files, such as ".CSV," ".JSON," and ".XML" files, together with MySQL as a relational database to perform transformation operations for different dataset sizes. As transferring data from a NoSQL database to a relational database, we observed exceptional performance as compared to the state-of-the-art approaches used at the time. Our proposed system can be best described as an ETL (Extraction, Transformation, Loading) tool that basically loads the data from NoSQL files to relational databases. It contains two major elements, Schema analyzer and dynamic data mapper. The schema analyzer analyzes the input file schema thoroughly and then decides and defines the schema of it in the relational database. After this process is complete, the next step is data mapping where the data from the file is mapped with the table in the relational database. By mapping the data from these NoSQL files into relational databases, the user can query the data and extract essential insights from it with Structured Query language. Using relational databases, we can run SQL queries over the data which is mapped from these files. So, our model provides better access and more usability to these files and the crucial information stored in them.

1.2 PROBLEM STATEMENT:

In the contemporary data management landscape, NoSQL databases have become popular due to their flexibility, scalability, and performance in handling large volumes of unstructured or semi-structured data. However, organizations often face the need to migrate their data to a Relational Database Management System (RDBMS) to leverage advantages such as strong consistency, ACID transactions, and complex query capabilities. This migration process presents several challenges due to fundamental differences in data models, schema design, and data integrity constraints between NoSQL and RDBMS systems.

Key Challenges

Schema Translation: NoSQL databases often employ flexible, schema-less models, whereas RDBMS requires a predefined schema. Mapping complex, nested structures of NoSQL databases to the tabular format of relational databases can be challenging.

Data Integrity and Relationships: Ensuring data integrity and accurately representing relationships between data entities during the migration process, including the translation of embedded documents or key-value pairs to foreign key relationships in RDBMS.

Data Type and Format Conversion: Converting various data types and formats from NoSQL to equivalents in RDBMS while preserving data accuracy and consistency.

Performance and Scalability: Managing the migration of potentially massive datasets with minimal impact on the performance of both the source NoSQL database and the target RDBMS.

Tool Usability and Flexibility: Providing a user-friendly, flexible tool that can accommodate various NoSQL and RDBMS platforms, enabling users to configure and execute the migration process according to their specific requirements.

Security and Privacy: Ensuring the security of the data during the migration process, including the handling of sensitive or personal information in compliance with data protection regulations.

1.3 OBJECTIVES:

The objective of a NoSQL to RDBMS converter project centers on creating a robust, efficient, and user-friendly tool or system that facilitates the migration of data from NoSQL files to relational database management systems (RDBMS). The key goals of this project include:

1. Data Compatibility and Integration

To enable seamless migration of data structures, entities, and relationships from NoSQL databases to an RDBMS format, ensuring that all data is accurately translated and preserved in the process.

2. Schema Translation

To automatically translate or assist in translating NoSQL schema designs to relational schema models, considering the differences in data modeling paradigms between NoSQL and RDBMS.

3. Data Transformation and Mapping

To convert and map NoSQL data types and structures (such as documents, key-value pairs, or wide-column stores) to the relational model, including tables, rows, and columns, while maintaining data integrity and relationships.

4. Performance and Scalability

To provide an efficient migration process that minimizes the impact on system performance, allowing for the migration of large datasets with minimal downtime.

5. Usability and Accessibility:

To offer a user-friendly interface that allows users with varying levels of technical expertise to configure and execute data migrations, potentially including features for custom mapping, conflict resolution, and preview of changes.

6. Interoperability:

To support a wide range of NoSQL and RDBMS platforms, maximizing the tool's applicability across different technological environments and use cases.

7. Data Integrity and Security

To ensure that all migrated data maintains its integrity, with comprehensive support for data validation, error handling, and security measures to protect sensitive information during the migration process.

8. Research and Innovation

To explore new methodologies and technologies for data migration between heterogeneous database systems, contributing to the academic and practical knowledge base in database management, data engineering, and software development.

By achieving these objectives, the project aims to provide a solution that addresses the challenges of migrating from NoSQL to RDBMS, facilitating the transition for organizations seeking the advantages of relational databases, such as robust transaction support, complex query capabilities, and standardized data integrity mechanisms.

CHAPTER 2
LITERATURE REVIEW

The advent of NoSQL databases has introduced a new paradigm in data storage and management, offering flexible and scalable solutions for various applications. However, the coexistence of NoSQL databases with traditional Relational Database Management Systems (RDBMS) presents significant challenges, often requiring data migration or conversion between the two. Due to the demerits of NoSQL which includes less flexibility of queries, it's inability to scale itself, it's immature behavior and so on encourages us to focus on the conversion of NoSQL to SQL databases. This literature review tells us about the present scenario of research that has been conducted in the area of NoSQL to RDBMS conversion, highlighting key approaches, tools, methodology, techniques, algorithms, scenarios and considerations. Before delving into NoSQL to RDBMS conversion, it's essential to understand the fundamental differences and characteristics of these database systems. This understanding makes it easier in the conversion of the databases as we are clear with the basic functionality and usage of both the databases and their advantages and disadvantages.

