International Journal of Applied Sciences and Smart Technologies

Volume 5, Issue 1, pages 1–16

p-ISSN 2655-8564, e-ISSN 2685-9432

Forensic Investigation in SQL Server Database Using Temporal Tables & Extended Events Artifacts

Shadi K. A. Zakarneh[1,*]

[1]*M.Sc. Student, Palestine Technical University- Kadoorei, Tulkarem, Palestine,*
[*]*Corresponding Author: zakarnehshadi@gmail.com*

(Received 21-09-2022; Revised 26-09-2022; Accepted 22-11-2022)

Abstract

Different Database management systems (DBMS) were developed and introduced to store and manipulate data. Microsoft SQL (MSSQL) Server one of the most popular relational DBMS used for large databases. With the increasing use of databases, intentional and unintentional accidents on databases are increasing dramatically. Therefore, there is a great need to develop database forensic investigation (DBFI) tools and models. The temporal table is a new feature introduced with MSSQL server 2012 for track changes, database audit, data loss protection, and data recovery. In addition, the extended events another new feature introduced with MSSQL server 2008 for database performance troubleshooting. This study focused on DBFI in the MSSQL server using temporal tables and extended events artifacts. The experiment is conducted and the results have presented the use of the temporal tables and extended events artifacts in analyzing and determining the internal unauthorized modification on the database.

Keywords: SQL Server, database forensic, forensic investigation, database management system.

1 Introduction

MSSQL Server is a relational DBMS. Microsoft in 1998 developed the first version of the MSSQL server, which is version 7.0. Microsoft continues in developing newer versions in the MSSQL server until the last version today which is the MSSQL server 2019 [1].

In the MSSQL server, there are two main types of databases, system databases that are used for SQL server operations. The second type is the user's databases, which are

Volume 5, Issue 1, pages 1–16
p-ISSN 2655-8564, e-ISSN 2685-9432

created by the users for their needs and business [1]. Each database in MSSQL Server mainly consists of two types of files. A transaction log file that is stores information about the transactions executed on the database. Each time the data modified, the transaction log information is used to undo or redo the changes, so the transaction log also used in the database recovery after the failure occurred [2]. The other file is the data file, which is used to store the data [2].

MSSQL Server consists of several services, which are database engine, agent, MSSQL server browser, and MSSQL server full-text search. Also, the MSSQL server provided other services called business intelligence services which are used for database analysis, these services are MSSQL server integration service, MSSQL server reporting service, and MSSQL server analysis service [3].

Since the stable and reliable MSSQL server 2019 is an improved version of the previous versions, the new version includes all the previous version's features in addition to a set of new features related to performance, security, availability, and big data [4]. The MSSQL server 2019 new features include intelligent query processing, which improves the query optimizer behavior, as a result, the performance will improve [4]. Accelerated database recovery (ADR) is another new feature, by adding this feature; the time required for the database recovery process has been greatly reduced [1]. AlwaysEncrypted with secure enclaves feature, by this feature, administrators are not allowed to access the decryption keys, while the transparent column encryption is allowed [1]. This feature enables MSSQL Server to encrypt part of the memory to perform computations on encrypted fields without exposing the unencrypted values to the rest of the operations [1]. To achieve this goal, secure enclave technology is used [5]. The other feature is memory-optimized tempdb metadata, in this feature; improvements have been made to the tempdb code by improving access to RAM so that metadata can rely completely on memory. Thus, the so-called bottleneck problem that large data is exposed to when using a large amount of tempdb is bypassed [5].

In recent decades, the use of databases has increased dramatically. The computerization of businesses and services and the use of applications in the daily lives of individuals have greatly increased the need for databases [6]. With this expansion in the use of databases, the need to improve security and privacy protection means increases

Volume 5, Issue 1, pages 1–16

p-ISSN 2655-8564, e-ISSN 2685-9432

dramatically, especially with the increase in security problems and security incidents that affect data confidentiality and users' privacy [6]. Because of the increase in security incidents on databases, there has become a great need for DBFI to identify digital evidence and perpetrators and improve information security [6].

MSSQL server one of the top ten relational DBMS due to the database engines ranking due to the popularity in 2020 as shown in Fig. 1 [7].

