Proposed framework for enhancing integrity technique using distributed query operation

Umar Farooq Khattak[1*] , *Aıtızaz* Alı[2] , *Hussein*[3] and *Ali* Hussein Al Naffakh[3]

[1]UNITAR International University, Malaysia, umar.farooq@unitar.my[1*]
[2]Asia Pacific University, Malayisa, Aitizaz.ali@apu.edu.my
[3]College of Health and Medical Techniques, University of Alkafeel, AlNajaf, Iraq,
nidhal.elabbadi@uokufa.edu.iq

Abstract. Database growth and Storage problems are main concern for large and small enterprises nowadays, which significantly influences the efficiency of database applications. Database archiving is one of the solutions available for management. Many issues have been identified as a result of the use of archive databases, including the elimination of inactive data from online transaction processing systems (OLTP), performance and integrity management. The aim of this paper is to propose a framework for storing OLTP and archiving data in a distributed database environment as part of an integrated system. Maintaining the integrity between an OLTP database and an archiving database is crucial, but even more critical is the performance of the required query. The main aspect of the proposed framework is that it will not only ensure data integrity for primary and unique keys between OLTP and archive database, but it will also improve query performance by introducing parallel processing and query execution plans to maintain that integrity.

1 Introduction

In the contemporary communication landscape, there is a growing trend of administrations storing increasing amounts of data in Online Transaction Processing (OLTP) systems for various reasons. Studies have indicated that this data is routinely processed and analyzed for analytical insights. Past occurrences have shown that larger databases have a significant impact on system applications, especially concerning loading and unloading processes. The performance of a database application, a critical aspect, is notably affected by the existence of a large volume of unused data within it. [1]. Database archiving is a subset of data archiving,

* Corresponding author: *umar.farooq@unitar.my*

encompassing a wide array of data types and functions. Although data exists in numerous formats and fulfills various roles, only a fraction is stored within databases. This includes documents in physical and digital forms, computer files, datasets, emails, and various file types. Ultimately, all such data necessitates archiving. [33]. Removal of selected data records from operational database which are not expected to be indicated or needed and storing then in an off line Data store is called Archive Database. By removal of data means that data deleted from operational Database when it is moved to data archive. After data is moved into archive stage, query and access to that data is not possible [28].

The primary objective of an archive database is to eliminate data records from OLTP databases that are unlikely to be referenced or used actively, storing them offline. Once archived, datasets are transferred to the archive database and are no longer accessible in the operational database. The responsibility of managing the integrity and accessibility of these database archives lies within the realm of database archiving, a process that shifts database information from the live production database to dedicated archives [4]. It is crucial to recognize the resemblance between database archiving and information archiving. Information Lifecycle Management (ILM) plays a pivotal role in treating data differently at various stages of its lifecycle. Database archiving, specifically tailored for structured data rather than flat files, serves as a practical means to implement ILM functions on databases. In essence, it efficiently navigates through different phases of data lifecycle, ensuring optimal handling and storage within the database archives [32].

In this context, the concept of an archive database sets it apart from a backup database as it involves the removal of novel data from the database, leaving the archive copy preserved for as long as necessary [3]. The retention period for archived data has gradually extended, with some productions opting for storage durations of up to 30 years. Given the rapid evolution of data and technology, merely increasing storage capacity proves inadequate to effectively manage the expanding volume of data [4]. In the current era, characterized by evolving data trends emphasizing speed, size, and mobility, organizations must ensure that their technology and software align with the evolving needs of their users. Creating a database stands as the initial step toward achieving these objectives [3].

Beyond the improvement in operations, the expansion of data storage brings about a range of adverse consequences, such as diminished system performance, prolonged response times, and heightened levels of complexity. This not only erodes the trust of loyal customers but also hinders the potential for generating new income opportunities. Hence, it is imperative to possess a dependable and integrated database capable of efficiently handling the escalating volume of data, which is doubling on an annual basis [5].

Numerous businesses leverage archived data to extract valuable insights about their customers and market trends, employing diverse data mining techniques for this purpose [8]. In an age characterized by an overwhelming influx of information, implementing database archiving emerges as a crucial and indispensable strategy for businesses.

