Journal of Engineering and Applied Sciences 12 (22): 5997-6005, 2017
ISSN: 1816-949X
© Medwell Journals, 2017

Materialized Views Optimal Selection for Data Warehouse Quality

[1]Mohanad Ahamed Salih and [2]Murtadha M. Hamad
[1]Department of Computer Science,
[2]Department of Information System, College of Computer Science and Information Technology,
University of Anbar, Anbar, Iraq

Abstract: For the success of any data warehouse, accurate and timely consolidated information along with quick and effective query response times is the basic fundamental requirement. The materialization of all views is practically impossible because of the materialized view storage space and maintenance cost constraint thus proper materialized views selection is one of the intelligent decisions in designing a data warehouse to get optimal efficiency. This study presents a framework for selecting best materialized view using algorithm Particle Swarm Optimization (PSO) this algorithm one of the stochastic algorithm so as to achieve the effective combination of good query response time, low query processing cost and low view maintenance cost. The results showed that the proposed method for selecting best materialized view using PSO algorithm is better than other techniques through compute the ratio of query response time and compare it to the response time of the same queries on the materialized views ratio of implementing the query on the base table takes eleven times more than time of the query implementation on the materialized views. Where the response time of queries through MVs access were found 0092 msec while through direct access queries were found 1039 msec. This show the performance of query through materialized views access is 1029.34% better than those directly access through data warehouse.

Key words: Data warehouse, select materialized view, OLAP, query processing, query access frequency, performance

INTRODUCTION

Data warehouse is a repository of large amount of data collected from multiple data sources. It is mainly used for processing of queries and detailed analysis of data that is useful for decision makers. Hence, to make this data available in less amount of time is essential. Here, comes the concept of materialize view. Materialized view stores result of queries which improve query performance (Kurzadkar and Bajpayee, 2015). Materialized view selection involved query processing cost and materialized view maintenance cost. Selecting views to materialize for the purpose of supporting the decision making efficiently is one of the most significant decisions in designing data warehouse. Selecting a set of derived views to materialize which minimizes the sum of total query response time and maintenance of the selected views is defined as view selection problem. Therefore, to select an appropriate set of a view is the major target that diminishes the entire query response time and also maintains the selected views. So many literatures try to make the sum of that cost minimal (Nalini *et al.*, 2012).

Literature review: In this paragraph, we will show some of the previous studies on the optimal select materialization view in the data warehouse and some of the algorithms used which is close to and connected to our study. These include studies, thesis and the following securities:

Mohod and Chaudhari (2013), they are proposing the idea of keeping in mind how to choose a set of views which materialized with the assistance of different parameters like: costs and frequency of query processing, space of storage and proposed methodology that determines which queries are more beneficial for creating the materialized view so as to realize a high performance of the query, the proposed framework is executed on the simulated student data warehouse model using list of query, to find the efficiency of the proposed approach in selection of materialized view.

Kurzadkar and Bajpayee (2015), this research presents the methodology of determining the queries which are more beneficial for the materialized view creation, so as to achieve the high query performance. The selection of query depends on the user need bases

Corresponding Author: Mohanad Ahamed Salih, Department of Computer Science,
College of Computer Science and Information Technology, University of Anbar, Anbar, Iraq

that changes the weighted factor given to different parameters, cost of query, cost of maintenance and space of storage. The designed framework is executed on the customer data warehouse model using list of query for finding the efficiency of the designed approach. The time required of query by selected MV as compared to that query directly selected by DW is very much less. One of the results shows that the queries performance in terms of access time from materialized view selection was found to 15 msec while those queries directly selected from data base/data warehouse was found to 32 msec. This shows that the performance of query through materialized views access is 113.33% better than those directly access through data base/data warehouse.

Gosain (2016), this research presented proposed framework for implemented Particle Swarm Optimization (PSO) algorithm on lattice framework for materialized view selection in DW. The experiment was conducted by running algorithms on TPC-H benchmark. The technique was by taking different frequency set and number of dimensions. The results proved effectiveness of PSO algorithm over genetic algorithm in selecting suitable set of materialized views with less query processing cost.

We conclude from the above query optimization is the ultimate target of enhancing the performance materialized views by considering all the essential constraints like response time of query, processing cost, low view maintenance cost and query access frequency. These are important factors for optimal selection of materialized views. By considering all the above factors will show drastic improvement in performance. Through previous studies, we come across the question: what all views must be materialized to get the most optimal (near optimal) solution? A possible solution is to select a set of derived views to materialize that minimizes the sum of total query response time and maintenance cost of the selected views.

