QUERY BY TABLE

Raghav Padmanabhan, George Nagy
ECSE Department, RPI, Troy NY-12180
padmar2@rpi.edu, nagy@ecse.rpi.edu

Abstract

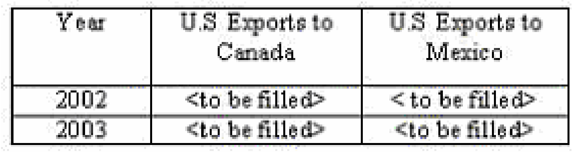
Querying any information system requires the knowledge of some formal language, making it inaccessible to computer-naïve potential users. We propose a new intuitive querying mechanism where the query is a (well-formed) table. We extract the underlying logical structure of the table to retrieve values from a database. Query tables are interpreted to perform simple SELECT & JOIN operations. We demonstrate that query tables with different layouts but with the same underlying logical structure yield correct answers. This approach can be extended to form complicated conditional queries and queries involving aggregates.

1. Introduction

Query Languages are high level programming languages that allow users to make queries into databases. Examples include SMARTS, the cheminformatics standard for a substructure search, XQuery, a query language for XML data sources and SQL for relational databases. The semantics of the query are defined by a formal syntax specific to that language. In contrast, Query By Example and Query By Browsing have a simple graphical user interface. We present here, Query by Table (QBT), an intuitive mechanism based on the idea that well-formed tables can represent queries to a relational database.

One interesting aspect of tables is that answers to certain kinds of queries seem most naturally expressed in tabular form. Consider, for example, the query: *"How do the volume of U.S. exports to Mexico compare to those to Canada for the years 2002 and 2003?"* This question can be formulated as a query in the form of the table.

Table I. Query Q1



Year	U.S Exports to Canada	U.S Exports to Mexico
2002	<to be filled>	< to be filled>
2003	<to be filled>	<to be filled>

QBT takes inputs in the form of a table shown above and fills in the values. The method is motivated by our ongoing research in Project TANGO (Table ANalysis for Generating Ontologies) – a collaborative project between Brigham Young University and Rensselaer Polytechnic Institute - where we have applied conceptual model-based data extraction and table recognition to understand a table's structure and its conceptual content for semi-automatic ontology generation [1]. We have used ideas drawn from the project to develop a simple paradigm for information retrieval. We expect to extend this method to retrieving information from the ontologies we create.

To use tables as queries to a database, we distinguish between a table's physical structure and its logical structure. Users may formulate a query table's layout in many ways yet request the same values. Our starting point is the formalism put forth by Wang in [2] to interpret the table, i.e., to recover category labels and the functions that map them to a set of domains. Table interpretation is not trivial. Experiments have shown that even human "experts" often disagree on the label-value pairs of a table [3]. The Wang notation for a query table is currently extracted by an interactive tool that requires very little user intervention for simple tables [4] and can be eventually automated.

The paper is organized as follows. We first describe the Wang notation for a table and derive it for an example table. We discuss the dimensional model of a database and its similarity to Wang Notation. The distinctions between query tables and normal tables are sketched and the QBT process is demonstrated. We conclude with examples of the different types of queries that the system has successfully answered,

Preprint submitted to 19th International Conference on Pattern Recognition.
Received April 5, 2008.

followed by a brief discussion of queries that QBT cannot handle yet.

2. Wang Notation

Wang defines an *abstract table* as an abstract data type and its layout structure as the *presentation form* of a table. The logical structure consists of entries and labels. The organization of the labels is called a *frame*, and the number of categories in the frame is the *dimension* of the abstract table. Applying a layout specification to an abstract table generates a *concrete table*. Tabular abstraction separates the logical structure of a table from its layout structure. The advantage of tabular abstraction is that the tables can be manipulated independently of their layout structure and we can easily alter the layout of a table by associating different topologies with the logical structure. Informally, the Wang Notation consists of two components (C, δ) where C is a finite set of labeled domains and δ is a mapping from the tree paths labels (or headers) to the possible values.

