Preparation of Data Set for Data Mining Analysis using Horizontal Aggregation in SQL

Vidya Bodhe
P.G. Student /Department of CE
KKWIEER Nasik, University of Pune, India

Prof. Jyoti Mankar
Assistant Professor/Department of CE
KKWIEER Nasik, University of Pune, Pune, India

Abstract— **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. Most data mining algorithms takes as input data set with a horizontal layout. Significant effort is required to prepare summary data set in a relational database with normalized tables. For preparing data sets suitable for data mining analysis from normalized tables, we have to write complex SQL queries, operation of joining tables and column aggregation. Horizontal aggregation can be performing by using operator, it can easily be implemented inside a query processor, much like a select, project and join. Two main ingredients in SQL code are joins and aggregations Standard aggregation returns one column per aggregated group and produce table with a vertical layout and Standard aggregations are hard to interpret when grouping attributes have high cardinalities. All these are limitations of standard aggregation. Because of these limitations, standard aggregation is not much suitable for preparation of data set for data mining analysis. Horizontal aggregation is a simple method which generates SQL code to return aggregated columns in a horizontal tabular layout and returns set of numbers instead of one number per row. This project is useful for building a suitable dataset for data mining analysis using horizontal aggregations in SQL. Four fundamental methods are used to evaluate horizontal aggregations: CASE, SPJ, and PIVOT and Left Outer Join.**

Index terms - CASE, data set, Horizontal Aggregation, SPJ, Pivot and vertical Aggregation

I. INTRODUCTION

Data mining is the process of analyzing data from different perspectives and summarizing it into useful information that can be used to increase revenue, cuts costs, or both. Data mining software is one of a number of analytical tools for analyzing data. It allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified. Technically, data mining is the process of finding correlations or patterns among dozens of fields in large relational databases. 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.

Preparation of data set for analyzing data in data mining project from relational database using standard aggregation function is time consuming task. Most data mining algorithm takes input a data set which is in horizontal layout i.e. in summarized form. Horizontal aggregations represent an extended form of traditional SQL aggregations, which return a set of values in a horizontal layout, instead of a single value per row. Horizontal aggregations are a new class of aggregations that have similar behavior to SQL standard aggregations, but which produce tables with horizontal layout. Horizontal aggregations have been evaluated using CASE, SPJ, and PIVOT method.

Datasets are prepared for data mining analysis using standard aggregation functions. Data set prepared using standard aggregation produce dataset in vertical tabular layout as shown in table 2. And converting vertical data set into summarized form requires writing long SQL statements or customizing SQL code if it is generated by some tool. Significant effort is required for computing aggregations using available functions and clauses in SQL to convert data set into cross tabular form suitable for data mining analysis.

Let F be a table having a simple primary key K represented by an integer, p discrete attributes and one numeric attribute: F(K,D1, . . . , Dp, A). Using standard aggregation functions datasets are prepared from table F shown in table 1 and result is shown in table 2.

Table 1: Input table, F

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

Table 2: Vertical table, F_V

D1	D2	A

1	X	NULL
1	Y	10
2	X	8
2	Y	6
3	X	17

II. Related Work

Datasets are prepared for data mining analysis using standard aggregation functions. The most widely-known aggregation is the sum of a column over groups of rows. Some other aggregations return the average, maximum, minimum or row count over groups of rows. Using these aggregation functions datasets are prepared from input table F as shown in table 1.

Following query on table 1 gives result as shown in table 3 in vertical tabular form.

```
SELECT D1, D2, sum (A)
FROM F
GROUP BY D1, D2
ORDER BY D1, D2;
```

A standard SQL aggregation (e.g. sum ()) with the GROUP BY clause, which returns results in a vertical layout as shown in table 2.