MATERIALS AND METHODS

Concept of materialized view: Materialized views are a decision support/data warehousing system tool that is able to increase by many orders of magnitude the speed of queries that access a large number of records. In data warehouses, materialized views can be used to pre-compute and store aggregated data such as the average of sales. In this environment, materialized views are often referred to as summaries, for they are storing summarized data (Sainath, 2012). Also, materialized views using to reduce two costs these costs are namely: the query processing cost and materialized view maintenance cost. It is not recommended to materialize all possible

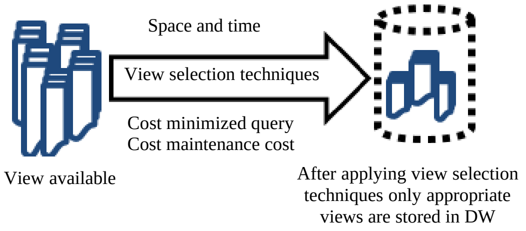

Space and time
View selection techniques
Cost minimized query
Cost maintenance cost
View available
After applying view selection techniques only appropriate views are stored in DW

Fig. 1: Space and time reduction scheme

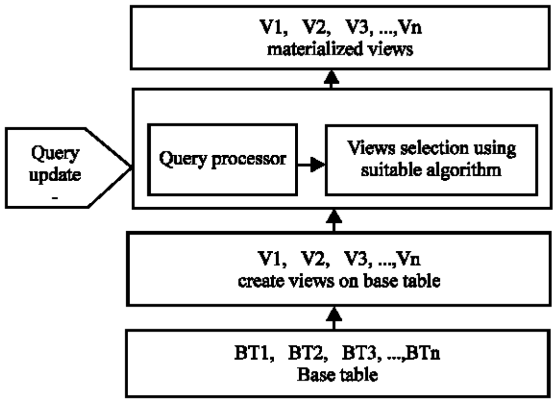

V1, V2, V3, ...,Vn materialized views
Query update
Query processor
Views selection using suitable algorithm
V1, V2, V3, ...,Vn create views on base table
BT1, BT2, BT3, ...,BTn Base table

Fig. 2: Materialized view selection process

views because of time constraints and it consumes a large space in memory. As shown in Fig. 1, the main goal of view selection operation is to minimize a cost function (Rashid and Islam, 2010).

Materialized view selection: The problem of view selection is choosing a set of views to materialize to achieve the best query performance. Typically view selection is under a maintenance cost constraint and/or a space constraint. Unlike answering queries using views that need to handle adhoc queries, In view selection scenarios the queries are known. Hence, most algorithms for view selection start from common sub-expressions among queries identification. These common sub-expressions work as the MV candidates. View selection has one practical fundamental issue that is there are multiple possibly competing factors to take into consideration during the phase of view selection such as query complexity, database size, query performance, etc. The process of selecting the suitable views to materialize in data warehouse is showed in Fig. 2 (Rashid and Islam, 2010).

The architecture above shows that the query processor interacts with the view selector. Based on the query processing plan it applies the notion of view relevance to select the views for a given set of queries (Karde and Thakare, 2010). MVS problem main objective is minimizing a cost function or constraint. A constraint can be user oriented (query response time constraint) or system oriented (space constraint). The basic objective of view selection problem is finding a set of views to minimize the expected cost of evaluating frequently used queries (Karde and Thakare, 2010).

Multiple View Processing Plans (MVPP): MVPP is a plan of global query processing for the entire set of queries and is formed by sharing and merging local processing plan of each query. AND-OR view graph is made by merging all possible execution plans of each query in the query set. Many algorithms were proposed for the problems of view selection, they include heuristic based algorithms. Greedy algorithms, stochastic algorithms such as genetic algorithm, simulated annealing, etc. Though greedy algorithm provides a close optimum solution, yet it becomes ineffective and slow for multi-dimensional problems. Particle Swarm Optimization (PSO) algorithm, one of the stochastic algorithms is a meta-heuristic global optimization algorithm. It is being used by wide ranges of applications in lots of computer science domains but it has not been studied extensively in the problem domain of materialized view selection in DW. Comparing PSO to other algorithms, it requires small number of basic operators and parameters and also very simple to use. In terms of speed and memory it is computationally inexpensive (Gosain, 2016).

Particle Swarm Optimization (PSO) algorithm: PSO algorithm is a meta-heuristic global optimization algorithm based on swarm intelligence theory. It behaves similar to the fish schools bird and flocks in searching of food. When the birds search for food, they transmit information to each other about the best possible way to find food and finally flock to that place. PSO, similarly, works as follows: it starts with a non-consistent initial population of n particles where each particle represents a candidate solution. Each particle has a position in D-dimensional space and some velocity (Marini and Walczak, 2015). The algorithm task is to optimize an objective function to the minimum value possible. Particles position and velocity are changed continuously according the best position found by whole population (global best) and by the particle itself (particle best) so far. Hence, particles learn from each other and thus, each particle persuades to the best particle of the swarm and reaches the idealist value of the objective function (Gosain, 2016).

