International Refereed Journal of Engineering and Science (IRJES)
ISSN (Online) 2319-183X, (Print) 2319-1821
Volume 3, Issue 6 (June 2014), PP.55-59

Object Oriented Database Management System for Decision Support System.

Ram Babu,
PhD Research Scholar –Bhagwant University, Ajmer, Rajasthan

Abstract:- To get best performance for an analytic system or data warehouse systems, two technologies, column oriented database management systems and main memory database management system can be combined to get advantages of these two. Both technologies give best performance to its opponent database system, for example Main memory database management systems are faster as they reside in main memory as compared to disk resident database systems. This is because main memory is faster in comparison to hard drive/disk. The performance of main memory database systems is 15-20 % higher than that of disk resident database systems. Whereas column based database systems are faster than the row based database because in column based database systems, data is stored in columns and indexed as compared to row based database systems. Performance of column based database systems is 15-30% higher than that of row based database systems.

It is important to mention about static drive which way faster than conventional hard drives and provide similar performance as of main memory can improve and give best performance if used as hard drive to have database created and maintained on it.

By combining these two technologies, we can achieve 30-50% higher performance for analytics systems they needs to be high performing in analysis and computation. Whereas by having databases & data warehouse created and maintained on static drives will further improve performance to ~50-60%

Keywords:- Memory, Database, Main memory database, real time database, database management, , Hard Disk, storage memory and products, Columns, Column Based Database, advantages, Analytics, data Warehouse

I. OBJECT ORIENTED DATABASES

Object oriented databases are the database in which information is stored and managed in the form of objects instead of data such as integer, characters, floats etc. Object oriented databases are also known as object databases. In object oriented databases, an object is first created, defined, named and then it is called at different point as per requirement similar to object oriented programming languages.

Objects basically consist of the following:

- Attributes - Attributes are data which defines the characteristics of an object. This data may be simple such as integers, strings, and real numbers or it may be a reference to a complex object.
- Methods - Methods define the behavior of an object and are what was formally called procedures or functions.

Because of attributes and methods, object oriented databases contains both executable as well as data in the form of objects. An object oriented database may contain classes which are nothing but a template of an object. Classes are used to define object and does not store any data or methods in it however the data and methods are stored in objects. The class is used to instantiate the object. Classes may be used in object databases to recreate parts of the object that may not actually be stored in the database. Methods may not be stored in the database and may be recreated by using a class.

With traditional databases, data manipulated by the application is transient and data in the database is persisted. In object databases, the application can manipulate both transient as well as persisted data.

These database systems are referred as In-Memory Columnar Database management systems
Two basic methods are used to store objects by different database vendors.

- Each object has a unique ID and is defined as a subclass of a base class, using inheritance to determine attributes.

- Virtual memory mapping is used for object storage and management.

Data transfers are either done on a per object basis or on a per page (normally 4K) basis.

1.1. ADVANTAGES of Object oriented databases

1.1.1. The objects do not require re-assembling from their component tables each time they are used thereby reducing processing overheads by increasing access speeds e.g. up to 100 times faster for some applications such as Sun Cattel benchmark

1.1.2. Paging is reduced

1.1.3. Versioning is easier

1.1.4. Navigation through the database is easier and more natural, with objects able to contain pointers to other objects within the database

1.1.5. Reuse reduces development costs

1.1.6. Concurrency control is simplified by the ability to place a single lock on an entire hierarchy

1.1.7. Better data model as based on the 'real world' instead of the 'flattened' relational model

1.1.8. Good for applications where the relationships between items in the database carry key information e.g. in the student database, we were particularly interested in what students studied (i.e. the STUDIES relationship). This is handled very efficiently by navigation.