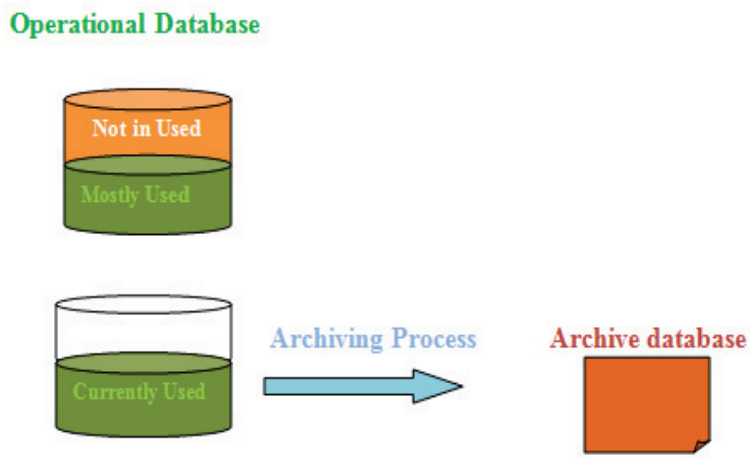

Fig .1. Operational and Archive database

Maintaining data integrity between the offline archive and OLTP database can be achieved through various methods. A conventional approach involves scrutinizing both the online database and offline archive database to eliminate data redundancy during update and insert operations. This implies that Data Manipulation Language (DML) operations not only verify the OLTP for inserting or updating a record but also undergo the same checking procedure in the offline archive database. This process, depicted in figure 2, introduces a time-consuming aspect to DML operations, making the insertion and updating procedures take longer compared to straightforward queries.

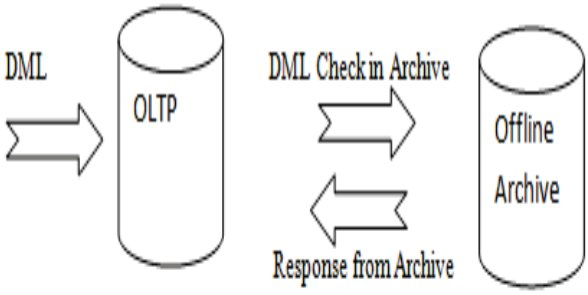

Fig .2. General Architecture for ensuring integrity

An alternative method proposed in the "Integrity Management between the Archive and OLTP Systems" involves preserving data integrity by extracting Primary and Unique keys from the active Offline Database Archive. Subsequently, these keys are inserted into two newly established tables within the OLTP system, each

designated for a specific key type (Primary key and Unique key), as illustrated in figure 3.

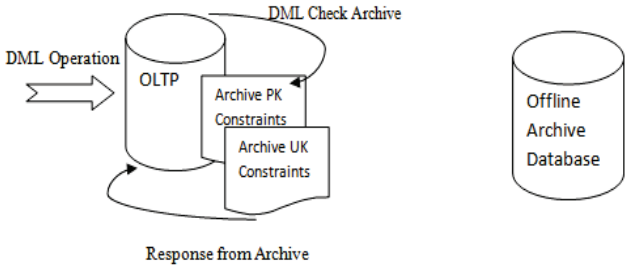

Fig .3. Integrity Management between Archive and OLTP systems

2 Related Work

Lifecycle Management adopts a comprehensive "best practice" strategy for managing database growth in the long run, and Database Archiving is a pivotal element of this approach[7]. In the mid-nineties, concerns arose regarding the extended preservation of digital assets for organizations, government agencies, scientific communities, and individual researchers. Various studies conducted during this period highlighted significant issues that required resolution to ensure reliable and efficient access to digital information over an extended duration. While maintaining data integrity is the primary objective, one of the challenges identified in these studies pertains to the performance of the Database Management System (DBMS) in guaranteeing data integrity between the Online Transaction Processing (OLTP) and Archive databases [1].

Forrester's estimates indicate that the contemporary database market is valued at $27 billion, covering new database licenses, technical support, facilities, and access. Furthermore, there is a projected increase to $32 billion in the coming years [4]. The database market is evidently expanding over time. However, the corresponding growth in data has significantly impacted performance, notably causing a considerable slowdown in query run time a critical consideration for the industry.

