DOI: http://dx.doi.org/10.31940/matrix.v13i2.106-114
URL: http://ojs2.pnb.ac.id/index.php/matrix

Optimizing transaction data performance in database management systems

Nahrun Hartono [1]****, Zulkarnaim Masyhur*** [2]

[1,2] *Universitas Islam Negeri Alauddin Makassar, Indonesia*
*Corresponding Author: *nahrunhartono@gmail.com*

Abstract: One indicator of the quality of an information system is the speed of data processing. A database's most common data processing operations are adding, displaying, changing, and deleting data. The amount of data stored in the database significantly impacts the performance of data processing and, therefore, the performance of information systems. The update command changes some or all of the data in a table. The update command works by retrieving the data in the table to be changed, entering the new data in a form, and then sending it back to the database. The update command is often combined with a condition specifying which data rows must be changed. This research is an experimental study that compares the use of the update command with a stored procedure to the use of the update command without a stored procedure. The results showed that the average processing time for the update command with the stored procedure was 348.896 milliseconds for the minimum data category, 266.462 milliseconds for the medium data category, and 279.543 milliseconds for the maximum data category. The average processing time for the update command without a stored procedure was 297.132 milliseconds for the minimum data category, 747.670 milliseconds for the medium data category, and 1256.273 milliseconds for the maximum data category. These results suggest that the update command with a stored procedure is more efficient than the one without a stored procedure. This is because the stored procedure can pre-compile the SQL statement, which reduces the time it takes to execute the statement.

Keywords: database, stored procedure, update

History Article: Submitted 22 May 2023 | Revised 1 June 2023 | Accepted 24 July 2023

How to Cite: N. Hartono and Z. Masyhur, "Optimizing transaction data performance in database management systems", *Matrix: Jurnal Manajemen Teknologi dan Informatika*, vol. 13, no. 2, pp. 106-114, 2023.

Introduction

The database is a collection of related. The database is the rules set for processing data into information to be useful for information system users according to their needs [1], [2]. The information generated from the database is obtained from several interconnected data [3]. There are several benefits of using a database, namely 1) Ease of storing data, 2) Minimizing data redundancy, 3) Data accuracy, 4) Availability of data according to user needs, 5) Data Security, 6) Availability of user sharing, 7) Providing data completely [4]. An application called the Database Management System is used to process data in the database, including MySQL, PostgreSQL, Ms. Access, SCL Server, and others [5], [1]. The database and database management system cannot be separated from each other and form a single unit called the database system. There are several database system components, namely 1) hardware, 2) operating system, 3) database, 4) database management system, 5) users, and 6) application interface [6], [7].

Database design and implementation, application design and implementation, and administrative procedures influence information systems [8]. Database design is the first factor that influences information systems. This means that a good database structure will greatly affect the performance of information systems. Nine aspects need to be considered in designing a database: the ability to be integrated, reach, level of detail, correctness, consistency, relevance, completeness, minimalism, and readability [9].

Several factors affect database performance, namely 1) response time, which is the duration of the database in processing a given command; 2) throughput, which is the ability of hardware and software to process data; 3) resources, hardware and software, 4) memory, the amount of memory used in completing the given command [10]. The amount of data to be processed by the database will help application performance if optimization is carried out [11].

Database optimization can be done by adjusting the hardware with the software and configuring the database application, but this method requires more expensive resources. Another method is optimizing the database design, which can only be done at the beginning of application development. Database optimization can also be done by optimizing database queries/commands used in applications [12]. In addition to database performance objectives, as mentioned earlier, optimization is also carried out for data security purposes to prevent interface exploitation of the database [13]. Communication errors between the application interface and the server will cause security holes in the stored data. This is called SQL injection. SQL Injection takes advantage of invalid data errors entered into the database through the application interface [14]. One method to optimize database security is by embedding data encryption algorithms in the application, resulting in stored data being different from the inputted data. This is done to minimize data misuse [15].

Generally, the most frequently performed data processes in the database are adding, displaying, changing, and deleting data. The large amount of data stored will greatly affect the performance of data processing and, of course, the performance of information systems. In the information system design process, there are database design stages. Understanding data flow, procedures, and organizational mechanisms for processing data is crucial in designing database systems. The basic data is designed in such a way as to meet the information needs of its users. Generally, database design is done using Entity Relationship Diagram (ERD). The designed ERD is then transformed into a table relation.

