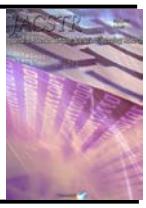

A Dynamic Warehouse Design Based on Simulated Annealing Algorithm

Murtadha M. Hamad[1,a] Yussra Atalla Turky[1,b]

[1]College of Computer - Anbar University – Anbar – Iraq
[a]mortadha61@yahoo.com, [b]white_rossa_ya@yahoo.com

ISSN: 2231-8852

ABSTRACT

The amount of information available to large-scale enterprises is growing rapidly. New information is being generated continuously by operational systems. Decision support functions in a warehouse such as On-Line Analytical Processing (OLAP), involving hundreds or thousands of complex aggregate queries over large volumes of data. A data warehouse can be seen as a set of materialized views defined over some relations. In this paper, when a query is posed to answer, here will be used the suitable materialized views with tables in order to produce best views and tables which will be used for constructing any new query. In order to achieve and implement the Dynamic Warehouse Design, creating three complex OLAP queries with join and aggregation operation, creating views and updating them by using windows task scheduler and batch files based on base table updating, creating lattice of views by using multiple view processing plan operation, simulated annealing(SA) algorithm was developed and introduced for query re writing by replacing dynamically suitable views instead of tables and introducing best tables and views that will used by user to construct the suitable query. The main goals of this work are to show the utilization of derived data such as materialized views for run time re optimization of aggregate queries (quick response time), effective, transparency and accuracy are important factors in the success of any data warehouse.

Keywords: On-line analytical processing (OLAP), materialized view, dynamic warehouse, simulated annealing, data warehouse

1. Introduction

A Data warehouse (DW) can be defined as a relational database that used for storing information which is collects from multiple operational databases in order to enable complex business analysis queries like summarization aggregates etc. The purpose from using data warehouse is to reproduce new data across the relevant tables and views as quickly as possible [1]. DW features are" subject oriented, integrated, non- volatile, and time-variant"[2].When a business environment evolves or updates several changes may occur in the content and structure of the underlying data sources. Thus DW is always developing and growing up environment. This developing takes a lot of attentions and many researchers have addressed these issues from different point of view [3][4]. DW can be seen as a set of materialized views (MVs) defined over a set of base relations, materialized views are the

query results or some intermediate results that store physically in order to avoid repetition computation operations [5]. On-Line Analytical Process (OLAP) and Decision Support System (DSS) applications used complex grouping/aggregation queries in order to making a decision and answering the query. When the base relations change, the MV at the DW views needs to be updated [1]. A number of developed algorithms have been proposed for optimizing queries to speed up data warehouse like Multiple View Processing Plan (MVPP) and classical algorithms for selection of materialized view like simulated annealing, genetic algorithm and A* algorithm etc.

The rest of the paper is organized as follows. In Section 2, the dynamic data warehouse is described. Section 3 explains the details steps of simulated annealing algorithm. In Section 4, the process of answering queries using views or rewriting queries using views has been elaborated. The details of the proposed system are explained in Section 5. In section 6 the details implementation and the results are demonstrated. Conclusion is presented in the final section, 7.

2. Dynamic Data Warehouse

DWs are dynamic entities have the ability to develop or update continuously over time. When time passes, new queries need to be answered by DW. Some of the new queries can be answered using views already materialized in the DW. But other new queries for answering by the DW need to materialization of new views. In any case, if a query to be answerable by the DW, there must exist a complete rewriting of the query over the (old and new) materialized views. Query rewriting can be doing over the old views, or over the new views, or partially over the new and partially over the old views. When new views need to be materialized, extra space needs to be allocated for materialization in order to answering the new queries [7].Many algorithms have been used to select set of views to materialized like genetic, A*, greedy and simulated annealing algorithms.

3. Simulated Annealing Algorithm

Simulated Annealing (SA) is single-solution based Meta heuristic applied successfully for many combinatorial optimization problems such as travelling salesman problem. SA method is useful for a problem that is characterized with a very large discrete arrangement space, which has too large search space, over which an objective cost function is to be minimized (or maximized) [9]. In this paper Simulated Annealing (SA) algorithm was introduced and applied. The motivation for using SA algorithm is to solve the materialized view selection problem was based on the data warehouse can have a large number of views and the queries may change frequently. Therefore, materialized view selection is considered as a complex problem with a very large state space.