SQL has been around since its inception and being used widely for interacting with relational databases both for storing and retrieving data. The SQL provides all kinds of constructs such as projections, selections, aggregations, joins and sub queries. Query optimization and using the result of query further is an essential task in database operations. As part of queries, aggregations are used to get summary of data. Aggregate functions such as SUM, MIN, MAX, COUNT, and AVG are used for obtaining summary of data [5]. These aggregations produce a single value output and can't provide data in horizontal layout which can be used for data mining operations. In other words, the vertical aggregations can't produce data sets for data mining. Association rule mining is one of the problems pertaining to OLAP processing [6]. SQL aggregate functions are extended for the purpose of association rule mining in [7]. The aim of this is to support data mining operations efficiently. The drawback of this is that it is not capable of producing results in tabular format with horizontal layout convenient for data mining operations. In [5] a clustering algorithm is explored which makes use of SQL queries internally. It is capable of showing horizontal layout for further mining operations. For performing spreadsheet like operations, alternative SQL extensions are proposed in [8]. They have optimizations too for joins and they do not have optimizations for partial transposition of resultant groups. Joins can be avoided using CASE and PIVOT constructs. Traditional relational algebra [9] has to be adapted to generate new class of aggregations known as horizontal aggregations for generating data sets for data mining operations. This is the focus of our work. The problem of optimizing outer joins is presented in [10]. However, it is not suitable for large queries. Traditional query optimizations [11] use tree-based plans for optimization. This is similar to SPJ method. CASE is also used with SQL optimizations. PIVOT in SQL is used for pivoting results. Lot of research has been around on aggregations and optimizations of SQL operations. They also include cross tabulation and explored much in [12] in case of cube queries. Unpivoting relational tables is also explored in [13] where each input row is used to calculate the decision trees. The result contains multiple rows with attribute – value pairs that behave like an inverse operator for horizontal aggregations. Many SQL operators are available to transform data from one format to another format [14]. The TRANSPOSE operator is similar to unpivot operator which produces many rows for each input row. TRANSPOSE can reduce the number of operations when compared with PIVOT. These two are having inverse relationship as the results are proving this. For data mining operations that produce decision trees, vertical aggregations can be used while the horizontal aggregations produce more convenient horizontal layout that is best suited for data mining operations. In SQL Server [15] both pivot and unpivot operations are made available.

Horizontal aggregations are explored to some extent in [16] and [17] with some limitations. It does mean that the result of these can't be efficiently used for further data mining operations. The proposed horizontal aggregations are different from the built in aggregations that come with SQL. Our operators such as CASE, PIVOT and SPJ are extensions to corresponding SQL operators. For instance CASE is our programming construct that is based on the CASE of SQL; PIVOT is our programming construct that is based on SQL pivoting operation; and the SPJ construct is built using standard SQL queries only.

III. Objectives & Overview Of The Proposed MECHANISM

A. Objectives

The basic objective regarding this paper is to prepare data set for data mining analysis using horizontal aggregation method and evaluate this horizontal aggregation method using CASE, SPJ, PIVOT and left outer join. Mining activities cannot be done directly on the regular databases. In order to perform data mining it is required to prepare datasets that will be useful for mining process. Preparing datasets manually for data mining is a challenging task as its needs aggregation, complex SQL queries. Building a suitable data set for data mining purposes is a time-consuming task. This task generally requires writing long SQL statements or customizing SQL code if it is automatically generated by some tool. There are two main ingredients in such SQL code: joins and aggregations. This paper proposed horizontal aggregation which can be useful for preparation of dataset in less time without any extra efforts.

B. Overview of the proposed Mechanism

The proposed method focuses on minimizing effort and time that is spent in preparing and cleaning a data set for data mining algorithms in data mining project. A big part of this effort involves deriving metrics and coding categorical attributes from the data set in question and storing them in a tabular (observation, record) form for analysis so that they can be used by a data mining algorithm.

IV. EFFICIENT HORIZONTAL AGGREGATION

Horizontal aggregation is a new class of aggregate functions that aggregate numeric expressions and transpose results to produce a data set with a horizontal layout. Functions that belongs to this class is called are called horizontal aggregations. Horizontal aggregations represent an extended form of traditional SQL aggregations, which return a set of values in a horizontal layout, instead of returning a single value per row. Horizontal aggregations are a new class of aggregations that have similar behavior to SQL standard aggregations, but which produce tables with a horizontal layout.