Analysts report an annual growth rate of 125% for enterprise databases, with a noteworthy observation that 80% of the data within these databases remains unused and is, therefore, ripe for archiving [9]. The question arises: why is there such an immense production of data? While technological advancements allow for diverse methods of capturing and storing data, it's essential to recognize that the rapid growth of data cannot be solely attributed to technology [10].

Annually, organizations invest significant financial resources in maintaining and updating business-critical applications that rely on intricate relational databases [26]. These databases play a crucial role in storing substantial volumes of data essential for business operations and decision-making processes. Consequently, databases often experience overload, leading to subpar performance and constraints on the availability of desired functionalities. Ironically, a substantial portion of this data remains online in production databases but is rarely accessed [26].

During August 2007, a media-conducted survey on behalf of Bridgehead Software targeted IT executives, disclosing that 59 percent of respondents encountered issues where the size of the data they were obligated to back up was causing disruption or had the potential to disrupt business activities [35]. Furthermore, 93% of the surveyed individuals reported a consistent growth in the size of routine backups. The survey outlined various advantages associated with minimizing the volume of data regularly subjected to backup processes:

• Decreased time allocated by IT for backup and other business operations (69 percent)

• Lesser impact of backup and replication on network utilization and capacity (60 percent)

• Lower allocation of disk resources for data snapshots, replication, and mirroring (58 percent)

• Minimized disturbance to the live application environment (45 percent) [31]

The majority of business operations are obligated to retain and store data for extended periods within operational databases. However, this practice can lead to a more intricate setup concerning database performance and usability, especially as operational databases continue to expand. This growth places additional strain on transaction processing, resulting in suboptimal query performance. As a general rule, the efficiency of transactions running against older dates in the Online Transaction Processing (OLTP) system tends to decrease. Modern industry analysts and professionals advocate for utilizing a record database as the solution to handle substantial data volumes within a database system. Acknowledging the significance of data integrity in database systems, which guarantees the precision, coherence, and dependability of data, the paramount principle is to maintain data integrity when implementing any archival database approach. [35].

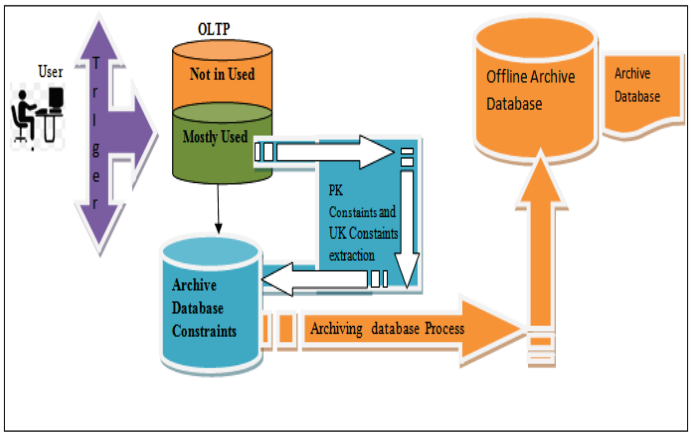

Fig .4. Database Archiving Process over Distributed database environment

3 Proposed Framework

The initial step in the suggested methodology involves archiving data from the Online Transaction Processing (OLTP) system that is deemed unnecessary for an organization. The quantity of data archived is contingent upon the organization's policy, which varies across different entities. Some organizations opt to archive data after one year, while others follow a six-year timeframe. Generally, data archiving policies are time-based, measured in years, rather than randomly selecting records within OLTP systems. Alternatively, certain organizations define their archiving policies based on the number of records. For instance, an organization might choose to archive ten thousand records every six months as exemplified in the policy.

A distributed database refers to a collection of data spread across various sites within a network, where each site possesses independent processing capabilities and the ability to execute local applications. In this network, each site responds to sub-queries of a global query through a communication subsystem. The enhancement of operational processing is achieved through parallel processing, allowing the simultaneous execution of various operations such as index building, data loading, and query evaluation. Despite the distribution of data across different sites, the performance is optimized by adhering to distribution rules. Managing increased database sizes and meeting performance demands becomes more manageable within a parallel processing environment. The distributed database environment, as illustrated in figure 4, clearly delineates the distribution of the database for parallel processing.

The proposed framework involves executing queries through parallel processing. This approach ensures integrity in the Online Transaction Processing (OLTP) system on one side while simultaneously managing integrity in the archive database within a distributed environment in the event of an insert or update query.

