Similarity Aware SQL Based Group-By Operators for Multidimensional Relational Data

Anup Wadhekar\(^1\) Dr. Emmanuel M\(^2\)
\(^1\)PG Student \(^2\)Professor

\(^2\)Department of Information Technology
\(^1,2\)PICT, Pune, India

Abstract— The SQL group-by operator plays crucial role in summarizing large datasets in a data analytics. Allowing similarity aware grouping provides a more rational view on real-world data that could lead to better insights. Existing similarity-based grouping operators primarily focus on one-dimensional attributes. However correlated attributes, such as in spatial data, are processed independently. Hence, groups in the multi-dimensional space are not detected properly. To address this problem, two new SGB operators for multi-dimensional data are introduced. The first operator is the clique (or distance-to-all) nSGB, where all the tuples in a group are within some distance from each other. The second operator is the distance-to-any SGB, where a tuple belongs to a group if that tuple is within some distance from any other tuple in the group. Since a tuple may satisfy the membership criterion of multiple groups, we introduce three different semantics to deal with such a case: (i) eliminate the tuple, (ii) put the tuple in any one group, and (iii) create a new group for this tuple. The experimental study, based on TPC-H and a social check-in data proposes that the proposed algorithms can achieve up to three orders of magnitude improvement in performance over baseline methods developed to solve the same problem.

Key words: Multi-dimensional data, relational database, similarity operators, group by, tuples

I. INTRODUCTION

The deluge of data accumulated from sensors, social networks, computational sciences, and location-aware services calls for advanced querying and analytics that are often dependent on efficient aggregation and summarization techniques. The SQL Group-By operator is one main construct that is used in conjunction with aggregate operations to cluster the data into groups and produce useful summaries. Grouping is usually performed by aggregating into the same groups tuples with equal values on a certain subset of the attributes. However, many applications are often interested in grouping based on similar rather than strictly equal values. It is widely recognized that the move from exact semantics of data and queries to imprecise and approximate semantics is one of the key paradigm shifts in data management. Many application scenarios, e.g., marketing analysis, sensor networks, and biological applications, can greatly benefit from the identification and processing of similarities in data. Some techniques have been proposed to extend certain data operations, e.g., join and selection, to make use of data similarities [1].

Clustering is a well-known technique for grouping similar data items in the multi-dimensional space. Clustering groups data instances into subsets in such a manner that similar instances are grouped together, while different instances belong to different groups. The instances are thereby organized into an efficient representation that characterizes the population being sampled. In most cases, clustering is performed outside of the database system. Moving the data outside of the database to perform the clustering and then back into the database for further processing results in a costly impedance mismatch. Moreover, based on the needs of the underlying applications, the output clusters may need to be further processed by SQL to filter out some of the clusters and to perform other SQL operations on the remaining clusters. Hence, it would be greatly beneficial to develop practical and fast similarity group-by (SGB) operators that can be embedded within SQL to avoid such impedance mismatch and to benefit from the processing power of all the other SQL operators.

Fig. 1: The semantics of similarity predicates $\equiv 3$ [2]

SQL-based Similarity Group-By operators have been proposed in to support several semantics to group similar but not necessarily equal data. Although several applications can benefit from using existing SGB over Group-By, a key shortcoming of these operators is that they focus on one-dimensional data. Consequently, data can only be approximately grouped based on one attribute at a time. In this paper, author has introduced new similarity-based group by operators that group multi-dimensional data using various metric distance functions. More specifically, two SGB operators are proposed, namely SGB-All and SGB-Any, for grouping multi-dimensional data. SGB-All forms groups such that a tuple or a data item, say o, belongs to a group, say g, if o is at a distance within a user-defined threshold from all other data items in g. In other words, each group in SGB-All forms a clique of nearby data items in the multi-dimensional space. For example, all the two-dimensional points (a-e) in Fig. 1.1(a) are within distance 3 from each other and hence form a clique. They are all reported as members of one group as they are all part of the output of SGB-All. In contrast, SGB-Any forms groups such that a tuple or a data item, say o, belongs to a group, say g, if o is within a user-defined threshold from at least one other data item in g. For example, all the two dimensional points in Fig. 1.1(b) form one group. Point a is within Distance 3
from Point c that in turn is within Distance 3 from Points b, d, and f. Furthermore, Point e is within Distance 3 from Point d, and so on. Therefore, Points a-h of Fig. 1b are reported as members of one group as part of the output of SGB-Any [2].