Table II is a 3-dimensional table. The categories are Year, Term and Mark. The Wang category notation for the table is

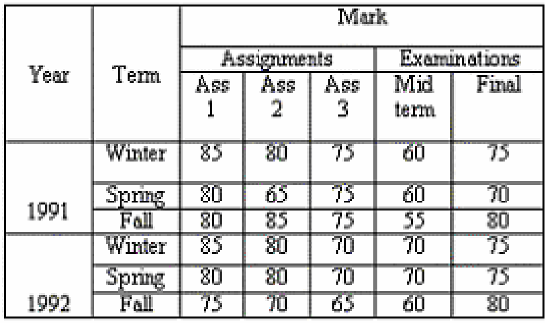
(Year, {(1991,φ), (1992,φ)})
(Term, {(Winter,φ), (Spring,φ), (Fall,φ)})
(Mark, {(Assignments, {(Ass1,φ), (Ass2,φ),
(Ass3,φ)}), (Examinations, {(Midterm,φ),
Final,φ})}))

Year is the first category with 1991 and 1992 as sub-categories. Term is the next category with *Winter*, *Spring* and *Fall* as the subcategories. *Mark* is the last category with two subcategories (*Assignments and Examinations*). *Assignments* and *Examinations* have their own subcategories. The delta notation shows which category cells are related to each of the individual values within the table. The δ notation for the first row of the table is

δ({Year.1991,Term.Winter,Mark.Assignments.Ass1})=85
δ({Year.1991,Term.Winter,Mark.Assignments.Ass2})=80
δ({Year.1991,Term.Winter,Mark.Assignments.Ass3}) =75
δ({Year.1991,Term.Winter,Mark.Examinations.Midterm})=60
δ({Year.1991,Term.Winter,Mark.Examinations.Final}=75

Some tables lack spanning labels. For example, Table II would still be considered "valid" even if the header "Mark" were absent from the table. But the logical structure requires a root for the column header tree paths. This requires the addition of what Wang called "virtual header". Virtual headers are implicit headers added by the user in order to complete the logical structure of the table to obtain its abstract notation. The absence of explicit headers is the major difference between high-level and low-level table interpretation.

Table II. Wang table



Year	Term	Mark				
		Assignments			Examinations	
		Ass 1	Ass 2	Ass 3	Mid term	Final
1991	Winter	85	80	75	60	75
	Spring	80	65	75	60	70
	Fall	80	85	75	55	80
1992	Winter	85	80	70	70	75
	Spring	80	80	70	70	75
	Fall	75	70	65	60	80

Lopresti et al. in [5] discuss the similarity of the relational paradigm to Wang's idea of abstract tables. A database table is a two-dimensional table with *attributes* in one dimension and *tuples* in the other. The Wang notation captures the idea of attributes in its C notation. The tuples are specified in the δ notation.

3. Dimensional Database

Query By Table retrieves values from a relational database, which is modeled dimensionally. The following paragraphs explain the concepts of Dimensional Data Modeling.

A dimensional database is like a database cube of *n* dimensions. Users can access a slice of the database along any of its dimensions. The dimensional model is also called the *star-join* schema. The central table is the only table in the schema with multiple joins connecting it to all the other tables. The central table is called the *fact table* and the other tables are called *dimension tables*. The dimension tables have a single join that attaches them to the fact table. The fact table stores the measures or facts, which represent quantitative or factual data about the subject. The measures are generally numeric and correspond to the *how much* or *how many* aspects of a question. A dimension represents a single set of objects or events in the real world. Each dimension we identify for the data model is implemented as a dimension table. Dimensions are the qualifiers that give meaning to the fact table measures. They answer the *what*, *when*, and *where* aspects of a question. A dimensional approach simplifies access to data that we want to summarize or compare. We chose tables from the Canada

Statistics Website to form the simple dimensional database. The dimensions of this database were Region, Year. The facts were identified as Population, Median Total Income, Number of Abortions, Infant Mortality Rate, Number of Homicides, Abortion Rate, Number of Divorces and Number of Trips made by Canadians to some Region.

4. Query By Table

A query table is a well-formed user-constructed table that encapsulates all the information internal to its structure and uses exact attribute names for the facts to be retrieved. A query table does not use any headings, captions or titles to convey information. The Query by Table system is currently implemented in MATLAB. It accepts an MS Excel query table as an input and retrieves the values requested by the user. The process has five steps:

- Derive the Wang Notation for the query table.
- Parse the Wang Notation of the input table.
- Identify the facts and dimensions of the query.
- Form SQL queries from the facts and dimensions.
- Plug the results back into the query table.