A new class of aggregations that have similar behavior to SQL standard aggregations, but which produce tables with a horizontal layout as shown in table 2. Horizontal aggregations just require a small syntax extension to aggregate functions called in a SELECT statement. Alternatively, horizontal aggregations can be used to generate SQL code from a data mining tool to build data sets for data mining analysis. Proposed syntax is as follows.

```
SELECT (L1… Lj), H (A BY R1 …Rk)
FROM F
GROUP BY (L1… Lj);
```

Table 3: Horizontal table, F_H

D1	D2X	D2Y
1	Null	10
2	8	6
3	17	Null

Architectural Design

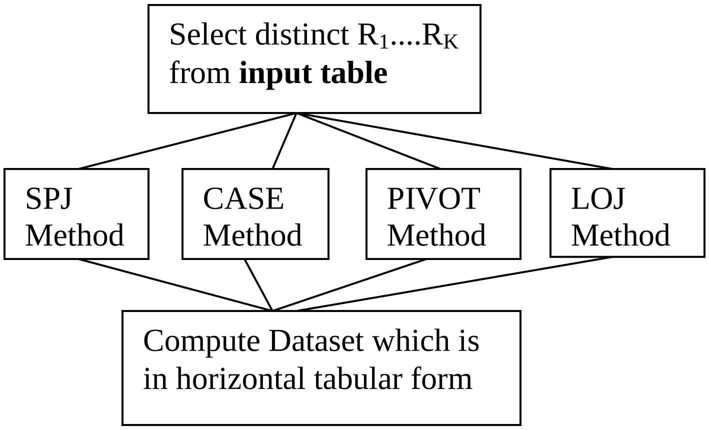
Select distinct R_1....R_K from **input table**
SPJ Method
CASE Method
PIVOT Method
LOJ Method
Compute Dataset which is in horizontal tabular form

Figure 1: System architecture

SELECT DISTINCT R_1...R_k FROM F returns a table with d distinct rows and each row is used to define one column to store an aggregation for one specific combination of dimension values.

In a horizontal aggregation there are four input parameters to generate SQL code:

1) The input table F,
2) The list of GROUP BY columns L_1. Lj
3) The column to aggregate (A),
4) The list of transposing columns R_1. R_k.

SPJ method

In this SPJ method first we create one table with a vertical aggregation for each result column, and then join all those tables to produce F_H. The d projected tables with d Select-Project-Join-Aggregation queries are aggregated from input table F. Each table FI corresponds to one sub grouping combination and has {L1…Lj} as primary key and an aggregation id done on A as the only non-key column. It is necessary to introduce an additional table F0 that will be outer joined with projected tables to get a complete result set.

Case Method

In this method the "case" programming construct which is available in SQL is used. The case statement returns a value selected from a set of values based on Boolean expressions.

Horizontal aggregation queries can be evaluated by directly aggregating from F and transposing rows at the same time to produce FH. First, we get the unique combinations of R1…. Rk, those define the matching Boolean expression for result columns.

Pivot Method

The PIVOT method uses the built-in PIVOT operator, which transforms rows to columns (e.g. transposing). PIVOT operator is a built-in operator in a commercial DBMS. The PIVOT method internally needs to determine how many columns are needed to store the transposed table and it can be combined with the GROUP BY clause.

Left Outer Join Method

In this SPJ method first we create one intermediate table F_V from input table F. Then we create one table with a vertical aggregation for each result column, and then join all those tables to produce F_H using Left Outer Join.

Output Table

It gives Dataset which is in horizontal tabular form suitable for data mining analysis FH with a horizontal layout having n rows and j+d columns, where each of the d columns represents a unique combination of the k grouping columns.