Strauch, C., Martens, A., & Rumpe, B. [1] provides us with the Classification and Survey of NoSQL Databases survey which categorizes NoSQL databases and their various data models, which is crucial for comprehending the diversity of NoSQL systems. They have given us the top-down overview of the field. They proposed a comparative model which classifies and relates the functional as well as the non-functional requirements along with their algorithms which are employed in NoSQL databases. The Toolbox provides us with the ability to derive a simple and easy decision tree which helps the researchers.

The paper by Kaur, M., & Kumar, S. [2] explores the challenges involved in migrating data from NoSQL to RDBMS systems. It tells about all the implementations of the various NoSQL databases. It can work as a supplementary or a replacement tool which helps to cope up with the increase in the need to access as well as manage a large amount of data which is structured and unstructured. The given paper is to discuss about the various challenges that are commonly faced by database administrators while migrating from a relational database to NoSQL. The paper also provides the insights on how the given problems can be completely solved.

Challenges of Data Migration Between NoSQL and SQL Databases of Alomari, M. M., Lechner, G., & Gal, A [3] discusses the problems faced by the developer in data migration between NoSQL and SQL databases. Several approaches and tools have been developed to address the conversion problem.

A few prominent references are Efficient Data Conversion and Loading for NoSQL Systems which explores efficient data conversion techniques for NoSQL databases proposed by Yu, H., Varghese, B., & DeWitt, D. J. [4].

Shah, S. H., & Patil, M. R. [5] describes the NoSQL to SQL: A Review which provides insights into existing tools and approaches for NoSQL to SQL data conversion. To emphasize the real-world relevance of NoSQL to RDBMS conversion, it's essential to showcase case studies and practical applications of such conversions.

Migrating to a NoSQL Database: An Empirical Study of Cassandra presents empirical insights into the challenges and benefits of migrating to a NoSQL database, specifically Cassandra presented by Chen, J., & DeWitt, D. J. [6].

Zain Aftab and Waheed Iqbal [7] proposed a model which is very efficient. It identifies the schema then extracts it, transforms it, and then finally loads the data from NoSQL to a relational database. A job manager is given the NoSQL to SQL job. The job helps in invoking the Schema Analyzer. The job manager identifies the schema of the provided source database which consists of big data. The parsing of schema file is done and a SQL query is created. The conversion is done according to the destination format of the source database. This results in the database creation. Once the creation of SQL database is completed, the ETL processes get activated for parallel processing. In the next step, the data is extracted from the source database. The extraction is done through ETL process in batches. After the extraction is completed, the data is processed. This results in the creation of queries. These queries are in format of the relational database. Then finally the data is loaded to the relational database.

Gour et al. [8] tells us about how the ETL process is performed in the data warehouses. It also listed the challenges which are implemented to improve the ETL processes. Extract, transform and load (ETL) is the internal process of data integration and is related to the data warehousing. ETL tools are used to extract the given data from a chosen source. Then transforms to some new formats which are according to the rules. It then loads it to the database. ETL is the process which brings all the data together in a similar environment. ETL is used to function as a reshaping agent which convert the relevant source data to some useful information which is to be stored in the data warehouse system. There would be no information present in the data warehouse if these functions are not present. If the data from source is not clean or transformed in the correct way, then query will not be processed. The paper is proposed with the process which will reduce the time taken to extract, transform and load the data in the data ware house with the help of query-cache. The response time of the process is reduced to the query used in the process. The performance of the process will also become much more efficient.

