Database Transformation to Build Dataset for Generation of Decision Tree and Extended ER Model

Archana A. Chaudhari
Dept. of Comp Engg
MMCOE, Pune
Maharashtra, India

Harmeet Kaur Khanuja
Dept. of Comp Engg
MMCOE, Pune
Maharashtra, India

ABSTRACT
In Data mining project most of the time consuming task is to prepare a required data set for data mining analysis because in general the relational database has collection of tables and views that must be joined, aggregated and transformed in order to build the required data set. As result, most of the complex SQL queries are written multiple times independently from each other and in a disorganized manner. Therefore, the database grows with many tables and views that are not present as entities in the ER model. Similarly existing SQL aggregations having some limitations to prepare normalized data sets because they return only one column per aggregated group. To address this issue, we propose simple methods to generate SQL code to return aggregated columns in a horizontal tabular layout, where every row corresponds to an observation and every column is associated to a one variable. This new class of functions is called horizontal aggregations. Horizontal aggregations is extension of standard SQL aggregation for building data sets with a horizontal denormalized layout, which is input for most of the data mining algorithms. By providing these standard normalized data-set as an input to the Decision tree generation algorithm for generating Decision tree, similarly we can generate extended ER model.

Keywords
Data mining, Transformation, Aggregation, Data preparation, pivoting, SQL.

1. INTRODUCTION
In general a data mining project consists of four major phases such as the first phase involves extracting, cleaning and transforming data for analysis is called as data preparation, in the second phase a data mining algorithm analyzes the prepared data set, third phase validates results, creates reports and tunes parameters, the first, second and third phases are repeated until satisfactory results are obtained, during the fourth phase statistical results are deployed on new data sets [6]. In relational database environment, most of the time is spend to build normalized data-set in order to use it as input for a data mining algorithm. As the most of the data mining algorithm required input the data-set in horizontal layout, where every row corresponds to an observation, instance (possibly varying over time) and every column is associated to an one variable. Generally, database has a collection of normalized tables that must be joined, aggregated and transformed in order to build the required unique data set. As a result many complex relational SQL queries are written multiple time which are independent of each other. Such relational queries increase the size of database with many temporary tables (static) or views (dynamic), which are not represented as entities in an existing ER model. Such collection of transformation tables and Sql queries, which are written as independently multiple times create complication in database management, software development and maintenance. Existing SQL aggregations having some limitations to prepare normalized data sets because they return only one column per aggregated group [2]. This approach is related to an extract-load-transform (ELT) process, in which tables are cleaned and transformed after they are loaded into the database, as compare to traditional Extract-Transform-Load (ETL) tools most data transformation happens outside the DBMS, before loading data [1].

This paper is organized as follows: section 2 discusses Literature Survey. Section 3 introduce Mathematical model of system, Section 4 introduces Scheme Description. Section 5 describe the experimental result, section 6 gives conclusions and directions for future work.

2. LITERATURE SURVEY
For extended SQL syntax there are many proposals and SQL extensions to define aggregate functions for association rule mining. Carlos Ordonez et al proposed framework for programming a clustering algorithm with SQL queries is explored in, which shows horizontal layout of the data set enables easier and simpler SQL queries, their optimization have the purpose of avoiding joins to express cell formulas, but are not optimized to perform partial transposition for each group of result rows [5]. Horizontal aggregations are related to horizontal percentage aggregations, the differences between both approaches are that percentage aggregations require aggregating at two grouping levels, require dividing numbers and need taking care of numerical issues (e.g. dividing by zero), horizontal aggregations are more general, have wider applicability and in fact, they can be used as a primitive extended operator to compute percentages [4]. The paper is proposed by Carlos Ordonez, et al, in which extend an ER model with new entities to represent database transformations and introduced an algorithm to automate the process, extended ER model has two kinds of entities: source entities and transformation entities, which correspond to normalized tables and temporary tables created with SQL queries, respectively [1]. From the query processing side, SQL has been extended with operators to transform tables for cube exploration and data mining, Horizontal aggregations represent a combination of pivoting and aggregation in a single query, which are useful to create data sets, it is essential to represent such data transformation in an abstract form, as we do in our extended ER model [2].