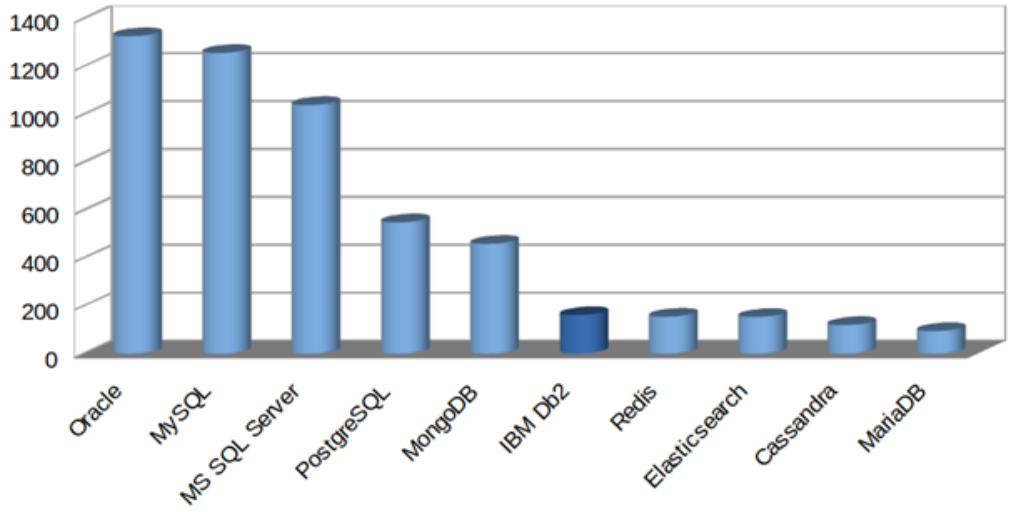


Figure 1. Database Ranking Score in Dec 2020 [7]

A new feature introduced in the MSSQL server named a temporal table. This feature was introduced with ANSI SQL 2011 [8]. Temporal Table started as a new feature with the MSSQL server 2016. Temporal tables are introduced for track changes, audit purposes, data loss protection, and data recovery in case of intentional or unintentional changes [9]. Another new feature that was developed in the MSSQL Server to help database developers troubleshoot performance issues during and after the development of the databases is the Extended events feature. This feature was first launched with MSSQL Server 2008 and then it was improved in MSSQL Server 2012 [10].

With the high popularity and wide use of the MSSQL server. Therefore, DBFI in the MSSQL server is highly important to determine the artifacts that can be extracted from the MSSQL server's new versions with its new features, especially in security features.

Volume 5, Issue 1, pages 1–16
p-ISSN 2655-8564, e-ISSN 2685-9432

This study will focus on the DBFI in MSSQL server 2019 to determine the new artifacts according to the temporal table and extended events features.

2 Literature review

Database forensics is one of the branches of digital forensics [11]. DBFI is based on examining and retrieving the contents of databases and analyzing metadata to identify digital evidence related to incidents that the databases are exposed to [12]. Because of the wide use and spread of digital data and the heavy reliance on databases, DBFI has become necessary to investigate accidents affecting databases [11]. As well as in other cybercrimes, where the digital forensic investigation in many cases includes extracting digital evidence from the databases of systems and applications related to the committed cybercrime [11].

To perform DBFI, many tools are used to extract databases and their metadata for analysis and investigation [13]. Some of the features of these tools are the ability to clone a hard drive, compare files, and encrypt them. These tools also work to recover deleted or damaged data [14]. In addition to the ability to recover lost or deleted database components such as tables, views, keys, and stored procedures [13].

While the DBFI includes the identification, collection, preservation, reconstruction, analysis, and reporting of the investigation results and findings. However, the multiplicity and diversity of DBMS such as Oracle, MSSQL Server, and PostgreSQL, etc. made it difficult to have a specific model for DBFI. Several DBFI models were designed based on specific accidents that some types of databases were exposed to [11].

Where the DBFI aims to find the digital evidence in the database, different forensic investigation models were conducted [15]. Some of these models analyzed transactions and journal logs. While other models worked to recover deleted data, among these, the models that rely on transaction logs to recover data. While other models depend on the analysis of the database engine to bypass the problem of deleting records for the antiforensic, overwriting, or changing them every period [15].

The engine-based method for data recovery is based on raw-level data analysis. This method is often used in small DBMS such as SQLite, where the internal structure of

Volume 5, Issue 1, pages 1–16
p-ISSN 2655-8564, e-ISSN 2685-9432

the databases must be understood for the investigator to use this method [15]. Thus, to use the forensic investigation method based on database engine analysis on large DBMS such as MSSQL server, it is necessary to understand the internal structure of the database engine and its storage [15].

In MSSQL server DBFI different artifacts can be collected and analyzed such as transaction logs files, Data files, SQL server logs, database schemas [15]. While the log file can be deleted or modified, so the investigator may need to analyze the data file. To Analyze the data file, the investigator needs to understand how is the MSSQL Server storage engine stores the raw data and the internal structure of the data file [11].

