THINK INDIA JOURNAL

ISSN: 0971-1260

Vol-22-Issue-17-September-2019

Horizontal Aggregation in SQL to Prepare Knowledge Sets

Mrinalini Rana[1]
Assistant Professor, School of Computer Science and Engineering
Lovely Professional University
Kapurthala, India.
mrinalini.22138@lpu.co.in

Ujjwal Makkar[2]
Assistant Professor, School of Mechanical Engineering.
Lovely Professional University
Kapurthala, India.
ujjwal.14832@lpu.co.in

Abstract **— The most time consuming and tidies job is preparing a data set for analysis using table joins, blending and various queries. The traditional aggregations (SQL) have various limitations because the outcome is one column per group (aggregated group). Due to this, data sets are prepared manually, where ever horizontal aggregation is required. Horizontal aggregation is known as new class of functions, which is required in various data mining algorithms. This paper discussed various fundamental methods to evaluate horizontal layout (aggregation) such as SPJ with relational algebra operators, CASE using some programming CASE structures and PIVOT using some DBMS operators**

Keywords**—** RDBMS(Relational DBMS),SQL, Aggregation and Pivoting.

I. INTRODUCTION

Data mining procedures comprises of four stages. Extraction, cleaning and transforming is the first stage. This is called as "DATA PREPARATION". Second stage is analyzing the records set. Mainly investigation effort in data mining is done on the proposing resourceful algorithms, lacking concentrating the structure the data set. The next step verifies outcomes, generates reports and mapping of parameters. In the fourth step numerical results are deployed on primary data. The paper concentrates on data preparation stage.

Creating the right data mining knowledge collection is a time-consuming job. SQL code's two main options: join and aggregate [9]. Programmer can focus on join option. Traditional aggregation method is sum of the column upon group of lines. Some other aggregations outcomes are the avg(), max(), min() or count() over groups of rows.. Some valid methods are required to find aggregation in horizontal form (cross tabulation), suitable for data mining algorithms, based on some traditional clauses and functions in SQL. Such efforts are to be written, optimized and tested due to the amount and complexity of SQL. There are few other practical reasons for the performance of the horizontal (cross-tabular) format. Benchmark (vertical) aggregations are difficult to understand when the results contain more lines, particularly when there is high attribute cardinality.

In order to transpose tests, OLAP tools produce SQL code and are also known as PIVOT [16]. Transposition is more effective if a consumer combines aggregation and transposition.

With all of these limitations,[5] proposes a new aggregating function class that aggregates numeric terms and transposes marks to generate a straight records format set. Under this class, function is called Horizontal aggregations.

Consider the following query for a cell phone company database:

F

K	D_1	D_2	D_3	A_1
1	2	a	x	2
2	2	b	y	2
3	4	b	z	4
4	2	a	x	0
5	4	c	y	0
6	2	a	z	1
7	1	c	x	**null**
8	2	b	y	1
9	2	a	x	2
10	3	a	y	8

F_V

D_1	D_2	count(*)	A
1	c	1	**null**
2	a	4	5
2	b	2	3
3	a	1	8
4	b	1	4
4	c	1	0

ISSN: 0971-1260
Vol-22-Issue-17-September-2019

F_H

D_1	sum(A BY D_2=a)	sum(A BY D_2=b)	sum(A BY D_2=c)
1	null	null	**null**
2	5	3	null
3	8	null	null
4	null	4	0

```
/* F_V */
SELECT D1,D2,count(*),sum(A)
FROM F
GROUP BY D1,D2
ORDER BY D1,D2;
/* F_H */
SELECT D1,sum(A BY D2)
FROM F
GROUP BY D1
ORDER BY D1;
```

FIG. 1: EXAMPLE OF INPUT TABLE F, VERTICAL AGGREGATION FV AND HORIZONTAL AGGREGATION FH [3].

Horizontal Aggregation Pros:

- SQL programming eliminates guidebook work during the figures mining campaign training period.
- When generated automatically, the SQL code is more efficient. It will take less time to create the data sets.
- User can directly create the data sets in the DBMS.

II. RELATED WORK

Computing aggregation research is highly scalable. Aggregations or layouts are pivotal in data mining(KDD) [17] and Online Analytical Processing [18] applications.