4. Answering Queries using Views or Rewriting Queries Using Views

It is an operation to find efficient methods to rewrite the query using a set of materialized views earlier defined. Query rewriting is rewriting the query to refer to the materialized views

based on the information from query containment checking, query rewriting was studied under various query languages, such as (selection, projection and join) queries, and queries with aggregations [5]. It is mostly useful in a data warehouse environment, because it is a mechanism where applications from the end user or database transparently to improve query response time by rewriting the SQL query using the materialized view instead of accessing the original table's [10]. It takes MVPP tree as input and produces the output as a tree of sequence of views and tables which can be used for answering queries [8].

5. The Proposed and Designed System

In this paper, we proposed a Dynamic Warehouse Design and it consists of several phases as shown in Fig. 1. These phases are:

1. Using data warehouse snow flake schema.
2. Create views.
3. Loading data base objects views and tables.
4. Make change on data warehouse (add new query or object after change in data amount).
5. Calculate new costs.
6. Process simulated annealing algorithm.
7. Add new query.
8. Update views data.
9. Getting result for new query.

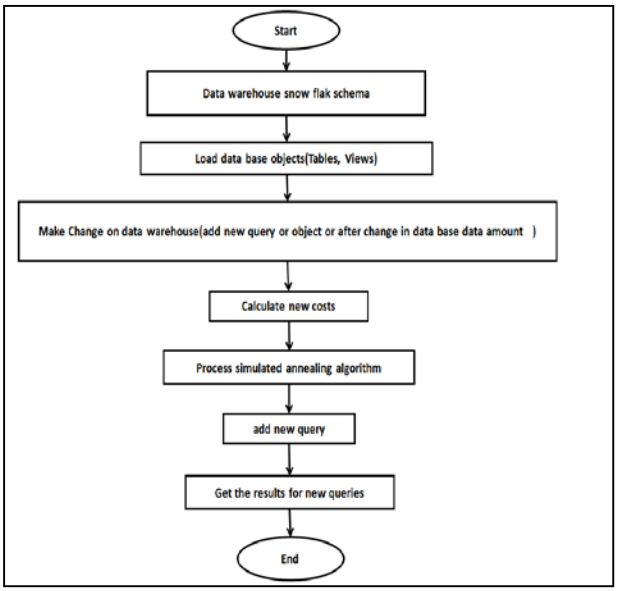

Fig. 1. The proposed dynamic warehouse system

In the first step of the proposed system which is using data warehouse snow flake schema in this stage sales system was taken as a case study, it is size 4.5 GB and it is contains one fact table (warehouse table) and seven dimension tables which are (items, clients, clients invoices, client invoice details, suppliers, suppliers invoices, suppliers invoices details), the relationship between fact and dimension tables is snow flake schema as shown in Fig. 2.

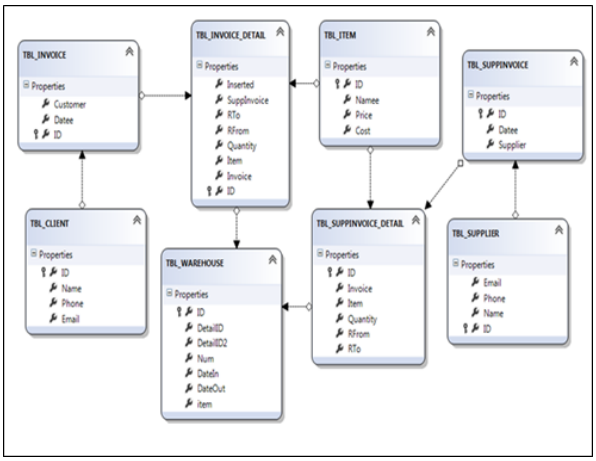

Fig. 2. Snow flake schema of the sales system data warehouse

Three OLAP queries are used in sales system these queries are

1. The first query is: Get a list of clients who bought specific Item (directly from base tables or by materialized views).
2. The second query is: Get list of suppliers whom having more Items sold (directly from base tables or by materialized views).
3. The third query is: Get list of best selling items based on season (directly from base tables or by materialized views).
4. If the query executed by using the base table it will take long time rather than when executing query by using materialized views as shown in table (1).