The data file in MSSQL Server consists of a set of pages. A page consists of a header, a data row, and a row offset array as shown in Fig. 2. The page size is 8192 bytes, of which 96 bytes are reserved for the header. The page metadata is stored in the first 64 bytes of the header and the rest of the header space is filled with 0x00. While record data in the tables stored in the data row, if the record size more than 8060 bytes the SQL server stores the record in multiple pages. The record location in the page is stored in the row offset array [15].

While DBMS record data changes in transaction logs and record information about transactions in audit logs such as what changed, when the changes were made, and who made the change. However, this data may not be sufficient for some systems such as financial systems that sometimes need to access a snapshot of the data at a certain time [2]. To solve this problem, temporal tables are defined by ANSI SQL 2011 to meet these requirements. Temporal tables are designed to keep a complete record of data changes and are easy to analyze on time. There are two types of temporal tables. The first type is system-versioned temporal tables, these tables keep a history of data changes based on the time changes occurred in the system [3]. Thus, system-versioned temporal tables provide a snapshot of the data that was on the system at a specific time. The second type is application-versioned temporal tables that provide data that is valid from a business point of view [3].

Volume 5, Issue 1, pages 1–16

p-ISSN 2655-8564, e-ISSN 2685-9432

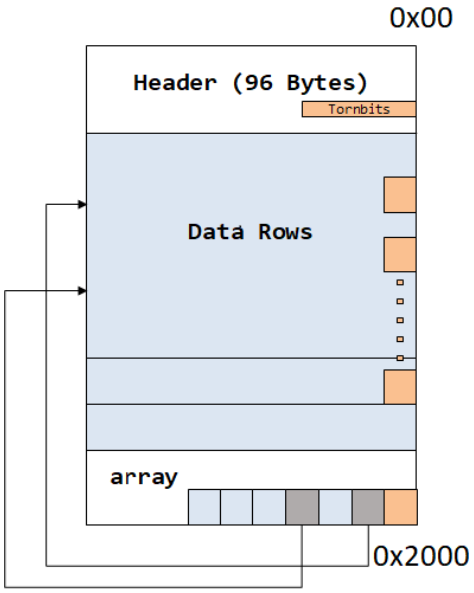


Figure 2. Page Structure [15]

Extended events feature was launched in MSSQL Server to collect as much data as needed to help database administrators or developers troubleshoot and identify database performance problems [16]. This feature was released with MSSQL Server 2008 and later versions. In MSSQL Server 2008 this feature was introduced without a GUI, so developers had to write large and complex queries to get the required data [5]. This feature has been further developed in MSSQL server 2012 and later to enable the user to set it up through the GUI and to give greater choices of data that can be collected to identify performance problems and to be used for troubleshooting by developers [5]. The extended events performance monitoring system is lightweight so it uses minimal performance resources. Extended events sessions can be created or modified, and the collected data is displayed and analyzed through the GUI provided by the MSSQL server management studio. The developer can choose the data he wants to collect and thus display it on the live monitoring screen, as well as store it in files or a table in a database for follow-up and analysis at any time [16].

Many previous studies were researched in the DBFI field. A study entitled: Development and validation of a Database Forensic Metamodel, this study aimed to present a model for DBFI called database forensic metamodel [17]. where the study

Volume 5, Issue 1, pages 1–16
p-ISSN 2655-8564, e-ISSN 2685-9432

analyzed a set of models used in DBFI to reach this model, which consists of four phases as identification, artifacts collection, artifacts analysis, and documentation and presentation [17]. Another study entitled: Detecting Database File Tampering through Page Carving, The study focused on presenting a component that detects modifications in the database file, this component relies on forensic investigation to identify discrepancies between indexes and tables in the database [14]. A study entitled: Duel Security-Detection of Database Modification Attack and Restore Facility from Unauthorized Access, this study proposed a model for dual security to identify and prevent attacks on the database by monitoring web and database requests [18]. In the proposed model the modified data can be restored using the MD5 algorithm [18].

While the previous studies focused on reviewing DBFI models and presenting a proposed model. Other studies focused on data recovery using DBFI techniques and tools by collecting and analyzing the log files. While other studies focused on analyzing the data files to recover the deleted or tampered data. This study aims to analyze and detect internal unauthorized modifications on the MSSQL server Database using temporal tables and extended events artifacts. In addition, the modified data will be recovered using the temporal table. The study used a DBFI model consist of four phases (identification, artifacts collection, artifacts analysis, and documentation and presentation).