.

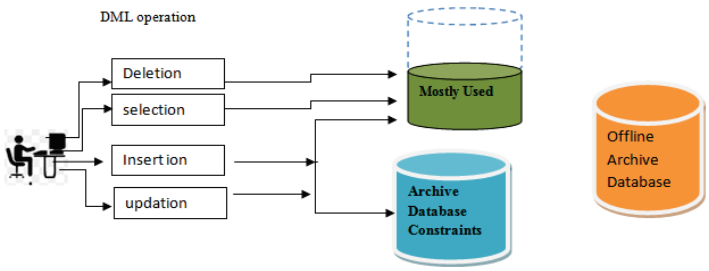

Fig .5. Parallel DML Operation over Distributed database environment for integrity Management

A Database level trigger will be established to detect insertion and update Data Manipulation Language (DML) operations for each table. Additionally, an Information schema view will be defined to identify table-specific details such as table name, value, value type, constraint type, etc. Following the identification of these parameters, query execution plans will be executed to manage integrity effectively.

The primary advantage of this framework lies in the query execution plan, outlining the path for integrity management checks. Additionally, the integrity management checks between the Online Transaction Processing (OLTP) and Archive

database will occur concurrently, enhancing the performance of Data Manipulation Language (DML) operations in terms of integrity checking. Accessing archived database data is expedited, eliminating the need for an extensive restoration procedure compared to archiving databases.

3.1 Data Archiving Process & Storing of Integrity Constraints

Archiving data from the Online Transaction Processing (OLTP) system, which is no longer necessary for an organization, constitutes the initial phase of the proposed architecture. To anticipate the potential need for archived data in the future, the proposed architecture preserves archived data across a distributed environment without constraints, ensuring accessibility. Some organizations adopt a data archiving policy after one year, while others follow a six-year timeframe. Generally, data archiving policies are time-based, measured in years, rather than selecting records randomly within the OLTP system. Alternatively, some organizations base their archiving policies on the predefined number of records. For instance, an organization archiving ten thousand records every six months exemplifies the aforementioned policy.

3.2 Parallel Processing in Distributed database

A distributed database is an assemblage of data dispersed across various computers within a computer network. Each site in the network possesses independent processing capabilities and the capacity to execute local applications. Furthermore, each site actively engages in at least one global application, necessitating data access across multiple sites facilitated by a communication subsystem. Parallel processing aims to enhance performance through the parallelization of diverse operations such as index building, data loading, and query evaluation. While data may be stored in a distributed manner in such a system, the distribution is primarily governed by performance considerations. Managing the escalation of database sizes and meeting increasing performance demands becomes more feasible within a parallel processing environment. Significant system overhauls are infrequently required, with expansions typically addressed by augmenting processing and storage capabilities. Given that Distributed and parallel Database Management Systems (DBMSs) incorporate replicated components, they are designed to enhance reliability by eliminating single points of failure.

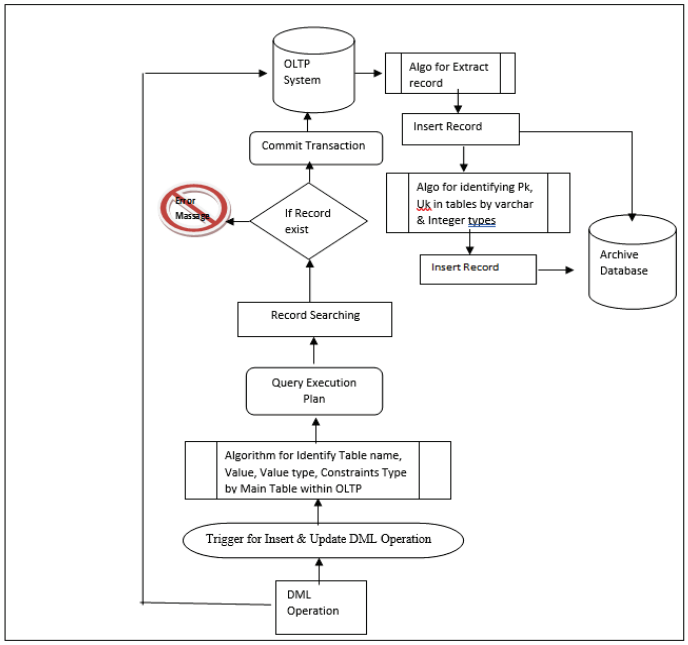

Fig .6. Flow Chart of Proposed Framework

In the proposed system, query execution will be conducted through parallel processing. This approach ensures integrity in the Online Transaction Processing (OLTP) system on one side while simultaneously managing integrity in the archive database within a distributed environment in the event of an insert or update query.