Table 1. Execution time of query

No. of query	Using Materialized Views	Using Base Table in data warehouse
1	1.512	10.875
2	2.751	14.325
3	0.881	12.920

Fig. 3 presents the execution time of query.

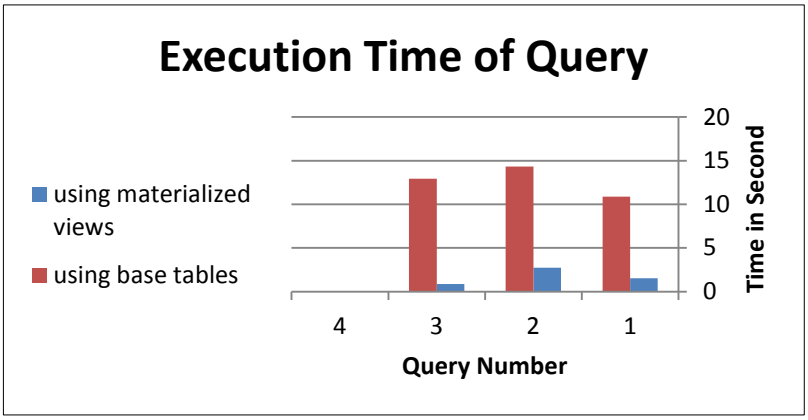

Fig. 3. Difference query execution time

MVPP was used like in [9] for creating query search space as shown in Fig. 4. In this figure creating query execution plan based on selecting data from base tables, but in Fig. 5 query execution plan creating based on selecting data from materialized views which substituted

instead of base tables.

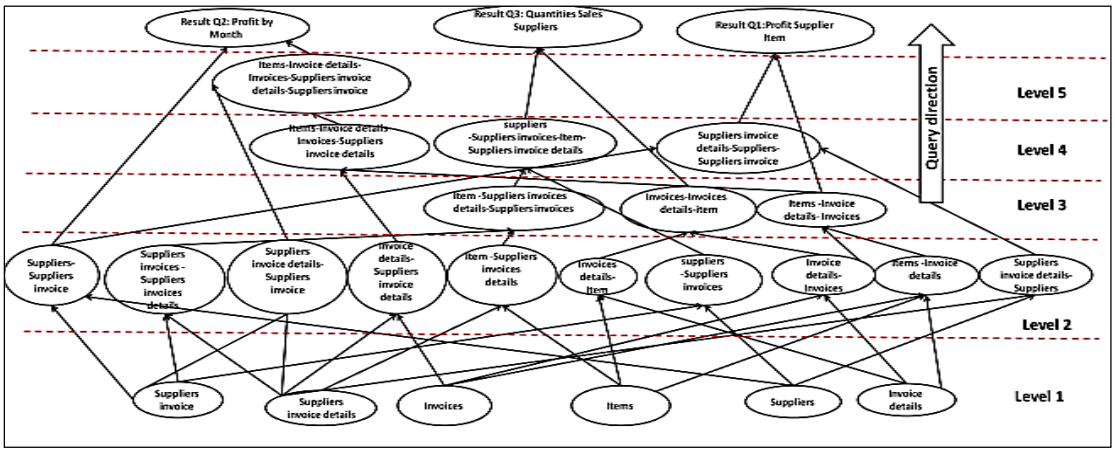

Fig. 4. Lattice illustrates the levels of query optimization before using MV

There is local access plan for each individual queries and then will be merged based on the shared operations on common data sets. The query travel between levels of the hierarchies by operations (drill down or roll up) and will answer based on collecting the result from the search space levels begin from the base level up to the upper level.

Views are created manually by selecting the active rows from base table and active procedures are used for updating views tables and requesting views tables. This procedures handled by tasks running from windows scheduler by running batch files that run the procedure, The goal is to keep the materialized views always updated. So the scheduler keep remove and fill the tables on time bases.