3 Research methodology

Two basic steps to complete this study. The first step is to collect data and information from literature studies, as a literature study is a method of collecting data through reading books, thesis, journals, and other related resources. The second step is to design a scenario to implement the DBFI in the MSSQL server database.

Through the designed scenario, the DBFI model is followed in this study is clarified. In this study, MSSQL Server 2019 is installed, and a proposed database is designed, implemented, and prepared for the DBFI model phases to be implemented to analyze and determine any internal unauthorized modification that occurred in the database.

Volume 5, Issue 1, pages 1–16
p-ISSN 2655-8564, e-ISSN 2685-9432

In the simulation process, the used DBFI model consists of four stages: identification, artifacts collection, artifacts analysis, and documentation and presentation as shown in Fig. 3 [11]. All stages were taken to obtain valid and admissible evidence.

Figure 3. Database forensic investigation model

At the identification stage, the incident type and nature of the target databases are understood. The techniques and means needed in DBFI are identified, the forensic environment is prepared. The database server is isolated from the production environment and networks [11]. In the process of artifacts collection, all relevant data in the compromised database are collected from the database server. The collected data will be analyzed, and then the evidence will be extracted and determined. All the conducted processes during the three stages are documented including who, where, and when conducted. All the findings and data related to the evidence are documented in a timely manner [11].

4 Experiment

The experiment was conducted by installing MSSQL Server 2019 standard version on a workstation with a windows 10 pro operating system. The experiment database was built on the MSSQL server. A temporal table for customer's information is built in the database with its historical table as shown in Fig. 4.

International Journal of Applied Sciences and Smart Technologies

Volume 5, Issue 1, pages 1–16

p-ISSN 2655-8564, e-ISSN 2685-9432

```
USE [test]
GO

CREATE TABLE dbo.Customers(
    [CustID] int  IDENTITY(1,1) NOT NULL PRIMARY KEY,
    [CustName] nvarchar(50) NOT NULL,
    [Address] nvarchar(50) NULL,
    [Tel] nvarchar(50) NULL,
    [ValidFrom] datetime2 GENERATED ALWAYS AS ROW START NOT NULL,
    [ValidTo] datetime2 GENERATED ALWAYS AS ROW END NOT NULL,
    PERIOD FOR SYSTEM_TIME ([ValidFrom], [ValidTo])
    )
WITH (
SYSTEM_VERSIONING = ON ( HISTORY_TABLE = dbo.Temporal_Customers_History )
)
GO
```

Figure 4. Create a Customer Temporal Table with its Historical

The extended events session created and the data that need to be collected are identified. The created session is configured to be saved to a file automatically as shown in Fig. 5.

After the extended events session started, a number of rows were inserted into the customer table. SQL quires executed to show the changes in temporal and historical tables as shown in Fig. 6. The results of the SQL query shown in Fig. 7 and Fig. 8. Then using another client computer, many rows were inserted, modified, and deleted. After these transactions on the table, the temporal and historical tables data are viewed to show the impact of the insert, update, and delete transactions on the historical table. The extended events session is viewed and the events reviewed to check the data are collected during the execution of the transaction on the table.

