International Journal of Modern Trends in Engineering and Research
www.ijmter.com
e-ISSN No.:2349-9745, Date: 2-4 July, 2015

A Survey on Iceberg Query Evaluation strategies

Kale Sarika Prakash[1], P.M.J.Prathap[2]

[1] Computer Science and engineering, St.Peters University,Chennai,sarikazaware@gmail.com [2]Information Technology,RMD Engineering College,Chennai,drjoeepratap@gmail.com

Abstract—Extracting small set of data from large or huge database is a challenge in front of data warehouse (DW) systems. The queries typically executed on data warehouse is of the nature aggregation function followed by having clause. The queries which are having this type of nature are called as Iceberg(IB) queries. Present database system execute it just like normal query so it take more time to execute. It is challenging task to extract interesting and important information quickly from huge database. Lots of research has been done to increase the speed of IB query .Initially researchers uses tuple scan approach to execute IB query which is time consuming and require more memory . To overcome these problems researchers proposed IB query evaluation using Bitmap Indexing techniques. This technique avoid complete table scan so time required to execute IB query is reduced and memory requirement is also reduced. But these techniques suffer from empty Bitwise AND operation and Bitwise XOR operation. This affects the performance of IB query. The challenge in front of current researcher in this field is to overcome the drawback of IB query evaluation using Bitmap Indexing techniques .In our research by using preprocessing of bitmap vector we are trying to overcome problems occurred in previous research.

Keywords- Data warehouse(DW),,Iceberg(IB) queries, bitmap index, Bitwise AND operation, Bitwise XOR operation, Aggregate function: COUNT,SUM,MIN,MAX ,AVG.

I. INTRODUCTION

The users of the DW are the decision makers, business analyst and knowledge workers organization. They make use of data from data warehouse to predict some issues about their business. Accordingly they take decisions about their business. Once they have taken decision they concentrate on implementation of the same. The information which they collect from data warehouse is small information from huge dataset. To extract such a type of data the query executed is of the nature aggregation of some value on some specified condition or threshold. This type of query is called as IB query.

IB queries were first studied by scientist named Min Fang et.al [1]. According to him, an iceberg query has lot of application in data-warehousing, data mining and information retrieval systems.Iceberg Query defined as the type of query which perform aggregation function on some set of attribute followed by having clause on some condition or on threshold value. Because of having clause the aggregate function which does not satisfy threshold condition will get eliminate from the result. In this way the output of iceberg query is small set of data from huge data set as input. Because of this type of nature it is called as iceberg .

Syntax of an Iceberg queries on a relational table R1 (C1, C2… Cn) is stated below:

SELECT Ci, Cj, …, Cm, AGG(*) FROM R1,
GROUP BY Ci, Cj…, Cm,
HAVING AGG (*) > = T

Where Ci, Cj,…,Cm represents a subset of attributes in R1 which are aggregate attributes. AGG represents an aggregation function such as COUNT, SUM,AVG, MIN and MAX. The greater than

equal to(>=),less than or equal to (<=) or is equal to (= =) are a special symbol used as a comparison predicate in having clause.

Due to threshold condition iceberg query returns very small distinct data set as output which looks like the tip of an iceberg. As output of iceberg query is very small compare to input ,it can answer quickly even on huge data set. However, current database systems do not fully take advantage of this feature of iceberg query. The relational database systems like Oracle, SQL server,DB2,Sybase, MySQL, PostgreSQL, and column-oriented databases are all using general aggregation algorithms [2], [3] and [4] to execute iceberg query. They did not consider it as special query and so the time required to execute such queries on these databases are more. They answer iceberg queries by first aggregating all tuples and then evaluating the HAVING clause to generate final iceberg result.Existing optimization techniques for processing iceberg queries [1], [5] can be categorized as the tuple-scan-based method , which requires minimum one table scan to read data from disk. They concentrate on reducing the number of table scan/passes required when dataset size is large. They did not make use of iceberg query property for efficient query processing. Due to this a tuple-scan-based method generally takes a long time to answer iceberg queries, when the dataset is very large.