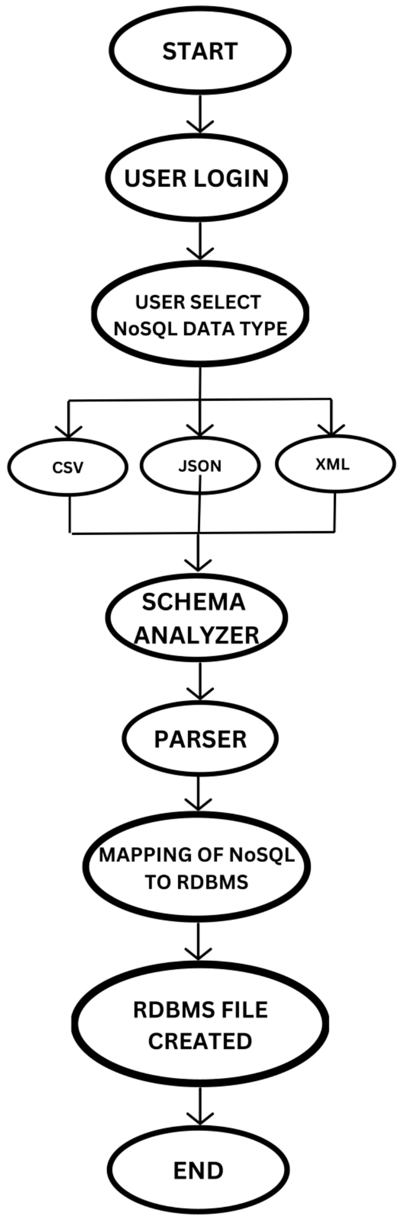
Skoutas and Simitsis [9] in his research work presented the methodology of automatic designing of ETL processes. They used attribute mapping and identification of ETL transformations for this work. One of the most essential tasks which is performed in early stage of a data warehousing is the understanding of the structure and component of the present data sources and their mapping with a model which contain common data .The paper tells us about mapping of data sources attributes and the data warehouse attributes. The model selected must be efficient for the redefinition efforts. They generally occur at the initial phase of a data warehouse. It functions as a means of communication. between parties which are involved. The authors argue about the ontologies which provide a very efficient and easy model for data warehouse. It also shows us how their usage can allow a high level of automation. It helps in the further development of the ETL designs.

Ramzan et al. [10] told us about the module which consists of data transformation along with cleaning to transfer the data from the relational to the NoSQL databases. The paper gives us an approach to transform the relational database to NOSQL databases. The paper provides us with two modules for the conversion process. The first module is data transformation and the second module is data cleansing. The first module is for the conversion of the relational database to the Oracle NoSQL database. The conversion will be based on model transformation. The second module is for the data cleansing. It provides the ability to make the data quality better to the maximum extent. It also prepares the data for the big data analytics. The experiments conducted by them show the proposed approach. The approach successfully converts the relational database to a NOSQL database with an improved data quality. It increases the value of data to a great extent.
Pore and Pawar [11] provided us with the research in which they discussed about the differences in SQL and NoSQL databases, providing us with various properties that are supported by the SQL and not by NoSQL, which includes ACID properties, transactions support, schema, and normalization. Converting data from NoSQL to RDBMS involves overcoming numerous challenges, including schema differences, data models, and query languages. The paper also identifies the feasibility of the process and technique of conversion of NoSQL to RDBMS databases.

This literature overview provides a comprehensive understanding of the challenges, existing conversion tools, and practical applications of NoSQL to RDBMS converters. As organizations continue to manage diverse data ecosystems, bridging the gap between NoSQL and RDBMS will remain an essential topic, necessitating ongoing research and development in this field.

