Implementation of Sales Data Warehouse Using Materialized View

Thuzar Aung
University of computer studies, Yangon
zinwar.@gmail.com
white.girl.angel.@gmail.com

Abstract

A distributed database is not stored in its entirety at a single physical location. Instead, it is spread across a network of computers that are geographically dispersed and connected via communications links. A distributed database allows faster local queries and can reduce network traffic. It allows applications to access data from local and remote database. In distributed environments, materialized views are used to replicate data at distributed sites and synchronize updates done at several sites with conflict resolution methods. The user should not need to know where a piece of data is stored physically. If materialized views are used, Database Administrator (DBA) can easily manage the database without too much effort. Utilizing the materialized views can not only reduce system errors but also improve system performance. So, this system is intended to develop Sales Data warehouse by using updateable materialized views.

Keywords: Distributed Database, Data warehouse, materialized views

1. Introduction

Database today, irrespective of whether they are data warehouses, data marts or OLTP systems, contain a wealth of information waiting to be discovered and understood. A data warehouse consists of a set of materialized views defined over a number of data source, collects copies of data from remote, distributed, autonomous and heterogeneous data sources into a central repository to enable analysis and mining of the integrated information.

Data warehousing and online analytical processing (OLAP) are essential elements of decision support, which has increasingly become a focus of the database industry. However, finding and presenting this information in a timely fashion can be a major issue, especially when vast amounts of data have to be searched.

Materialized views help solve this problem, by providing a means to access and report on this data very quickly. Materialized views were first introduced in Oracle 8i and they are part of a component known as Summary Management. The materialized view should be thought of as a special kind of view, which physically exists inside the database; it can contain joins and or aggregates and exists to improve query execution time by pre-calculating expensive joins and aggregation operations prior to execution.

Today, organizations using their own summaries waste a significant amount of time manually creating summaries, identifying which ones to create, indexing the summaries, updating them and advising their users on which ones to use. Now the DBA will only have to initially create materialized view, it can then be automatically updated whenever changes occur to its data source. There is also Summary Advisor component which will recommend to the DBA which materialized views to create, delete and retain.

2. Related Works

New and novel applications for materialized views and view maintenance techniques are emerging. We describe a few of the novel applications here, along with a couple of traditional ones. Materialized views are likely to find applications in any problem domain that needs quick access to derived data, or where recomputing the view from base data may be expensive or infeasible.

Materialized views provide a framework within which to collect information into the warehouse from several databases without copying each database in the warehouse. Queries on the warehouse can then be answered using the materialized views without accessing the remote databases. Provisioning, or changes, still occurs on the remote databases, and are transmitted to the warehouse as a set of modifications.

Incremental view maintenance techniques can be used to maintain the materialized views in response to these modifications. While the materialized views are available for view maintenance, access to the remote databases may be restricted or expensive. Self-Maintainable views are thus useful to maintain a data warehouse [9]. For cases where the view is not self-maintainable and one has to go to the remote databases, besides the cost of remote accesses, transaction management is also needed [10].

Materialized views are used for data integration in [11,9]. Objects that reside in multiple databases are integrated to give a larger object if the child objects match." Matching for relational tuples using outer-joins and a match operator is done in [9], while more general matching conditions are discussed in [11]. The matching conditions of [11] may be expensive to compute.

By materializing the composed objects, in part or fully, the objects can be used inexpensively. [12] presents another model of data integration. They consider views defined using some remote and some local relations. They materialize the view partially, without accessing the remote relation, by retaining a reference to the remote relation as a constraint in the view tuples. The model needs access to the remote databases during queries and thus differs from a typical warehousing model.

3. Materialized Views

A view is a derived relation defined in terms of base (stored) relations. A view thus defines a function from a set of base tables to a derived table; this function is typically recomputed every time the view is referenced.

A view can be materialized by storing the tuples of the views in the database. Index structures can be built on the materialized view. Consequently, database accesses to the materialized view can be much faster than recomputing the view. A materialized view is thus like a cache- a copy of data that can be accessed quickly.