Ferro et al. [6] designed a two-level bitmap index which can be leveraged for processing iceberg queries. But, it suffers from the massive empty bitwise-AND results problem. The research proposes in [7] provides dynamic pruning and vector alignment strategy to overcome this empty bitwise AND operation problem. But empty Bitwise XOR operation and futile queue pushing problem occurred in this research. The researchers [8][9] tries to solve this empty bitwise XOR problem. But they concentrate only on COUNT aggregate functions and also suffers from Bitwise AND operation. It is a challenge in front of researcher to overcome drawbacks of [7][8][9] and proposes a efficient IB query evaluation technique. The remaining sections of this paper are structured as follows: In Section II we discussed Literature review of IB query evaluation technique. Section III describes comparative study between different algorithms and section IV concludes the paper.

II. LITERATURE REVIEW OF ICEBERG QUERY EVALUATION TECHNIQUES

DW is a subject oriented, time variant, integrated and Nonvolatile. It is a set of huge and historical database. DW is not update frequently. Information extracted from DW is summarized and consolidated. To extract this information the queries to be executing on the warehouse is of the nature of IB query. As DW is not update frequently so IB query execution using bitmap indexing technique is appropriate for DW.

In this section we are highlighting the iceberg query execution strategies developed by different researchers and application of bitmap index to execute iceberg query.

The IB query processing is first described by Min Fang[1] in 1998.In this author proposed techniques for threshold which are the building block for their algorithm. These techniques are: Sampling and Coarse count/Probabilistic count.

IB query processing is proposed by [5] in 2000. This research concentrate on evaluation of AVERAGE function in Iceberg query by using partitioning technique. In this they proposed Basic Partitioning(BAP) and Postponed Partitioning (POP)algorithm to compute AVARAGE function of IB query. Main concept behind these algorithms is to partition database logically to find candidate set of tuple which are satisfying threshold condition of IB query. They work in two phases-Partition relation and selecting candidate and second phase which computes average value of candidate set. It has been proved that performance of above algorithms depends upon data order of tuple and memory size. If the table is in sorting order then the performance of above strategies are excellent without regards to memory size.

Framework for IB query processing has proposed by[10] in 2004. In this paper researchers developed different Iceberg query processing technique and suggest algorithms that can be implemented in query optimizer to choose more relevant algorithm for IB Query Evaluation.
Above mentioned approaches suggested by[1][5][10] are comes under the group of tuple scan based methods. None of the above research consider properties of Iceberg query for its evaluation. These algorithms focus on reducing number of tuple scan but none of them makes use of iceberg query property.
Bitmap index is usually a better choice for querying the massive and multidimensional scientific datasets. It has significantly increases data accessing time and reduced the query response time on both high and low cardinality values with a number of techniques [11]. Generating the bit map index of attribute will not affect on the performance of query because generated bitmap by database system is in compressed mode [12]. Use of bitmap index to execute iceberg query is important factor as it helps to evaluate query in following manner:
1.Bit map avoids massive disk access on complete tuple. It access bitmap index of attributes of GROUP BY clause only.
2.It operate on bits rather actual tuple values. Bitwise operations are very fast to execute as they directly supported by hardware.[13]
3.Bit map can leverage the antimonotone property of IB query very easily. This helps for index pruning in IB Query Evaluation.[14,15]
By considering applicability of Bitmap indexing [6] in 2009 make use of bitmap indexing for IB Query Evaluation. This is the first research which makes use of Bitmap Indexing for IQ evaluation. This research explain detail algorithm for COUNT aggregate function and proposes extension of the same for SUM,MAX and MIN function. But this strategy suffers from empty bit wise AND result problem. This problem is minimized by [7] using dynamic pruning and vector alignment approaches .Both this approaches makes use of the antimonotone property of iceberg query.
Research [8] try to handle empty X-OR operation problem but did not able to solve fruitless Bit wise AND operation problem. Similarly research [9],[16] and [17] also tries to minimize fruitless AND operations. They also not able to minimize XOR operation. All these uses the indexing concept on number of 1 present in bitmap vector. For ordering queue they uses number of 1 bit present in bitmap vector. Accordingly they take decision to prune the bitmap vector.
All above [7],[8],[9],[16] and [17] uses priority queue concept but in this futile queue pushing problem occurs. The queue maintenance cost is involved in each pass. All faces fruitless bitwise AND and XOR problem. To handle all above problems we are proposing novel Iceberg query evaluation strategy. Based on the query the attribute bitmap vector is pre-processed. Due to this the vectors which are having highest probability of to be a part of final answer are evaluated first. Then comparison of the result with next highest vector is done so the chances of pruning the vector in advance will be more. This process avoids fruitless Bitwise AND and XOR operations. None of above research [7],[9],[16] and [17] work on other aggregate functions like MIN,MAX and AVERAGE