Materialized view selection using (PSO): Materialized view selection aim is to select an optimal set of views within space limits minimizing the query processing cost due to the set of user queries. PSO algorithm searches through the whole space and moves in the direction of the best found solution in order to find the minimized cost value to run the user queries and return results in minimum possible time. The following equation represent (PSO) Algorithm 1.

Algorithm 1:

$$p = p + v$$

with

$$V = \underbrace{V}_{\text{Diversification}} \quad \underbrace{+C1\times rand\times(pBest\text{-}p)+C2\times rand\times(gBest\text{-}p)}_{\text{Intensification}}$$

Where:

p	=	Particle's position
v	=	Path direction
c1	=	Weight of local information
c2	=	Weight of global information
pBest	=	Best position of the particle
gBest	=	Best position of the swarm
rand	=	Random variable
Intensification	=	Explores the previous solutions, finds the best solution of a given region
Diversification	=	Searches new solutions, finds the regions with potentially the best solutions in PSO

The problem: Choosing which view will be selected first to be materialized in the DW is the DW administrator/designer's main problem. Maintain materialized views for each query impractical as the MV are realized physical table until the disk-space requirements and thus the consumption is very large and/or large update cost having. A possible solution is to choose a group of derived views to materialize, that decrease the summation of total query maintenance cost and response time of the selected views. That is the view selection problem. In this research presented the methodology that determines which queries are more beneficial for the creation of materialized view so as to achieve the high query performance. Using algorithm (PSO) to be ready to select the best solution when directing new queries to the system.

Materialized views selection framework: This study elaborates the created framework approach for the selection of materialized view. The ultimate aim behind the proposed materialized view selection framework is to materialize the user views by taking into consideration of query frequency, query processing cost and storage requirement of query. Accordingly, we have built the data warehouse tables (company system) in SQL Server 2014 environment, filling these tables of large number of records; these tables are created based on the needs of any company system. Clients where all the information of the clients are saved, suppliers is the same but for suppliers, invoices are form in two types, supplier

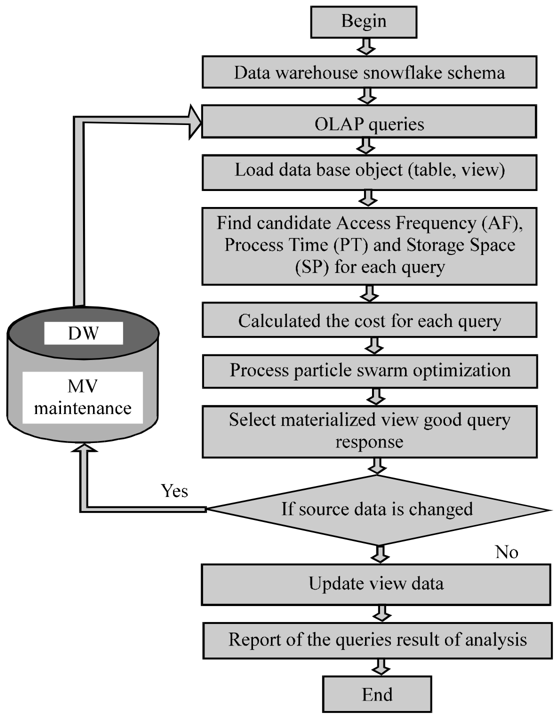

Begin
Data warehouse snowflake schema
OLAP queries
Load data base object (table, view)
Find candidate Access Frequency (AF), Process Time (PT) and Storage Space (SP) for each query
Calculated the cost for each query
Process particle swarm optimization
Select materialized view good query response
If source data is changed
Yes
No
Update view data
Report of the queries result of analysis
End
DW
MV maintenance

Fig. 3: Flowchart of materialized view using (BS-MV) algorithm

invoices and client invoices, each of those has two tables, main table and details (invoices) and (invoices-details) for example. After creating the company system data warehouse, it becomes ready for importing in to the visual basic.net 2013 environment for completing the next steps of the proposed system.

The proposed and designed system: In this study, the main steps (phases) of the proposed design of Best Selection Materialized View (BS-MV) algorithm have been explained and demonstrated. Figure 3 shows the main phases of the proposed system.

Suggested algorithm: In this study, materialized views method has been proposed which is called best selection materialized view (BS-MV) it has a low processing time, storage cost and good query response. The flowchart above was constructed according to the following Algorithm 2.