A materialized view is a database object that contains the results of a query. They are local copies of data located remotely, or are used to create summary tables based on aggregations of a table's data. Materialized views, which store data based on remote tables are also, know as snapshots. A materialized view can query tables, views, and other materialized views. Collectively these are called master tables (a replication term) or detail tables (a data warehouse term). For replication purposes, materialized views allow you to maintain copies of remote data on your local node. These copies are read-only. For data warehousing purposes, the materialized views commonly created are aggregate views, single-table aggregate views, and join views.

Like a cache, a materialized view provides fast access to data; the speed difference may be critical in applications where the query rate is high and the views are complex so that it is not possible to recomput the view for every query. Materialized views are useful in new applications such as data warehousing, replication servers, chronicle or data recording systems, data visualization, and mobile systems. Integrity constraint checking and query optimization can also benefit from materialized views.

Materialized views reduce system CPU/IO resource requirements by pre-calculating and storing results of intensive queries. It allow for the automatic rewriting of intensive queries. They are transparent to the application. They have storage/maintenance requirements. It can understand complex data relationships. It can be refreshed on demand or on a schedule.

A materialized view definition can include aggregation, such as SUM, MIN, MAX, AVG, COUNT(*), COUNT (X), COUNT (DISTICT), VARIANCE or STDDEV, one or more tables joined together and a GROUP BY. It may be indexed and partitioned and basic DDL operations such CREATE, ALTER and DROP may be applied.

Materialized views can be refreshed either on demand or at regular time intervals. Alternately, materialized views in the same database as their master tables can be refreshed whenever a transaction commits its changes to the master tables.

A materialized view is a replica of a target master from a single point in time. The master can be either a master table at a master site or a master materialized view at a materialized view site. Whereas in multimaster replication tables are continuously updated by other master sites, materialized views are updated from one or more masters through individual batch updates, known as a refreshes, from a single master site or master materialized view site.

3.1. Updatable materialized view

Updatable materialized view allows users to insert, update and delete rows of the target master table or master materialized view by performing these operations on the materialized view. An updatable materialized view may also contain a subset of the data in the target master.

Updatable materialized views are based on tables or other materialized views that have been set up to support replication.

Updatable materialized views have the following properties. They are always based on a single table, although multiple tables can be referenced in a sub query. They can be incrementally (or fast) refreshed. Oracle propagates the changes made to an updatable table or master materialized view. The updates to the master then cascade to all other replication sites.

Updatable materialized views provide the following benefits: Users allow to query and update a local replicated data set even when disconnected from the master site or master materialized view site. Resources requires fewer than multi master replication, while still supporting data updates. Materialized views can reduce the amount of stress placed on network resources because materialized views can be refreshed on demand, while multi master replication propagates changes at regular

intervals. In addition, because materialized views can reside in a database that contains far less data, the disk space and memory requirements for materialized view clients can be less than the requirements for an Oracle server containing master sites.

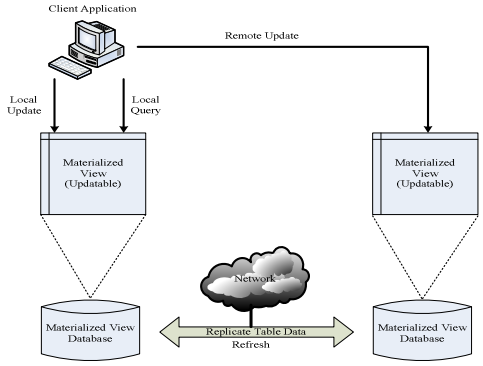

Figure 1. Updatable materialized view

4. Proposed System

This system implements the different materialized views on Sales Data Warehouse of Watch Gallery product on different distributed sites. Sales Data Warehouse of Watch gallery product contains one fact table and six dimension tables. They are sales fact table, item dimension, type dimension, brand dimension, location dimension, branch dimension, and time dimension.

A materialized view is a database object that contains the results of a query. They are local copies of data located remotely, or are used to create summary tables based on aggregations of a table's data. A materialized view can query tables, views, and other materialized views.