III. COMPARATIVE STUDY BETWEEN DIFFERENT ALGORITHMS

In this section we are showing the comparative analysis of strategies developed in [7],[8] and our proposed bitmap preprocessing strategy. We are considering examples given in [7],[8] and we have solved it with Bitmap Indexing[6],Dynamic Pruning Strategy and Vector Alignment Strategy[7][8] and we compare their results with our proposed Bitmap preprocessing strategy.

SELECT X,Y,COUNT(*) FROM R1 GROUP BY X,Y HAVING COUNT(*)>3;

Table 1 Relation R1

X	Y	Z
X1	Y2	500
X3	Y1	600
X2	Y2	1000
X3	Y1	500
X2	Y2	1000
X3	Y1	600
X1	Y2	500
X2	Y2	600
X1	Y1	1000
X2	Y2	500
X3	Y1	600
X2	Y3	1000
X1	Y1	600
X3	Y3	1000
X3	Y2	1000

Table 2 Bbitmap Indices for x, y

X1	X2	X3	Y1	Y2	Y3
1	0	0	0	1	0
0	0	1	1	0	0
0	1	0	0	1	0
0	0	1	1	0	0
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
1	0	0	1	0	0
0	1	0	0	1	0
0	0	1	1	0	0
0	1	0	0	0	1
1	0	0	1	0	0
0	0	1	0	0	1
0	0	1	0	1	0

After solving we calculate number of Bitwise AND operation required by each strategy. It is shown in following table 3. By this analysis we conclude that by solving above queries manually we require less bitwise AND operations.

Table 3 Comparison Table

Logical Operation / IB Query Evaluation	Number of Bitwise AND Operations required
Bitmap Indexing	9
Dynamic Pruning Strategy	5
Vector Alignment Strategy	5
Proposed Strategy: Bitmap Preprocessing	3

CONCLUSION

Decision support and knowledge discovery systems often compute aggregate values of interesting attributes by processing a huge amount of data in very large databases and/or warehouses. In particular, it executes a query which contain aggregation function followed by having clause such a query is called as Iceberg query. It is a special type of aggregation query that computes aggregate values above a user-provided threshold. Usually, only a small number of results will satisfy the threshold condition. Means we have to extract small information from huge database. The huge research has been done on this area which is of tuple scan based and bit map indexing based. Out of this bitmap index based is efficient compare to tuple scan based. But still bitmap index based suffer

from empty bitwise AND and bitwise XOR operation problem and they concentrate only on COUNT and SUM aggregate function. It is challenge in front of researcher in this area to find solution to overcome above problem and develop a framework for other aggregate functions like MIN,MAX and AVERAGE. In our research we are concentrating on above aspects and we are using Bitmap Preprocessing strategy to solve above problem. Our research will support MIN,MAX,COUNT and SUM aggregate function as they all support antimonotone property of IB Query. But AVERAGE function is not supporting antimonotone property of IB Query as it require all the tuples from huge dataset. So our system will not support AVERAGE function.

