International Journal of Engineering & Technology, 7 (2.7) (2018) 791-793

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research Paper

A Survey: Computing Iceberg Queries

Pallam Ravi[1]*, D. Haritha [2], P.Niranjan [3]

[1]Scholor KLEF ,Guntur
[2]Professor in KLEF
[3]Professor in KITS Warangal
**E-mail: Satishpallam@gmail.com*

Abstract

Knowledge discovery, tasks are deals with huge no of records. Queries are needed to identify unique attributes values and their aggregate that is above a predefined threshold from this huge number of records. This type of queries is called iceberg queries. Iceberg queries requires huge amount of main memory and takes longer time to answer the query. As computer system has limited amount of main memory, the processing of iceberg queries is a challenging task. This paper discusses different methods that are in literature for processing iceberg queries , we also explore pros and cons of these methods and future scope.

Keywords: *Aggregation functions, Bitmap Index , Iceberg Queries, Anti monotone , Bit map Numbers .*

1. Introduction

For understand trends in business needs to find insights, for it needs to perform some analytics on business data(historical),it reveals the relations among data features(attributes),this process called as Knowledge discovery process),for identify the relation among attributes need summarized information, which calculated with aggregation functions on one or more attributes ,we interested on those have above the user provided threshold value, this type of queries called iceberg queries. Aggregate value above threshold value gives more important information.
For example in college principal wants find relationships between section ,course on students marks data,he his interested in which section ,course get pass more than 80 students are pass from a section, this will represent as iceberg query as :select section, course, COUNT(*) from marks group by section, course where COUNT(*)>80 and result="pass". Here COUNT() is aggregation function, threshold value is 80,the results set include only section, course groups exceeding 80.
The general form of iceberg queries as follow
SELECT A1,A2 … An FROM R
GROUP BY Ai,Aj,… Ak
HAVING AGG() >T

Where R is relation which contain A1,A2 … An attributes and T is user provided threshold
Example 1: the college principal wants which branches have above 50 students admissions from admission data base
Select Branch_Name, COUNT() from A
GROUP BY Branch_name
HAVING COUNT()>50
Iceberg queries are widely used many application using such as data warehouse , data mining, multimedia databases and embedded systems
The iceberg query characteristics 1) aggregation function on one or more attributes computation 2)execute on large data set 3) contain large unique attributes combination (domain size) 4)small result set which aggregate values is above threshold.
Problems facing iceberg queries while executing are 1) need to execute within limited memory means domain size is greater than memory 2) Aggregation values computation takes huge time . The global research objective of iceberg queries execution is reduce time takes to execute query within limited memory only
Rest of paper are structure as follows, in section2 discussed related work done on iceberg queries, different authors contribution with respective aggregation functions, data scan and computing environment discussed in section 3,in section 4 discussed about different method with respective aggregation functions, data scan and computing environment ,section 5 gives future scope in iceberg queries.

2. Related Work

The first study about iceberg queries by fang et al[1] in it coarse count and sampling methods are used to answer the iceberg queries ,but it suffers false negatives and hybrid and multi bucket algorithm proposed in [1],these are extended with probability techniques whag et al [2].
But [1][2] are not work for average queries, Bae and Lee[3] are proposed partition algorithms(BOP and POP) for average iceberg queries computation, disadvantage of these algorithm is many data scans are required, leela et al[4] comparative study using sort merge aggregate, ORACLE and hybrid hash aggregate methods, which reveals sort merge aggregate gives better performance .
Ferro *et al.*[5] use bitmap index which suffer massive empty Bitwise AND operation ,to reduced mass empty bitwise And operations Ferro *et al.*[6] proposed dynamic pruning algorithms, to reduce number of Bitwise and operations , shanker at al proposed different algorithms [7]differing push and pop operations [8],cache based,[9]check point mechanism which reduce number of Bitwise AND operations.

we [10] introduce bitmap number, which generated by use of bitmap index, which use for computing average queries on large amount of data . Reduce i/o operation in iceberg query sarika prakash et al[11] used tracking pointer(LP) and look ahead matching strategy(LMS),Yue Cui et al [12] use p-trees develop efficient computing aggregation function algorithms , shanker et al[13] iceberg queries on distributed databases

Above development on Iceberg query computation goal in direction of usage of aggregation functions properties, type data base scan and computing environment to reach global goal of iceberg queries