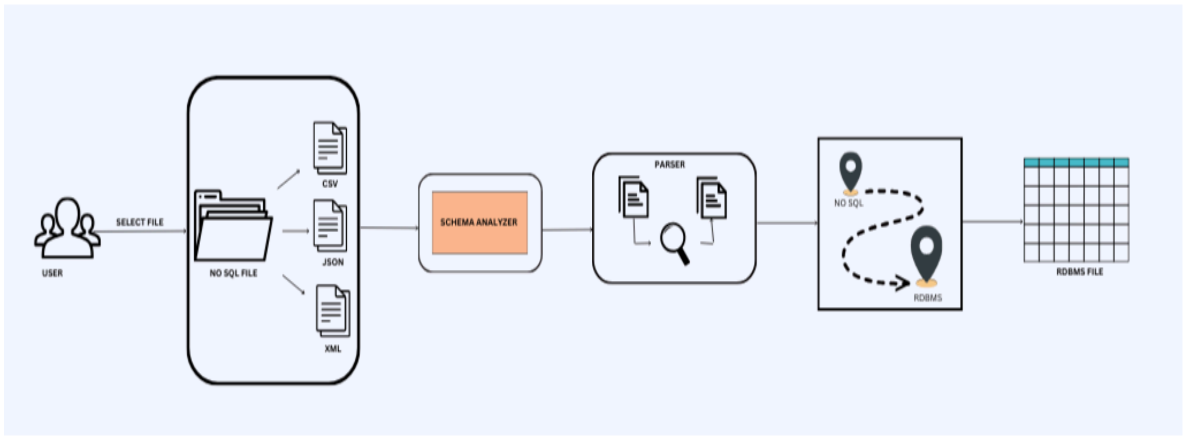
CHAPTER 3
PROPOSED METHODOLOGY

3.1 FLOWCHART



3.2 ALGORITHM PROPOSED

Our proposed system can be best described as an ETL (Extraction, Transformation, Loading) tool that basically loads the data from NoSQL files to relational databases. It contains two major elements, Schema analyzer and dynamic data mapper. The schema analyzer analyzes the input file schema thoroughly and then decides and defines the schema of it in the relational database. After this process is complete, the next step is data mapping where the data from the file is mapped with the table in the relational database. By mapping the data from these NoSQL files into relational databases, the user can query the data and extract essential insights from it with Structured Query language. Using relational databases, we can run SQL queries over the data which is mapped from these files. So, our model provides better access and more usability to these files and the crucial information stored in them.



System Overview

The system will consist of several core components that work together to facilitate the migration from NoSQL to RDBMS:

Schema Translation Module: This component is responsible for analyzing the schema or data structure of the NoSQL database and translating it into an equivalent relational schema that can be implemented in an RDBMS. It handles the mapping of document-based, key-value, or column family data models to a tabular format.

Data Transformation Engine: This engine converts the data from NoSQL formats (e.g., JSON, BSON) into data formats suitable for RDBMS (e.g., SQL tuples). It includes functionalities for transforming complex, nested structures into a relational format while preserving data integrity and relationships.

Migration Execution Framework: This framework oversees the actual migration process, including data extraction from the NoSQL database, transformation by the Data Transformation Engine, and loading into the RDBMS. It ensures efficient data transfer, minimal downtime, and scalability for large datasets.

User Interface (UI): A graphical user interface that allows users to configure the migration process, including selecting source and target databases, mapping schemas, and initiating the migration. The UI is designed to be intuitive and accessible to users with varying levels of technical expertise.

Validation and Verification Module: This module checks the migrated data for integrity, consistency, and correctness. It includes tools for error detection, reporting, and, if possible, automatic correction of common issues encountered during migration.

Security Layer: This component ensures that all data handled by the system is securely processed and transferred, implementing encryption, access control, and compliance with data protection regulations.

CHAPTER 4
TECHNOLOGY USED

XAMPP:

XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. It simplifies the process of installing and configuring these components to create a local web server for testing and development purposes.

Here's a brief overview of its components:

1. Apache: A widely used web server software that is responsible for serving web pages to users' browsers.
2. MariaDB: A database management system, derived from MySQL, used to store and manage data for your site in a relational database.
3. PHP: A popular server-side scripting language used for web development to create dynamic web pages.
4. Perl: Another scripting language that is often used for web development and network programming.

XAMPP is available for Windows, Linux, and macOS. It's designed to be very easy to install and to use, making it a popular choice for developers who want to develop and test their web applications on their local machines before deploying to a live server. The "X" in XAMPP stands for cross-platform, meaning it can run on any operating system that supports these components.

XAMPP also comes with additional tools such as phpMyAdmin for database management, FileZilla FTP Server, Mercury Mail for mail serving, and Tomcat for Java servlet and JSPs. It's a versatile package that can support a wide range of web development needs.

PHP:

PHP, which stands for "PHP: Hypertext Preprocessor," is a popular general-purpose scripting language that is especially suited to web development. It was originally created by Rasmus Lerdorf in 1994, and its open source. PHP scripts are executed on the server, and the result is returned to the browser as plain HTML.

Features of PHP:

1. Server-Side Scripting: PHP is mainly used for server-side scripting. This means that you can create dynamic page content, collect form data, and send and receive cookies, among other things.
2. Easy to Learn: PHP syntax is quite similar to C and Java, which makes it easy for individuals with background in these languages to pick up PHP.
3. Cross-Platform: PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.) and can be integrated with many servers (Apache, IIS, etc.).
4. Database Integration: PHP supports a wide range of databases, including MySQL, PostgreSQL, Oracle, Microsoft SQL Server, and others. This makes it a great choice for web applications that need database interaction.
5. Embedded: PHP code can be easily embedded within HTML code by using the special PHP tags. This allows for the creation of dynamic content without needing to call external files for processing.

Applications of PHP:

1. Web Pages and Web-Based Applications: Creating dynamic and interactive web pages and applications.
2. Content Management Systems (CMS): Many popular CMS, like WordPress, Drupal, and Joomla, are built with PHP.
3. E-commerce Applications: PHP is widely used in developing e-commerce platforms.
4. Web Frameworks: Frameworks like Laravel, Symfony, and CodeIgniter are based on PHP, facilitating rapid application development.
5. Data Analysis and Visualization: With the right libraries, PHP can be used for data analysis and generating visual reports.