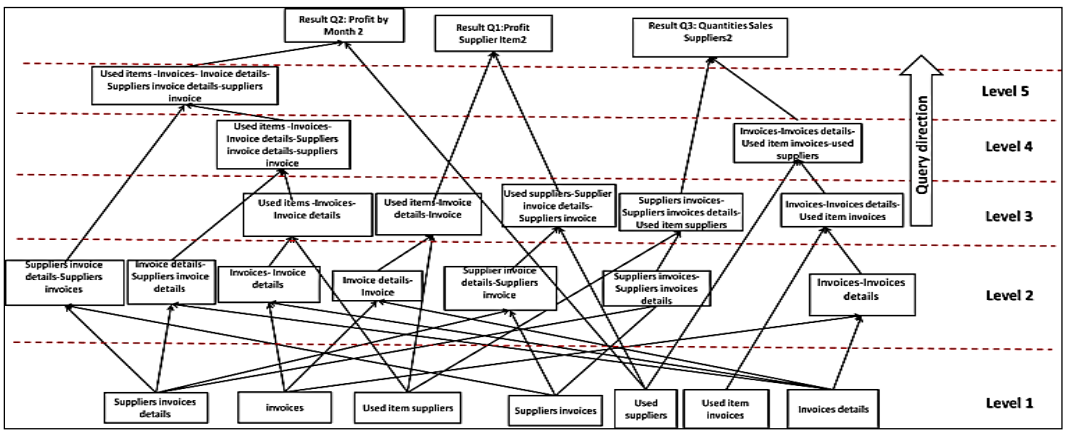

Fig. 5. Lattice illustrates the levels of query optimization after using MV

As illustrated in the previous schema the numbers of nods will decreased when using MV on the contrary when using the base tables, so this will decrease the search space that will lead to decrease query requesting time.

The operations of replacing view instead of tables when creating query is doing manually, this operation will take a long time, effort and the query may be not written in exact form, so the query re writing or query re optimization operation was introduced and implemented.

After specific circumstances the database reporting procedures needs to update in order to change query writing and so the query execution plan is changed and the query speed is enhanced. In some reports the speed without using the enhanced query is even unusable. The circumstances that need to upgrade the queries writing method is like:

- Adding new materialized views to the database.
- Accumulating more records that affect the current database working speed.

So an application need must have the ability to reach the needed information to suggest new queries from calculating the costs of the tables that construct the query or the costs of calling data from materialized views. The information that the Warehouse upgrading application is acquiring is:

- The cost of getting all rows from table.
- The materialized views that representing specific tables and the tables that could be substituted by.
- The relationships among the tables as well as with materialized views.
- The objects needed for each query.

So all these information are embedded within the application classes and interfaces to be acquired from the user who should be a database administrator or someone who is in charge of the management of database management system. The information also acquired directly from the database.

(Query, Join, Table) is represented by classes and it contains the needed attributes to calculate the best layout of objects inside the queries. The main challenge in the work was to find the calculate-able form of tables inside the query to make it possible to apply artificial intelligence functions to it, which are all needs a basic form of objects to be calculated. The method used to calculate the best formation of queries in each query is simulated annealing. Each query is entered as a construction of tables that is important to the query to work with them. But the materialized views are added to the system as tables that can be substituted with tables. So during the calculations it will appear like the same table but with better performance. The simulated annealing is a method that's based for solving unconstrained and bound-constrained optimization problems. The method models the physical process of heating a material and then slowly lowering the temperature to decrease defects, thus minimizing the system energy. So two factors is needed one is the current Performance counter and the other is the factor in which the Performance counter goes down. This represents some form of looping for a large amounts that enable the systems that's needed to have unknown number of probabilities to reach stable state or the best state. And in our case the cost of the query that's created by each time is the thing that we compare to check if the case is suitable or not. In each loop a swap operation is done to a random table from the query, and then the query is checked weather it's suitable and compatible with the needs of origin query and having all the needed tables inside it or not .And then to check the cost. After Hundreds or thousands of probabilities the query will reach the best state that will be chosen. Table 2 represents the cost for each table and view, (RPT) represented as view, (TBL) represented as table.