Notice that in the SGB-All operator, a data item may qualify the membership criterion of multiple groups. For example, data item c in Fig. 1a forms a clique with two groups. In this case, we propose three semantics, namely, on-overlap join-any, on-overlap eliminate, and on-overlap form-new-group, for handling such a case. This paper provides efficient algorithms for computing the two proposed SGB operators over correlated multi-dimensional data. The proposed algorithms use a filter-refine paradigm. In the filter step, a fast yet conservative check is performed to identify the data items that are candidates to form groups. Some of the data items resulting from the filter step will end up being false-positives that will be discarded. The refinement step eliminates the false-positives to produce the final output groups. Notice that for the case of SGB-Any, a data item cannot belong to multiple groups. For example, consider a data item, say o, that is a member of two groups, say g1 and g2, i.e., o is within distance epsilon from at least one other data item in each of g1 and g2. In this case, based on the semantics of SGB-Any, Groups g1 and g2 merge into one encompassing bigger group that contains all members of g1, g2 and common data item o. Specifically, main focus is on two and three dimensional data space, leaving higher dimensions for future work.

The contributions of this paper are as follows:
1) Introduction of two new operators, namely SGB-All and SGB-Any, for grouping multi-dimensional data from within SQL.
2) An extensible algorithmic framework to accommodate the various semantics of SGB-All and SGB-Any along with various options to handle overlapping data among groups. We Effective optimizations for both operators are introduced.
3) The two operators are prototyped inside PostgreSQL and study their performance using the TPC-H benchmark. The experiments demonstrate that the proposed algorithms can achieve up to three orders of magnitude enhancement in performance over the baseline approaches. Moreover, the performance of the proposed SGB operators is comparable to that of relational Group-by, and outperform state-of-the-art clustering algorithm (i.e., K-means, DBSCAN and BIRCH) from one to three orders of magnitude.

II. LITERATURE SURVEY

Earlier work on similarity-aware query processing mentioned the theoretical groundwork and query optimization issues for similarity-aware query operators. A number of similarity operators are described and a complete conceptual analysis model is introduced for similarity queries. In addition, we present a rich set of transformation rules that permit cost-centric query optimization of similarity queries. [3], [4] introduce similarity algebra that extends relational algebra operations, e.g., joins and set operations, with similarity semantics. The mentioned algebra extends the relational algebra alternatively than begin from scratch, accordingly making it present a relational interface to non-relational structures - consequently, it’s consistent with prevailing trends in industry to add “datablades”, “extenders”, and “cartridges” to relational DBMS systems to extend their capabilities. Also they have got developed methods of enhancing queries over such similarity centered retrieval engines. Similarity queries and their optimizations comprise algorithms for similarity range search and k-Nearest Neighbor (KNN). Similarity queries in the form of range queries or k-nearest neighbor queries are the most important query forms for metric databases the place a metric distance perform is defined for pairs of database objects. Such queries play a foremost role in functions corresponding to multimedia systems, decision support systems and data mining [5], similarity join [6], and similarity aggregates [7]. Most of work focus on semantic and transformation ideas for query optimization independently from specific algorithms to understand similarity-aware operators. In contrast, our focus is on the latter.

Clustering methods generates groups of similar data for the purpose of learning hidden knowledge. Clustering methods and algorithms have been extensively studied in the literature. The main clustering methods are partitioning, hierarchical, and density-based. K-means [8] is a extensively used partitioning algorithm by many applications that uses several iterations to improve the output clusters. Hierarchical methods creates clusters either divisive (i.e., top-down ) equivalent to in BIRCH, or agglomerative (i.e., bottom-up) similar to in CURE. Density-based methods, e.g., DBSCAN, cluster data based on local criteria, e.g., density reachability among data elements. The key variations between our proposed SGB operators and clustering are: (1) The proposed SGB operators are relational operator which can be integrated in a relational query evaluation pipeline with more than a few grouping semantics. For that reason, they prevent the impedance mismatch experienced with the aid of standalone clustering and knowledge mining packages that make it mandatory to extract the information to be clustered out of the DBMS. (2) Unlike standalone clustering algorithms, the SGB operators can also be interleaved with other relational operators. (3) Standard relational query optimization techniques that apply to the standard relational group-by are also applicable to the SGB operators. This is not feasible with standalone clustering algorithms. Also, improved performance can be achieved by using database access methods that process multi-dimensional data.