PHP Versions:

Over the years, PHP has evolved significantly, with each new version bringing performance improvements, new features, and better security. It's important to keep PHP versions up to date to take advantage of these improvements and to maintain security.

PHP is a pivotal technology in web development, and its widespread use across CMS platforms and web applications underscores its importance. Whether you're building a simple website or a complex web application, PHP offers a rich ecosystem of tools and frameworks to support your development process.

MySQL:

MySQL is an open-source relational database management system (RDBMS) that relies on Structured Query Language (SQL) for processing the data in the database. MySQL is widely used for web applications and acts as the database component of the LAMP, MAMP, and XAMPP web development stacks, among others, with LAMP standing for Linux, Apache, MySQL, and PHP/Perl/Python.

Key Features of MySQL:

1. Cross-Platform Support: MySQL can be run on various platforms, including Linux, Unix, Windows, and macOS, making it versatile for different environments.
2. Reliability and Performance: Known for its reliability and high performance, MySQL is used in demanding production environments. Advanced caching, replication, and batch processing enhance its performance.
3. Scalability: It can handle a wide range of application demands, scaling from small to large enterprise operations with many concurrent users.
4. Security: MySQL provides strong data protection mechanisms, including encrypted connections and authentication processes that ensure secure data transactions.
5. Comprehensive Application Development: One of MySQL's advantages is its architecture, which allows for full flexibility in application development, offering features like stored procedures, triggers, and views.
6. Wide Range of Application: It's used in a wide range of applications, from web databases to logging applications, e-commerce, and data warehousing.

Applications of MySQL:

1. Web Databases: The most common use of MySQL is for web databases, where it stores data for web applications, such as user information, posts, and comments.
2. Content Management Systems (CMS): MySQL serves as the backend database for popular CMS platforms like WordPress, Joomla, and Drupal.

3. E-commerce Applications: Online stores and e-commerce platforms use MySQL to store product inventory, customer information, and transaction histories.
4. Data Warehousing and Business Intelligence (BI): Organizations use MySQL for data warehousing purposes, where they can store large volumes of data and perform complex queries for business insights.
MySQL Versions:
MySQL has undergone significant evolution, with improvements in performance, security, and features over the years. MySQL 8.0 is one of the latest versions, introducing enhanced JSON support, window functions, and CTEs (Common Table Expressions), among other features. Keeping your MySQL version up to date is crucial for benefiting from these improvements and maintaining security.
MySQL competes with other RDBMS systems like PostgreSQL, Microsoft SQL Server, and Oracle Database. However, its simplicity, speed, and compatibility with major web technologies make it a preferred choice for many developers and businesses.

VISUAL STUDIO:

Visual Studio is an integrated development environment (IDE) from Microsoft used for developing computer programs, websites, web apps, web services, and mobile apps. It offers developers a single application in which they can use various development tools, languages, and libraries to build a wide range of applications. Visual Studio supports development in several programming languages including, but not limited to, C#, Visual Basic .NET, C++, JavaScript, Python, and more. Its capabilities can be extended by developers and third-party companies using plugins.

Key Features of Visual Studio

1. Multiple Language Support: Supports development in C#, VB.NET, C++, F#, JavaScript, Python, and more, making it versatile for different types of development projects.

2. Powerful Debugging and Diagnostics: Offers robust debugging features, including the ability to set breakpoints, step through code, inspect variables, and analyze memory and CPU usage.
3. Integrated Development Environment (IDE): Provides a comprehensive environment with features like code editor, debugger, GUI design, database schema design, and version control integration.

4. Extensibility: Visual Studio can be extended with a vast array of extensions available through the Visual Studio Marketplace, allowing developers to add new features and support for different languages and frameworks.

5. Collaboration and Version Control: Integrated support for Git and Team Foundation Version Control (TFVC) enables developers to collaborate on projects and manage code changes efficiently.

6. Mobile Development: With Xamarin integration, developers can build and debug cross-platform mobile apps for Android, iOS, and Windows using C#.

7. Web Development: Offers extensive tools for building web applications, including ASP.NET development, JavaScript editing, and powerful frameworks like .NET Core.

8. Cloud Development: Direct integration with Microsoft Azure allows developers to build, test, manage, and deploy cloud applications directly from the IDE.

Editions of Visual Studio

Visual Studio is available in several editions, catering to different needs and budgets:

1. Visual Studio Community: A free edition for individual developers, open-source projects, academic research, and small teams. It offers many of the features of the more expensive editions.

2. Visual Studio Professional: Offers more advanced features and tools for professional developers and small teams, with subscription-based or standalone licensing.

3. Visual Studio Enterprise: The most comprehensive edition, providing advanced debugging, AI-driven code recommendations, testing tools, and more, for large development teams and enterprises.