One aspect that influences the successful implementation of information systems is the design of tables in the database [16]. Using the right data types, using keys, and applying some table constraints will guarantee data quality. One benchmark for data quality is the absence of data duplication, whether it occurs within the same table or across different tables. Data duplication can lead to data inconsistencies, resulting in information discrepancies [17]. Applying normalization to tables will also affect data quality. Insert anomaly, delete anomaly, and update anomaly can be minimized by applying normalization rules. Management and data processing in the database can be done by running several commands. Structured Query Language (SQL) is a command for managing and processing database data. The SQL language is grouped into several categories, namely Data Definition Language (DDL), Data Manipulation Language (DML), and Data Control Language (DCL).

Insert, update, and delete commands are included in the DML category. Insert is used to add data to the table. Delete is used to delete some or all of the data in the table, while the update command is used to change some or all of the data in the table. The update command works by retrieving data in the table to be changed, entering it in the form, and then sending it back to the database; generally, the update command is combined with a condition where the data rows to be changed are by the specified conditions. This study aims to test the update command using a stored procedure in the database. The stored procedure is a subroutine that makes it possible to execute SQL statements that have been prepared [18]. There are several benefits of using stored procedures, including 1) effectively used for commands that are executed many times, 2) using stored procedures, the load on resources will be divided, and 3) better security [19].

Methodology

This research is experimental research. Experimental research is a type of research conducted by testing several conditions [20]. This research was conducted on a database containing tables, which have a number of rows of data. These data rows will be changed using the update command without a stored procedure and the update command using a stored procedure. Testing is done by running each command using the terminal/command prompt. The amount of data used is 30000, 70000, and 100000 data, while the research flow can be seen in Figure 1 below:

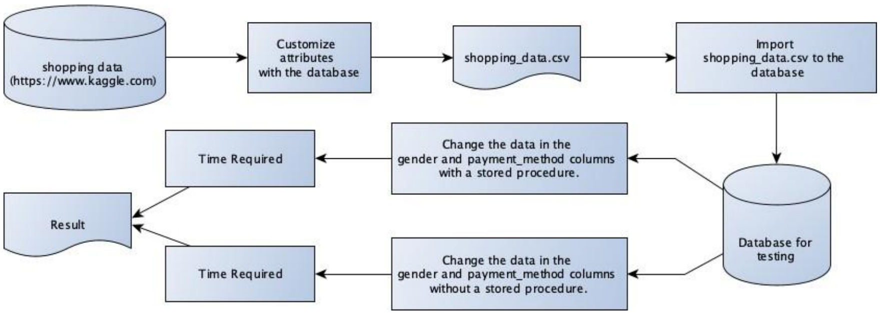

shopping data (https://www.kaggle.com)
Customize attributes with the database
shopping_data.csv
Import shopping_data.csv to the database
Database for testing
Change the data in the gender and payment_method columns with a stored procedure.
Change the data in the gender and payment_method columns without a stored procedure.
Time Required
Time Required
Result

Figure 1. Research Flow

The data used in this research is customer shopping data downloaded from Kaggle (https://www.kaggle.com/datasets/mehmettahiraslan/customer-shopping-dataset). It needs to be adjusted before testing some of the attributes in the data. The test was performed 10 times by running the update command with a different amount of data. Each test will record the processing time, and the results will then be averaged.

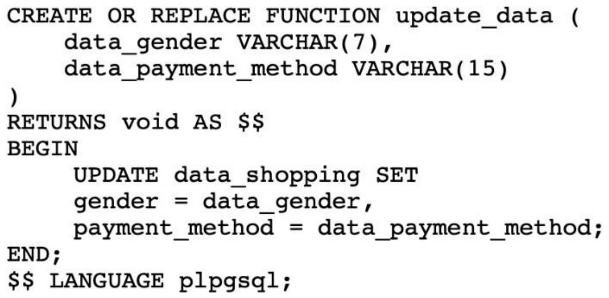
Testing is carried out by changing the data in the database with different command methods, namely using stored procedures and without using stored procedures with different amounts of data, namely the minimum amount of data with a total of 30,000 data, the amount of medium data with a total of 70,000 data and the maximum number data with a total of 100000 data. Before making changes to the data, creating a database and table structure to store data needs to be done. The structure of the test table used in this study is shown in Table 1 below:

Table 1. Table Structure for Testing

Field	Type Data/Length	Constraint
invoice_no	Character (10)	Primary Key
customer_id	Character (10)	Not Null
gender	Variable Character (7)	Not Null, Check (Male, Female)
age	Integer	Not Null
category	Variable Character (25)	Not Null
quantity	Integer	Not Null
payment_method	Variable Character (15)	Not Null, Check (Cash, Debit Card)
invoice_date	Date	Not Null
shopping_mall	Variable Caharcter (45)	Not Null

Table 1 above shows the structure of the test table, and it can be seen that there are 9 columns/fields with data types and constraints that are adjusted to the test data.

