C. Poongodi *et al*, International Journal of Computer Science and Mobile Applications,
Vol.2 Issue. 2, February- 2014, pg. 111-117 **ISSN: 2321-8363**

Dynamic Optimization of Generalized SQL Queries with Horizontal Aggregations Using K-Means Clustering

Mrs. C. Poongodi[1], Ms. R. Kalaivani[2]
[1]PG Student, [2]Assistant Professor, Department of Computer Science and Engineering
PRIST University, Trichy District, India
[1] cpoongodimca@gmail.com

Abstract

Data mining is widely used domain for extracting trends or patterns from historical data. However, the databases used by enterprises can't be directly used for data mining. It does mean that Data sets are to be prepared from real world database to make them suitable for particular data mining operations. However, preparing datasets for analyzing data is tedious task as it involves many aggregating columns, complex joins, and SQL queries with sub queries. More over the existing aggregations performed through SQL functions such as MIN, MAX, COUNT, SUM, AVG return a single value output which is not suitable for making datasets meant for data mining. In fact these aggregate functions are generating vertical aggregations. This paper presents techniques to support horizontal aggregations through SQL queries. The result of the queries is the data which is suitable for data mining operations. It does mean that this paper achieves horizontal aggregations through some constructs built that includes SQL queries as well. PIVOT method is much faster method and offers much scalability. Partitioning large set of data, obtained from the result of horizontal aggregation, in to homogeneous cluster is important task in this system. K-means algorithm using SQL is best suited for implementing this operation.

Keywords: Aggregation, Data Mining, Structured query language (SQL), PIVOT, K-means algorithm

I. INTRODUCTION

Horizontal aggregation is new class of function to return aggregated columns in a horizontal layout. Most algorithms require datasets with horizontal layout as input with several records and one variable or dimensions per columns. Managing large data sets without DBMS support can be a difficult task. Trying different subsets of data points and dimensions is more flexible, faster and easier to do inside a relational database with SQL queries than outside with alternative tool. Horizontal aggregation can be performing by using operator, it can easily be implemented inside a query processor, much like a select, project and join. PIVOT operator on tabular data that exchange rows, enable data transformations useful in data modelling, data analysis, and data presentation There are many existing functions and operators for aggregation in Structured Query Language. The most commonly used aggregation is the sum of a column and other aggregation operators return the average, maximum, minimum or row count over groups of rows. All operations for aggregation have many limitations to build large data sets for data mining purposes. Database schemas are also highly normalized for On-Line Transaction Processing (OLTP) systems where data sets that are stored in a relational database or data warehouse. But data mining, statistical or machine learning algorithms generally require aggregated data in summarized form. Data mining algorithm requires suitable input in the form of cross tabular (horizontal) form, significant effort is required to compute aggregations for this purpose. Such effort is due to the amount and complexity of SQL code which needs to be written, optimized and tested.

Data aggregation is a process in which information is gathered and expressed in a summary form, and which is used for purposes such as statistical analysis. A common aggregation purpose is to get more information about particular groups based on specific variables such as age, name, phone number, address, profession, or income. Most algorithms require input as a data set with a

horizontal layout, with several records and one variable or dimension per column. That technique is used with models like clustering, classification, regression and PCA. Dimension used in data mining technique are point dimension.
There are several advantages for horizontal aggregation. First one is horizontal aggregation represent a template to generate SQL code from a data mining tool. This SQL code reduces manual work in the data preparation phase in data mining related project. Second is automatically generated code, which is more efficient than end user written SQL code. Thus datasets for the data mining projects can be created in less time. Third advantage is the data sets can be created entirely inside the DBMS K-means clustering algorithms are used to cluster the attribute, that attribute is the result of horizontal aggregation. The rest of the paper is organized as follows. Next part presents clustering of aggregated dataset and different methods existing for aggregation and Conclusion.

II. Horizontal Aggregations

For describing the methods pertaining to the proposed horizontal aggregations such as PIVOT, CASE and SPJ, input table as shown in Fig. 1 (a), traditional vertical sum aggregation as shown in Fig 1 (b) and horizontal aggregation as shown in Fig. 1 (c) are considered.