Visual Studio Code:

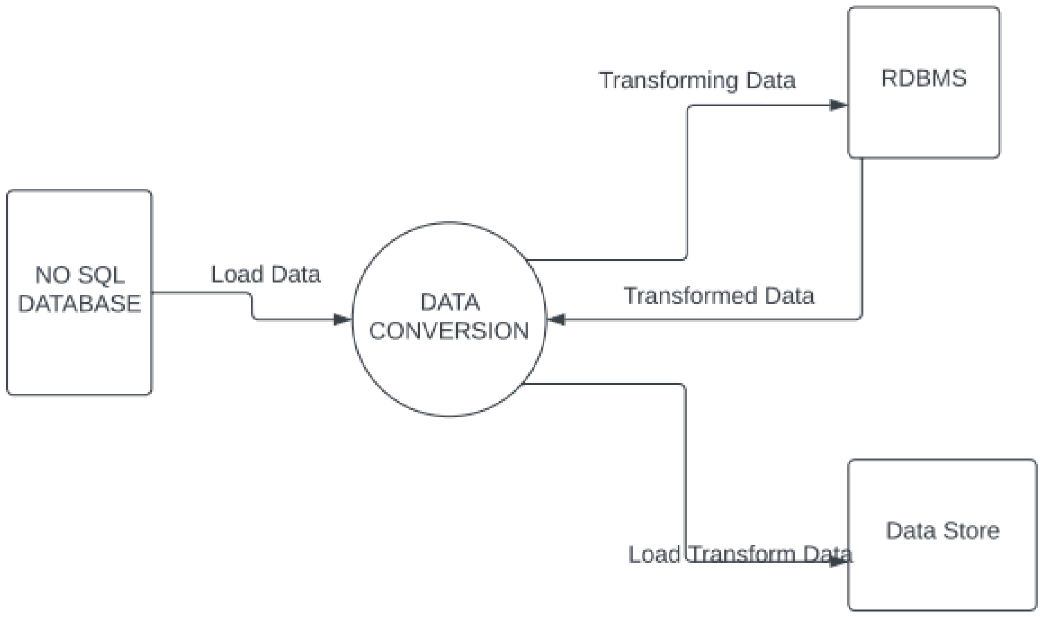
It's worth mentioning Visual Studio Code (VS Code), a free, open-source editor also developed by Microsoft. VS Code is lighter and more customizable than Visual Studio IDE, supports development in multiple languages through extensions, and is available on Windows, Linux, and macOS. While VS Code shares part of its name with Visual Studio, it's a separate product focusing more on being a powerful code editor rather than a full IDE.