The main issue of incorporating data mining algorithms into RDBMS is the research proposed. Comparing horizontal aggregation with alternative methods to work with transposition or pivoting.

There are plenty of ideas that have expanded the syntax of SQL. The closest problem associated with the storage of OLAP data mining is association rule mining[19]. In [20], aggregate functions for association rule mining are implemented with some SQL extensions.

Unfortunately, as transactions are mentioned in vertical layout, there is no notation of transposing results. Some other SQL extensions have been added in [21] to perform spreadsheet-like (excel) operations.

There is no membership in the procedure for CASE and PIVOT.

Optimizing joints have been implemented in [23] by reordering operations and by applying transformation rules. In practice, a commonly used SQL feature is the CASE construct to optimize queries that have not been studied in depth before a list of similar CASE statements is available.

The TRANSPOSE operator produces multiple columns for a single input line compared to the un-pivot operator.

In terms of horizontal aggregations, both UNPIVOT and TRANSPOSE is reverse operators.

A generic format may provide aggregations and group-by queries with more robust data mining techniques such as FP tree, decision tree, but is usually less effective than a cross-tabulation model. Parallel aggregations are equal to horizontal aggregations percentage [6]. Instead of one and DBMS storage problems such as generating and locking SQL code, three calculation methods are now considered.

III. BACKGROUND

A data source is a collected data or information, usually in tabular form. Each column is a common factor. Every row corresponds to the data set of a given member. It lists values such as the altitude and mass of an object for each of the variables. Each value is referred to as a datum.

ISSN: 0971-1260
Vol-22-Issue-17-September-2019

The facts set may contain records corresponding to the number of rows for one or more members.

Data mining systems are widely used in horizontal (cross-tabular) layout for the efficient analysis of data. Vertical (standard) tables for RDBMS are generally administered. Aggregated columns return numbers in a horizontal (cross-tabular) format.

The system uses one super table and various sub tables, and then procedures are performed on multiple tables loaded raw facts. PIVOT operator is used for the measurement of RDBMS cumulative operations.
PIVOT operator is used to measure cumulative operations provided by RDBMS. System PIVOT is much simpler and offers a lot of scalability.

IV. HORIZONTAL AGGREGATION

It is an introduction of a new class of aggregation that has same functionality to SQL traditional code aggregation and results in tables with horizontal display. On the other side, using traditional SQL aggregations represent the vertical representation. In the SELECT command, level aggregation requires a syntax extension to aggregate functions (sum, max, min, etc.). Otherwise, SQL code can be created from a data mining device to create data source for data withdrawal examination. Explain how to automatically create SQL code.

A. SQL Code Generation:

First goal is to define a standard template for aggregating and transposing / pivoting code generation.
The second objective is to enhance the SELECT command by introducing and aggregating it.
For Illustration:
"Select L1 to Lm sum(B)
From F
Group By L1 to Lm;"

The aggregation order would arise from a table of "m+1" attributes, with single class for apiece inimitable grouping of L1 ... Lm values and an aggregated assessment per category (sum (B) in the example in question). To test this query, the question optimizer enters three constraint:

1. Tabular representation
2. Columns ($L_1 \ldots L_{m)}$
3. Layout or aggregation

Thus, in horizontal design there are four input constraint to generate code:

1. F is participation tabular representation,
2. Column L1 to Lj is list of GROUP BY columns
3. Aggregate (A) column
4. Transposing Column R1 to Rk.

Horizontal aggregations maintain semantic assessment of standard aggregations. The foremost variation is to return a table, possibly with additional nulls, with a horizontal layout. The definition makes it possible to transpose a simple generalization of several comprehensive columns, each among a diverse column list being transposed.

V. Collection of SQL Code: table Locking and Effect Definition

Discuss now how to create active SQL code automatically to test horizontal aggregations. Changing the query optimizer's internal data structures has more scalability, but refer to some pointers. Begin by discussing the result table structure and then look for methods of optimization to fill it.

A. Locking

To get a reliable request evaluation, locking [22] must be used.
The actual values of aggregation in FH will change. Using tabular locks acquired on F, FV, and FH before beginning and releasing the first claim after populating FH to return clear responses. To recapitulate, the intact set of declarations becomes a larger transaction.

.