Table 1: comparative study of existing papers

Sr. No.	Title	Advantages	Disadvantages
1	Extending ER models to capture database transformations to build data sets for data mining [1]	It bridges the gap between a logical database model represented by a standard ER model and a physical database model represented by SQL.	It adding a new query instead of modifying exiting one.
2	Horizontal Aggregations in SQL to Prepare Data Sets for Data Mining Analysis [2]	They represent a template to generate SQL code from a data mining tool	Horizontal aggregations do not introduce any conflict with vertical aggregations
3	Extended aggregations for databases with referential integrity issues[3]	It repair the original database to convert it into a consistent database.	The original database is updated and the user loses track of which data elements represent either repaired data or correct data
4	Integrating K-Means clustering With a Relational DBMS Using SQL [5]	Integrating data mining algorithms with a relational DBMS	Data set needs to be scanned several times

3. MATHEMATICAL MODEL

Let A be the system is represented as a 4-tuples: {U, Q, T, X} where,

- U be the relational database denoted as U(S, I) where,
  - S = {S_1, S_2, ... , S_m} be the set of m tables representing the existing source entities. Each table having a primary key K represented by an integer, p discrete attributes, D as dimension of look up table, and one numeric attribute (i.e. A) is: S (K, D_1, . . . ,D_p, A).
  - I is a set of integrity constraints. A referential integrity constraint, belonging to I, between two tables S_i and S_j is a statement of the form: Si(K)→ S_j(K), where Si is the referencing table, Sj is the referenced table, K is a foreign key (FK) in S_i and K is the primary key or a candidate key of S_j.
- Q = {Q_0, Q_1, Q_2, . . ., Q_n} be a sequence of n + 1 relational queries producing a chain of database transformations, where Q0 determines the universe of data mining records to be analyzed.
- T = {T_0, T_1, T_2, . . .,T_n} be the sequence of transformation tables, where X = T_n. Table T_0 will be used to build the final data set X with left outer joins.
- X be the desired normalized dataset, which is in the form of an entity X (K, A) with two attribute sets, where K is the primary key (possibly composite) and A is a set of non-key attributes.

4. SCHEME DISCRIPTION

4.1 SQL Code Generation: Query Evaluation

This approach, extending the standard SQL aggregation function and build a new class of aggregation called Horizontal aggregation, which produce tables with a horizontal layout. HA (Horizontal Aggregation) can be implemented by using three different methods:

- SPJ (Select-Project-Join) method relies on standard relational operators.
- CASE method relies on the SQL CASE construct.
- PIVOT method uses a built-in operator in a commercial DBMS that is not widely available.

Figure 1 shows the system control flow of entire system for getting data-sets.

4.1.1 SPJ Method:

The Select-project-Join-Aggregation (SPJ) method is based on relational operators only. The basic idea is to create one table with a vertical aggregation for each result column, and then join all those tables to produce S_H [2]. Horizontal aggregation using SPJ method require four input parameters to generate SQL code [9]:-

(i) The input table S

(ii) The list of GROUP BY columns L1, . . . ,Lm

(iii) The column to aggregate (A) and

(iv) The list of transposing columns R1, . . .,Rk.

The optimized SPJ method code is as follows:

