Automated Generation of Object Summaries from Relational Databases: A Novel Keyword Searching Paradigm

Georgios John Fakas

Department of Computing and Mathematics,
Manchester Metropolitan University
Manchester, UK.
g.fakas@mmu.ac.uk

***Abstract*— This paper introduces a novel keyword searching paradigm in Relational Databases (DBs), where the result of a search is a ranked set of Object Summaries (OSs). An OS summarizes all data held about a Data Subject (DS) in the Database. More precisely, it is a tree with a tuple containing the Keyword as a root and neighboring tuples as children. In contrast to traditional Relational Keyword Search (R-KwS), an OS comprises a more complete and therefore semantically meaningful set of information about the enquired DS.**

The proposed paradigm is based on two key concepts: Affinity and Importance. The system investigates and quantifies the Affinity of relations in order to automatically create OSs and OS Importance (Im(OS)) in order to rank them. Im(OS)s considers the weight (i.e. PageRank) of tuples, Affinity and size of OS.

Experimental evaluation on TPC-H and Northwind DBs so far verifies the searching quality of the proposed paradigm.

I. INTRODUCTION

The Keyword Search paradigm has been successfully used in Relational Databases (R-KwS). Such paradigms have significant usability advantages as they liberate users from technical details such as Database Schemata and Query Languages. In general, R-KwSs facilitate users to do advanced search using a set of keywords; e.g. "Peacock Fuller" which will return trees of nodes containing information associating the two keywords such as Orders of Customer Peacock prepared by Employee Fuller (in the context of Nothwind DB). Yet, unlike Web Kw Search (W-KwS) results (namely a web page), R-KwS results fail to provide a complete and meaningful set of information about a particular DS e.g. "Peacock"; since additional information is required to comprise a meaningful result for "Peacock"; i.e. his Nation, Orders etc.

In this paper, a novel Kw Search paradigm is proposed that produces results which are comprised of a more complete set of information about the DS in interest; namely a ranked set of Object Summaries (OS). For instance, when searching information about "Peacock" the proposed paradigm will produce a ranked set of OSs containing Kw "Peacock" at their root nodes. More precisely in the Northwind DB context, an OS will be comprised of an Employee tuple (with Employee.LastName="Peacock") as a root and child nodes including additional information about his Nation, Regions, Orders he served etc. Similarly, to R-KwS, it liberates users from Query language and schema technical details. In order to produce an OS, the proposed paradigm traverses the data-graph as follows: it starts from a tuple containing the Kw (denoted as t^{DS}) and continues traversing neighboring tuples as long as the data traversed is relevant to t^{DS}.

Challenges

The proposed search paradigm faces several challenges. The primary challenge is the classification of neighboring data as relevant or irrelevant to t^{DS}. For this reason, the semantic of Affinity of surrounding relations to the Relation of t^{DS} (denoted as R^{DS}) is investigated and quantified so as to select which relations to traverse. These Affinity scores in combination with a threshold provided by the DBA (or users) will facilitate the decision on which relations to retrieve in the context of an OS and therefore liberate the user from the schema details. The Affinity is calculated based on the combination of schema design and data distribution.

The ranking of OS results is another challenging problem, since existing ranking semantics of traditional R-KwS are completely inappropriate for OS ranking. This is because in R-KwS, generally a result of a small size has a higher ranking semantic than another result of a larger size [1, 4, 5, 6]. In contrast, in the proposed paradigm an OS containing many and well connected tuples should have certainly greater importance than an OS with less tuples. For instance, a Customer or Employee OS involved in many Orders or an Author authored many important papers and books. Therefore, an efficient ranking technique is required that will weight the Importance of OSs based on these criteria but at the same time limit user's input to only a Kw.

Contributions

The novel contributions of this paper are the following:

- The formal definition of the novel Searching Paradigm which automatically produces a ranked set of OSs. The novelty of this paradigm is that it requires minimum

contribution from the user (i.e. only a Kw) and does not require any prior DB registration, prior knowledge of the DB schema or any query language.

- The formal definition and calculation of Affinity among Relations of Relational DBs and the proposition of the Affinity formula (i.e. the building block of the paradigm). The Affinity formula considers both Schema Design and Data distributions. The excellent Precision, Recall, F-score (P/R/F) and Affinity Ranking Correctness results of OSs validate the quality of the Affinity Formula.
- The proposition of a novel ranking paradigm based on a *Combine Function* that considers the (1) weight (e.g. PageRank) of tuples, (2) Affinity and (3) size of OS in order to calculate Im(OS). The consideration of tuples' centrality (i.e. PageRank) satisfies the requirement of minimum user's input (i.e. only a Kw).
- The pre-computation and indexing of Im(OS)s values for each potential OS improves significantly systems performance during user's search.