B. Result Table Definition

FH has columns for aggregation, plus its primary key. FH (the result table) must therefore have the grouping column set (L1 to Lj) as the primary key and all existing combinations of values R1 to Rk as non-key columns. The horizontal aggregation function H) (returns a set of values for each class L1 to Lj, rather than a single value. Assume that the table of results is FH. Retrieved the separate combination meaning of 'R_1 to $R_{k'}$, using the following statement.

"Select distinct R1 to Rk from F;"

Let the table be FH and primary key syntax is as follows:
.
"Create table F_H(a_1 int, a_2 int,…., a_n real) primary key($a_1 \ldots a_n$);"

VI. Generation of SQL Code: Query Evaluation Methods

ISSN: 0971-1260
Vol-22-Issue-17-September-2019

For test horizontal aggregations, the paper suggests three methods. The first method is based solely on related operations, i.e. only selecting, projecting, joining, and aggregating queries; it is called the SPJ method. The instant form is based on the "case" structure; it is called the CASE method. Each table has some serial numbers on its primary key to decide to blend effectively. There are also no external indexing mechanisms to step up query analysis. The further move towards uses the built-in SQL revolve operator, converting rank into line (e.g., transposing).

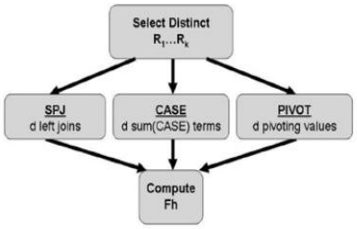

Fig 2. Unoptimized 5].

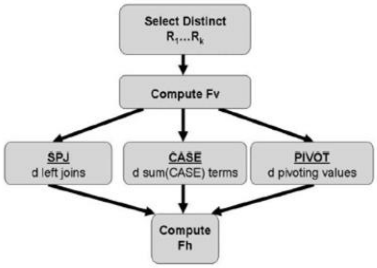

Fig. 3. Optimized[5].

A. SPJ Method:

Projected tables of d queries from F (input table) to d Select-Project-Join-Aggregation, column number for analysis, array, forecast. Each FI table corresponds to one subgroup variation and is the only non-key column (L1; ..; Lj) as the primary key and an A aggregation. In order to obtain a complete collection of results and suggest two simple FH measurement sub-strategies, it is compulsory to implement an supplementary Table F0 that will be joined externally with expected tables. The first of these aggregates directly from F. In a temporary table FV (vertical aggregation table) grouping by L1 to Lj and R1 to Rk, the second measure the corresponding vertical aggregation.

Then parallel design can be instead executing from F_V, which is a squashed version of F, since standard aggregations are allocated [14].

```
INSERT INTO FH
SELECT
F0.L1; F0.L2; . . . ; F0.Lj,
F1.A; F2.A; . . . ; Fd.A
FROM F0
LEFT OUTER JOIN F1
ON F0.L1 = F1.L1 and . . . and F0.Lj = F1.Lj
LEFT OUTER JOIN F2
ON F0.L1 = F2.L1 and . . . and F0.Lj = F2.Lj
. . .
LEFT OUTER JOIN Fd
ON F0.L1 = Fd.L1 and . . . and F0.Lj = Fd.Lj;
```

In the above formula left join is usd.

B. CASE Method

In this "case" approach, the form of programming available in SQL is used. The case statement precedes a significance selected from a set of Boolean expressions based values. This is similar to a fundamental question of projection / aggregation from the theoretical point of view of a relational database. By aggregating unswervingly from F and rearranging tables, horizontal accumulation could be decided simultaneously to construct FH. User may mainly have a blend of "R1 to Rk" that defines the corresponding expression of the Boolean line.
Select Distinct R1 to Rk from F;
Insert into F_H

C. PIVOT Method

Remember the PIVOT operator who is an advanced DBMS operator. Because this operator can carry out transposition and can assist in estimating straight aggregations. The PIVOT process internally requires determining how many columns the transposed table must be stored and can be combined with the GROUP BY clause.
The vital syntax to utilize the PIVOT operator is as follows:

"Select Distinct R1 from F;
Select L1 to Lj into Ft
From F PIVOT(V(A) from R1 in(v1 to vd) as P;"

Additional optimized query for large table, because the query inside the query orderly F from line that are not later on desirable.

