Preparation of Data Sets for Data Mining Analysis using Horizontal Aggregation in SQL and Optimization of Horizontal Aggregation

Ms Vidya Bodhe, Prof Jyoti Mankar

P.G.Scholar, *Department of Computer Engineering KKWIEER,Nasik*
Asst. Professor, Department *of Computer Engineering KKWIEER,Nasik*

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 such as age, name, phone number, address, profession, or income. 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, 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. PIVOT operator on tabular data that exchange rows, enable data transformations useful in data modeling, data analysis, and data presentation

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. Three fundamental methods are used to evaluate horizontal aggregations: CASE, SPJ, and PIVOT. This project will evaluate horizontal aggregation using K-means algorithm for query optimization.

Keywords: CASE, data set, Horizontal Aggregation,, Optimization, SPJ, Pivot

I. INTRODUCTION

Preparation of data set for analyzing data in data mining project from relational database using standard aggregation function is time consuming task. Most data miming 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.

Motivation:-

The basic objective regarding this project is to prepare data set for data mining analysis using horizontal aggregation method and evaluate this horizontal aggregation method using CASE, SPJ, PIVOT and K-means algorithm for query optimization.

Existing systems:-

Datasets are prepared for data mining analysis using standard aggregation functions. Most data mining algorithm requires input a datasets with a horizontal layout with several records and one variable or dimensions per column. But data set prepared using standard aggregation produce dataset in vertical tabular layout. 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.

Concept or seed idea:-

In this section, we suggest some alternatives which will make it possible to convert data sets into horizontal layout easily using horizontal aggregation function. The 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.

DEFINITIONS

Tables that will be used to explain SQL queries are defined here.

F(K,D_1,……..,D_p,A) F is a fact table with primary key K, D1,……..,Dp are dimensions ,A is measure column that is passed to standard SQL aggregations.

F is a temporary table as shown in table 1

Table 1: Input Table, F

K	D_1	D_2	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

F is used as input table. Here for explaining some examples of analyzing store database transationLine table is used. transactionLine table is a transaction table of store database. Table transactionLine has three dimensions namely product hierarchy, location and time .these dimensions are used for grouping rows. Table transactionLine also contains three measures- itemQty, salesAmt and costAmt are used to pass as arguments in aggregation function.

Table transaction Line is assumed to have star schema as shown in figure 2

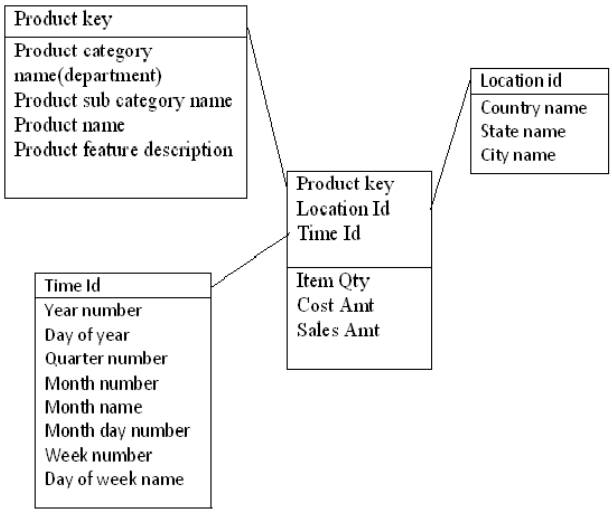


Figure 1: transaction Line table

II. LITERATURE REVIEW

This chapter will introduce us with the previous system and its analysis. It also includes the comparison of existing system with proposed system. This will give us the detail idea about the need of proposed system.

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.

Table 2: Vertical Table, F_V

D_1	D_2	A
1	X	NULL
1	Y	10
2	x	8
2	y	6
3	x	17

Analysis of Exiting System:

Analysis of system is given using examples with a store (retail) database that requires data mining analysis.

Store database contains following tables,

Product hierarchy (product ID, name, category)

Location (location id, city, state, country)

Time (date, month, quarter, year)

transactionline (product hierarchy, location, time) and three measures itemQty, costAmt and salesAmt as shown in figure 1.