For replication purposes, materialized views allow to maintain copies of remote data on local node. These copies are read-only. If you want to update the local copies, you have to use the Advanced Replication feature. For data warehousing purposes, the materialized views commonly created are aggregate views, single-table aggregate views, and join views.

In this system, all the information for the Watch Gallery is stored in two distinct databases. They are Site A database and Site B database. It uses the Apache database server. In Site A, it includes Mater Database and two materialized views. In Site B, it includes Master Database and one materialized view. If you want to show one site data, you will check the site alive or not. This system allows force refresh of all databases for every 10 minutes by using updatable materialized views. Each database of this system maintains materialized views that contain a complete or partial copy of target masters from the other sites. Updatable materialized views enable users to decrease the load on master sites because users can make changes to the data at the materialized view site. This system will provide user quick access without passing through network communication and gives low query evaluation cost.

Every master site and materialized view site in a replication environment has a replication catalog. A replication catalog for a site is a distinct set of data dictionary tables and views that maintain administrative information about replication objects. Every server participating in a replication environment can automate the replication of objects using the information in its replication catalog.

4.1 System Flow Diagram

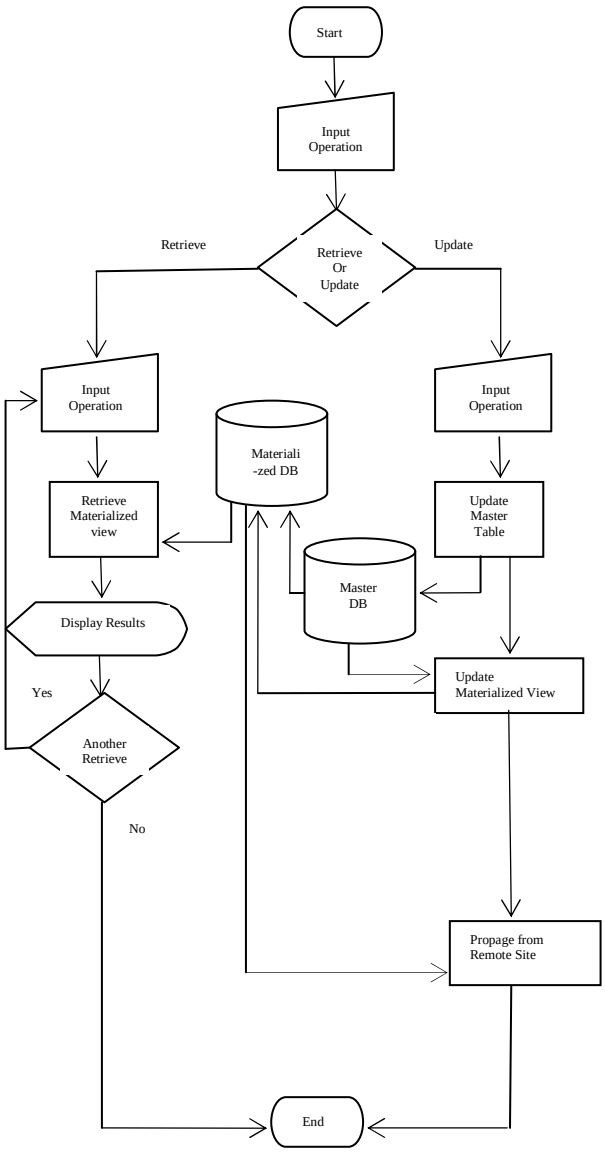

Figure 2. System Flow Diagram

This system includes two site, Site A and Site B. In site A, it is two materialized view. They are by type view and by branch view. In site B, it is one materialized view. It is by branch view. They are separate database. When Site A is mater site,

Site B is remote site. In figure 2, it shows only one site. The user can stay master site or remote site. Wherever the user stay master or remote site, he can use retrieve or update operation. If the user stays remote site, he can use the materialized view on remote site for retrieve purpose.