International Journal of Applied Sciences and Smart Technologies

Volume 5, Issue 1, pages 1–16

p-ISSN 2655-8564, e-ISSN 2685-9432

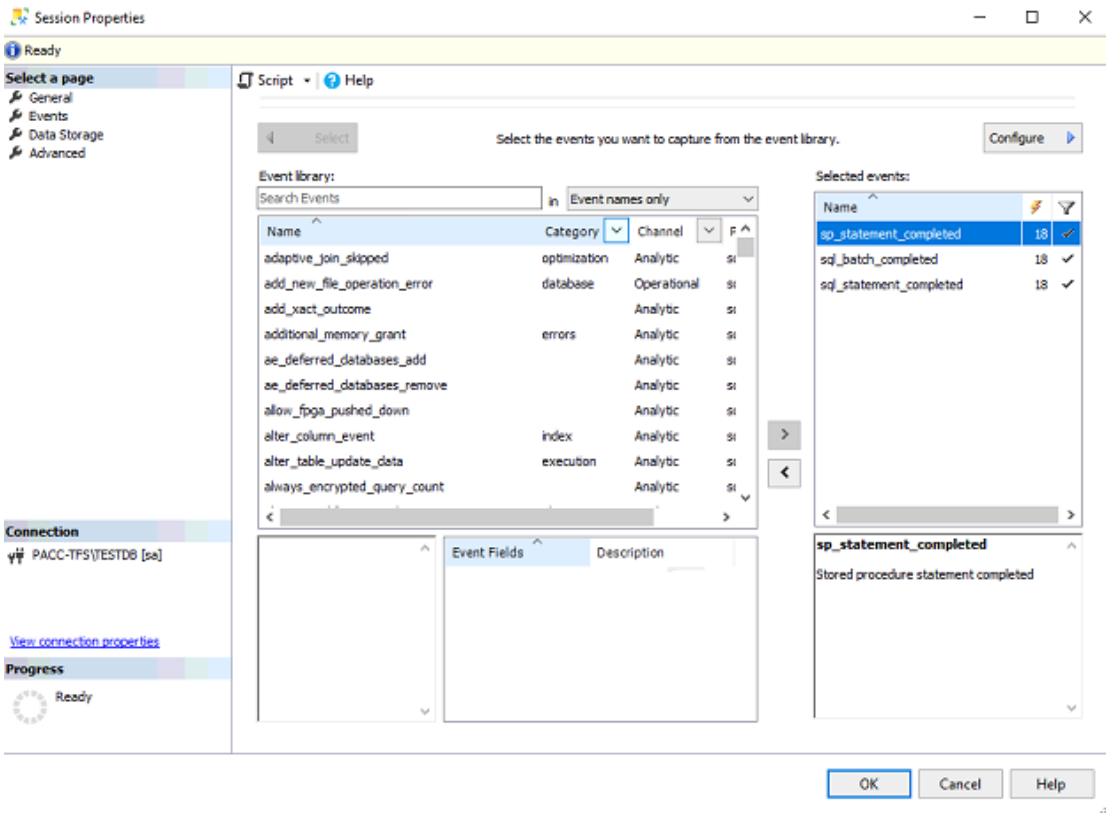


Figure 5. Extended events session properties

```
SELECT  CustID
      ,CustName
      ,Address
      ,Tel
      ,ValidFrom
      ,ValidTo
  FROM Test.dbo.Customers
SELECT  CustID
      ,CustName
      ,Address
      ,Tel
      ,ValidFrom
      ,ValidTo
  FROM Test.dbo.Customers
  FOR SYSTEM_TIME BETWEEN '2021-06-20 19:02:43.9992811' AND '2021-06-24 19:02:43.9992811'
```

Figure 6. SQL queries to retrieve data from the temporal and historical tables

International Journal of Applied Sciences and Smart Technologies

Volume 5, Issue 1, pages 1–16

p-ISSN 2655-8564, e-ISSN 2685-9432

	CustID	CustName	Address	Tel	ValidFrom	ValidTo
1	1	Customer1	France	0101235467	2021-06-23 16:19:55.5971923	9999-12-31 23:59:59.9999999
2	2	Customer2	Jermany	0201234568	2021-06-23 16:24:34.1475024	9999-12-31 23:59:59.9999999
3	3	Customer3	Italy	030456231	2021-06-23 16:24:34.1943972	9999-12-31 23:59:59.9999999
4	4	Customer4	Sweden	0405632145	2021-06-23 16:24:34.2256511	9999-12-31 23:59:59.9999999
5	5	Customer5	Spain	0705633345	2021-06-23 16:24:34.2568903	9999-12-31 23:59:59.9999999

Figure 7. Retrieved Data from Temporal Table

	CustID	CustName	Address	Tel	ValidFrom	ValidTo
1	1	Customer1	France	0101235467	2021-06-23 16:19:55.5971923	9999-12-31 23:59:59.9999999
2	2	Customer2	Jermany	0201234568	2021-06-23 16:24:34.1475024	9999-12-31 23:59:59.9999999
3	3	Customer3	Italy	030456231	2021-06-23 16:24:34.1943972	9999-12-31 23:59:59.9999999
4	4	Customer4	Sweden	0405632145	2021-06-23 16:24:34.2256511	9999-12-31 23:59:59.9999999
5	5	Customer5	Spain	0705633345	2021-06-23 16:24:34.2568903	9999-12-31 23:59:59.9999999

Figure 8. Retrieved data from Historical Table

According to a suspected unauthorized modification on the database, a comparison between the current data in the customer temporal table and its historical table is conducted. By comparison, the historical data and current data are viewed and rows dates, and times are collected. The extended events collected data are reviewed due to the collected date and time from the historical table. From the extended events collected data, the executed transaction is determined with its date and time, the number of affected rows, the user who has performed the modification, the client machine, and executed the SQL statement text, etc.

5 Results and discussion

The results of the study that was conducted successfully in collecting the temporal and historical table's data and extended events logs artifacts to determine the unauthorized modification evidence.