Algorithm 2; Suggested algorithm called best selection materialized view:

```
Input:  Data warehouse snowflake schema
Output: Report of the queries result of analysis
Step 1: Generation set OLAP queries by operation aggregation (sum, max, min, count ...)
Step 2: Find candidates Access Frequency (AF), Processing Time (PT) and Storage space (SP) for each query in the data warehouse
Step 3: Calculate the materialized view creation cost (SQC), for each Frequency Cost (FC), Processing Cost (PC) and Storage Cost (SC), following formula [SQC = w1*AFC +w2*PTC+w3 (1-SPC)], W1, W2 and W3 are the impact weight specified by the materialized view selection analyzer
Step 4: Using Swarm Optimization (PSO) algorithm, find the best queries that need to materialize to optimize the complex query processing time
Step 5: Select materialized views which have a low processing, good query response and low storage cost
```

On Line Analytical Processing (OLAP): The first step of the Algorithm 2, it generation set of OLAP queries; it is an approach used to answer analytical queries, in business intelligence in relational database, report writing and data mining. In this study, 25 complex SQL OLAP queries with drill down operation, aggregation operations like (COUNT, SUM, MIN, MAX), join, selection, filtering operation like (using condition where) and GROUP BY operation doing have supposed to select data from tables in the company system.

Selection parameters of queries: The second step of the algorithm above was used to find processing time, storage space information and queries frequency. For each query in Query Set (QSET) they find the storage space, frequency and processing time that can be stored in query Information List (IL) in the form of Query Information of Parameters (QIP). The query access frequency refers to how many times queries are being fired on a particular cube, it is like 4 out of 20 times this query is being fired. The selection most prominent parameters of queries using following Algorithm 3.

Algorithm 3; This algorithm is used selection to parameters of queries:

```
Assumptions:
QSET →Given set of queries
 AF →Query Access Frequency
 SP →Query Storage Space
 PT →Query Processing Time
 IL →Query Information List
    QIP →Query Information  Parameters
Algorithm:
begin:
Repeat for I ←1 to QSET
QIP ←find  AF
QIP←find  SP
QIP←find  PT
 IL←    QIP
end
```

Selection cost of queries: The third step of the algorithm is used for calculating the cost of selection for each query. Here, the Two algorithm output, i.e., IL is used as an input to calculate the query frequency, query processing and query storage cost. Using weighted combination of query frequency, query processing and query storage cost, materialized view selection cost will be calculated for each query. Then after adding all parameter's cost we arrived at selection cost. Designed formula to be used in order to find the selection cost is given by:

$$SQC = w1*AFC+w2*PTC+w3\ (1-SPC)$$

where, AFC is the frequency cost that can be calculated as frequency of particular query/max frequency from all queries. SPC is calculated as multiplication of row and column of query and SPC that can be calculated as storage space to be selected of particular query/maximum storage space from all queries.

PTC is the processing time that can be calculated as processing time of particular query/maximum processing time from all queries where, weight 1-3 are weights given to AFC, SPC and PTC such that their sum = 1. The selection cost of queries using Algorithm 4 as:

Algorithm 4; This algorithm is used to selection cost of queries:

```
Assumptions:
SQ  → Selected Query MAF → Maximum of
        query Frequency
  MPT → Maximum of
          query Processing Time
  MSP → Maximum Query Storage Space
PTC →Query Processing Time Cost
  SPC →Query Storage Space Cost
  AFC →Query Access FRequency      Cost
QCT →Query Cost Table
SQC → Selection Query Cost
IL →Query Information List
QIP →Query Information Parameters
NRIL →Number of Rows in query Information List
w1, w2 and w3 →Weighted constant values in between 0-1
Algorithm:
begin:
Repeat for I ←0 to NRIL -1
QIP   ← IL [i]
AF    ← QIP
 PT   ← QIP
 SP  ← QIP
AFC   ←AF \/MAF
PTC   ←PT/MPT
SPC   ←SP/MSP
QCT    ← AFC
QCT    ← PTC
QCT    ← SPC
 [Find selection cost]
Repeat for I←1 to QCT
SQC = w1*AFC+w2*PTC+w3 (1-SPC)
QCT ←SQC
end repeat
End
```

Materialized view selection by using (PSO) algorithm: The fourth step of the algorithm above is used to find the best queries that need to be materialized for optimizing the complex query processing time. After each query selection cost is calculated, the next step is using Particle Swarm Optimization (PSO) algorithm in order to find the appropriate materialized view as shown in Algorithm 5.

Algorithm 5; Selection materialized view using Particle Swarm Optimization (PSO) algorithm:

```
 Input: tables, views, query
 Output: best representation of query
1. Initialize for each of the N particles
   a. Initialize position xi
   b. Set particle best position Pi (0) = xi (0)
   c. Evaluate fitness value of each particle and the best
      value is assigned to global best
2. Repeat step 2 until the stopping criteria is met
   a. Compare current fitness value of each particle with its pbest and update
      pbest if needed
   b. Compare gbest with each particle's current pbest and update gbest if
      needed
   c. Update velocity of the particle
3. Global best particle is the best optimized solution
```

