EXPLORATIONS ON ENGINEERING LETTERS (EEL)
VOLUME 1, ISSUE 1 (2016):PP.279-285
SANA ACADEMIC PRESS

Virtual Denormalization Based on Hybrid OLTP & OLAP Main Memory Database System: Hyper

S.Usha[1], P.Chidambaranathan[2], R. Latha[3]
[1]Research Scholar, St.Peter's University, Chennai.
[2]Asst.Prof. & Head., Dept. of Computer Applications, St.Peter's University, Chennai
[3]Prof. & Head., Dept. of Computer Applications, St.Peter's University, Chennai
ushashiva12@gmail.com

Abstract*— *Hyper is an advance database which will reduce the limitation of traditional database; Hyper is a main-memory-based relational DBMS for mixed OLTP and OLAP workloads A customer having critical transaction with high rates splits their data into two disunited systems, one database for OLTP and one so-called data warehouse for OLAP, this separation has many limitations including data rawness issues due to the delay caused by only periodically initiating the Extract Transform Load-data staging and excessive resource obsess due to maintaining two separate information systems. We present an efficient hybrid system, called Hyper. During parallel execution of OLAP multiple query sessions and OLTP transactions, here the main memory will guarantees the ACID properties. Our research focus on the virtual denormalization based system in hybrid main memory database.

Keywords: *OLAP, OLTP, ACID, HyperDB.*

I. INTRODUCTION

Online transaction processing (OLTP) systems are the heart of today's enterprises for daily operation. They provide the applications for the business processes of an enterprise and record all business movements, e.g. sales and purchase orders, production, billing, and payments. For strategic insights to make business decisions for the future and to monitor the performance of their business, enterprises utilize online analytical processing (OLAP) systems. Since OLTP and OLAP present very different demanding for database architectures and transaction throughput is essential for OLTP, they have been separated into different systems.

Our goal was to architect a main-memory database system that can.

- process OLTP transactions at rates of tens or hundreds of thousands per second as efficiently as dedicated OLTP main memory systems such as VoltDB or TimesTen, and, at the same time,
- process OLAP queries on up-to-date snapshots of the transactional data as efficiently as dedicated OLAP main memory DBMS such as MonetDB or TREX.

This challenge is sketched in Figure 1.

The efficient hybrid system architecture, called HyPer, that can handle both OLTP and OLAP simultaneously by using hardware-assisted. HyPer is a main-memory database system that guarantees the ACID properties of OLTP transactions. In particular, we devised logging and backup archiving schemes for atomicity, durability, and fast recovery. In parallel to the OLTP processing, HyPer executes OLAP query sessions (multiple queries) on the same, arbitrarily current and consistent snapshot.

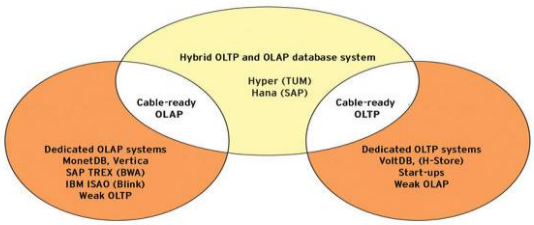

Fig. 1. Hybrid OLTP&OLAP Database Architecture

This snapshot is kept consistent via the implicit OS/processor-controlled lazy copy-on-write mechanism. The utilization of the processor-inherent support for virtual memory management (address translation, caching, copy on update) accomplishes both in the same system and at the same time: unprecedentedly high transaction rates of millions of transactions per minute as high as any OLTP-optimized database system and ultra-low OLAP query response times as low as the best OLAP-optimized column stores. These numbers were achieved on a commodity desktop server. Even the creation of a fresh, transaction-consistent snapshot can be achieved in subseconds.