V. PERFORMANCE EVALUATION

In order to compare the performance of the proposed system, the system is checked with dataset generated by TPC-H generator having input table lineitem with 700 records, |F|=700 and following parameters as shown in table 4.

Table 4: Summary of Grouping Columns from TPC-H Table Lineitem (N=700).

L1(grouping column)	**R1(transposing column)**	**n (answerset size)**	**d (no.of dimensions)**
suppkey Suppkey Suppkey Suppkey	linestatus Weekday Month Brand	50 50 50 50	2 7 12 24
partkey Partkey Partkey partkey	linestatus Weekday Month Brand	100 100 100 100	2 7 12 24
orderkey orderkey orderkey orderkey	linestatus Weekday Month Brand	200 200 200 200	2 7 12 24

Table 5: Query Optimization (N =700) for different methods Times In milliseconds

n	**d**	**SPJ**	**CASE**	**PIVOT**	**LOJ**
50	2	1653	94	47	63
	7	3884	94	78	78
	12	4914	62	62	93
	24	10497	78	47	94
100	2	1747	62	47	78
	7	3915	78	94	124
	12	5446	78	63	93
	24	1073	47	62	78
200	2	2106	94	94	125
	7	4025	78	78	94
	12	5848	93	78	94
	24	10856	78	171	109

Table 5 shows time required for SPJ, CASE, PIVOT and LOJ(Left Outer Join) method in milliseconds for parameters shown in table 4.