Paper Organization

The rest of the paper is structured as follows. Section II presents the proposed searching paradigm and Affinity semantics while section III discuses OSs ranking. Section IV presents the experimental results and section V concludes the paper.

II. The Proposed Searching Paradigm: OS

The proposed paradigm is illustrated with examples from the Microsoft Northwind DB (see schema below).

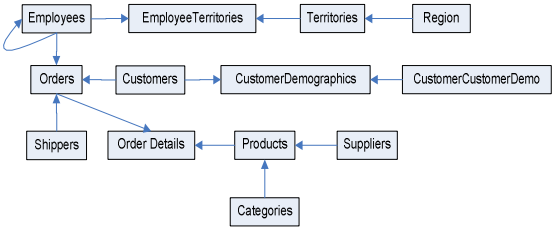

Fig. 1 The Northwind Database Schema

A. OS Generation

In order to construct OSs, the proposed approach combines the use of Graphs and SQL. The rationale is based on the fact that some relations, denoted as R^{DS} (where $t^{DS} \in R^{DS}$ includes a Kw), hold information about Data Subjects (DSs) and the relations linked around R^{DS}s contain additional information about the particular DS. For each R^{DS}, a Data Subject Schema Graph (G^{DS}) is automatically generated; namely a Directed Labeled Tree that captures a subset of the database schema with $\mathbf{R^{DS}}$ as a root. Affinity measures of relations in G^{DS} are investigated, quantified and annotated on the G^{DS}. G^{DS} is also annotated with Cardinality, Relative Cardinal etc. (Fig. 2). Provided an Affinity threshold θ (either by the DBA or user) a subset of G^{DS} can be produced; denoted as $G^{DS}(\theta)$. Finally, by traversing the $G^{DS}(\theta)$ we can now proceed with the generation of OSs.

For instance, for the keyword search "Janet Peacock" and θ=0.7 the system will automatically generate the report presented in Fig. 3. Since the Kw is found in tuple e_3 (i.e. the t^{DS} which belongs to the Employees relation) then Employees G^{DS} will be used (Fig. 2).

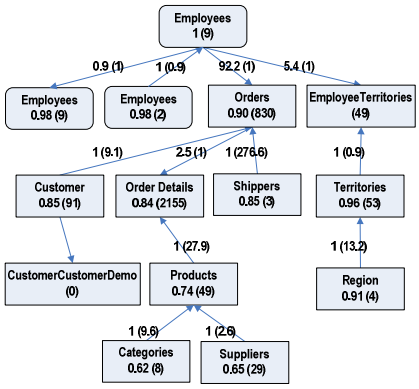

Fig. 2 Employees G^{DS}s

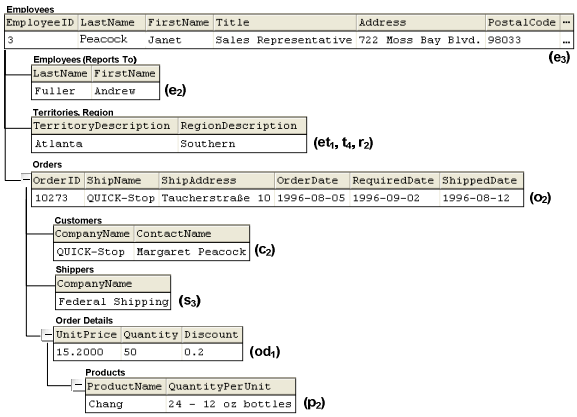

Fig. 3 The OS for "Janet Peacock"

B. Affinity Calculation

The semantic of Affinity $\mathrm{Af}_{R_i \to R^{DS}}$ (or Af_{R_i}) between a relation R_i to R^{DS} in relational DBs considers the following metrics.

1. Relations' Distance.

The primary metric for closeness between R_i to R^{DS} is their distance (denoted as ld_i) on the DB schema, namely the length (i.e. the number of relationships) of a path from R_i to the R^{DS}. The shorter the distance is the bigger the affinity between the two relations is.