K	D1	D2	A
1	3	X	9
2	2	Y	6
3	1	Y	10
4	1	Y	0
5	2	X	1
6	1	X	NULL
7	3	X	8
8	2	X	7

D1	D2	A
1	X	Null
1	Y	10
2	X	8
2	Y	6
3	X	17

D1	D2 X	D2Y
1	NULL	10
2	8	6
3	17	NULL

Fig. 1 – Input table (a), traditional vertical aggregation (b), and horizontal aggregation (c)

As can be seen in fig. 1, input table has some sample data. Traditional vertical sum aggregations are presented in (b) which is the result of SQL SUM function while (c) holds the horizontal aggregation which is the result of SUM function.

Steps Used in All Methods

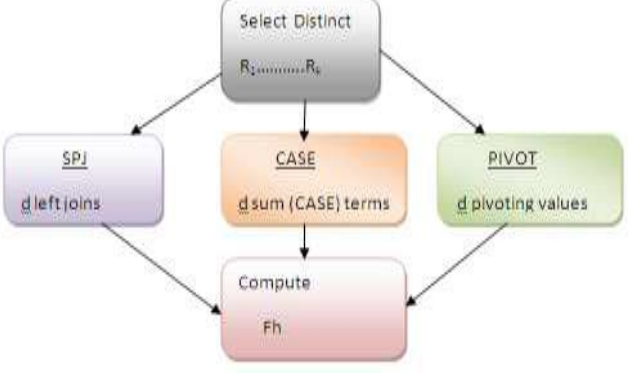


Fig. 2 shows steps on all methods based on input table

As can be seen in fig. 2, for all methods such as SPJ, CASE and PIVOT steps are given. For every method the procedure starts with SELECT query. Afterwards, corresponding operator through underlying construct is applied. Then horizontal aggregation is computed.

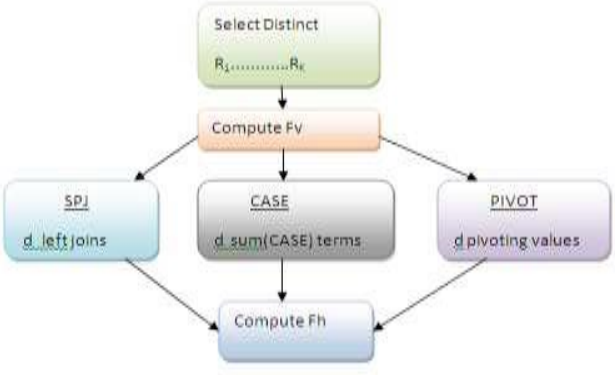


Fig. 3 shows steps on all methods based on table containing results of vertical aggregations

As can be seen in fig. 2, for all methods such as SPJ, CASE and PIVOT steps are given. For every method the procedure starts with SELECT query. Afterwards, corresponding operator through underlying construct is applied. Then horizontal aggregation is computed.

II.1 SPJ Method

This method is based on the relational operators only. In this method one table is created with vertical aggregation for each column. Then all such tables are joined in order to generate a table containing horizontal aggregations. The actual implementation is based on the details given in [18].

II.2 PIVOT Method

This aggregation is based on the PIVOT operator available in RDBMS. As it can provide transpositions, it can be used to evaluate horizontal aggregations. PIVOT operator determines how many columns are required to hold transpose and it can be


```
SELECT L1,..,Lj
,sum(CASE WHEN R1= v11 and .. and Rk = vk1
THEN A ELSE null END)
..
SELECT DISTINCT R1
FROM F; /* produces v1; . . . ; vd */
SELECT
L1; L2; . . . ; Lj
,v1; v2; . . . ; vd
INTO Ft
FROM F
SELECT L1; L2; . . . ; Lj;R1, A
FROM F) Ft
PIVOT(
V (A) FOR R1 in (v1; v2; . . . ; vd)
) AS P;
```

Listing 1 – Shows optimized instructions for PIVOT construct

As can be seen in listing1, the optimized query projects only the columns that participate in computation of horizontal aggregations.

II.3 CASE Method

This construct is built based on the existing CASE struct of SQL. Based on Boolean expression one of the results is returned by CASE construct. It is same as projection/aggregation query from relational point of view. Based on some conjunction of conditions each non key value is given by a function. Here two basic strategies to compute horizontal aggregations. The first strategy is to compute directly from input table. The second approach is to compute vertical aggregation and save the results into temporary table. Then that table is further used to compute horizontal aggregations. The actual implementation is based on the details given in [18].

III. INTEGRATING K-MEANS ALGORITHM WITH HORIZONTAL AGGREGATION

Clustering methods partition a set of objects into clusters such that objects in the same cluster are more similar to each other than objects in different clusters according to some defined criteria. Data mining applications frequently involve categorical data. The biggest advantage of these clustering algorithms is that it is scalable to very large data sets.