REFERENCES

[1] M. Fang, N. Shivakumar, H. Garcia-Molina, R. Motwani, and J.D. Ullman, "Computing Iceberg Queries Efficiently," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 299-310, 1998.

[2] G. Graefe, "Query Evaluation Techniques for Large Databases," ACM Computing Surveys, vol. 25, no. 2, pp. 73-170, 1993 .

[3] W.P. Yan and P.-_A. Larson, "Data Reduction through Early Grouping," Proc. Conf. Centre for Advanced Studies on Collaborative Research (CASCON), 1994.

[4] P.A. Larson, "Grouping and Duplicate Elimination: Benefits of Early Aggregation," Technical Report MSR-TR-97-36, Microsoft Research,1997.

[5] J. Bae and S. Lee, "Partitioning Algorithms for the Computation of Average Iceberg Queries," Proc. Second Int'l Conf. Data Warehousing and Knowledge Discovery (DaWaK), 2000.

[6] A. Ferro, R. Giugno, P.L. Puglisi, and A. Pulvirenti, "BitCube: A Bottom-Up Cubing Engineering," Proc. Int'l Conf. Data Warehousing and Knowledge Discovery (DaWaK), pp. 189-203, 2009.

[7] Bin He, Hui-I Hsiao, Ziyang Liu, Yu Huang and Yi Chen, "Efficient Iceberg Query Evaluation Using Compressed Bitmap Index", IEEE Transactions On Knowledge and Data Engineering, vol 24, issue 9, sept 2011, pp.1570-1589.

[8] C.V.Guru Rao, V. Shankar,"Efficient Iceberg Query Evaluation Using Compressed Bitmap Index by Deferring Bitwise- XOR Operations" 978-1- 4673-4529-3/12/$31.00c 2012 IEEE

[9] C.V.Guru Rao, V. Shankar, "Computing Iceberg Queries Efficiently Using Bitmap Index Positions"DOI: 10.1190/ICHCI- IEEE.2013. Publication Year: 2013 ,Page(s): 1 – 6.

[10] K.P. Leela, P.M. Tolani, and J.R. Haritsa, "On Incorporating Iceberg Queries in Query Processors," Proc. Int'l Conf. Database Systems for Advances Applications (DASFAA), pp. 431-442, 2004.

[11] Ying Mei, Kaifan Ji*, Feng Wang," A Survey on Bitmap Index Technologies for Large-scale Data Retrieval" 978-1-4799-2808-8/13$26.00 © 2013 IEEE

[12] F. Deliege and T.B. Pedersen, "Position List Word Aligned Hybrid: Optimizing Space and Performance for Compressed Bitmaps," Proc Int'l Conf. Extending Database Technology (EDBT), pp. 228-239, 2010

[13] ONeil, E., ONeil, P., Wu, K, "Bitmap index design choices and their performance implications",Database Engineering and application Symposium ,2007, IDEAS, 2007.

[14] K. Stockinger, J. Cieslewicz, K. Wu, D. Rotem, and A. Shoshani."Using Bitmap Index for Joint Queries on Structured and Text Data". Annals of Information Systems, 3:123, 2009.

[15] Kesheng Wu, Ekow J. Otoo and Arie Shoshani Lawrence Berkeley National Laboratory Berkeley,"Compressing BitmapIndexes for Faster Search Operations", IEEE Computer Society,2002.

[16] Sandeep Reddy Chidirala , Shankar Vuppu, " Efficient Evaluation of Iceberg Queries Using Priority Queues "© 2013, IJARCSSE, Volume 3, Issue 9, September 2013 ISSN: 2277 128X.

[17] Ankam Praveen, Vuppu Shankar ,"Array Indexing Technique: AnsweringThe Iceberg Queries Efficiently " IJES,Volume 2,Issue8,Pages: 41- 49,2013, ISSN(e): 2319 – 1813 ISSN(p): 2319 –1805.