2. Connectivity.

A secondary metric of closeness between R_i to R^{DS} is R_i's connectivity on both the DB schema, denoted as Schema Connectivity (Co_i) and the data-graph, denoted as Relative Cardinality ($RC_{i \to j}$). For instance, the relation CustomerDemographic which has Co_i=1 (and ld_i=1) is closer to Customer R^{DS} than Orders which has Co_i=4 (and ld_i=1). Let $RC_{i \to j}$ represent the Relative Cardinality of R_i and R_j, namely the average number of tuples of R_i that are connected with each tuple from R_j (this concept was also used in [7]). Now,

let Reverse Relative Cardinality, denoted as $\overline{RC_{i\to j}}$, be the reverse of $RC_{i\to j}$ (i.e. $\overline{RC_{i\to j}} = RC_{j\to i}$).

3. Penalisation of Lateral Data (from Hub Relations).

Analysing further Relational DB schemata, we realize that 'hub' relations give lateral data to the R^{DS} [1, 3]; such paths containing 'hubs' have the following structure: $R^{DS}\ldots\leftarrow R_{hub}\rightarrow R_2$. For instance in the TPC-H database, $R_{Supplier}$ with ld_i=2 from the $R^{Customers}\leftarrow R_{Nation}\rightarrow R_{Supplier}$ path will result to a big set of Suppliers coming from the same nation as the Customer; something not directly relevant to Customer. This can be penalised by increasing the impact of ld_i (e.g. ld_i=ld_i*h) and Relative Cardinality.

Let the Affinity Descriptor of R_i to R^{DS} be a list of weighted metrics; namely, $DAf(R_i)$={(m_1, w_1), (m_2, w_2),.. (m_n, w_n)}, where Σw_i=1. Then, metrics m_1...m_n are as follows: $m_1=f_1(ld_i)$, $m_2=f_1(\log(10*RC_i))$, $m_3=f_1(\log(10*\overline{RC_i}))$, $m_4=f_1(\log(10*Co_i)$ in the case of a hub-child $m_1=f_1(ld_i*h_i)$ and $m_2=f_1(RC_i)$ (i.e. penalization of hub-child relations), where $f_1(\alpha)=(11-\alpha)/10$.

Definition 1 (Affinity): *The Semantic of Affinity of R_i to R^{DS}, denoted as $Af_{R_i\to R^{DS}}$ (or Af_{R_i}), with respect to a schema and a database conforming to the schema, can be calculated with the following formula:*

$$Af_{R_i\to R^{DS}} = \sum_j m_j w_j \cdot Af_{R_{Parent}\to R^{DS}} \quad (1)$$

where j ranges over all metrics, $Af_{R_{Parent}\to R^{DS}}$ is the Affinity of the R_i's Parent to R^{DS} or is 1 if $R^{Parent}\equiv R^{DS}$.□

III. OSs Ranking and Presenation

The proposed ranking paradigm ranks OS descending their Importance Scores, denoted as *Im(OS),* which is calculated by employing a *Combine function* that considers the (1) PageRank of each tuple (2) Affinity and (3) size of OS. The following Combine functions is considered where $Im(t_i)$ is the Importance of t_i (i.e. PageRank) that belongs to OS, |OS| is the amount of tuples in OS, and $Af_R(t_i)$ is the affinity of R that t_i belongs to.

$$Im(OS)= \frac{\sum Im(t_i) * Af_R(t_i)}{log(|OS|)+1} \quad (2)$$

For usability and efficiency reasons (since the amount and size of OS results may be large), OSs may also be presented partially to users (rather than completely) and then the user selects which *Partial OSs* to completely expand. Partial OSs may contain up to 5 tuples and the selection of these tuples is rather simple; i.e. one tuple from each Relation with the highest Affinity. This option is in accordance with W-KwSs where Web Page answers are presented with their titles and short text fragments associating them with Kws. Similarly to W-KwS results, a "lucky" user may satisfy his/her query with this summarized information.

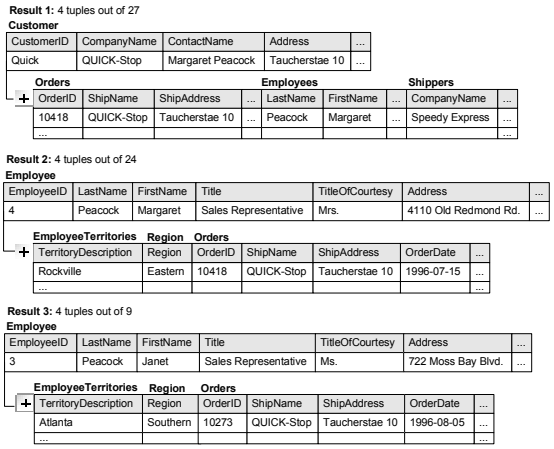

Fig. 4 The Ranked OSs for "Peacock"

A way to speed up the generation and therefore the presentation of partial results to users during search time is to employ a pre-computed index that stores all possible Im(OS) values for each tuple t^{DS}. Such an index will facilitate the generation of ranked Partial OS results without actually implementing time consuming Complete OS results.