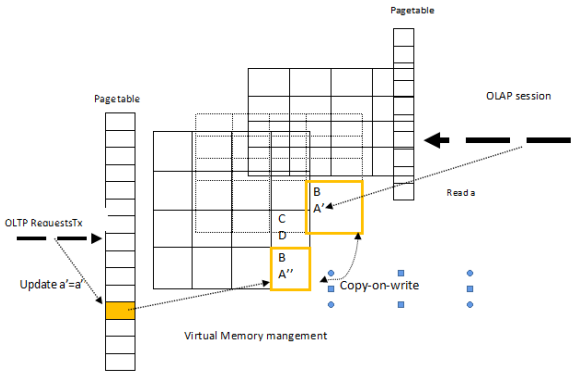

Fig 2. Virtual Memory array table implemented Hybrid OLTP&OLAP Database

The objective of database normalization is to eliminate data redundancy, so as to save storage space and avoid update abnormality. It is usually achieved by decomposing a large relation into several small ones, connected by foreign keys. Star and snowflake schemas are typical forms of normalization, in which data is decomposed into fact tables and dimension tables. Despite its benefits, normalization introduces performance penalty to a DBMS, as it has to process multiple relations separately and perform expensive joins to integrate the intermediate results. Therefore, denormalization is sometimes applied to optimize the performance of a DBMS. Denormalization reverses the process of normalization by joining multiple relations back into one, such that query processing can be conducted on a single table. It simplifies query execution plans and eliminates expensive join operations.

Denormalization does not mean chaos. The development of properly denormalized data structures follows software engineering principles that insure that information will not be lost. If the table is read-only (periodically refreshed from the records) then the rules are looser. Star schemas and hyper-cubes are read-only denormalizations. If the data is to be distributed and/or segmented and added-to, changed, or deleted from then the reconstruction described below must be followed. Fundamentally a single principal must be

followed. If the individual physical table is updated in more than one system, it should be possible to reconstruct the original table as if the data was never reformatted or taken apart.

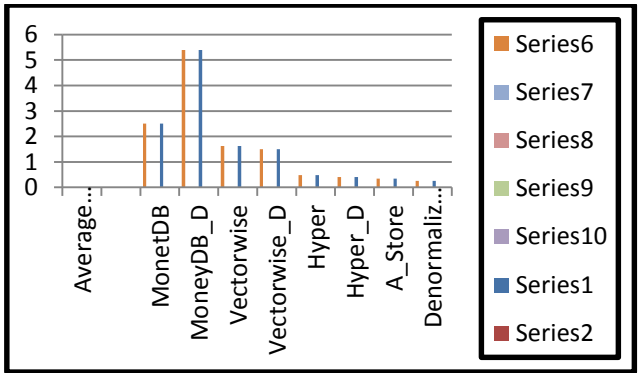

Fig. 3. Denormalization versus normal MMDBs

II. MATERIAL AND METHODS

The word 'denormalization' is beneficial to present changes to the table design that cause the physical tables to differ from the institutionalized quantity relationship diagram.'Denormalization' does not mean that anything goes. Denormalization does not mean chaos. The growth of properly denormalized data structures follows software engineering principlesthat insure that information will not be gone. If the table is read-only (periodically refreshed from the system-of-record) then the rules are nerdy. Star method and hyper-cubes are read-only denormalizations. If the data is to be distributed and/or disconnected and added-to, modified, or erased from then the reconstruction presented below must be hunted. Fundamentally a single principal must be hunted. If the separate single table is updated in more than one system, it should be possible to redesign the original table as if the data was never reformatted or taken seperately.

DENORMALIZATION

There are many methods for denormalizing a relational database invent. These include –

- Duplicated data
- Derived data
- Surrogate keys
- Horizontal segmentation
- Stored Joins
- Recurring data groups (vector data)
- Horizontal Segmentation

Rule of Redesign When the Rule of Reconstruction is overlooked and the data updated, the data is corrupted. Codd's Rule of Reconstruction is a coefficient to 'Lossless-Join Decomposition'. Lossless Decomposition is a method for creating well-designed normalizations from non-normalized database constructions. The Rule of Reconstruction is basically the same idea in reverse. Well designed non-normalized structures are made from normalized tables.

Read-only database (data warehouses, data marts, etc.) gain elastisity and superiority if they also be similar to this approach. The designer should think in details of a series of transformations via SQL. If the physical databank design is not based on a well-designed logical database construction you can't know if the Rule of Reconstruction is being followed or not. There seems to be some kind of data entropic law here. If data is not carefully designed and managed through time, it slides into chaos. Constructions based upon algebra and set theory are fragile. The Relational Model, the operators used on databases are Insert, Replace, Delete, Retrieve, Select, Project, and Join (and to be complete –Product, Union, Intersection, Difference, and Divide). This causes some confusion with those of us that know SQL. The Retrieve, Select, Project, and Join functions are all performed by the SQL SELECT operator. Just to reduce the amount of re-statement and/or translation from Codd's original source, the following uses the Relational Model terminology.

The following is the Relational Model's definition of Select, Project, and Join:

Select – The Select function takes whole rows from a single table and creates an answer set consisting of some (or all) of the rows.