According to a suspected unauthorized modification on the database, the DBFI model was used to collect the artifacts and determine the evidence. At the identification stage, the MSSQL server version and the tampered database is determined, the accident information is identified, and the Database server is isolated. The tampered database and

Volume 5, Issue 1, pages 1–16

p-ISSN 2655-8564, e-ISSN 2685-9432

the extended events files were collected and transferred to the Database forensic workstation in the artifacts collection stage.

At the artifacts analysis stage, the database is attached to the MSSQL server in the database forensic workstation. The current data from the customer temporal table are viewed using a SQL select statement, and the historical table data are viewed using a SQL select statement with a system time period condition to return the historical data as shown in Figs. 9 - 11.

```
SELECT  CustID
      ,CustName
      ,Address
      ,Tel
      ,ValidFrom
      ,ValidTo
  FROM Test.dbo.Customers
SELECT  CustID
      ,CustName
      ,Address
      ,Tel
      ,ValidFrom
      ,ValidTo
  FROM Test.dbo.Customers
  FOR SYSTEM_TIME BETWEEN '2021-06-20 19:02:43.9992811' AND '2021-06-24 19:02:43.9992811'
  Order BY CustID ASC
```

Figure 9. SQL queries to retrieve data from the temporal and historical tables

	CustID	CustName	Address	Tel	ValidFrom	ValidTo
1	1	Customer 1	France	0101235467	2021-06-23 16:19:55.5971923	9999-12-31 23:59:59.9999999
2	2	Customer2	Jermany	0201234568	2021-06-23 16:24:34.1475024	9999-12-31 23:59:59.9999999
3	3	Customer 100	Italy	030456231	2021-06-23 16:40:53.8654999	9999-12-31 23:59:59.9999999
4	5	Customer5	Spain	0705633345	2021-06-23 16:24:34.2568903	9999-12-31 23:59:59.9999999
5	6	Customer99	UK	0101235467	2021-06-23 16:39:41.7544971	9999-12-31 23:59:59.9999999

Figure 10. Retrieved Data from Temporal Table

	CustID	CustName	Address	Tel	ValidFrom	ValidTo
1	1	Customer 1	France	0101235467	2021-06-23 16:19:55.5971923	9999-12-31 23:59:59.9999999
2	2	Customer2	Jermany	0201234568	2021-06-23 16:24:34.1475024	9999-12-31 23:59:59.9999999
3	3	Customer 100	Italy	030456231	2021-06-23 16:40:53.8654999	9999-12-31 23:59:59.9999999
4	3	Customer3	Italy	030456231	2021-06-23 16:24:34.1943972	2021-06-23 16:40:53.8654999
5	4	Customer4	Sweden	0405632145	2021-06-23 16:24:34.2256511	2021-06-23 16:41:30.4440715
6	5	Customer5	Spain	0705633345	2021-06-23 16:24:34.2568903	9999-12-31 23:59:59.9999999
7	6	Customer99	UK	0101235467	2021-06-23 16:39:41.7544971	9999-12-31 23:59:59.9999999

Figure 11. Retrieved data from Historical Table

Volume 5, Issue 1, pages 1–16
p-ISSN 2655-8564, e-ISSN 2685-9432

By comparing the temporal table with the historical table, the investigator found the differences between the data in the two tables. The first difference was due to the CustID field with the value 3 repeated in the historical table with different CustName field Values and different ValidFrom and ValidTo fields Values, which means there is a modification executed on this row. In addition, the second difference was the CustID field value 4 Found in the historical table and not presented in the temporal table which means the row was deleted.

According to the determined modifications and deletion that were presented by viewing the temporal and historical data, the extended events session file was analyzed due to the system dates from the historical table. In the analysis stage, the SQL statement completed events determined and viewed. The collected events details include the client application name, the client hostname, the affected database name, number of affected rows, the database server instance name, the user name who modified the data, the used database user name, and the executed SQL statement text as shown in Fig. 12.

As shown in Fig. 12, the modification happened from client host named “SHADI-PC”, the user name that do the modification was “PACC\shzakarneh”, and the affected database name was “test”.