3.3 Ensuring Integrity management between Archive and OLTP system

Database integrity is synonymous with the accuracy and consistency of data, representing an essential facet of database protection. Security, on the other hand, pertains to safeguarding data from unauthorized operations. The concept of integrity is closely tied to the quality of data and is upheld through the implementation of integrity constraints. Ensuring data integrity is crucial, as it guarantees that the data is of superior quality, accurate, consistent, and readily accessible for the intended end-users of the database. Thus, achieving data integrity is of paramount importance.

Within the proposed system, data integrity is upheld through a query execution plan that systematically checks the integrity in the archive database of the distributed system in parallel with the Online Transaction Processing (OLTP) system. The initial trigger is activated upon the occurrence of an insert or update query in the Database Management System (DBMS). Following the determination of the insert or update query, the contents are scrutinized against the Information Schema to identify column names, constraint types, and data types.

Upon detecting the column name, constraint type, and data type, a query execution plan is executed based on the specified constraint type and data type for the particular column. If the constraint type is a Primary key and the data type is varchar, integrity management is exclusively checked in the PrimaryKeys table.

Alternatively, if the constraint is a Unique key and the data type is varchar, data integrity is verified in the UniqueKeys table in the archive database within a distributed environment.

In cases where the data type is integer in the desired constraint (Primary or Unique key), integrity management is examined in the Local Monitor within the OLTP system. If the required item is identified in the LM Table within the OLTP system, a trigger is activated to determine a specific archive in the AAM table where the given item is present. Subsequently, it checks the ADM table for the duplication of that particular item once the specific archive is found in the AAM table. The transaction is terminated in case of item detection in these specified tables. The flowchart illustrating the query execution plan is depicted in Figure 6.

4 Conclusion

It is evident that substantial database volumes can detrimentally affect database efficiency. Database applications experience prolonged durations for tasks such as loading, unloading, searching, retrieving, reorganizing, recovering, and backing up the requisite database. Various challenges linked to data archiving, such as integrity management, query performance, and the accessibility of archived data, have been brought to the forefront. While the primary focus in archiving databases is on integrity management, the significance of query performance in maintaining that integrity cannot be overstated.

The chosen approach prioritizes ensuring the integrity of primary and unique keys between Online Transaction Processing (OLTP) and archive databases within a distributed environment. It also guarantees swift query performance to uphold that integrity through the initiation of a query execution plan. On one front, the query execution plan establishes a direct path for checking integrity, while concurrently, integrity is verified between OLTP and Archive databases over a distributed setting. The proposed framework facilitates data availability by maintaining both archive data and OLTP in a distributed environment, eliminating the need for extensive restoration procedures.

5 Future Research Dırectıon

The approach presented offers ample opportunities for expansion. Beyond the focus on primary and unique keys, this concept can be extended to incorporate referential integrity. Exploring a new approach becomes imperative for effectively maintaining referential integrity between the Archive and Online Transaction Processing (OLTP) systems. Another crucial avenue for exploration involves the ongoing synchronization of schemas between the archive and OLTP systems. Ensuring that any changes in the OLTP schema are mirrored in the archive database schema represents a vital new direction.

References

1. Hamidah Ibrahim , Ali Amer Alwan, Nur Izura Udzir, Checking Integrity Constraints with Various Types of Integrity Tests for Distributed databases.