INSERT INTO SH

SELECT S0.L1, S0.L2,. . . . ,S0.Lm,

S1.A, S2 .A,..., Sm .A

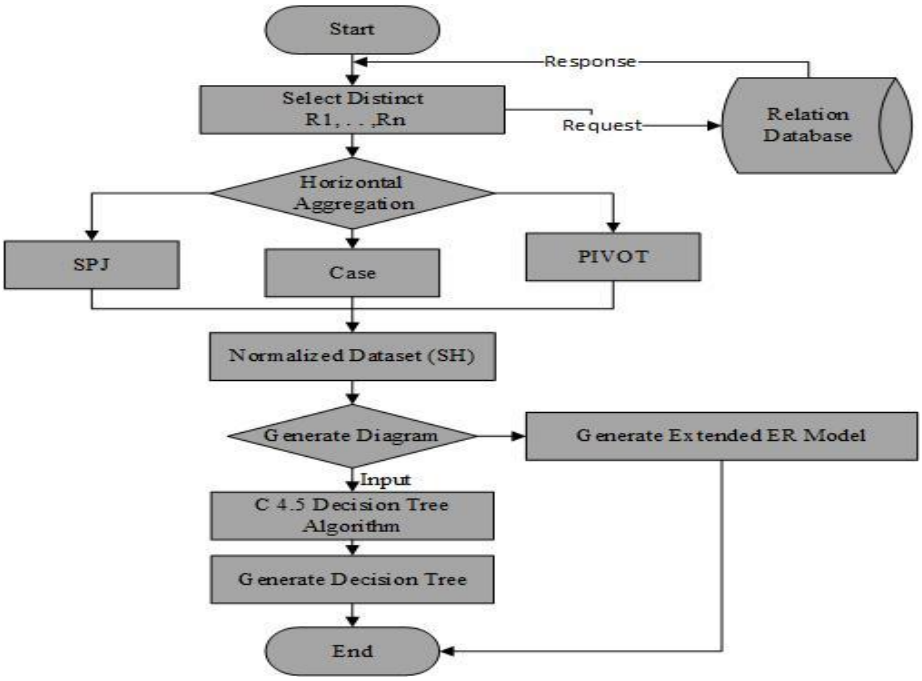

Fig 1: System Control Flow

```
FROM S0
LEFT OUTER JOIN S1
ON S0. L1= S1. L1 ...S0. Lm= S1. Lm
LEFT OUTER JOIN S2
ON S0. L1= S2. L1 ...S0. Lm= S2. Lm
. . . .
LEFT OUTER JOIN Sm
ON S0. L1= Sm. L1 ...S0. Lm= Sm. Lm;
```

4.1.2 CASE Method:

In this method, 'case' programming construct available in SQL is used. If Boolean expressions is satisfied then case statement returns a selected value from a group of values. From a relational database this is equivalent to doing a simple projection/aggregation query where each non-key value is given by a function that returns a number based on some conjunction of conditions [2]. The optimized case method code is as follows, where V () is a standard SQL aggregation that has a 'case' statement as an argument.

```
SELECT DISTINCT R1, . . .,Rk
FROM S;
INSERT INTO SH
SELECT L1, . . . ,Lm
, V (CASE WHEN R1 = v11 and . . . Rk = vk1
THEN A ELSE null END)
...
, V (CASE WHEN R1 = v1d and . . .Rk = vkd
THEN A ELSE null END)
FROM S
GROUP BY L1, L2, . . . ,Lm;
```

4.1.3 PIVOT Method:

The pivot operator is a built-in operator which transforms row to columns. Since this operator can perform transposition it can help in evaluating horizontal aggregation. The optimized set of queries which reduces the intermediate transposed table is as follows:

```
SELECT DISTINCT R1
FROM S;          /*produces v1,, vd*/
SELECT L1, L2,, Lm,v1,v2,.,vd
INTO SH
FROM (
SELECT L1, L2, …, Lm, R1,A
FROM S
) T
PIVOT (
V (A) FOR R1 in (v1, v2, vd)
) AS P;
```

5. EXPERIMENTAL RESULT

In our project we take weather Database which is created in Microsoft SQL Server Management Studio 2008.The Microsoft SQL Server 2008 Database Engine is a service for storing and processing data in either a relational (tabular) format or as XML documents. Datasets are prepared from the output of all these three (SPJ,CASE,PIVOT) method, so by applying these three methods on our Weather database we get Dataset in Attribute Relation File Format (.arff) as shown in figure 2. On that dataset Entropy and Information gain are calculated and a suitable decision rules are generated. Based on these rules and calculation decision tree is generated by

using C4.5 algorithm in WEKA [6] as shown in figure 3. Decision tree generation are implemented in WEKA algorithm and linked with java file.