CHAPTER 5

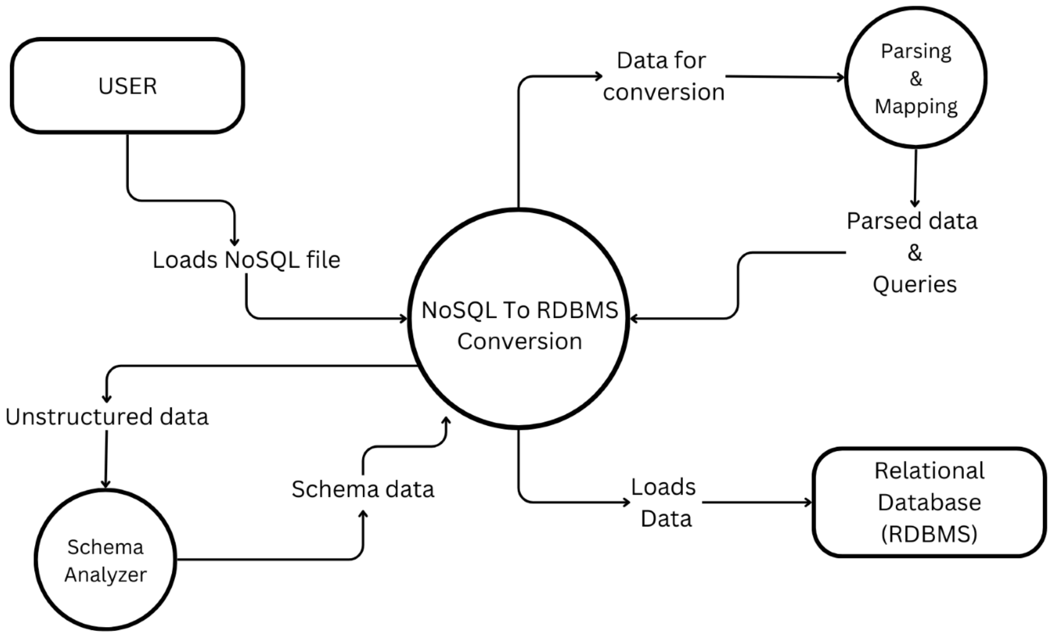
DIAGRAMS

5.1 DFD LEVEL 0 AND DFD LEVEL 1

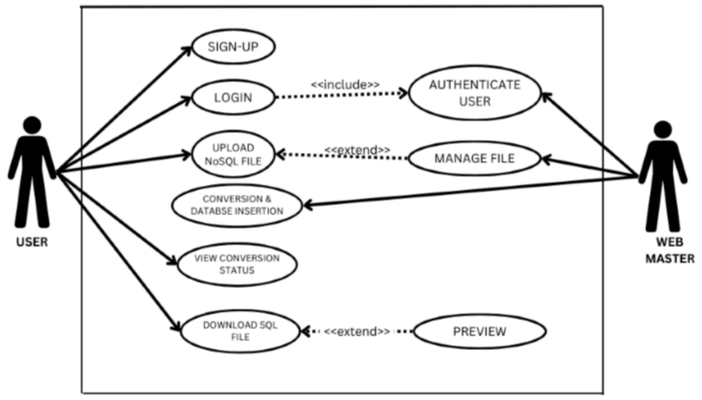
LEVEL 0



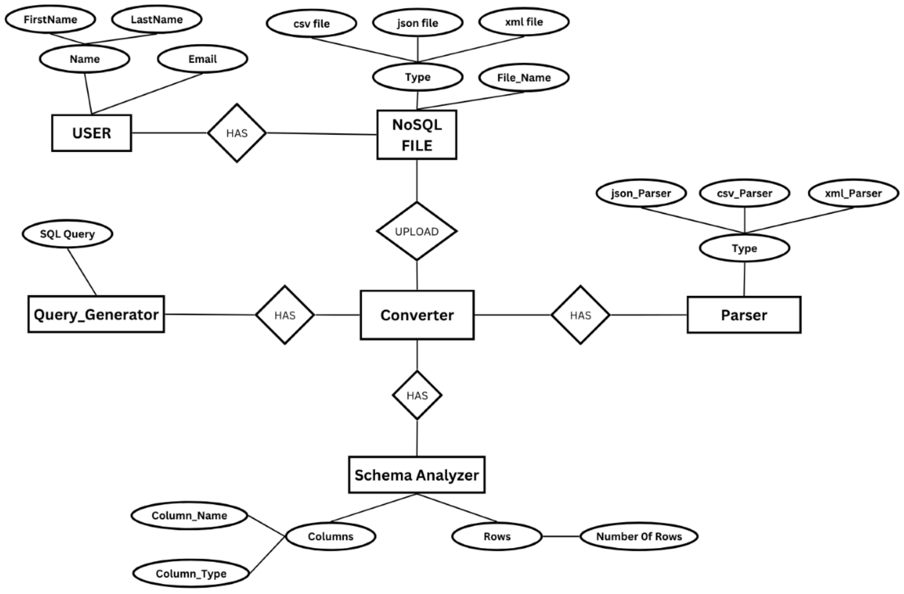
LEVEL 1


USER
Loads NoSQL file
NoSQL To RDBMS Conversion
Data for conversion
Parsing & Mapping
Parsed data & Queries
Unstructured data
Schema Analyzer
Schema data
Loads Data
Relational Database (RDBMS)

5.2 USE CASE DIAGRAM



5.3 ER DIAGRAM



CHAPTER 6
CONCLUSION

Conclusion:

The NoSQL to RDBMS conversion project represents a significant step towards bridging the gap between flexible, schema-less NoSQL databases and the structured, transaction-oriented world of relational databases. By developing a comprehensive tool that automates the migration process, this project addresses a crucial need for organizations looking to leverage the benefits of RDBMS for certain applications, such as enhanced data integrity, complex query capabilities, and robust transaction support. The emphasis on user accessibility, security, performance optimization, and support for a wide range of database technologies ensures that the solution is practical, efficient, and adaptable to various use cases and requirements.

In conclusion, the NoSQL to RDBMS conversion project can become an indispensable tool for organizations navigating the complexities of modern database technologies. By continuously evolving in response to technological advancements and user feedback, the project can expand its scope to offer more sophisticated, efficient, and versatile data migration solutions, catering to the ever-changing landscape of database management and data-driven decision-making.

REFERENCES

[1] Strauch, C., Martens, A., & Rumpe, B. (2017). "A Classification and Survey of NoSQL Databases". ACM Computing Surveys, 50(2), 1-38.