Project – The Project function takes whole columns for a single table and creates answer set consisting of some (or all) of the columns.

Join – The Join function takes whole rows form two or more tables and creates an answer set consisting of concatenated rows that pass the join criteria. The join criteria is usually that two or more columns in the source tables' rows are equal. Any combination of relational operators can be applied to re-form how the data is distributed, provided that the total transformation is reversible. The issue is verifying this reversibility. To not cause inaccuracies and anomalies, each updateable physical data store must be reversible to the well-formed normalized data model. The following is not intended to be exhaustive and consists of only a few of examples of well-formed denormalizations.

OVER NORMALIZATION

This is achieved via the project rescindable operator. Each projection to be stored in some table, possibly another database, must include the primary key of the relation from which the projection is made. Then, each and every projection is well-designed with no duplicate rows. A table can be divide into any number of projections. Note that, at any time, the original table can be renewed using equi-join with respect to the primary keys of the new denormalized tables being co-joined. In applying rescindable technology to the management of a deconventionalized database, it is essential that it be possible the rescindable operators can be beneficial to decompose relations in the global database into relations for the various deconventionalized target tables. In other words you should be able to use SQL for the decomposition. This doesn't mean that you must use SQL for every decomposition, only that you could. You might have a non-SQL Extract Transform and Load (ETL) engine interfacing expressly OLTP systems with an Operational Data Store. If the key in the above example was very large and multiple, a surrogate key could be deputed. And one of the vertical segments could be stored with the surrogate key only.

HORIZONTAL SEGMENTATION

The select interpersonal operator is beneficial to insert some rows of a table into some other table, and other rows to another tables. The collection of rows cannot be arbitrary. The collection of rows must be

made using the choose operator. When a table is disconnected by rows to be saved in other tables, union is used to recover the original table.

STORED JOINS

The beneficial of stored co-joins as a derescindable will formally offend the rule of reconstruction, if the connect-table is updated. In view of this is a very accepted, if not the most common derescindable some discussion is take over. If you must update a stored join, non- interpersonal programmatic measures must be taken to see that data integrity is unmolested. A simple example of this is to not update the replicate columns. It is always hazardous to have the data integrity rules outside the data area. When another update programs are joined to the system they must also include the non-interpersonal data honestly handle. It is good software engineering to have all the data integrity rules in the DBMS in the form of constraints, triggers, and stored procedures. Quality almost always implies simplicity in usage and a 'minimum number of moving parts.' If you have to write complex code with a interpersonal database to get the right answer you are permitted to leap to the conclusion that the database construction does not map to the business. form of constraints, triggers, and stored procedures. Quality almost always implies simplicity in usage and a 'minimum number of moving parts.' If you have to write complex code with a interpersonal database to get the right answer you are permitted to leap to the conclusion that the database construction does not map to the business.

In Summary, Any combination of relational operators can be applied sequentially provided that the total transformation is reversible. While the project and select operators can be used to split a single table from a database into several tables, the join, union, relational intersection, relation difference, and relation divide operators can be used to combine several relations from a database into one table. If the transformations performed on tables from the database are to be reversible, these operators must be applied with considerable care. One can imagine over normalization and horizontal segmentation being applied to the same table. Of course the same restriction applies. If the reversing relational operators are applied, and you cannot tell they table was not always in Third Normal Form, the denormalization was well-formed. We should start the physical tuning of our databases at design time with well-formed denormalizations. To delay tuning until after implementation is a mistake.

III. LITERATURE REVIEW

1. A-Store, a main memory OLAP engine customized for star/snowflake schemas. Instead of generating fully materialized denormalization, A-Store resorts to virtual denormalization by treating array indexes as primary keys. This design allows us to harvest the benefit of denormalization without sacrificing additional RAM space. A-Store uses a generic query processing model for all SPJGA queries. It applies a number of state-of-the-art optimization methods, such as vectorized scan and aggregation, to achieve superior performance. In this experiments (Yansong Zhang et al, 2015) show that A-Store outperforms the most prestigious MMDB systems significantly in star/snowflake schema based query processing.