Even though the existing system presented the computation of the values for different attributes, it has some drawbacks. In the research of the horizontal aggregation, the existing systems are not well defined for the different fact tables that need better indexing and extraction.

Multiple fact tables: Constructing new data sets within the range of a discrete set of known data points we need different attributes from different fact tables. In many applications one often has a number of data values, obtained by experimentation, which stored on limited number of databases. It is often required to extract the particular useful attributes from the different fact tables and perform aggregation.

K-means: K-means is initialized from some random or approximate solution. Each step assigns each point to its nearest cluster and then points belonging to the same cluster are averaged to get new cluster centroids. Each step successively improves

cluster centroids until they are stable. This is the standard version of K-Means technique used. Optimized K-means computes all Euclidean distances for one point in one I/O, exploits sufficient statistics, and stores the clustering model in a single table. Experiments evaluate performance with large data sets focusing on elapsed time per iteration.
The main issue here addressed is how to make efficient indexing of horizontal aggregation. Initially an aggregation operation is performed horizontal layout are creating by using pivot operator. In this a k-means algorithm are implementing to create datasets with horizontal layout as input.

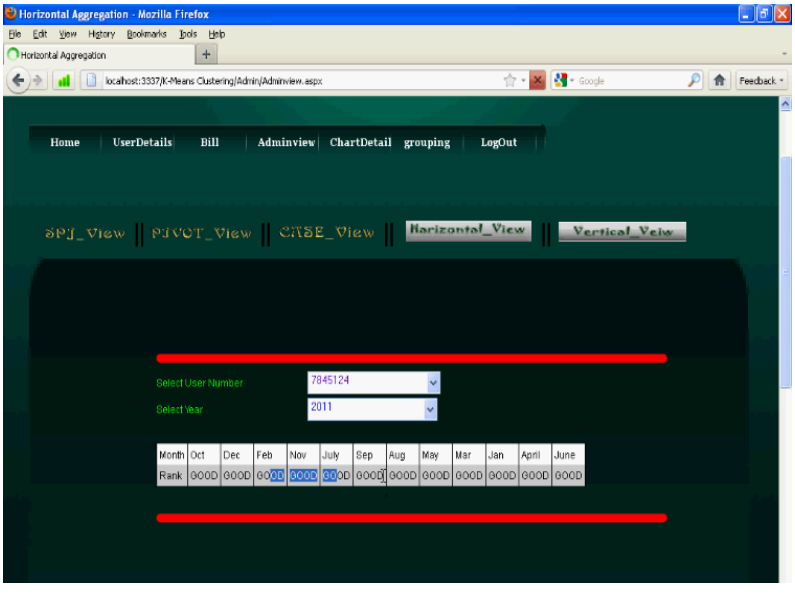
A. ALGORITHM DESIGN

The algorithm is designed as follows:
K-means algorithm based on classification technique uses horizontal aggregation as input. Pivot operator is used to calculate the aggregation of particular data values from distinct fact tables. Optimization provides for PIVOT for large number of fact table. The database connectivity and choosing different tables with .mdb extension is the first step in this system. Horizontal aggregation can be evaluated by choosing transpose column and aggregate operation .Pivot operator automatically transforms table to horizontal layout.

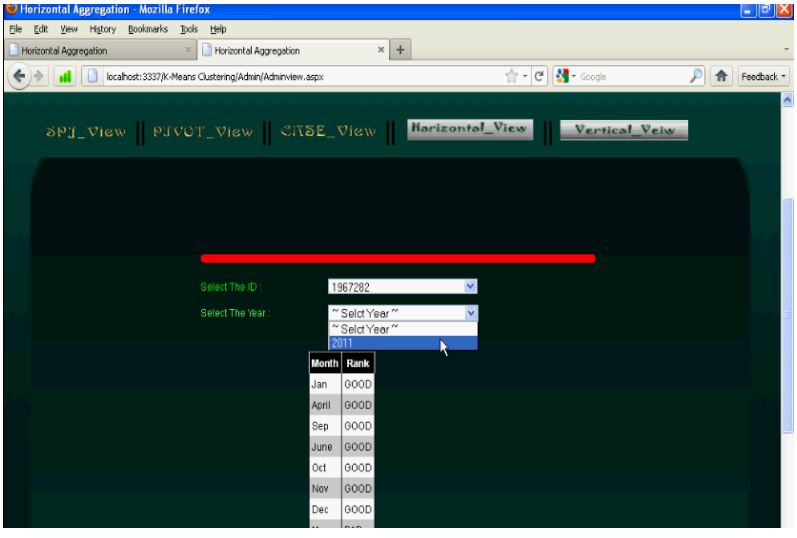
This is the main advantage of this particular algorithm ,the k-means algorithm is the best-known squared error based clustering algorithm with input as horizontal aggregation The algorithm consist of mainly four steps.1) Selection of the initial k means for k clusters from attribute of datasets obtained from horizontal aggregation operation.2) Calculation of the dissimilarity between an object and the Mean of a cluster.3) Allocation of an object to the cluster whose mean is nearest to the object.4) Re-calculation of the mean of a cluster from the objects allocated to it so that the intra cluster dissimilarity