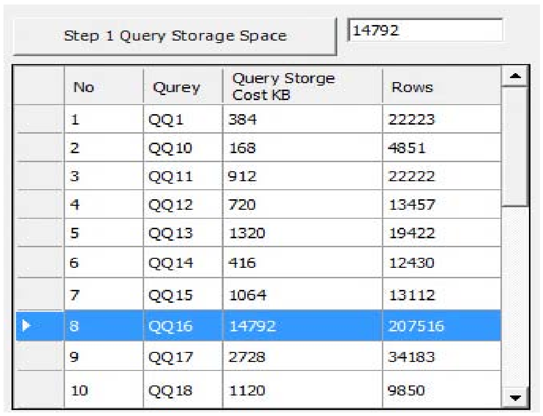
Select best materialized view: The 5 step of the algorithm is used to select materialized view that has low processing time, good query response and low storage cost. Selection minimum value using summation of all the selection queries cost using (PSO) algorithm divided by number of selected queries, then deducting the value of the cost each query of minimum query cost to determine the error rate for queries cost are then build the MV for the selected query by selection less error rate. They are selected best query that need to be materialized to optimize the complex query processing time using Algorithm 6.

Algorithm 6; This algorithm is used to selection best MV:

```
Assumptions:
SQ   → Selected Query
BSMV →Best Select MV
ER   → Error Rate
ERT  →Error Rate Table
QCT  →Query Cost Table
  SQS → Query Select cost of Swarm
LQC → Less Query Cost
N    → Number of rows in query
  E → Number of rows in Error rate
   QVL →Query Value Location
QC   → Query Cost
Algorithm:
[Selection less query cost ]
LQC = Σ SQS/N               Repeat for i ← 0 to N-1
SQS←QCT [i]
ER²← LQC-SQS
End repeat
[ Selection best MV ]
Repeat for j ← 0 to E-1
ER²← ERT [ j]
if (ER ≥ LQC) then
[Build the MV for the selected query (BSMV)]
BSMV ←QVL+SQ+QC
else
Discard the query
End if
End repeat
```

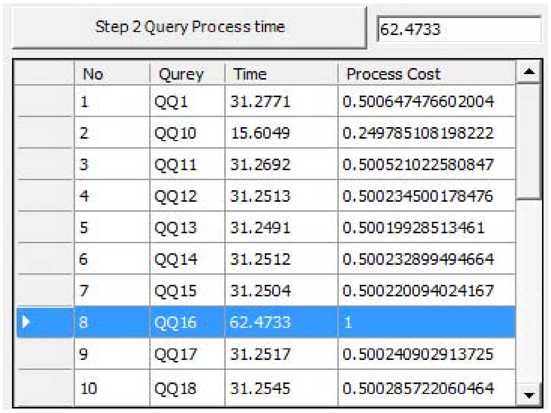
RESULTS AND DISCUSSION

Implementation and results of suggested algorithm (BS-MV): We start our data warehouse by calling the



No	Qurey	Query Storge Cost KB	Rows
1	QQ1	384	22223
2	QQ10	168	4851
3	QQ11	912	22222
4	QQ12	720	13457
5	QQ13	1320	19422
6	QQ14	416	12430
7	QQ15	1064	13112
8	QQ16	14792	207516
9	QQ17	2728	34183
10	QQ18	1120	9850

Fig. 4: Interface find candidate query storage space



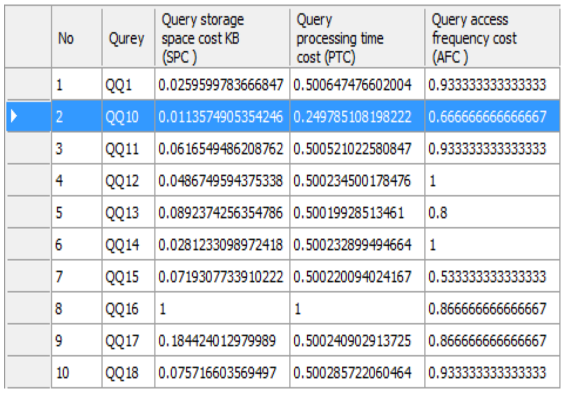
No	Qurey	Time	Process Cost
1	QQ1	31.2771	0.500647476602004
2	QQ10	15.6049	0.249785108198222
3	QQ11	31.2692	0.500521022580847
4	QQ12	31.2513	0.500234500178476
5	QQ13	31.2491	0.50019928513461
6	QQ14	31.2512	0.500232899494664
7	QQ15	31.2504	0.500220094024167
8	QQ16	62.4733	1
9	QQ17	31.2517	0.500240902913725
10	QQ18	31.2545	0.500285722060464

Fig. 5: Interface find candidate query processing time

tables according to our prototype "company" which contains on several tables (items, supplier invoice details, invoices details and warehouse) and in this study, we suppose there are 25 complexes SQL OLAP queries with operations aggregation like (COUNT, SUM, MIN, MAX), join, selection, filtering operation like (using condition where) and GROUP BY operation doing to select data from tables in the company system.