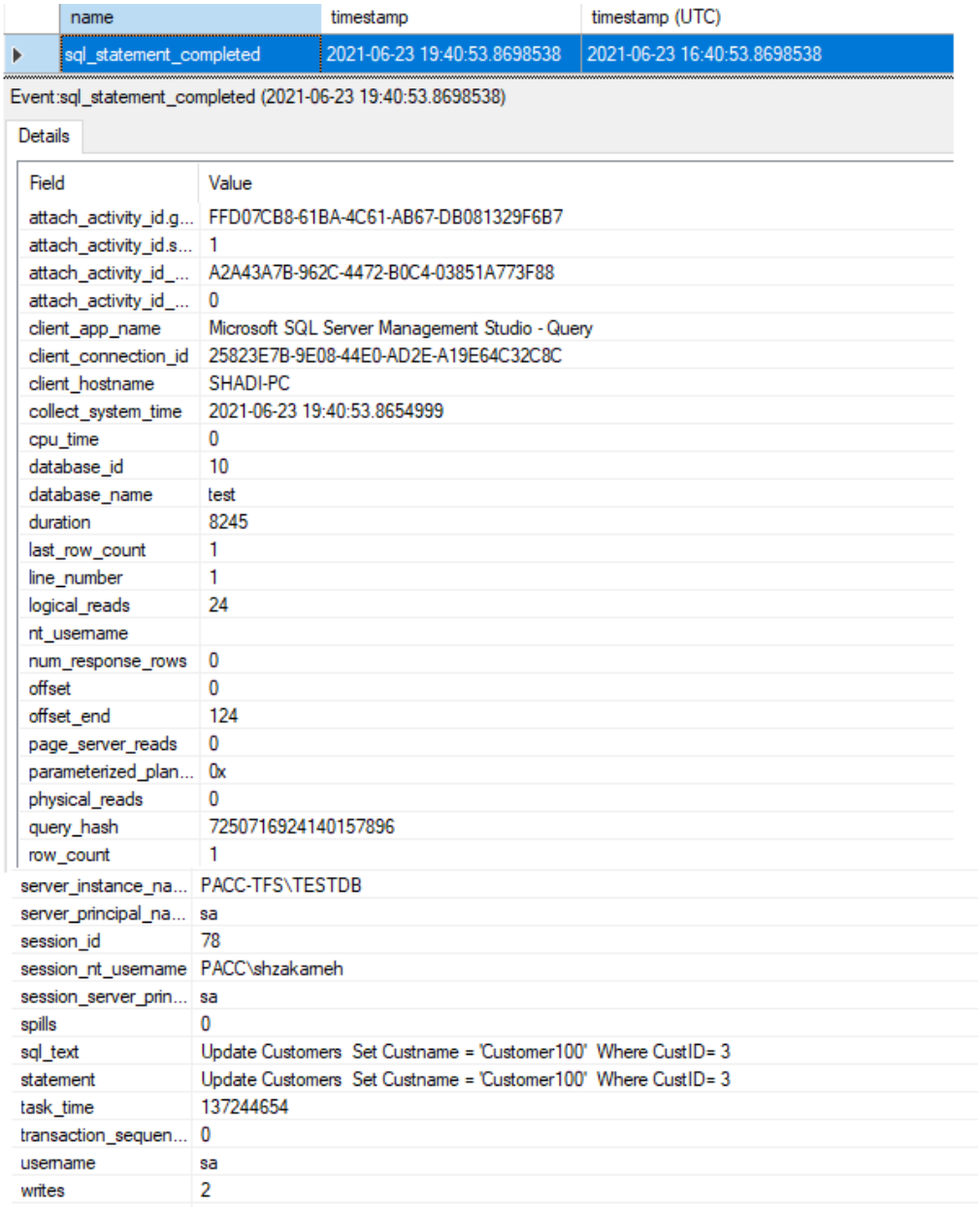
While the temporal table store the current data, the historical table store the historical data includes the current rows, deleted rows, and the old and modified rows in addition to the two system times columns that are indicated to the created date and time of the rows. By comparing the two tables, any suspected modification will be determined. Table

In addition, by linkage the date and time column from the historical data with a timestamp in the extended events session, the evidence will be extracted and determined from the details of the event.

International Journal of Applied Sciences and Smart Technologies

Volume 5, Issue 1, pages 1–16

p-ISSN 2655-8564, e-ISSN 2685-9432



name	timestamp	timestamp (UTC)
sql_statement_completed	2021-06-23 19:40:53.8698538	2021-06-23 16:40:53.8698538

Event:sql_statement_completed (2021-06-23 19:40:53.8698538)