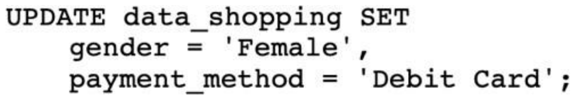
The next step in this research is to enter data into the table that has been made. Then the testing; the first test is carried out on minimum category data, then medium category data, and maximum category data. Before entering data into the table, it is necessary to make data adjustments. The data adjustments change all data in the gender column to 'Male' and data in the payment_method column to 'Cash'. After that, testing is carried out by executing updated data with a stored procedure and updating data without a stored procedure. The two commands will change the data in the gender column, which previously had the value 'Male', to 'Female', and the data in the payment_methode column, which previously had the value 'Cash', to 'Cash'. Debit Card'. The stored procedure commands used in testing can be seen in Figure 2 below:



```
CREATE OR REPLACE FUNCTION update_data (
    data_gender VARCHAR(7),
    data_payment_method VARCHAR(15)
)
RETURNS void AS $$
BEGIN
     UPDATE data_shopping SET
     gender = data_gender,
     payment_method = data_payment_method;
END;
$$ LANGUAGE plpgsql;
```

Figure 2. Stored Procedure Update

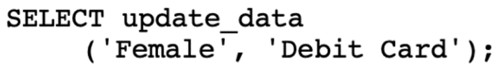
As for the command to change data without a stored procedure, it can be seen in Figure 3 below:



```
UPDATE data_shopping SET
    gender = 'Female',
    payment_method = 'Debit Card';
```

Figure 3. Update Without Stored Procedure

In contrast to changing data with a stored procedure, changes without a stored procedure can be executed immediately, whereas to execute data changes with a stored procedure, it is necessary to call the previously created stored procedure, as shown in Figure 3. The calling commands can be seen in Figure 4 below:



```
SELECT update_data
     ('Female', 'Debit Card');
```

Figure 4. Calling Command Stored Procedure

Suppose the command in Figure 4 is executed. In that case, it will execute a stored procedure called update_data where in the stored procedure update _data, there are commands to change the data in the gender and payment_method columns; the values 'Female' and 'Debit Card' are new values that will replace the old values. Namely 'Male' and 'Cash'.

Results and Discussions

Experimental research trials were conducted on both hardware and software systems with the following specifications: 1) Intel Core i5 dual-core 2,3 GHz, 2) RAM 8 gigabytes, 3) Solid State Drive 128 gigabyte, 4) Intel Iris Plus Graphics 640, 5) Operating System: Mac OS Ventura, 6) PostgreSQL Version 14.8. The first trial was carried out by running the update command with the stored procedure; the command execution was performed 10 times for each amount of data. Next, run the update command without a stored procedure and do it 10 times. The results of testing the minimum data category with a total data of 30,000 can be seen in Table 2 below:

Table 2. The Result from the Minimum Data Category

Experiment To -	Update with Stored Procedure (milliseconds)	Update without Stored Procedure (milliseconds)
1	327,442	304,699
2	339,585	308,567
3	343,934	306,887
4	384,772	308,828

5	352,058	278,808
6	321,442	290,663
7	393,636	339,163
8	328,377	296,573
9	375,645	279,559
10	322,076	257,582

The graph of the experimental results for the minimum data category can be seen in Figure 5 below:

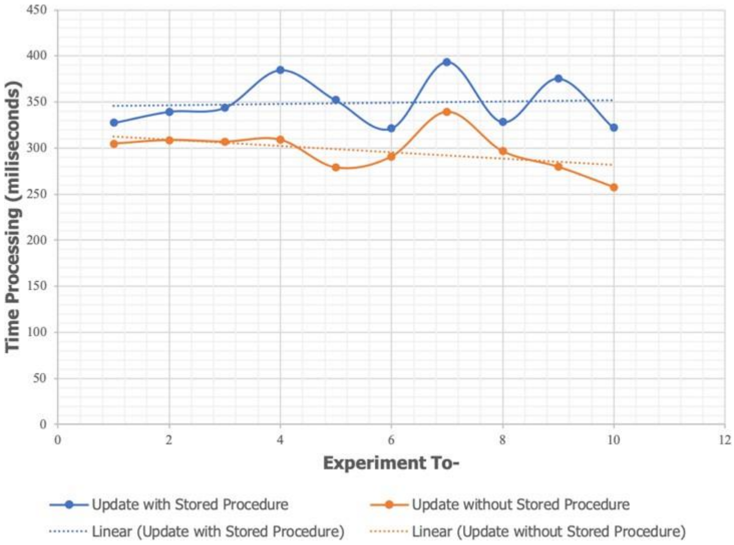


Figure 5. Graph Result in Minimum Data Category

The test results for the medium data category with a total data of 70,000 can be seen in Table 3 below:

Table 3. The Result from the Medium Data Category

Experiment To -	Update with Stored Procedure (milliseconds)	Update without Stored Procedure (milliseconds)
1	326,921	774,549
2	239,468	713,015
3	323,322	750,009
4	251,495	768,340
5	259,701	729,498
6	224,897	727,768
7	251,202	770,247
8	263,754	764,007
9	250,137	723,504
10	273,726	755,772

The graph of the experimental results for the medium data category can be seen in Figure 6 below:

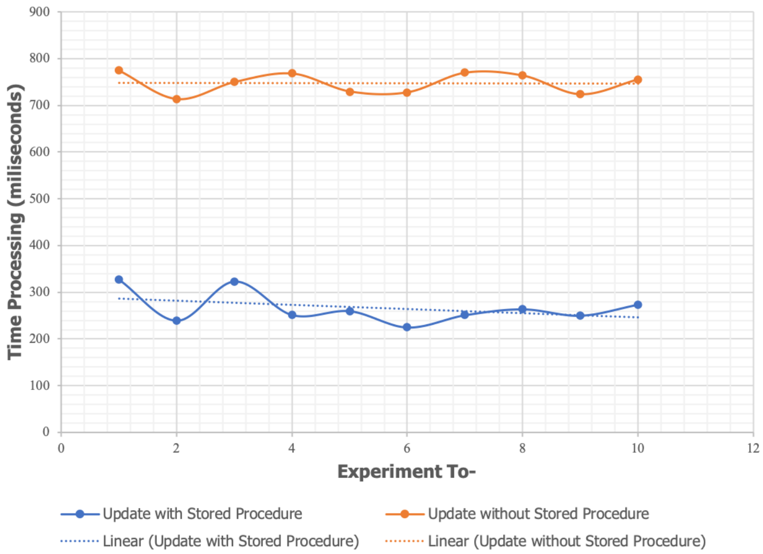


Figure 6. Graph Result in Medium Data Category

Figure 5 shows that changing data with a stored procedure is better than changing data without using a stored procedure in the medium data category with a total of 70,000 data. The experimental results for the maximum data category with a total data of 100000 can be seen in Table 4 below:

Table 4. The Result from the Maximum Data Category

Experiment To -	Update with Stored Procedure (milliseconds)	Update without Stored Procedure (milliseconds)
1	254,917	1156,595
2	297,792	1005,891
3	250,609	1007,575
4	228,304	1380,146
5	258,102	1101,780
6	540,740	1067,539
7	233,316	1695,356
8	255,274	1051,303
9	223,324	1116,453
10	253,055	1980,094

The graph of the experimental results for the maximum data category can be seen in Figure 7 below:

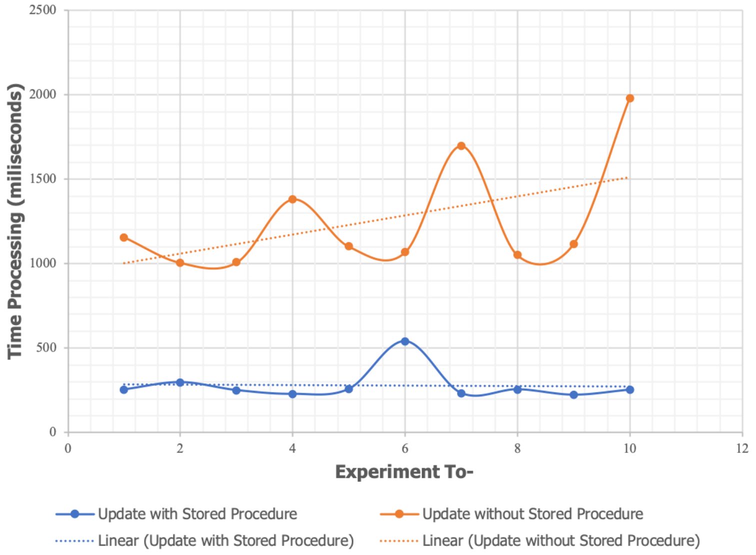


Figure 7. Graph Result in Maximum Data Category

Figure 7 shows that changing data with a stored procedure is faster than changing data without a stored procedure in the category maximum with 100000 data. Data changes using stored procedures were better in the medium and maximum data categories of the three experiments conducted in this study. Table 5 displays the outcomes concerning the percentage representation of data modification process time when employing stored procedures, while Figure 8 illustrates the graphical depiction of the comparative average data modification process time using stored procedures versus data modification without the utilization of stored procedures:

Table 5. Percentage of Data Modification Process Time with Stored Procedure

Experiment To -	Minimal	Medium	Maximum
1	9%	12%	9%
2	10%	9%	11%
3	10%	12%	9%
4	11%	9%	8%
5	10%	10%	9%
6	9%	8%	19%
7	11%	9%	8%
8	9%	10%	9%
9	11%	9%	8%
10	9%	10%	9%

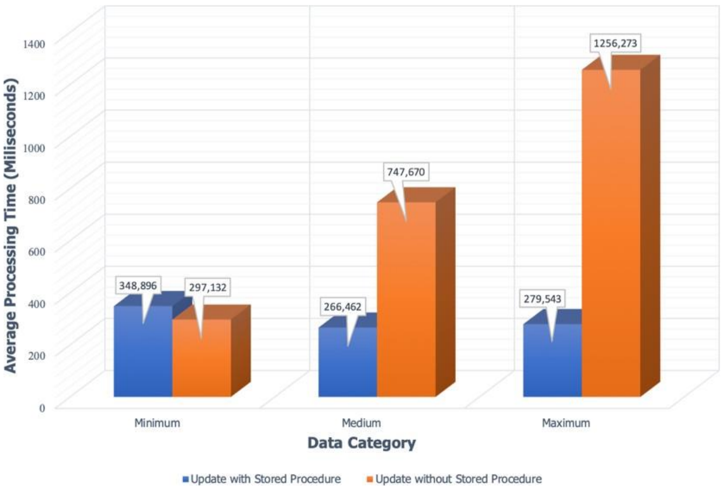


Figure 8. Average Processing Time Based on Data Category

Table 5 presents the percentage of data modification process time using stored procedures. The table indicates that as the percentage value decreases, the data modification process time becomes faster. Figure 8 shows the average processing time of the three data categories. It can be seen that for the minimum data category with the amount of data 30000, the use of updating data without stored procedures resulted in an average processing time of 297.132 milliseconds, while for the medium and maximum data categories with the amount of data 70000 and 100000 the use of updating data without stored procedures produced an average processing time of 747.670 milliseconds and 1256.273 milliseconds. The test results show that the use of updating data with stored procedures is better used in the medium data category with an amount of data of 70000 with an average processing time of 266.462 milliseconds and the maximum data category with an amount of data 100000 with an average processing time of 279.543 milliseconds.

Conclusion

This study is an experimental research that compares data conversion methods using stored procedures and without stored procedures. The data used is categorized based on the amount of data, namely the minimum, medium, and maximum data categories. The results showed that changing data using a stored procedure was better for the amount of data in the medium and maximum categories while changing data without a stored procedure was better for the minimum data category. This implies that the utilization of stored procedures to optimize the performance of data transactions is more suitable for data quantities falling within the categories of medium and maximum. The results of this study are very useful for developing applications that use data conversion with large amounts of data.

This research is limited to the comparison of query processing times. It is highly recommended for future research to not only compare processing time but also investigate how queries function to identify the underlying factors contributing to the slow processing time of stored procedures with minimum data category. The researcher realizes that this research is still far from perfect because the researcher's understanding still needs to be improved, and the researcher's time is limited. For further research, researchers suggest testing on a larger amount of data and directly implementing it in the application.

References

[1] Connolly, T. M. dan Begg, C., Database Systems: A Practical Approach to Design, Implementation, and Management. Science, 2015.

[2] Raut Dr, A. B., "NOSQL Database and Its Comparison with RDBMS," vol. 13, no. 7, pp. 1645–1651, 2017.

[3] G. Susilo, "Keamanan Basis Data pada Sistem Informasi diera Global," J. Transform., vol. 12, no. 2, 2016, doi: 10.56357/jt.v12i2.70.