IV.EXPERIMENT AND RESULT

We performed several experiments to test the proposed algorithm and evaluate its performance against different attacks and experienced various results as follows



A . Horizontal view results



B. vertical view results

V. CONCLUSIONS AND FUTURE WORK

This system extended the horizontal aggregations with k-means algorithm to cluster the aggregated column which help preparing datasets for data mining related work. Optimized k-means is significantly faster because of small data set run clustering outside the DBMS. Input to the system is data from multiple tables rather than single table used in traditional horizontal aggregation. Include Euclidean distance computation, pivoting a table to have one dimension value per row. Data manipulating operator Pivot is easy to compute for wide set of values. Pivot is an extension of Group By with unique restrictions and optimization opportunities, and this makes it easy to introduce incrementally on top of existing grouping implementation.

In future, this work can be extended to develop a more formal model of evaluation methods to achieve better results. Also then we can be developing more complete I/O cost models.

References

[1] C. Ordonez, "Data Set Preprocessing and Transformation in a Database System," Intelligent Data Analysis, vol. 15, no. 4, pp. 613-631, 2011.
[2] C. Ordonez, "Statistical Model Computation with UDFs," IEEE Trans. Knowledge and Data Eng., vol. 22, no. 12, pp. 1752 -1765, Dec. 2010.
[3] C. Ordonez and S. Pitchaimalai, "Bayesian Classifiers Programmed in SQL," IEEE Trans. Knowledge and Data Eng., vol. 22, no. 1 ,pp. 139-144, Jan. 2010.
[4] J. Han and M. Kamber, Data Mining: Concepts and Techniques, first ed. Morgan Kaufmann, 2001.
[5] C. Ordonez, "Integrating K-Means Clustering with a Relational DBMS Using SQL," IEEE Trans. Knowledge and Data Eng., vol. 18, no. 2, pp. 188-201, Feb. 2006.
[6] S. Sarawagi, S. Thomas, and R. Agrawal, "Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '98), pp. 343-354, 1998.
[7] H. Wang, C. Zaniolo, and C.R. Luo, "ATLAS: A Small But Complete SQL Extension for Data Mining and Data Streams," Proc. 29th Int'l Conf. Very Large Data Bases (VLDB '03), pp. 1113- 1116, 2003.
[8] A. Witkowski, S. Bellamkonda, T. Bozkaya, G. Dorman, N. Folkert, A. Gupta, L. Sheng, and S. Subramanian, "Spreadsheets in

RDBMS for OLAP," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '03), pp. 52 -63, 2003.
[9] H. Garcia-Molina, J.D. Ullman, and J. Widom, Database Systems: The Complete Book, first ed. Prentice Hall, 2001.
[10] J. Clear, D. Dunn, B. Harvey, M.L. Heytens, and P. Lohman, "Non- Stop SQL/MX Primitives for Knowledge Discovery," Proc. ACM SIGKDD Fifth Int'l Conf. Knowledge Discovery and Data Mining (KDD '99), pp. 425-429, 1999.
[11] C. Cunningham, G. Graefe, and C.A. Galindo-Legaria, "PIVOT and UNPIVOT: Optimization and Execution Strategies in an RDBMS," Proc. 13th Int'l Conf. Very Large Data Bases (VLDB '04), pp. 998-1009, 2004.
[12] C. Ordonez, "Horizontal Aggregations for Building Tabular Data Sets," Proc. Ninth ACM SIGMOD Workshop Data Mining and Knowledge Discovery (DMKD '04), pp. 35-42, 2004.
[13] C. Ordonez, "Vertical and Horizontal Percentage Aggregations," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '04), pp. 866-871, 2004.