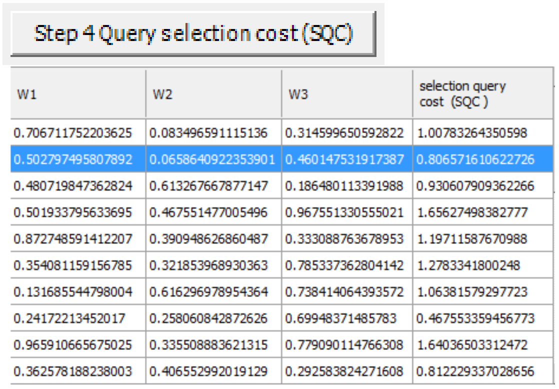
Interface implementing the algorithm Selection parameter of queries: In this study of BS-MV algorithm, the Storage Space (SP) of each query has been calculated by using button "query storage space" and then stores the biggest value of storage space from all queries. Figure 4 which shows how calculate storage space.

Then query Processing Time (PT) of each query has been calculated by using button "query processing time" and then stores the biggest value of processing time from all queries. Figure 5 which shows how to calculate query processing time.



No	Qurey	Query storage space cost KB (SPC)	Query processing time cost (PTC)	Query access frequency cost (AFC)
1	QQ1	0.025959978366847	0.500647476602004	0.933333333333333
2	QQ10	0.0113574905354246	0.249785108198222	0.666666666666667
3	QQ11	0.0616549486208762	0.500521022580847	0.933333333333333
4	QQ12	0.0486749594375338	0.500234500178476	1
5	QQ13	0.0892374256354786	0.50019928513461	0.8
6	QQ14	0.0281233098972418	0.500232899494664	1
7	QQ15	0.0719307733910222	0.500220094024167	0.533333333333333
8	QQ16	1	1	0.866666666666667
9	QQ17	0.184424012979989	0.500240902913725	0.866666666666667
10	QQ18	0.075716603569497	0.500285722060464	0.933333333333333

Fig. 6: Interface calculates the cost for each query



W1	W2	W3	selection query cost (SQC)
0.706711752203625	0.083496591115136	0.314599650592822	1.00783264350598
0.502797495807892	0.0658640922353901	0.460147531917387	0.806571610622726
0.480719847362824	0.613267667877147	0.186480113391988	0.930607909362266
0.501933795633695	0.467551477005496	0.967551330555021	1.65627498382777
0.872748591412207	0.390948626860487	0.333088763678953	1.19711587670988
0.354081159156785	0.321853968930363	0.785337362804142	1.2783341800248
0.131685544798004	0.616296978954364	0.738414064393572	1.06381579297723
0.24172213452017	0.258060842872626	0.69948371485783	0.467553359456773
0.965910665675025	0.335508883621315	0.779090114766308	1.64036503312472
0.362578188238003	0.406552992019129	0.292583824271608	0.812229337028656

Fig. 7: Shows how to calculate the cost of MV for each query

Then query Access Frequency (AF) of each query has been calculated through how many times queries are being fired on a particular cube, it is like 4 out of 20 times this query is being fired.

Implementing the algorithm selection cost of queries: In this study of BS-MV algorithm, the cost for each query selection (Frequency Cost (FC), Processing Cost (PC), Storage cost (SP)) has been calculated through the divided the values query access frequency for each query, on the maximum query frequency of all the frequencies as well as for Processing Cost (PC) and storage cost. Figure 6 shows how to calculate the cost for each query.

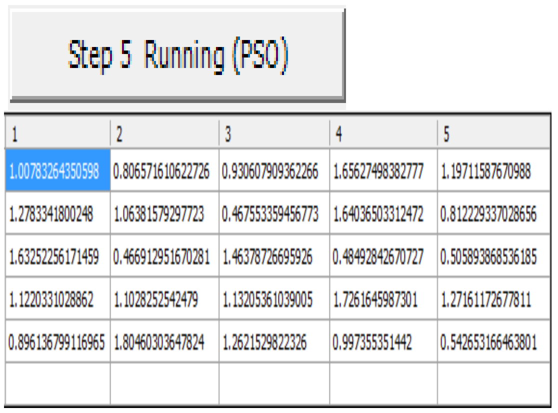
By using button "Query Selection Cost (SQC)" the cost of materialized view for each query selection has been calculated through following equation:

$$[SQC = w1*AFC+w2*PTC+w3\ (1-SPC]$$

Figure 7 shows how to calculate the cost of materialized view for each query.