"Select L1 to Lj , v1 to vd into FH
From (select L1 to Lj, Rk , A from F) Ft PIVOT(V (A) For R1 in(v1 to vd) as P;"

ISSN: 0971-1260

Vol-22-Issue-17-September-2019

VII. CONCLUSION

Data mining tools are commonly used in horizontal tabular format to analyze data effectively. Typically, RDBMS manages vertical tables. Aggregated columns return numbers, rather than one number per row, in a horizontal tabular format. Device PIVOT is a lot simpler and offers a lot of scalability. So, the concluding points are as follows:

Generally horizontal aggregations are useful in generating horizontal format data sets, as the data mining algorithms usually require. But a horizontal multiplication returns for each group, analogous to a multidimensional matrix, a collection of numbers as an alternative of a single number. From a hypothetical point of view, the SPJ method be useful because it focuses on selecting, projecting, and joining i.e. The SPJ's comments.

The large-scale experiments demonstrate that the efficiency of the proposed parallel aggregations evaluated using the CASE approach is the same as the integrated PIVOT operator[5].

REFERENCES

[1] Carlos Ordonez, San Diego, *"Horizontal Aggregation for Building Tabular Data Sets",* Proc.ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD),pp.35-42,2004.

[2] M.Stella Inba Mary,V.Kalaivani, *"QueryOptimization using SQL Approach for Data Mining Analysis",* IJCA Proceedings on International Conference in Recent trends in Computational Methods, Communication and Controls (ICON3C 2012) ICON3C(3) ,pp. 17-21,2012.

[3] Carlos Ordonez, Javier García-García, Zhibo Chen, San Diego, *" Dynamic Optimization of Generalized SQL Queries with Horizontal Aggregations"*, Proc. ACM SIGMOD Conference, pp.35-42 ,2012.

[4] Carlos Ordonez, "*Optimization of Linear Recursive Queries in SQL"*, IEEE Transactions on Knowledge and Data Engineering (TKDE Journal). 22(2):264-277, 2010.

[5] Carlos Ordonez, Zhibo Chen, *"Horizontal Aggregation in SQL to prepare Data Sets for Data Mining Analysis"* IEEE ,VOL.24 , Issue 4 , pp. 678-691,2012.

[6] C. Ordonez, *"Vertical and Horizontal Percentage Aggregations," Proc*. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '04), pp. 866-871, 2004.

[7] 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.

[8] C. Ordonez, *"Statistical Model Computation with UDFs,"* IEEE Trans. Knowledge and Data Eng., vol. 22, no. 12, pp. 1752-1765, Dec. 2010.

[9] C. Ordonez, *"Data Set Preprocessing and Transformation in a Database System"*, Intelligent Data Analysis, vol. 15, no. 4, pp. 613-631, 2011.

[10] 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.

[11] S. Chaudhuri and U. Dayal, *"An overview of data warehousing and OLAP technology"*, ACM SIGMOD Record, 26(1):65-74, 1997.

[12] J. Clear, D. Dunn, B. Harvey, M.L. Heytens, and P. Lohman. Non-stop SQL/MX primitives for knowledge discovery. In ACM KDD Conference, pages 425-429, 1999.

[13] G. Graefe, U. Fayyad, and S. Chaudhuri, *"On the efficient gathering of sufficient statistics for classification from large SQL databases*" ,In Proc. ACM KDD Conference, pages 204-208, 1998.

[14] J. Gray, A. Bosworth, A. Layman, and H. Pirahesh, *"Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-total"*, In ICDE Conference, pages 152-159, 1996.

[15] J. Han, J. Pei, G. Dong, and K. Wang, *"e-Client computation of iceberg cubes with complex measures"*, In ACM SIGMOD Conference, pages 1-12, 2001.

[16] 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.

[17] U. Fayyad and G. Piateski-Shapiro, *"From Data Mining to Knowledge Discovery"*, MIT Press, 1995.

[18] J. Widom, *"Research poblems in data warehousing"*, In ACM CIKM Conference, pages 25{30, 1995.

[19] 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.

[20] 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.

[21] 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.

[22] H. Garcia-Molina, J.D. Ullman, and J. Widom, *"Database Systems: The Complete Book"*, first ed. Prentice Hall, 2001.

[23] 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.