If we compute query "summarizes sales for each store by each day of the week" gives results as shown in table 4.

```
SELECT storeId, dayofweekNo, sum (salesAmt)
FROM TransactionLine
GROUP BY storeId, dayweekNo
ORDER BY storeId, dayweekNo;
```

Table 3: Vertical Table

storeId	Dayofweekno	salesAmt
10	Mon	2345
10	Thu	5777
10	Wed	3445
10	Thu	6868
10	Fri	90
10	Sat	8089
10	Sun	4543
32	Mon	3343
.	.	.
.	.	.
.	.	.
32	Sun	6767
56	mon	32314

If there are 200 stores, 30 store departments and stores are open 7 days a week, the first query returns 1400 rows which may be time-consuming to compare with each other each day of the week to get trends. This query can be answered with standard SQL, but additional code needs to be written or generated to return results in horizontal tabular form as shown in table 3.

Horizontal aggregation just requires a small syntax expression to aggregate functions called in a SELECT statement and produce datasets in horizontal layout which is the desired layout for most data mining algorithms. They represent a template to generate SQL code from a data mining tool automates writing SQL queries, optimizing them and testing them for correctness. This method reduces manual work in the data preparation phase in a data mining project. The data set can be created entirely inside the DBMS. As horizontal aggregations produce tables with fewer rows and more columns, query optimization techniques used for standard aggregations are inappropriate for horizontal aggregations.

What is 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.

Table 4: Horizontal Table, F_H

D_1	D_2X	D_2Y
1	NULL	10
2	8	6
3	17	NULL

If we want to summarize sales information with one store per row for one year of sales means we need the

sales amount broken down by day of the week, the number of transactions by store per month, the number of items sold by department and total sales. The query is

```
SELECT
storeId,
sum (salesAmt BY dayofweekName),
count (distinct transactionid BY salesMonth),
sum (1 BY deptName),
sum (salesAmt)
FROM transactionLine
, DimDayOfWeek, DimDepartment, DimMonth
WHERE salesYear=2009
AND transactionLine.dayOfWeekNo
=DimDayOfWeek.dayOfWeekNo
AND transactionLine.deptId
=DimDepartment.deptId
AND transactionLine.MonthId
=DimTime.MonthId
GROUP BY storeId;
```

This query gives output as shown in table below.

Table 5: Horizontal table for data mining algorithms

storeId	SalesAmt Mon Tue…Sun	countTransactios Jan Feb ……Dec	countItems dairy Meat.	Total salesAmt
10	120 111 200	2011 1807 4200	34 57	25025
32	7 65 98	802 912 1632	32 65	14022
56 . .	77 92 99	555 1000 950	55 26	20500

III. SYSTEM ARCHITECTURE

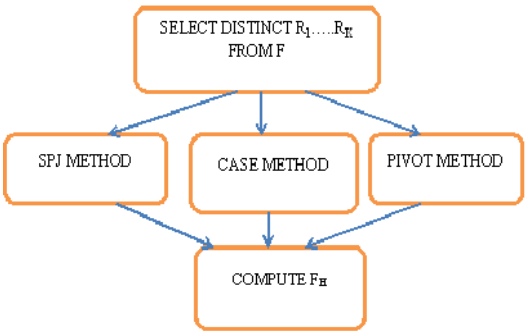


Figure 2 System Architecture diagram

SELECT DISTINCT R_1... Rk 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 L1. Lj,
3) The column to aggregate (A),
4) The list of transposing columns R1. . . Rk.

Syntax for horizontal aggregation is as follows.

```
SELECT L1, L2…Lj, H (A BY R1…. Rk)
FROM F
GROUP BY L1, L2… Lj;
```

1 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.

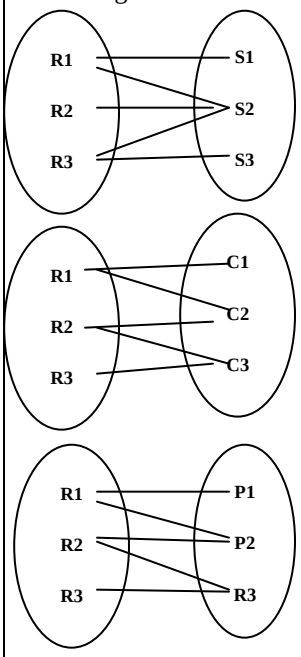
2 Case Method
In this method the"case" programming construct which is available in SQL is used. The case statement returns a value selected from a set of values based on Boolean expressions.

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.

3 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.