1.1.9. Relationships and constraints on objects can be stored in the server application rather than the client application therefore any changes need only be made in one place thus reducing the need for and risks involved in making multiple changes

1.1.10. Fit in well with client/server and distributed architectures

1.2. DISADVANTAGES of Object oriented databases

1.2.1. Poor for applications where the values of items in the database carry key information e.g. if we had been more interested in student age (e.g. to calculate the mean age) than the courses they study then relational database would clearly be more efficient

1.2.2. Speed of access may be reduced by late binding which may cause extensive searches through the inheritance hierarchies

1.2.3. Present lack of standards including the lack of a common query language such as SQL (though OQL on its way?)

1.2.4. There are as yet no formal semantics for ODBMS. Relational databases can be 'proved' correct by means of set theory and relational calculus

1.2.5. The simplicity of relational tables is lost Object Oriented Databases

1.2.6. The object oriented paradigm shift can make the move to ODBMS difficult

II. PRODUCTS

Some of products available in market which uses combined technologies (Columnar database and in-memory database technologies also known as In-Memory Columnar Database System) are given below

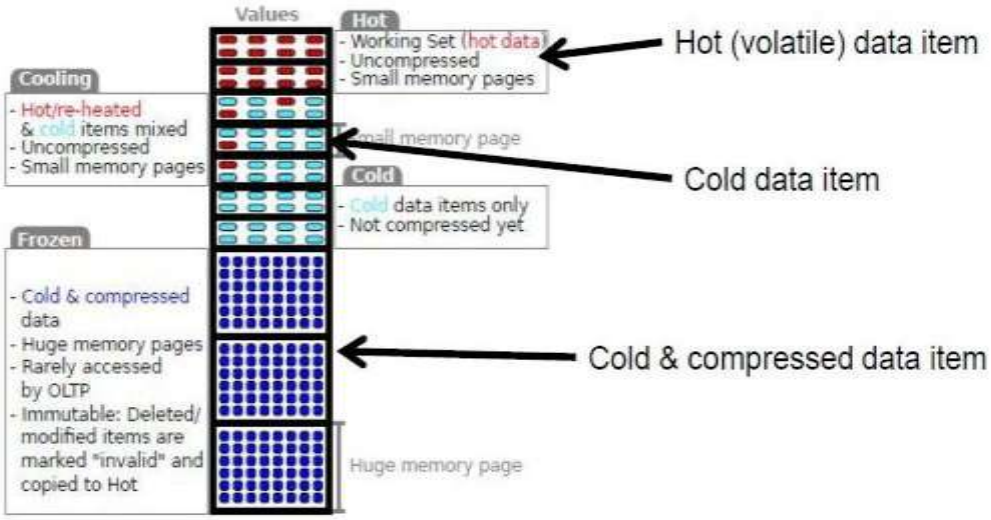
a. HyPer

It is a hybrid OLTP&OLAP main memory database system. And it is columnar in order to achieve best possible query execution performance for OLAP applications.

HyPer provides best performance due to Data Clustering & Compression and its design choices.

III. DATA CLUSTERING & COMPRESSION

HyPer's compression approach in hybrid OLTP & OLAP column stores is based on the observations that while OLTP workloads frequently modify the dataset, they often follow the working set assumption: only a small subset of the data is accessed and an even smaller subset of this working set is being modified.



Hot/cold clustering is an elegant solution to this problem, as the cold bulk of the data can be stored on huge memory pages while the hot, frequently modified working set remains on regular memory pages that can be replicated inexpensively. The frozen, huge data pages are never modified; if a frozen data object is changed, after all, it is invalidated in the frozen partition and reinserted into the hot working set.
Some of the design choices are described below

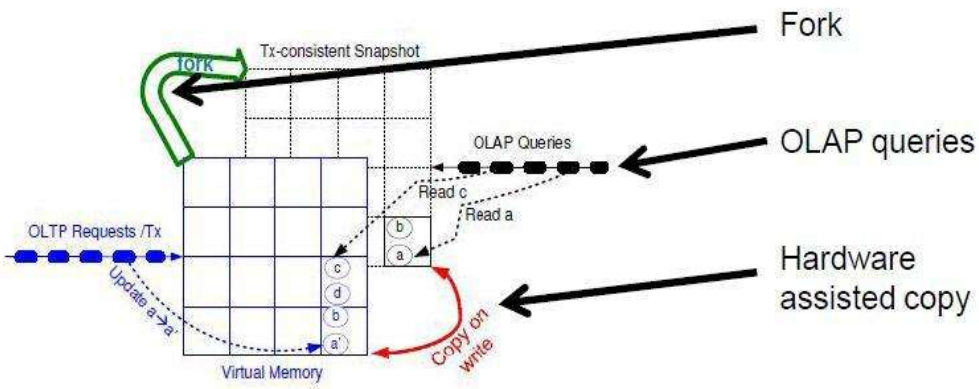
Design Choices 1

HyPer relies on in-memory data management without the ballast of traditional database systems caused by DBMS-controlled page structures and buffer management. The SQL table definitions are transformed into simple vector-based virtual memory representations – which constitutes a column- oriented physical storage scheme.