If you want to show other site data, you check the site alive or not. If other site is alive, you can view all materialized views. If you want to propagate the other site, you may check propagate site.

In our database, they are six tables. There are one fact table and five dimension tables. Sales table is fact table. Item dimension has 70 items, branch dimension has six branches, Type dimension has 3 type, brand dimension has five brand. Brand name are Rolex, Citizen, Seiko, Rado and Casio. Type are Man, Woman and Child. Branch are Lanmadaw, Latha, Pabedan, Kyauk-ta-da and Mingalataungnyunt. In this system, local data can be kept locally, while at the same time remote data can be accessed when necessary. This system can ease the network load because there is no need to be alive connection the whole time. The users can query the desired information as a standalone application.

Figure 3 shows the materialized view on master site A.

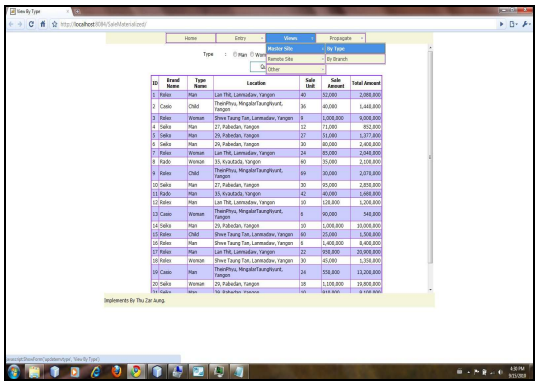

Figure 3. Materialized view on master site

In this system, the user can not compare the using materialized view query and normal query. But the user wants to combine query from master site data and other remote site data, the user easily use the materialized view data. Thus, the network traffic cost is reduced and quick response to user.

5. Conclusion

A materialized view is a view that actually exists as a table. This can be more efficient than re-computing the view's query each time it is accessed. It is also useful for summarizing, pre-computing, replicating and distributing data. MVs can be added or dropped without invalidating coded SQL and is transparent to end-users. In this system, materialized view is used for sales data warehouse to provide faster access for expensive and complex joins in the distributed database.

By using the materialized views, it reduces network loads and users can access data from the replication site that has the lowest cost. Moreover, materialized view can improve the performance of the system and users can access easily.

Reference

[1] G Coulouris, Jean Dollimore, Tim Kindberg, Distributed Systems concetps and design, Third edition, ADDISON WESLEY

[2] G.Chan, Qing Li, Ling Feng. Design and selection of materialized views in a data warehousing environment: A case study. 1999. Http://www .cs.cityu.edu.hk/ ~csqli/papers / DOLAP99.ps.gz.

[3] C.J. Date, An Introduction to DATABASE SYSTEMS, Six edition, The system programming series, ADDISON WESLEY 82458

[4] A. Gupta, H. V. Jagadish, and I. S. Mumick. Data integration using self-maintainable views. Technical Memorandum 113880-941101-32, AT&T Bell Laboratories, November 1994.

[5] K Loney, Kevin Loney Consulting, LLC, Creating Materialized views http://www.embaracadero.com/resources/tech_papers/matview.pdf

[6] J. Lu, G. Moerkotte, J. Schu, and V. S. Subrahmanian. E_cient maintenance of materialized mediated views. In SIGMOD 1995

[7] Tanebaum, Andrew S, Computer Networks, Fourth edition

[8] Y. Zhuge, H. Garcia-Molina, J. Hammer, and J. Widom. View maintenance in a warehousing environment. In SIGMOD 1995, pages 316-327.

[9] G. Zhou, R. Hull, R. King, J-C. Franchitti. Using Object Matching and Materialization to Integrate Heterogeneous Databases. In Proc. of 3rd Intl. Conf. on Cooperative Info. Sys., 1995.

[10] http://en.wikipedia.org/wiki/Materialized view

[11]http://download.oracle.com/doc/cd/B10500_01/server.920/a96567/re.pmview.htm

[12]http://www.orcle.com/technology/products/oracle9i/pdf/09i_mv_pdf