IV. MATHEMATICAL MODEL

Table 6: Table showing mathematical model

Sr.No.	Description	UML Design Observations
1.	Problem Description	
	s ={R,T,C,P,S|$Ø_s$} Where R=sets of records {r1,r2,r3|$Ø_r$} T=sets of tables {t1,t2,t3|$Ø_t$} C=sets of tables in horizontal tabular form using CASE method {c1,c2,c3|$Ø_c$} P=sets of tables in horizontal tabular form using PIVOT method {p1,p2,p3|$Ø_p$} S=sets of tables in horizontal tabular form using SPJ method {s1,s2,s3|Øs}	S holds list of actors
2	Activity1	
	$f_{RC}(R) \mid \rightarrow C$ $f_{RP}(R) \mid \rightarrow P$ $f_{RS}(R) \mid \rightarrow S$	f_R is the rules to generate H.A. from the input record set.
	$f_{TC}(R) \mid \rightarrow C$ $f_{TP}(R) \mid \rightarrow P$ $f_{TS}(R) \mid \rightarrow S$	F_T is the rules to generate H.A. from the input tables
3.	Venn diagram 	R is set of distinct rows C is set of tables in H.A. using CASE P is set of tables in H.A. using PIVOT S is set of tables in H.A. using SPJ

V. EXPERIMENTAL SETUP

- For designing and implementation purpose C# language is require.
- Also, require SQL Server V9 running on DBMS server running at 3.2GHz
- Large synthetic datasets generated by the TPC-H generator is required.
- Operating environment requires for implementation is as shown in table below

Table 7: Table showing experimental setup

Sr. No.	Name of content	Capacity
1	Operating System	Window XP
2	Processor	Dual core
3	Hard Disk	1TB
4	RAM	4 GB
5	Front End	C#
6	Back End	SQL Server V9 running on DBMS server

Test id	Test description	Expected input	Actual output
1.	Distinct Rows	Specific row from input table from which we want to summarize data for data mining analysis	Data set for data mining analysis
2.	SPJ method	Specific row from input table from which we want to summarize data for data mining analysis using SPJ method	Table in horizontal tabular form for data mining analysis
3.	CASE method	Specific row from input table from which we want to summarize data for data mining analysis using CASE method	Table in horizontal tabular form for data mining analysis
4	PIVOT method	Specific row from input table from which we want to summarize data for data mining analysis using PIVOT method	Table in horizontal tabular form for data mining analysis

VI. TEST CASES

Table 8: table showing test cases

VII. EXPECTED RESULT

Since TPC-H only creates uniformly distributed values, we will create a similar data generator for controlling probabilistic distribution of column values. SPJ will be slowest method as compared to PIVOT and CASE.PIVOT and CASE will have similar time performance. Time will grow as n(size of data set) grows for all methods but much more for SPJ. When d (dimensionality) increases for SPJ, small time growth will occur. Main performance factor for PIVOT and CASE method is n (data set size).But both n and d will impact the SPJ method. Use of two probabilistic distributions, uniform and skewed will be used for query optimization. Using skewed distribution at high d, PIVOT shows a slightly higher impact than the CASE method.SPJ will be slowest and will show bigger impact at high d.

VIII. CONCLUSION

Horizontal aggregations help in preparing data sets for data mining and OLAP cube exploration. Horizontal aggregations are useful to create data sets with a horizontal layout, as commonly required by data mining algorithms A horizontal aggregation returns a set of numbers instead of a single number for each group, resembling a multi-dimensional vector. Horizontal aggregations can be used as a database method to automatically generate efficient SQL queries with three sets of parameters: grouping columns, sub grouping columns and aggregated column. 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.

IX. REFERENCES

[1] Carlos Ordonez, Zhibo Chen Horizontal aggregations in SQL to prepare data sets for data mining analysis, *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 24(4):678-691 , 2011, IEEE Computer Society.

[2] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Francisco, 1st edition, 2001

[3]C. Galindo-Legaria and A. Rosenthal. Outer join simplification and reordering for query optimization. *ACM TODS*, 22(1):43–73, 1997.

[4]C.Cunningham, G.Graefe, and C.A. Galindo-Legaria. PIVOT and UNPIVOT: Optimization and execution strategies in an RDBMS. In *Proc. VLDB Conference*, pages 998–1009, 2004

[5] C.Galindo-Legaria and A. Rosenthal. Outer join simplification and reordering for query optimization. *ACM TODS*, 22(1):43–73, 1997.

[6] Gray, A. Bosworth, A. Layman, and H. Pirahesh. Data cube: A relational aggregation Operator generalizing group-by, cross-tab and subtotal. In *ICDE Conference*, pages 152–159, 1996.

[7] C. Ordonez. Data set preprocessing and transformation in a database system. *Intelligent Data Analysis (IDA)*, 15(4), 2011

[8] 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.

[9] C. Ordonez. Statistical model computation with UDFs. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 22, 2010.