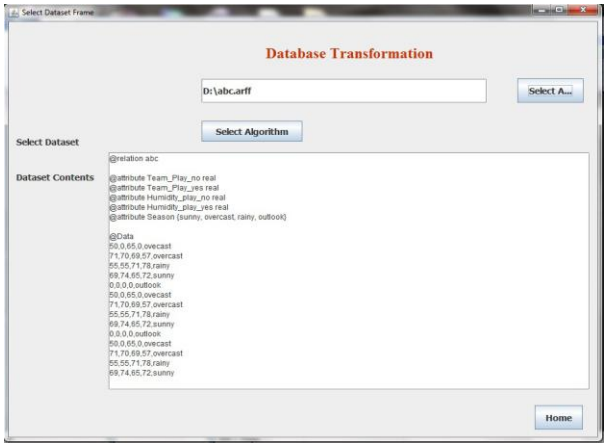

Fig 2: Data-set for weather database (.arff)

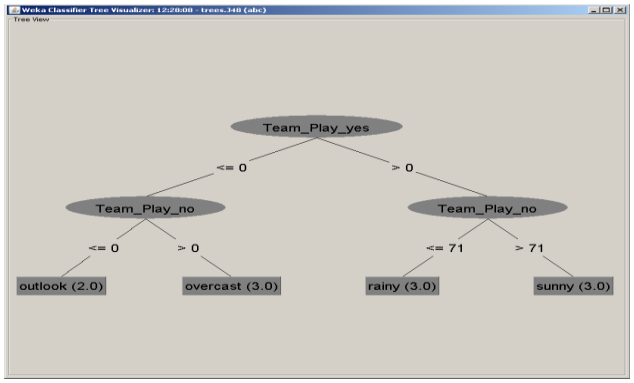

Fig 3: Decision tree using Dataset and C 4.5 Algorithm

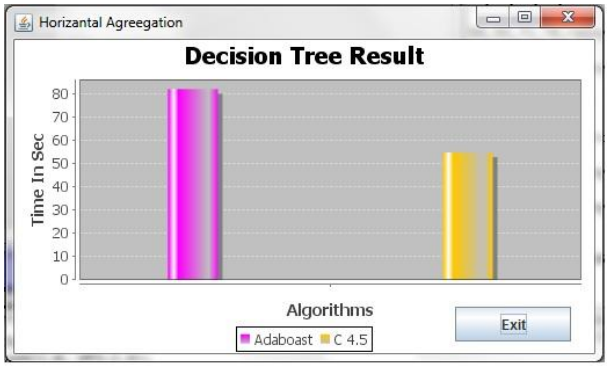

Fig 4: Comparative Result of Decision Tree Algorithm

Fig. 4 shows the time required for AdaBoost and C4.5 Decision tree algorithm. As database grows with many table and view which are not represent as entity in standard ER diagram, so we extending the ER model to generate the Extended ER-diagram as shown in figure 5.

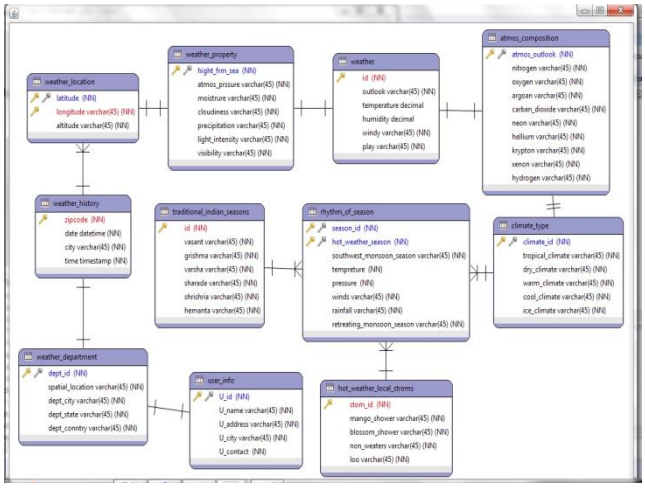

Fig 5: Extended ER-Diagram

Fig. 6 shows the impact of increasing the size of the fact table (N). The PIVOT and CASE methods show very similar performance but there is a big gap between PIVOT/CASE and SPJ for large data set.