An early work on similarity-based grouping is presented in [9]. It addresses the inconsistencies and redundancy encountered while integrating information systems with dirty data. Nonetheless, this work realizes similarity grouping by means of pairwise comparisons which incur immoderate computations within the absence of a correct index. Additionally, the offered extensions aren't built-in as first-class database operators. The work in [10] focuses on overcoming the limitations of the distinct-value group-by operator and introduces the SQL construct “Cluster By” that makes use of traditional clustering algorithms, e.g., DBSCAN, to achieve similarity grouping. Cluster By addresses the impedance mismatch due to the data being outside the DBMS to perform clustering. Our SGB operators are more generic as they use a collection of...
parameters and clauses to refine the grouping semantics, e.g., the distance relationships among the data elements that constitute the group and how intergroup overlaps are dealt with.

A number of DBMSs had been extended to help similarity operations. SIREN is a similarity retrieval engine that enables executing similarity queries over a relational DBMS. POSTGRESQL-1E is an image handling extension of PostgreSQL to support and achieve content-based image retrieval capabilities, e.g., supporting the image data type and responding to image similarity queries. While these extensions include various concepts of similarity into query processing, their main focus is on the similarity search operation. SimDB is a PostgreSQL extension that supports similarity-based queries and their optimizations. Several similarity operations, e.g., join and group-by, are implemented in as first class database operators. However, the similarity operators in SimDB focus on one-dimensional data and do not handle multi-dimensional attributes.

The proposed similarity-group-by operators, namely SGB-All and SGB-Any, need to index the processed group bounds (i.e., convex hull and rectangles), and data tuples on the fly, respectively. R-tree spatial index is used in the implementation. However, the R-tree can be replaced by other indexes such as a M-tree. While M-tree generally outperforms R-tree, trying different indexes to speedup SGB queries is not the main focus of this paper.

Finally, Hellerstein et al. [11] studied the online aggregation problem for grouping by operation. They proposed several techniques to guarantee that the aggregation outputs deterministic results regardless of the order in which the tuples are processed. In [12], the deterministic properties of the newly proposed SGB operator are analyzed and proved. Specially, the SGB-All (with overlap options ELIMINATE and FORM-NEW-GROUP) and SGB-ANY are deterministic operators.

### A. Comparison of Similarity Grouping Techniques

<table>
<thead>
<tr>
<th>Parameters v/s</th>
<th>Database</th>
<th>Grouping Attribute</th>
<th>Support for traditional Group by operator</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Similarity</td>
<td>1-dimensional data</td>
<td>1 attribute at a time</td>
<td>Yes</td>
<td>Less than SGB</td>
</tr>
<tr>
<td>Queries [1]</td>
<td>Multidimensional database</td>
<td>More than 1 attribute at a time</td>
<td>Yes</td>
<td>Better</td>
</tr>
<tr>
<td>Proposed SGB</td>
<td>Multidimensional database</td>
<td>More than 1 attribute at a time</td>
<td>No</td>
<td>Less than SGB</td>
</tr>
<tr>
<td>Operators [2]</td>
<td>Multimedia Image Database</td>
<td>More than 1 attribute at a time</td>
<td>Focus on selection and Join operations</td>
<td></td>
</tr>
<tr>
<td>Clustering</td>
<td>Multidimensional database</td>
<td>More than 1 attribute at a time</td>
<td>Yes without aggregation functions</td>
<td>Less than SGB</td>
</tr>
<tr>
<td>Algorithms</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Novel Similarity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Based Relational</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operators [4]</td>
<td>Multidimensional database</td>
<td>More than 1 attribute at a time</td>
<td>Yes without aggregation functions</td>
<td>Less than SGB</td>
</tr>
<tr>
<td>Similarity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Based variants</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>of grouping</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>and join operators</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[9]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Comparison of various similarity based grouping techniques

### III. CONCLUSION

In this paper, we have reviewed different similarity based grouping techniques for over single and multidimensional data. New similarity grouping operators are defined with a variety of practical and useful semantics to handle overlap. This system proposes extensible algorithmic framework to efficiently implement these operators inside a relational database management system under a variety of semantic flavors. The performance of SGB-All performs up to three orders of magnitude better than the naive All-Pairs grouping method. Moreover, the performance of the optimized SGB-Any is more than three orders of magnitude better as compared to naive approach. Finally, the performance of the proposed SGB operators is comparable to that of standard relational group by

### REFERENCES