3. Author Directions

This is section explore different authors contribution, we classified their contribution as aggregation function properties, type of data scan and computing environment

3.1 Aggregation functions properties:

Aggregation fuction like AVERAGE ,SUM,MAX,MIN and COUNT ..etc ,we these are divided into anti-monotone and non anti-montone, anti-monotone Aggregation functions use Apriori[14] property, on anti-monotone are not able to use Apriori property ,Example of anti-monotone aggregation fuctions are SUM,MAX,MIN and COUNT ,for non anti-monotone aggregation functions are AVERAGE,STDIV

Takes advantage of Apriori property for Computing anti-monotone iceberg queries(iceberg query with anti-monotone Aggregation functions) ,by this the pruning of computing Aggregation functions will take place due this reduce time for generate query result set

Non anti monotone aggregation iceberg queries are not takes advantages of threshold on AVERAGE values as SUM,MAX,MIN and COUNT Aggregation function(anti monotone aggregation functions) ,average iceberg queries need to compute AVERAG for all unique grouped attributes ,then apply threshold constraint on these AVERAG values .maintain a counter bucket for each unique grouped attributes. Normally required a counter bucket are not maintain in memory (characteristic of iceberg query) .

[2] proposed a partitioning two methods(BOP &POP).these methods sequentially partitioned data , number of unique values in partition data are less the maximum number counter bucket are handled in memory, each counter bucket have two tuples <sum,count> ,scan partition data, updates sum and count in counter bucket if exits ,else it create a new counter bucket with initialize<value,1> , produce results set by apply threshold constraint on a counter bucket with calculate AVERAGE by use its sum and count values (average=Sum/count), its have two disadvantages those are

1) two times need to compute AVERAGE value per one candidate unique group attributes(one for selecting candidate unique group attributes, other is to decide actual value of candidate meets threshold constraint)
2) many scan on data(equal to number of partitions)

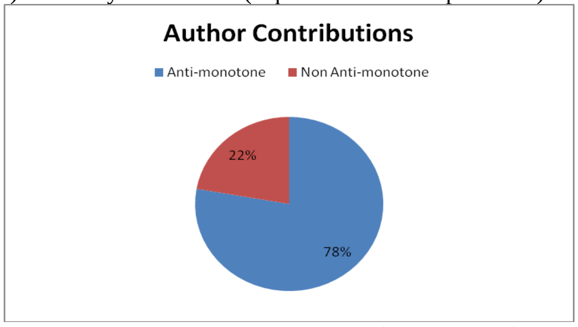

Fig. 1. Authors Contributions with direction of Aggregation functions

3.2 Type of Data Base Scan:

for data scan ,use tuple based and column based scanning data ,in tuple based iceberg query computing is support for small data sets not for large data sets, in column based it works for large datasets due to use bitmap indexing(BI),BI takes lesser memory to represent data, advantage with BI is quick indexed the record values

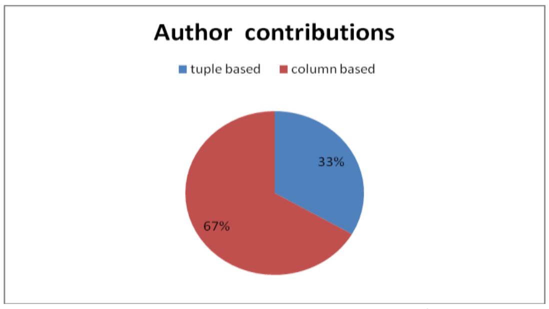

Fig. 2. Authors Contributions with direction of Data Scan

3.3 Computing environment:

The work done so for single processor except shanker et al[15] is focus on distributed environment, using data shipping and query shipping proposed different algorithms