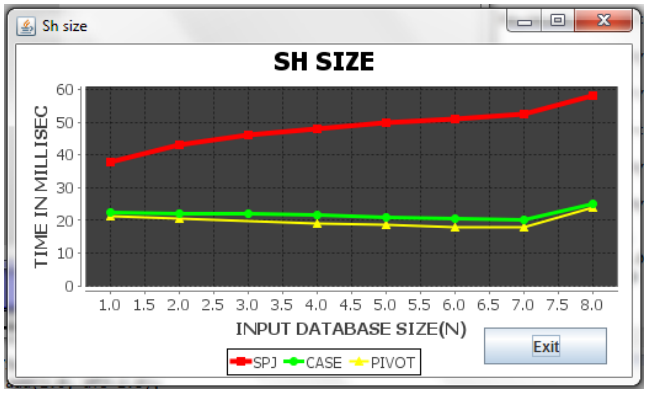

Fig 6: Statistical Result for Weather Database

6. CONCLUSION AND FUTURE WORK

In this approach a new class of aggregate functions in SQL is developed called as horizontal aggregation which is used for preparing data-sets for the data mining projects. Mainly, the existing SQL aggregations return results in one column per aggregated group, but horizontal aggregation returns a set of numbers instead of a single number for each group. Three query evaluation methods are discussed: the first method focuses on the relational operators in SQL, the second method focuses on the SQL case construct and third method focuses on the PIVOT built-in operator in a commercial DBMS. With the execution of methods of Horizontal Aggregation Datasets are created and these Datasets are used to generate Decision Tree. As database grows with many table and view which are not represent as entity in standard ER diagram, so we extending the ER model to generate the Extended ER-diagram.

In future work we can develop the most appropriate query optimization technique for the horizontal aggregation to achieve better results.

7. REFERENCES

[1] Carlos Ordonez, Sofian Maabout, David Sergio Matusevich, Wellington Cabrera, 2014 "Extending ER models to capture database transformations to build data sets for data mining", in Data and Knowledge Engineering.

[2] Carlos Ordonez and Zhibo Chen, 2012 "Horizontal Aggregations in SQL to Prepare Data Sets for Data Mining Analysis". In IEEE Transaction on Knowledge and Data Engineering.

[3] Javier Garca-Garcaa, Carlos Ordonez, 2010 "Extended aggregations for databases with referential integrity issues". In Data and Knowledge Engineering.

[4] Carlos Ordonez, 2004 "Vertical and Horizontal Percentage Aggregations". In Proc. ACM SIGMOD Intl Conf. Management of Data (SIGMOD 04)

[5] Carlos Ordonez, 2006 "Integrating K-Means Clustering with a Relational DBMS Using SQL". In IEEE Trans. Knowledge and Data Eng.

[6] Carlos Ordonez, 2004 "Horizontal Aggregations for Building Tabular Data Sets". In Proc. Ninth ACM SIGMOD Workshop Data Mining and Knowledge Discovery (DMKD 04).

[7] Hall, Mark, Eibe Frank, Geoffrey Holmes, Bernhard P fahringer, Peter Reutemann, and Ian H. Witten, "The WEKA data mining software: an update", ACM SIGKDD explorations newsletter, Vol. 11, no. 1, pp. 10-18, 2009.

[8] Archana A. Chaudhari, H. K. Khanuja, "Database Transformation to Build Data-set for Data Mining Analysis-A Review", Presented in ICCUBEA 2015 Sponsored by IEEE pune section Organized by Pimpri Chinchwad College Of Engineering(PCCOE), Pune .

[9] Archana A. Chaudhari, Harmeet Kaur Khanuja, "Extended SQL Aggregation for Database Transformation", International Journal of Computer Trends and Technology (IJCTT) , Vol 18, No. 6, pp 272-275, Dec 2014.