2. Our HyPer architecture is based on virtual memory supported snapshots on transactional data for multiple query sessions. Thereby, the two workloads OLTP transactions and OLAP queries – are executed on the same data without interfering with each other. The snapshot maintenance and the high processing performance in terms of OLTP throughput and OLAP query response times is achieved via hardware supported copy on demand (= write) to preserve snapshot consistency. The detection of shared pages that need replication is done efficiently by the OS with Memory Management Unit (MMU) assistance. The

concurrent transactional workload and the BI query processing use multi core architectures effectively without concurrency interference – as they are separated via the VM snapshot. In this way, HyPer achieves the query performance of OLAP-centric systems such as SAP's TREX and MonetDB and, in parallel on the same system, retains the high transaction throughput of OLTP-centric systems, such as Oracles's TimesTen, SAP's P*Time, or VoltDB's H-Store. (Alfons Kemper et al, 2014)

3. Hekaton is a new database engine optimized for memory resident data and OLTP workloads. Hekaton is fully integrated into SQL Server; it is not a separate system. To take advantage of Hekaton, a user simply declares a table memory optimized. Hekaton tables are fully transactional and durable and accessed using T-SQL in the same way as regular SQL Server tables. A query can reference both Hekaton tables and regular tables and a transaction can update data in both types of tables. T-SQL stored procedures that reference only Hekaton tables can be compiled into machine code for further performance improvements. The engine is designed for high con-currency. To achieve this it uses only latch-free data structures and a new optimistic, multiversion concurrency control technique.(Cristian Diaconu et al, 2014).

IV CONCLUSION

In this paper, we present Hybrid main memory, which applies the strategy of denormalization to accelerate analytical query processing in main memory. Hyper can also be regarded as a specialized system for multidimensional model and SPJGA queries, which are the most common model and queries in OLAP. Through Hyper, we show that denormalization can be a good optimization strategy for main memory databases. It is highly suitable for today's multicore platforms. a one-size-fit-all design will sometimes make database more complicated and result in low instruction efficiency; instead, by designing a specialized query processing model, e.g., by using virtual denormalization, we could achieve enhanced performance.

REFERENCES

[1]. (2014, Jul. 30). SolidDB. [Online]. Available. http://www-01.ibm.com/software/data/soliddb/
[2]. (2014, Mar. 15). EXtremeDB. [Online]. Available. http://www.mcobject.com /extremedbfamily.shtml
[3]. (2014, Oct. 12). [Online]. Available: http://hyper-db.de/
[4]. (2014, Oct. 12). [Online]. Available: http://hyper-db.de/
[5]. (2014, Oct. 12). TimesTen. [Online]. Available. http://www.oracle.com/ technetwork/products/timesten/overview/index.html
[6]. [Online].Available. http://www.cidrdb.org/cidr2005/papers/P19.pdf
[7]. Balkesen.C, G. Alonso, J. Teubner, and M. Tamer € Ozsu, "Multicore,main-memory joins: Sort vs. hash revisited," Proc. VLDB Endowment, vol. 7, no. 1, pp. 85–96, 2013.
[8]. Boncz P. A, M. Zukowski, and N. Nes. (2005, Jun.). MonetDB/X100: Hyper-pipelining query execution. in Proc. 2nd Biennial Conf. Innovative Data Syst. Res., pp. 225–237.
[9]. Boncz P. A, T. Neumann, and O. Erling, "TPC-H analyzed: Hidden messages and lessons learned from an influential benchmark,"in Proc. TPC Technol. Conf. Perform. Eval. Benchmarking,Aug. 2013, pp. 61–76.

[10]. Diaconu,.C, C. Freedman, E. Ismert, P.-A Larson, P. Mittal, R. Stonecipher, N. Verma, and M. Zwilling, "Hekaton: SQL server's memory-optimized OLTP engine," in Proc. Int. Conf. Manage. Data, Jun. 2013, pp. 1243–1254.
[11]. Doppelhammer.J, T. H¨oppler, A. Kemper, and D. Kossmann, "Database performance in the real world - TPC-D and SAP R/3," in SIGMOD, 1997.
[12]. Hector G.-M. and S. Kenneth, "Main memory database systems: An overview," IEEE Trans. Knowl. Data Eng., vol. 4, no. 6, pp. 509–516, Dec. 1992.
[13]. Kemper.A and T. Neumann, "HyPer: A hybrid OLTP&OLAP main memory database system based on virtual memory snapshots," in Proc. Int. Conf. Data Eng., Apr. 2011,
[14]. Kemper.A and T. Neumann, "HyPer: A hybrid OLTP&OLAP main memory database system based on virtual memory snapshots," in Proc. Int. Conf. Data Eng., Apr. 2011 pp. 195–206.
[15]. O'Neil.P, B. O'Neil, and X. Chen. (2007). The star schema benchmark (SSB). [Online]. Available: http://www.cs.umb.edu/ _poneil/StarSchemaB.PDF.