[4] H. A. Mumtahana, "Optimization of Transaction Database Design with MySQL and MongoDB," Sink. J. Dan Penelit. Tek. Inform., vol. 7, no. 3, Art. no. 3, Jul 2022, doi: 10.33395/sinkron.v7i3.11528.

[5] E. Susena, E. Utami, dan A. Sunyoto, "Perencanaan Strategis Sistem Informasi Smart Campus Untuk Meningkatkan Pelayanan Di Politeknik Indonusa Surakarta," vol. 1, 2015.

[6] E. Utami dan S. Raharjo, RDBMS dengan PostgreSQL di GNU/Linux. Penerbit Andi, 2006.

[7] N. Hartono, "Keuntungan Penggunaan External Function pada Database PostgreSQL," CSRID Comput. Sci. Res. Its Dev. J., vol. 7, no. 1, pp. 39, Feb 2015, doi: 10.22303/csrid.7.1.2015.39-47.

[8] P. Rob dan C. Coronel, Database Systems: Design, Implementation, And Management. Course Technology., 2009.

[9] O. Herden, "Measuring Quality of Database Schemas by Reviewing – Concept, Criteria and Tool," COOP Work Quant Approaches Object-Oriented Softw Eng QAOOSE 2001, 2001.

[10] M. Silalahi dan D. Wahyudi, "Perbandingan Performansi Database MongoDB dan MySQL Dalam Aplikasi File Multimedia Berbasis Web," Comput. Based Inf. Syst. J., vol. 6, no. 1, pp. 63–78, 2018, doi: https://doi.org/10.33884/cbis.v6i1.574.

[11] Y. Y. Putra, O. Purwaningrum, dan R. H. Winata, "Perbandingan Performa Respon Waktu Kueri MySQL, PostgreSQL, dan MongoDB," J. Sist. Inf. Dan Bisnis Cerdas, vol. 15, no. 1, Art. no. 1, Jul 2022, doi: 10.33005/sibc.v15i1.7.

[12] Q. Li, X. Honglin, dan G. Yan, "Research on Performance Optimization and Implementation of Oracle Database," dalam 2009 Third International Symposium on Intelligent Information Technology Application, Nov 2009, pp. 520–523. doi: 10.1109/IITA.2009.7.

[13] N. Hartono, E. Utami, dan A. Amborowati, "Migrasi dan Optimalisasi Database Sistem Informasi Manajemen Universitas Cokroaminoto Palopo," J. Buana Inform., vol. 7, no. 4, Art. no. 4, Okt 2016, doi: https://doi.org/10.24002/jbi.v7i4.766.

[14] E. Utami, "The Advantages of Using CHECK Constraints in The Academic Database Tables," J. Softw. Acad. Publ., vol. 9, no. 2, pp. 382–388, Feb 2014.

[15] P. Irfan, R. Hammad, A. S. Anas, Fatimatuzzahra, dan N. Samudra, "Application of the Blowfish Algorithm in securing patient data in the database," Matrix J. Manaj. Teknol. Dan Inform., vol. 12, no. 2, pp. 102–108, Jul 2022, doi: 10.31940/matrix.v12i2.102-108.

[16] M. R. A. Yudianto, A. Rahim, dan E. Utami, "Analisis Aspek Kualitas Skema Basis Data (Studi Kasus Pada Aplikasi LaundryPOS)," vol. 8, no. 1, pp. 8, 2019.

[17] R. Hammad, A. C. Nurcahyo, A. Z. Amrullah, P. Irfan, dan K. A. Latif, "Optimization of data integration using schema matching of linguistic-based and constraint-based in the university database," Matrix J. Manaj. Teknol. Dan Inform., vol. 11, no. 3, pp. 119–129, Nov 2021, doi: 10.31940/matrix.v11i3.119-129.

[18] N. Suver, "A Database Tuning Framework For Improving Stored Procedure Performance," ProQuest Diss. Publ., pp. 18, 2020.

[19] I. Warman dan W. Wildani, "Analisa Kinerja Query Stored Procedure Pada Database Management System (DBMS) MySQL," J. Sains Dan Teknol. J. Keilmuan Dan Apl. Teknol. Ind., vol. 21, no. 1, pp. 58, Jun 2021, doi: 10.36275/stsp.v21i1.366.

[20] Z. A. Hasibuan, Metodologi Penelitian pada Bidang Ilmu Komputer dan Teknologi Informasi. Jakarta: Fakultas Ilmu Komputer Universitas Indonesia, 2017.