Details

Field	Value
attach_activity_id.g...	FFD07CB8-61BA-4C61-AB67-DB081329F6B7
attach_activity_id.s...	1
attach_activity_id_...	A2A43A7B-962C-4472-B0C4-03851A773F88
attach_activity_id_...	0
client_app_name	Microsoft SQL Server Management Studio - Query
client_connection_id	25823E7B-9E08-44E0-AD2E-A19E64C32C8C
client_hostname	SHADI-PC
collect_system_time	2021-06-23 19:40:53.8654999
cpu_time	0
database_id	10
database_name	test
duration	8245
last_row_count	1
line_number	1
logical_reads	24
nt_username	
num_response_rows	0
offset	0
offset_end	124
page_server_reads	0
parameterized_plan...	0x
physical_reads	0
query_hash	7250716924140157896
row_count	1
server_instance_na...	PACC-TFS\TESTDB
server_principal_na...	sa
session_id	78
session_nt_username	PACC\shzakameh
session_server_prin...	sa
spills	0
sql_text	Update Customers Set Custname = 'Customer100' Where CustID= 3
statement	Update Customers Set Custname = 'Customer100' Where CustID= 3
task_time	137244654
transaction_sequen...	0
username	sa
writes	2

Figure 12. Extended Events Session Details

6 Conclusion

The database is used to store data for businesses and individuals. MSSQL server one of the most popular DBMSs in the world used for small and large data. With the increased use of the database, the DBFI becoming more important to determine the evidence and data recovery. Different techniques and models were used in the DBFI using

Volume 5, Issue 1, pages 1–16
p-ISSN 2655-8564, e-ISSN 2685-9432

transaction log files and data file recovery. This study focused on DBFI using temporal tables and extended events features in MSSQL server new version as new artifacts. The study experiment was conducted using a DBFI model in MSSQL server 2019 with the four stages, identification, artifacts collection, artifacts analysis, and documentation and presentation. In the study results, the evidence was extracted and determined using the temporal tables and extended events artifacts.

References

[1] P. A. Carter, Pro SQL Server 2019 Administration, (2019).

[2] D. Korotkevitch, Pro SQL Server Internals, (2016).

[3] L. Davidson, Pro SQL Server Relational Database Design and Implementation, (2021).

[4] B. Ward, SQL Server 2019 on Linux, (2019).

[5] B. Nevarez, Performance SQL Server, (2021).

[6] A. P. Jamdar, M. B. Bhangire, S. G. Shahari, and K. G. Matere, an Efficient Framework for Database Forensic Analysis., Int. J. Adv. Eng. Res. Dev., 4 (5) (2017) 12634–12637

[7] M. Kamaruzzaman, "Top 10 Databases to Use in 2021," Towards Data Science, https://towardsdatascience.com/top-10-databases-to-use-in-2021-d7e6a85402ba (accessed Jun. 11, 2021), (2021).

[8] Ghanayem, Mark, W. Rohm, and J. Parente, Temporal Tables, Microsoft. https://docs.microsoft.com/en-us/sql/relational-databases/tables/temporal-tables?view=sql-server-ver15 (accessed Jun. 17, 2021). (2016).

[9] P. Jayaram, Temporal Tables in SQL Server, SQL Shake, https://www.sqlshack.com/temporal-tables-in-sql-server/, (2019).

[10] S. Johnson, Introducing SQL Server, (2015).

[11] A. Al-Dhaqm et al., Database forensic investigation process models: A review, IEEE Access, 8 (2020) 48477–48490.

[12] R. Bria, A. Retnowardhani, and D. N. Utama, Five Stages of Database Forensic Analysis: A Systematic Literature Review, Proc. 2018 Int. Conf. Inf. Manag.

Volume 5, Issue 1, pages 1–16

p-ISSN 2655-8564, e-ISSN 2685-9432

Technol. ICIMTech 2018, no. September, (2018) 246–250,

[13] B. Narwal, A Walkthrough of Digital Forensics and its Tools, March (2020) 13757–13764, 2020.

[14] J. Wagner, A. Rasin, K. Heart, T. Malik, J. Furst, and J. Grier, Detecting database file tampering through page carving, Adv. Database Technol. - EDBT, 2018-March, (2018) 121–132.

[15] H. Choi, S. Lee, and D. Jeong, "Forensic Recovery of SQL Server Database: Practical Approach," IEEE Access, 9 (2021) 14564–14575.

[16] R. Jason, A. Wolter, and M. MSFT, Extended events overview, Microsoft, https://docs.microsoft.com/en-us/sql/relational-databases/extended-events/extended-events?view=sql-server-ver15, (2019).

[17] A. Al-Dhaqm, S. Razak, S. H. Othman, A. Ngadi, M. N. Ahmed, and A. A. Mohammed, Development and validation of a database forensic metamodel (DBFM), 12(2) (2017).

[18] V. K. Gupta, J. Bonde, A. Gorad, and P. Joshi, Duel Security-Detection of Database Modification Attack and Restore Facility from Unauthorized Access, 7(6) (2020) 983–987.