Design Choices 2

The OLAP processing is separated from the mission critical OLTP transaction processing by fork-ing virtual memory snapshots. Thus, no concurrency control mechanisms other than the hardware assisted VM management are needed to separate the two workload classes.

Design Choices 3



Transactions and queries are specified in SQL and are efficiently compiled into LLVM assembly code. The transactions are specified in an SQL scripting language and registered stored procedures. The query evaluation follows a data-centric paradigm by applying as many operations on a data object as possible in between pipeline breakers.

HyPer's approach to compression in hybrid OLTP & OLAP column stores is based on the observation that while OLTP workloads frequently modify the dataset, they often follow the working set assumption: only a small subset of the data is accessed and an even smaller subset of this working set is being modified.

a. ObjectStore database

Object Store is a specialized type of database designed to handle data created by applications that use object-oriented programming techniques. It is inspired by the Static database originally developed at Symbolics. ObjectStore is innovative in its use of the C++ language to make database access transparent. Objects can be created in a database by overloading the operator new(). In this way, one can store C++ objects directly in the database and these persistent objects look and behave just like normal C++ objects. By making use of signals, Object Store traps pointer exceptions and transparently brings objects in from the database. In addition, by use of a technique called swizzling, the database can be accessed from different platforms, with pages being 'swizzled' as they are brought into memory on page faults to correct big endian versus little endian platform issues as well as virtual function table layout.

Object Store has recently expanded its use beyond the object database market to target use as a database for real-time computing, specifically designed for RFID data management, and as a cache for relational databases

b. Objectivity Database

Objectivity Database allows applications to make standard C++, Java, Python or Smalltalk objects persistent without having to convert the data objects into the rows and columns used by a relational database management system (RDBMS). Objectivity/DB supports the most popular object oriented languages plus SQL/ODBC and XML. It runs on Linux, LynxOS, UNIX and Windows platforms. All of the languages and platforms interoperate, with the Objectivity/DB kernel taking care of compiler and hardware platform differences.

c. GemStone

Gemstone builds on the Smalltalk programming language. GemStone systems serve as mission-critical applications. GemStone frameworks still see some interest for web services and service-oriented architectures.

A recent revival of interest in Smalltalk has occurred as a result of its use to generate Javascript for e-commerce web pages or in web application frameworks such as the Seaside web framework. Systems based on object databases are not as common as those based on ORM or Object-relational mapping frameworks such as TopLink or Hibernate. In the area of web application frameworks, JBoss and BEA Weblogic are somewhat analogous to GemStone.

d. DB4O Object Database

DB4O represents an object-oriented database model. One of its main goals is to provide an easy and native interface to persistence for object oriented programming languages. Development with DB4O database does not require a separate data model creation; the application's class model defines the structure of the data in DB4O database. DB4O attempts to avoid the object/relational impedance mismatch by eliminating the relational layer from a software project. For more information see db4o features.

Developers using relational databases can also benefit from using DB4O, which can be viewed as a complementary tool. The DB4O -RDBMS data exchange can be implemented using DB4O Replication System (dRS). dRS can also be used for migration between object (DB4O) and relational (RDBMS) technologies.

As an embedded database db4o can be run in application process. It is distributed as a library (jar/dll).

IV. SUMMARY

There has been always a requirement of a database management system which are faster in querying and analyzing data for business purpose. As the size of databases are increasing day by day, the query execution and performance of these databases is getting slower and slower or the maintenance cost is going higher and higher.

To overcome these issues, we can take advantages of two technologies called column based database and Main Memory Database management systems. This way we can achieve a database management system having 30-50% higher performance in comparison of DRDB and RODBs.

ACKNOWLEDGMENT

My sincere thanks to Dr. Raghav Mehera to his encouragement and guidance in writing research paper on database management systems which in turn allowed me to focus on a technology, in Memory database systems and column oriented database systems to bring its best.

REFERENCES

[1]. In Memory Database definition, WhatIs.com
[2]. Column based database, WhatIs.com
[3]. wikipedia.org and wikimedia.org
[4]. In-Memory Columnar Databases – HyPer by Arto Kärki