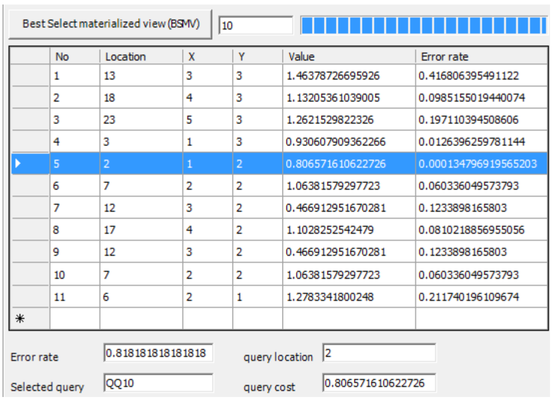
Table 1: Execution OLAP query

Base table in data warehouse	Materialized views
SELECT*FROM New_TBL_WAREHOUSE where [DetailID] = 649462	SELECT*FROM QQ1 where [DetailID] = 649452
SELECT*FROM New_TBL_WAREHOUSE where [Nun] = 96	SELECT*FROM QQ10 where [Nun] = 96
SELECT*FROM New_TBL_WAREHOUSE where [Itm] = 5850	SELECT*FROM QQ12 where [Itm] = 5850
SELECT*FROM New_TBL_WAREHOUSE where [Nun] = 7819	SELECT*FROM QQ13 where [Nun] = 7819
SELECT*FROM New_TBL_WAREHOUSE where [Nun] = 6374	SELECT*FROM QQ16 where [Nun] = 6374
SELECT*FROM New_TBL_WAREHOUSE where [Nun] = 5043	SELECT*FROM QQ19 where [Nun] = 5043
SELECT*FROM New_TBL_WAREHOUSE where [DetailID2] = 5431827	SELECT*FROM QQ2 where [DetailD2] = 5431827
SELECT*FROM New_TBL_WAREHOUSE where [Nun] = 73	SELECT*FROM QQ20 where [Nun] = 73
SELECT*FROM New_TBL_WAREHOUSE where [Nun] = 676	SELECT*FROM QQ9 where [Nun] = 676
SELECT*FROM New_TBL_WAREHOUSE where [Nun] = 2874	SELECT*FROM QQ6 where [Nun] = 2874
SELECT*FROM New_TBL_WAREHOUSE where [Nun] = 8680	SELECT*FROM QQ3 where [Nun] = 8680



1	2	3	4	5
1.00783264350598	0.80657161062272 6	0.930607909362266	1.65627498382777	1.19711587670988
1.2783341800248	1.06381579297723	0.467553359456773	1.64036503312472	0.812229337028656
1.63252256171459	0.466912951670281	1.46378726695926	0.484928426707 27	0.505893868536185
1.1220331028862	1.1028252542479	1.13205361039005	1.7261645987301	1.27161172677811
0.896136799116965	1.80460303647824	1.2621529822326	0.997355351442	0.542653166463801

Fig. 8: Interface two-dimensional matrix of the all queries



	No	Location	X	Y	Value	Error rate
	1	13	3	3	1.46378726695926	0.416806395491122
	2	18	4	3	1.13205361039005	0.0985155019440074
	3	23	5	3	1.2621529822326	0.197110394508606
	4	3	1	3	0.930607909362266	0.0126396259781144
▶	5	2	1	2	0.806571610622726	0.000134796919565203
	6	7	2	2	1.06381579297723	0.060336049573793
	7	12	3	2	0.466912951670281	0.1233898165803
	8	17	4	2	1.1028252542479	0.0810218856955056
	9	12	3	2	0.466912951670281	0.1233898165803
	10	7	2	2	1.06381579297723	0.060336049573793
	11	6	2	1	1.2783341800248	0.211740196109674
*						

Error rate: 0.818181818181818
query location: 2
Selected query: QQ10
query cost: 0.806571610622726

Fig. 9: Select best the materialized view

Implementing the algorithm MVS using (PSO) algorithm: In this study of BS-MV algorithm, the best queries to optimize the complex query processing time using Swarm Optimization (PSO) algorithm have been found. By using button "running PSO", the (PSO) algorithm starts work where all queries are taken in the form of two-dimensional matrix. Figure 8 shows two-dimensional matrix of all queries.

Implementing the algorithm selection best materialized view: After selecting the best queries by (PSO) algorithm. In this study of BS-MV algorithm are used to select materialized view having good query response, low processing and storage cost by using the button "BSMV". Where the values of cost are collected for all queries divided on the number of rows to find out the minimum error ratio, then it stored in label "error ratio". After that the cost of each query has been subtracted from the value of least error ratio then the results have been square to get them positive. After that and according to the supposed algorithm, the positions of least error ratio have been chosen from all queries and it saved in the label of "query location". finally, the query that has less cost and high frequency, according to the selected position for less error ratio will be stored in "select query" label as well as the chosen query cost will be saved in "query cost" label. Figure 9 shows select best the materialized view.

Response time of query in the materialized view: The query response time in the OLAP and decision support systems is critical and very important. Therefore, the implementation of the query on the summary table (materialized views) provides us with fast response time and speeds up decision making, for example eleven of the complex queries implemented outside the base tables in the data warehouse as show in Table 1.