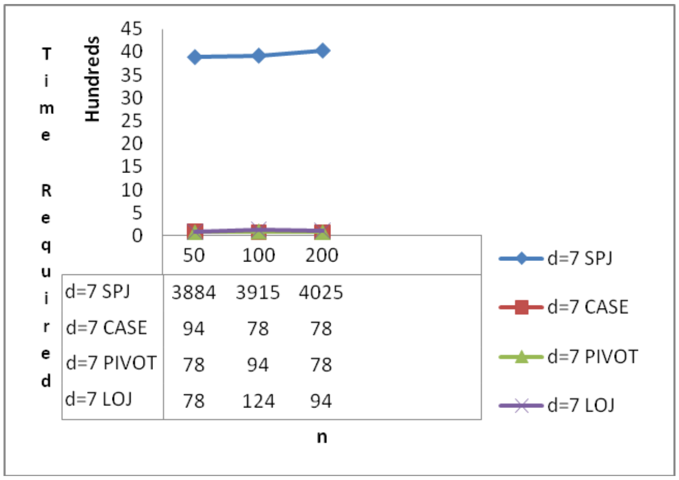

Figure 2: Graph of result for d=7

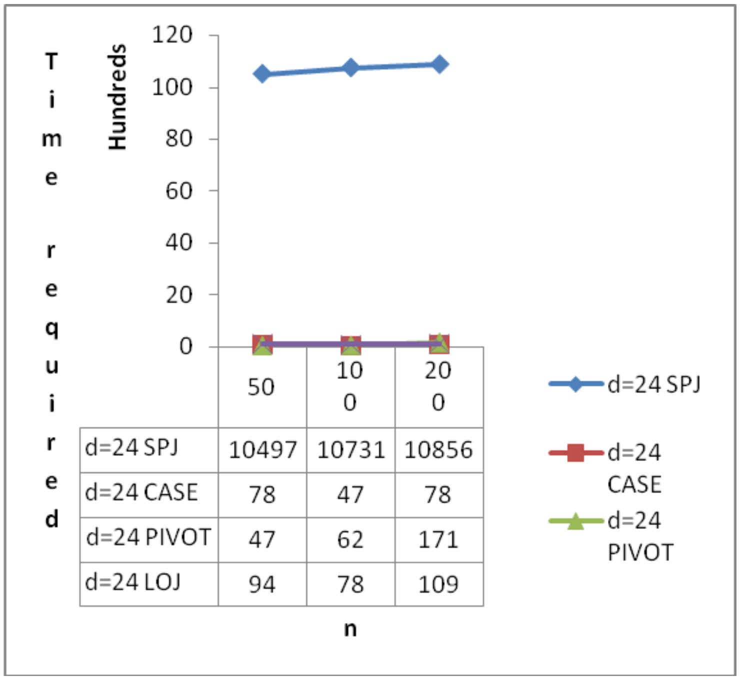

Figure 3: Graph of result for d=24

As shown in figure 2 and figure 3, the graphs shows result of Horizontal aggregation for four different methods. The result is obtained from input table Lineitem which is generated from TPC-H generated dataset of size 700 and using grouping columns as shown in table 4. As shown in the above graphs, time required for generating data set using PIVOT, CASE and Left Outer Join method is near about same and time required for generating data set using SPJ is very large. Therefore our method Left Outer Join is efficient for producing dataset for data mining analysis.

VI. Conclusion

We are introduced a new class of extended aggregate functions, called a horizontal aggregations which are help to preparing datasets for OLAP cube exploration and data mining. In particularly, horizontal aggregations are useful to create data sets with a horizontal layout. Mainly a horizontal aggregation returns a set of numbers instead of one number per each group. For a query optimization perspective, we are proposed the four fundamental query evaluation methods. The first method is SPJ. It relies on standard relational operators. The second is CASE. It relies on the case construct. PIVOT is the third method. The pivot is a built in operator. Generally PIVOT operator is shows the table in two ways (Narrow, wide tables). It is a built-in operator in a commercial database. Fourth method is Left Outer Join in which we joins table using left outer join for producing output.SPJ methods is important from a theoretical point of view because it is based on select, project and join queries. CASE, PIVOT and Left Outer Join evaluation methods are significantly faster than the SPJ. All methods produces same result.SPJ method takes more time than CASE, PIVOT and Left Outer join for producing result.

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] C. Galindo-Legaria and A. Rosenthal, “Outer Join Simplification and Reordering for Query Optimization,” ACM Trans. Database Systems, vol. 22, no. 1, pp. 43-73, 1997.

[11] G. Bhargava, P. Goel, and B.R. Iyer, “Hypergraph Based Reordering of Outer Join Queries with Complex Predicates,” Proc. ACM SIGMOD Int’l Conf. Management of Data (SIGMOD ’95), pp. 304-315, 1995.

[12] J. Gray, A. Bosworth, A. Layman, and H. Pirahesh, “Data Cube: A Relational Aggregation Operator Generalizing Group -by, Cross- Tab and Sub-Total,” Proc. Int’l Conf. Data Eng., pp. 152-159, 1996.

[13] G. Graefe, U. Fayyad, and S. Chaudhuri, “On the Efficient Gathering of Sufficient Statistics for Classification from Large SQL Databases,” Proc. ACM Conf. Knowledge Discovery and Data Mining (KDD ’98), pp. 204-208, 1998.

[14] 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.

[15] 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.

[16] 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.

[17] C. Ordonez, “Vertical and Horizontal Percentage Aggregations,” Proc. ACM SIGMOD Int’l Conf. Management of Data (SIGMOD ’04), pp. 866-871, 2004.

[18] Carlos Ordonez and Zhibo Chen, “Horizontal Aggregations in SQL to Prepare Data Sets for Data Mining Analysis”, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 24, NO. 4, APRIL 2012.

Authors Profile

Vidya Bodhe received the **B.E.** degree in Computer Technology from the Chandrapur engineering college, chandrapur, from Nagpur University, India. Currently doing **M.E.** in computer engineering from K. K. Wagh Institute of Engineering education and research, Nasik in Pune University, India. Her research interest includes Data mining, relational database, computer security.

Prof. Jyoti Mankar Assistant professor Department of CE, KKWIEER Nasik, Pune University received the bachelor’s degree in Computer Technology from Nagpur University, Nagpur, India and Master degree in computer science and engineering, from G H R C E Nagpur, India, Her research interests include in, image processing, pattern recognitions, computer security and networking.