2. Vinita Gupta, Databases for mission critical apps.
3. Noel Yuhanna, Enterprise Database Management Systems, June 30, 2009
4. Vinita Gupta, Databases for mission critical apps.
5. Charles P. Pfleeger and Shari Lawrence Pfleeger, Security in Computing, Prentice Hall Professional Technical Reference, Upper Saddle River,New Jersey, 2003.
6. Raghu Ramakrishnan, Johannes Gehrke, Database Management System 2nd Edition
7. Noel Yuhanna, Enterprise Database Management Systems, June 30, 2009
8. Rick Noel, "Scale Up in Distributed Databases: A Key Design Goal for Distributed System,"
9. Craig S. Mullins, Database Archiving for Long-Term Data Retention, October 1, 2006
10. Thomas C. Redman, The impact of poor data quality on the typical enterprise. ACM, 2:79-82, 1998.
11. L.English. Plain English on data quality, information quality management: The next frontier. DM Review Magazine, April 2000.
12. R. Sandhu and S. Jajodia , Integrity mechanisms in database management system .In NIST-NCSC National Computer Security Conference, 1990.
13. Phathisile Sibanda, A Comparative Investigation and Evaluation of Oracle 9i and SQL Server 2000 with respect to Performance and Scalability, November 2005.
14. *Trevor Eddolls,* Database Archiving for the Future, June 12, 2008.
15. Noel Yuhanna, Mike Gilpin, Katie Smillie ,Database Archiving Remains An Important Part Of Enterprise DBMS Strategy, August 13, 2007 .
16. Stephanie Balaouras, Barry Murphy Simon Yates, Rachel Batiancila , Backup Versus Archiving: Firms Need Separate Strategies For Each, November 17, 2006
17. Bernard Liautaud, Chairman and CSO, Business Objects, Data Integrity is a Universal Issue October, 2008.
18. Andy Bitterer, Analyst, Gartner, Data Integrity is a Universal Issue October, 2008.
19. Mike Pratt, Data Integrity Manager, Business Link London (BLL), Data Integrity is a Universal Issue October, 2008
20. Sergio Flesca, Sergio Greco and Ester Zumpano, Active Integrity Constraints.
21. Charles P. Pfleeger and Shari Lawrence Pfleeger, Security in Computing, Prentice Hall Professional Technical Reference, Upper Saddle River, New Jersey, 2003.
22. Ovita Barretto, "Security Issues with Distributed Databases," http://students.depaul.edu/~obarratt/Security.htm, May 20, 2004.
23. Jhan JD, Shajio, Components of a Distributed Database System, http://www.fi/~hhyotyni/latex/Final/node44.html, May 24, 2004.

24. Eirikdahle , Helgeberg , DIBAS – A management system for distributed databases

25. M. Tamer O¨ zsu, Patrick Valduriez, Distributed and Parallel Database Systems

26. M. Tamer O¨ zsu, Patrick Valduriez, Distributed and Parallel Database Systems

27. Craig S. Mullins, Database Archiving for Long-Term Data Retention, October 1, 2006

28. Jim Lee, Reduce the cost of compliance: database archiving and Information Lifecycle Management Storage Management, Computer Technology Review, Dec 2003.

29. Database Archiving for Long- Term Data Retention Periods, by Craig S. Mullins, Published in www.TDAN.com October 2006.

30. Jack Olson, Database Archiving Basics, a New Function for Improving Data Management, Information Management Special Reports, September 22, 2009.

31. Database Archiving: Managing Data for Long Retention Periods, by Craig S. Mullins, Published in www.TDAN.com October 2006.

32. James Lee, Database Archiving: A Critical Component of Information Lifecycle Management, April 21, 2004.

33. David Hill, Database Archiving: A Necessity, Not an Option, April 28, 2006

34. *Trevor Eddolls,* Database Archiving for the Future, June 12, 2008

35. Khattak, U. F., Mustapha, A., Yaseen, M., Shah, M. A., & Shahzad, A. (2019). Enhancing Integrity Technique Using Distributed Query Operation. Recent Trends and Advances in Wireless and IoT-enabled Networks, 139-146.

36. Talpur, N., Abdulkadir, S. J., Akhir, E. A. P., Hasan, M. H., Alhussian, H., & Abdullah, M. H. A. (2023). A novel bitwise arithmetic optimization algorithm for the rule base optimization of deep neuro-fuzzy system. Journal of King Saud University-Comp

37. Abdullah, M. H. A., Aziz, N., Abdulkadir, S. J., Alhussian, H. S. A., & Talpur, N. (2023). Systematic Literature Review of Information Extraction From Textual Data: Recent Methods, Applications, Trends, and Challenges. IEEE Access. uter and Information Sciences, 35(2), 821-84.