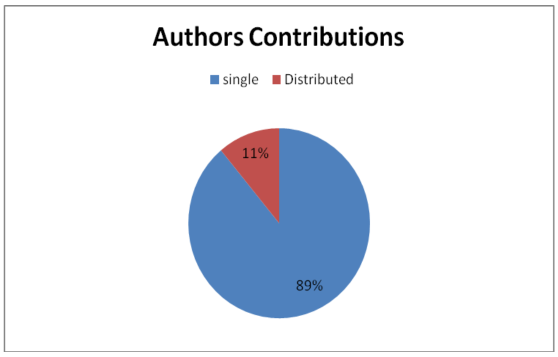

Fig. 2. Authors Contributions with direction of Environment

Table 1: Methods Comparison

Slno	Author	Method	Aggregation function	Type of scan	Environment	Disadvantage
1	fang et al	Coarse-count	Anti-monotone	Tuple	single	False negatives
2	whag et al	probability	Anti-monotone	Tuple	single	False negatives
3	Bae and Lee	Partition based	Non Anti-monotone	Tuple	single	Small data
4	leela et	sort merge aggregate, ORACLE	Anti-monotone	Tuple	single	Small data
5	Ferro *et al*	Dynamic purning	Anti-monotone	column	single	Bitwise AND operations
6	shanker at al	1,differing push and pop operations 2,cache based, 3,check point mechanism	Anti-monotone	column	single	Bitwise AND operations
		Data shipping and	Anti-monotone	column	Distributed	Bitwise AND oper-

		query shiping				ations
7	Pallam Ravi et al	BitMap Numbers	Non Anti-monotone	column	single	Bitwise AND operations
8	Yue Cui	P-trees	Anti-monotone	column	single	
9	Sarika prakash et al	Track pointer & look ahead Matching	Anti-monotone	column	single	

4. Methods Comparisons

We are presenting different method's Advantages and disadvantages, which helps to future scope. Representing method in table 1, which explain author's direction .

5. Conclusion

The paper explored the various methods for processing iceberg queries. This paper mainly focused the iceberg query processing with respect to aggregation function, data scan and computing environment. There is lot of research needed to focus on handling large value of data for non anti-monotone aggregation iceberg queries on single and distributed computing environments.

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]. Whang, K.Y., B.T.V. Zanden and H.M. Taylor, "Alinear-time probabilistic counting algorithm for database applications", ACM Trans. Database Syst., 15: 208-229, 1990.

[3]. J.Bae and S.Lee. "Partitioning algorithms for the computation of average iceberg queries." Proc. Second Intl Conf. Data Warehousing and Knowl-edge Discovery (DaWaK), pp. 276-286, 2000.

[4]. K.P. Leela, P.M. Tolani, and J.R. Haritsa, "On Incorporating Iceberg Queries in Query Processors", Proc. Intl Conf. Data-base Systems for Advances Applications (DASFAA), pp. 431-442, 2004.

[5]. J. Han, J. Pei, G. Dong, and K. Wang, "Efficient Computation of Iceberg Cubes with Complex Measures", Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 1-12, 2001.

[6]. B.He,H-I.Hsia,Z.Liu,Y.Huang and Y.Chen, "Efficent computing Iceberg queries using compresed bitmap index", IEEE TRANSACTION ON KNOWLEDGE AND DATA ENGINEERING,2012

[7]. Vuppu shanker et al ."Effective Iceberg Query Evaluation by Deferring Push and Pop Operations",IJAC, ISSN:2051-0845, Vol.36, Issue.2.2015

[8]. Vuppu shanker et al," Cache Based Evaluation of Iceberg Queries", ICCCT-2014 IEEE,2014

[9]. Vuppu shanker et al, "Answering Iceberg Queries Efficiently Using Check Point Mechanism",IJAC , Vol.46, Issue.2,2015

[10]. Pallam Ravi et al "COMPUTING ICEBERG QUERIES HAVING NON ANTI MONOTONE CONSTRAINS WITH BIT MAP NUMBER", JATIT,Vol. 8. No. 2 -- 2016

[11]. Kale Sarika Prakash et al,"Tracking Pointer and Look Ahead Matching Strategy to Evaluate Iceberg Query",JCS,2017

[12]. Y.Cui, W.Perrizo "Aggregate Function Computation and Iceberg Query-ing in Vertical Database",Computers and Their Applications,2006

[13]. Vuppu shanker et al, "Efficient iceberg evaluation in Distributed databases by Developing Deferred Strategies",2016

[14]. K.S. Beyer and R. Ramakrishnan, "Bottom-Up Computation of Sparse and Iceberg CUBEs", Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 359-370, 1999.

[15]. C.Y. Chan and Y.E. Ioannidis, "Bitmap Index Design and Evaluation", Proc. ACM SIGMOD Intl Conf. Management of Data, 1998.

[16]. F. Deliege and T.B. Pedersen, "Position List Word Aligned Hybrid: Optimizing Space and Performance for Compressed Bitmaps", Proc. Intl Conf. Extending Database Technology (EDBT), pp. 228-239, 2010.2.