IV. Experimental Evaluation

The proposed search paradigm has been evaluated with two databases, namely Northwind and TPC-H. As this is work still in progress we present the results produced so far. A survey was conducted with ten lecturers and students from our department and also from the School of Computer Science of the University of Manchester. The participants were given twelve G^{DS}s (i.e. six from TCP-H and six from Northwind DB) and were asked to define manually their own G^{DS}, denoted as G^{DS}(h)s. The results presented below compare the OSs produced by $G^{DS}(\theta)$ and participants' G^{DS}(h); i.e. the average of P/R/F (with θ=0.70 and w_1=0.5, w_2=0.4, w_3=0.05 w_4=0.05 and h=1.6).

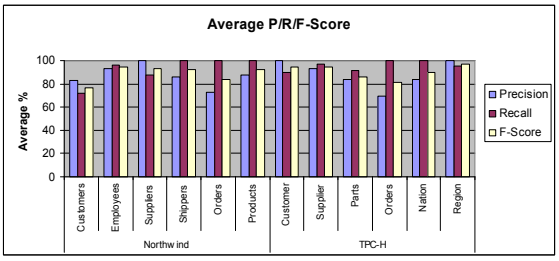

Fig. 5. Precision, Recall and F-Score

Another accurate measurement of the quality of our approach is the correctness of the ranking of Relations based on their Affinity. The results in Figure 6 depict the average of Affinity Ranking Correctness (ARC) of the proposed Affinity formula against evaluators' rankings. In summary, results so far are very encouraging. We are currently evaluating the OS ranking paradigm.

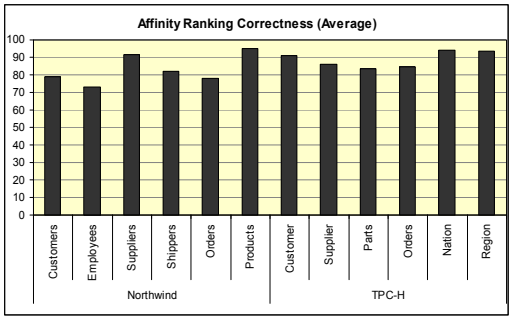

Fig. 6 Affinity Ranking Correctness

V. Conclusion and Future Work

This paper introduces the work in progress of a novel Keyword Searching Paradigm that facilitates the automated extraction of data held about DSs in a DB. According to the best of our knowledge, this KW searching paradigm has not been attempted before. Such a Searching paradigm, that liberates users from Schema and query Languages, will certainly be a great contribution especially now with the wider use of Web-Accessible DBs.

A direction of future work concerns top-k size of the OS or top-k results. Both problems are very challenging since the weights of new tuples (i.e. a function of PageRanks, Affinity and |OS|) comprising an OS according to the proposed Combine function are not monotonic (since a tuple's PageRank may increase while its Affinity decrease). In addition, alternative to PageRank weighting systems are currently investigated e.g. ObjectRanks [2, 5].

References

[1] B. Aditya, G. Bhalotia, S. Chakrabarti, A. Hulgeri, C. Nakhe, P. S. Sudarshan, "BANKS: Browsing and keyword searching in relational databases", *In VLDB*, 2002.

[2] A. Balmin, V. Hristidis, and Y. Papakonstantinou, "Objectrank: Authority-based keyword search in databases", *In VLDB,* 2004.

[3] R. Goldman, N. Shivakumar, S. Venkatasubramanian, and H. Garcia-Molina, "Proximity search in databases", *In VLDB*, pp. 26–37, 1998.

[4] V. Hristidis, and Y. Papakonstantinou, "DISCOVER: Keyword search in relational databases", *In VLDB*, 2002.

[5] V. Hristidis, L. Gravano, and Y. Papakonstantinou, "Efficient IR-style keyword search over relational databases", *In VLDB*, 2003.

[6] A. Markowetz., Y. Yang, and D. Papadias. "Keyword search on relational data streams", In SIGMOD, 2007.

[7] C. Yu, H.V. Jagadish, "Schema Summarization", *In VLDB*, 2006.