6. Implementation and Results

The proposed and designed system has been executed by using (visual basic.net 2013 and SQL 2008 R2 express programming languages). So the implementation of the system is performed on phases. The system includes interface to execute it easily and to support the manager or decision maker in the process creating queries. The input to the query optimizer system is set of tables and views while the output is the best tables and views will be used to construct the query, for example:
Input: table1, table2, table3
Output: table1, view1, table3

Table 2. Execution time of tables and views

Table and view	Average work time (Sec)	0Average elapsed time (Sec)
RPT_CLIENTS	227010	243490
RPT_ITEM_SUPPINVOICE	29723	56250
RPT_ITEMS_CLIENTS	12581	12582
RPT_ITEMS_INVOICES	17498	23395
RPT_ProfitSuppItemDate	9164	24874
RPT_VENDOR	11134	18100
RPT_VENDORS_ITEMS	253	253
RPT_WAREHOUSE_INSERTED	346667	465582
TBL_CLIENTS	191193	302015
TBL_INVOICE_DETAILS	330294	488234
TBL_INVOICE	13084	234097
TBL_ITEMS	18995	44850
TBL_RANGES	266858	410311
TBL_SUPPINVOICE_DETAILS	447351	628555
TBL_SUPPINVOICE	66769	91763
TBL_SUPPIERS	33443	81064
TBL_WAREHOUSE	33443	81064

7. Conclusions

After the implementation of the proposed Design Dynamic Data Warehouse system and through the execution of the proposed algorithm simulated annealing algorithm for warehouse upgrading when entering new queries to select the best solution from multiple choices, from obtained results, we concluded that database used in different field of works, due to the multiple and different using of data bases, it needs to enhance operations to change its structure In terms of queries or indexes to speed up and increase the efficiency of paging operations. Besides, the most common way to improve database performance is to change the way of writing the query. Also, an artificial intelligence algorithm on one hand is needed and on the other hand is a simple way of writing the query and delivers them with a meaningful manner, so that the proposed system can find a better way to rewrite the queries. The simulated annealing is an artificial intelligence algorithm that is highly efficient to find a stable solution from among multiple options for solutions. Lastly, when using materialized views for answer queries, time will decrease in 7 % when using base tables.

References

[1] Payal Pahwa, Rashmi Chhabra, “An Object Oriented Data Warehouse Design”, International Journal of Soft Computing and Engineering (IJSCE), IV (2), pp. 2231-2307, 2014.

[2] Jin-Hyuk Yang, In-Jeong Chung, “ASVMRT: Materialized View Selection Algorithm in Data Warehouse”, International Journal of Information Processing Systems, II (2), pp. 57-75, 2005.

[3] Jing Hu, “Optimizing Queries Using a Materialized View in a Data Warehouse”, College of the Oklahoma State University, Oklahoma, USA, 2005.

[4] William Inmon, Derek Strauss, Genia Neushloss, The Architecture for the next generation of DW, Genia Neushloss, 2008.

[5] Songting Chen, “Efficient Incremental View Maintenance for Data Warehousing”, Thesis of a doctor degree in Computer Science, 2005.

[6] Sergio Lujan, Juan Trujillo, "Acomprehensive Method for Data Warehouse Design", In Proceedings of 5th International Workshop on Design and Management of Data Warehouse(DMDW’03) , pp. 1-14, 2003.

[7] Phan Quoc Nghia, “Building OLAP Application to Exploit Database of Rice Pests”, International Journal of Emerging Technology and Advanced Engineering, III (11), pp. 130-137, 2013.

[8] Mukesh Mohania, Guozh Dong, “Algorithms for Adapting Materialised Views in Data Warehouses", International Symposium on Cooperative Database Systems for Advanced Applications, 2000.

[9] Roozbeh Derakhshan, Frank Dehne, Othmar Korn and Bela Stantic “Simulated Annealing for Materialized View Selection in Data Warehousing Environment ", Springer, pp.89-94, 2005.

[10] Bilal Adil Mahdi," Design and Implementation of Materialized Views Tool for Data Warehouse Structure", Thesis of a master degree in Computer Science, 2013.