[2] Kaur, M., & Kumar, S. (2015). "Challenges in Data Migration from NoSQL Databases to RDBMS". International Journal of Computer Applications, 116(10), 1-7.

[3] Alomari, M. M., Lechner, G., & Gal, A. (2019). "Challenges of Data Migration Between NoSQL and SQL Databases." In Proceedings of the 2019 International Conference on Management of Data (COMAD), 145-152.

[4] Yu, H., Varghese, B., & DeWitt, D. J. (2016). "Efficient Data Conversion and Loading for NoSQL Systems". In Proceedings of the VLDB Endowment, 9(10), 684-695.

[5] Shah, S. H., & Patil, M. R. (2017). "NoSQL to SQL: A Review". International Journal of Computer Applications, 159(4), 23-29.

[6] Chen, J., & DeWitt, D. J. (2017). "Migrating to a NoSQL Database: An Empirical Study of Cassandra." In Proceedings of the VLDB Endowment, 10(12), 1879-1890.

[7] Zain Aftab, Waheed Iqbal, Khaled Mohamad Almustafa, Faisal Bukhari and Muhammad Abdullah (2020).” Automatic NoSQL to Relational Database Transformation with Dynamic Schema Mapping.” Hindawi Scientific Programming.

[8] V. Gour, D. S. S. Sarangdevot, J. R. N. R. Vidyapeeth, G. S. Tanwar, and A. Sharma (2010). “Improve performance of extract, transform and load (ETL) in data warehouse,” International Journal on Computer Science and Engineering, vol. 2, no. 3, pp.786–789.

[9] D. Skoutas and A. Simitsis(2006). “Designing ETL processes using semantic web technologies,” in Proceedings of the 9th ACM International Workshop on Data Warehousing and OLAP, DOLAP 06, ACM, New York, NY, USA, pp. 67– 74.

[10] M. Casters, R. Bouman, and J. Van Dongen, Pentaho Kettle Solutions (2010). Building Open Source ETL Solutions with Pentaho Data Integration, John Wiley & Sons, Hoboken, NJ, USA.

[11] S. Ramzan, I. S. Bajwa, B. Ramzan, and W. Anwar (2019). “Intelligent data engineering for migration to NoSQL based secure environments,” IEEE Access, vol. 7, pp. 69042–69057.

[12] A. K. Bhattacharjee, A. Mallick, A. Dey, and S. Bandyopadhyay (2013). “Enhanced technique for data cleaning in text file,” International Journal of Computer Science Issues (IJCSI), vol. 10, no. 5, p. 229.

[13] H. Mohamed, T. Leong Kheng, C. Collin, and O. Siong Lee (2011). “E-clean: a data cleaning framework for patient data,” in Proceedings of the 2011 First International Conference on Informatics and Computational Intelligence, pp. 63–68, IEEE, Bandung, Indonesia.

[14] B. Iqbal, W. Iqbal, N. Khan, A. Mahmood, and A. Erradi, “Canny edge detection and Hough transform for high resolution video streams using Hadoop and Spark,” Cluster Computing, vol. 23, no. 1, pp. 397–408, 202.

[15] A. Gupta, S. Tyagi, N. Panwar, S. Sachdeva and U. Saxena, "NoSQL databases: Critical analysis and comparison," 2017 International Conference on Computing and

Communication Technologies for Smart Nation (IC3TSN), Gurgaon, India, 2017, pp. 293-299, doi: 10.1109/IC3TSN.2017.8284494.

[16] LI Jun-shan, LI Jian-jun(20, “Research on NoSQL Database Technology,” 2nd International Conference on Management, Education and Social Science (ICMESS 2018).

[17] Ying-Ti Liao, Jiazheng Zhou, Chia-Hung Lu, Shih-Chang Chen, Ching-Hsien Hsu, Wenguang Chen, Mon-Fong Jiang, Yeh-Ching Chung, “Data adapter for querying and transformation between SQL and NoSQL database,” 2016, Vol 65, pp. 111-121.

[18] Jing Han, Haihong E, Guan Le, Jian Du, “Survey on NoSQL database,” 2011 6th International Conference on Pervasive Computing and Applications, 2011.

[19] Ágnes Vathy-Fogarassy, Tamás Hugyák, “Uniform data access platform for SQL and NoSQL database systems,” 2017, Vol 69, pp. 93-105.

[20] Srdja Bjeladinovic, “A fresh approach for hybrid SQL/NoSQL database design based on data structuredness,” Enterprise Information Systems, 2018, Vol 12(8-9), pp. 1202-1220.

[21] Sharvari Rautmare,D. M. Bhalerao, “MySQL and NoSQL database comparison for IoT application,” 2016 IEEE International Conference on Advances in Computer Applications (ICACA), 2016