Compute ratio of query response time and compare it with response time to the same queries on the materialized views ratio of implementation of the query on the base table takes eleven times more than time of the query implementation on the MV as show in Table 2.

After complete comparison execution time of query, the performance of query by direct access and access through materialized view are represented in Fig. 10.

From the performance any one judge the efficiency of MV queries over the queries directly access through data warehouse. The response time of queries through MV access were found 0092 msec while through direct

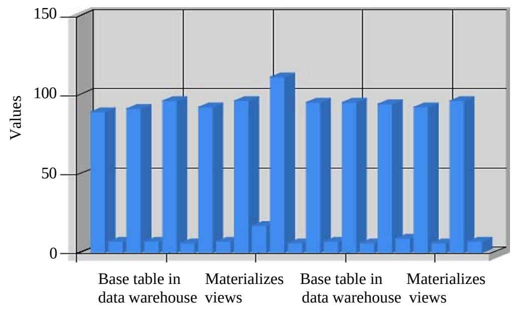


Fig. 10: The performance of query by direct access and access through materialized view

Table 2: Execution time of the query

	Response time of the query	
No. of queries	Base table in DW	Materialized views
1	0.0089	0.0007
2	0.0090	0.0008
3	0.0096	0.0006
4	0.0092	0.0009
5	0.0096	0.0017
6	0.0111	0.0008
7	0.0093	0.0007
8	0.0095	0.0008
9	0.0094	0.0009
10	0.0090	0.0006
11	0.0093	0.0007
Total time	0.1069	0.0092
Average	0.1039/0.0092 = 11.293	

access queries were found 1039 msec. And hence efficiency of queries through MVs access over direct access:

Efficiency = (Direct access of queries-MVs access of queries)/MVs access of queries $\times$ 100 = (1039-0092)/0092$\times$100 = 1029.34%

i.e., 1029.34% more efficient than access over direct access. This study gives the idea regarding best view selection for materialization and the effective incremental batch approach for materialized view maintenance.

The essential constraints that the proposed methodology determines for materialize view selection are: storage space, query frequency and the time of query processing. These queries are more beneficial using combination of query frequency, processing and storage cost for the creation of materialized view so as to achieve the quick query processing time.

Particle Swarm Optimization (PSO) algorithm compared to other algorithms is very simple to use and requires very few parameters and basic operators. Also, in terms of memory and speed, it is computationally inexpensive. That is highly efficient to find a stable solution from among multiple options for solutions.

CONCLUSION

Choosing which view will be selected first to be materialized in the DW is the DW administrator/ designer's main problem. Maintain materialized views for each query impractical as the MV are realized physical table until the disk-space requirements and thus the consumption is very large and/or large update cost having. A possible solution is to choose a group of derived views to materialize, that decrease the summation of total response time of the selected views. This study gives the idea regarding how to select a most important materialized view with the help of various major parameters like: query frequency cost, query storage cost and query processing cost. We have implemented algorithm Particle Swarm Optimization (PSO) which views are more valuable for the creation of materialized view so as to achieve the good query performance. Results obtained during the current work indicate the query optimization through the decreasing of query response time.

RECOMMENDATION

The future research will explore techniques to choose other structures (e.g., Join materialized views and indexes) for database design in addition to materialized views.

REFERENCES

Gosain, A., 2016. Materialized cube selection using particle swarm optimization algorithm. Procedia Comput. Sci., 79: 2-7.

Karde, P. and V. Thakare, 2010. Selection of materialized views using query optimization in database management: An efficient methodology. Intl. J. Manage. Syst., 2: 116-130.

Kurzadkar, S. and A. Bajpayee, 2015. Anatomization of miscellaneous approaches for selection and maintenance of Materialized view. Proceedings of the 2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO), January 9-10, 2015, IEEE, Coimbatore, India, ISBN:978-1-4799-6481-9, pp: 1-5.

Marini, F. and B. Walczak, 2015. Particle Swarm Optimization (PSO): A tutorial. Chemom. Intell. Lab. Syst., 149: 153-165.

Mohod, A. and M. Chaudhari, 2013. Efficient algorithms for materialized view selection in data warehousing environment. Intl. J. Comput. Sci. Netw., 2: 71-75.

Nalini, T., A. Kumaravel and K. Rangarajan, 2012. A comparative study analysis of materialized view for selection cost. Intl. J. Comput. Sci. Eng. Surv., 3: 13-22.

Rashid, A.B. and M.S. Islam, 2010. An incremental view materialization approach in ORDBMS. Proceedings of the 2010 International Conference on Recent Trends in Information, Telecommunication and Computing (ITC), March 12-13, 2010, IEEE, Kochi, India, ISBN:978-1-4244-5956-8, pp: 105-109.

Sainath, K., 2012. Analysis and evaluation of data mining integration into data warehousing. PhD Thesis, San Diego State University, San Diego, California, USA.