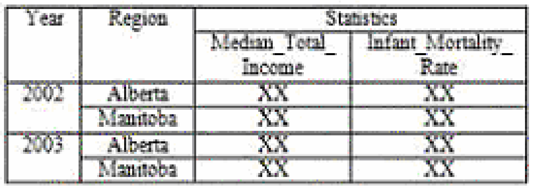
Consider the query shown in Table III. An Excel macro program converts the table into the ASCII format required by the Wang Notation Tool. The user interacts with the tool to produce the Wang Notation for the table. The Wang Notation for this 3-dimensional table is

C Notation:
(Year,{(2002,phi),(2003,phi)})
(Region,{(Alberta, phi),(Manitoba, phi)})
(Statistics,{(Median_Total_Income,phi),
(Infant_Mortality_Rate,phi)})

Delta Notation:
delta({Statistics.Median_Total_Income,
Year.2002, Region.Alberta})=XX
delta({Statistics.Infant_Mortality_rate, Year.2002,
Region.Alberta})=XX
…

The Wang Category Notation of the query table is parsed using regular expressions. Dimensions are stored as separate tables in the database, so we can query the database to determine if a header is actually a dimension for the data value, if it is a fact, or if it is merely a dummy header. We initially consider three values – Region, Year and Statistics. On querying the database for Region and Year, we conclude that both of them are keys. For each row in the delta notation, all the subcategories associated with the keys are stored separately. Querying the fact table for Statistics retrieves no values. We conclude that the term is a dummy header and examine its subcategories. The database is queried for the first subcategory of Statistics, Median_Total_Income. Since it is a column in the fact table, it must be a fact and a value to be retrieved. Similarly, another fact is identified to be Infant_Mortality_Rate for each combination of Region and Year. Now, we have the facts to be retrieved and the dimensions for the facts. This information is sufficient to form SQL queries to retrieve values from the database by looping through simple SELECT statements to perform a simple JOIN. The data is plugged back into the Excel sheet by making use of its explicit table layout.

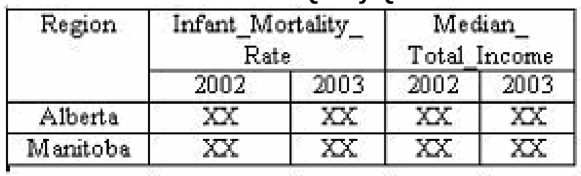
Table III. Query Q2



Year	Region	Statistics	
		Median_Total_Income	Infant_Mortality_Rate
2002	Alberta	XX	XX
	Manitoba	XX	XX
2003	Alberta	XX	XX
	Manitoba	XX	XX

4.1. Why Wang Notation for QBT?

Table IV. Query Q3



Region	Infant_Mortality_Rate		Median_Total_Income	
	2002	2003	2002	2003
Alberta	XX	XX	XX	XX
Manitoba	XX	XX	XX	XX

Table IV is a variation of Table III, with one of the dimensions hidden under the facts to be retrieved. This table requires two additional virtual headers: Year, which is a suitable header name for the sub- categories 2002 and 2003 and Statistics, which is a reasonable header name for the subcategories Infant_ Mortality_Rate and Median_Total_Income. Thus, after the addition of the Virtual Headers the Wang Notation becomes

(Year,{(2002,phi),(2003,phi)})
(Region,{(Alberta,phi),(Manitoba,phi)})
(Statistics,{(Median_Total_Income,phi),
(Infant_Mortality_Rate,phi)})

delta({Statistics.Median_Total_Income,Year.2002,
Region.Alberta})=XX

delta({Statistics.Infant_Mortality_Rate,Year.2002,
Region.Alberta})=XX
…

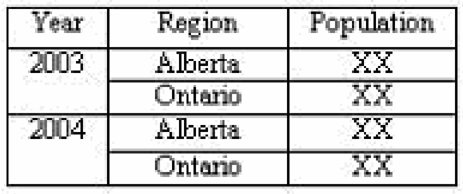
Although the layout of Table IV is different from the layout of Table III, the Wang Notation preserves the underlying logical structure. Even if the added virtual headers are different from Year and Statistics (say Period and Values), we can still retrieve the values by looking into their subcategories.

4.2. Example Queries

This section illustrates some of the queries that the system has successfully answered. We can present here only small examples. Some of our experimental query tables for Canada Statistics are much larger, but have similar structures. We first present some narrative queries and then their possible query table representations.

Query Q4:
"How do the populations of Alberta and Ontario compare with one another for the years 2003 and 2004?"

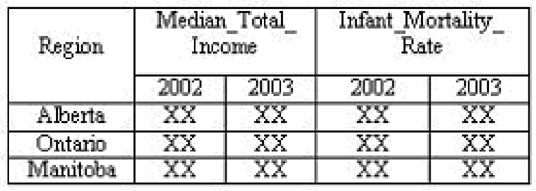
Table V. Query Q4



Year	Region	Population
2003	Alberta	XX
	Ontario	XX
2004	Alberta	XX
	Ontario	XX

Query Q5:
"For the provinces Alberta, Ontario and Manitoba, did the change in Median Total Income in the years 2002 and 2003 affect the Infant Mortality Rate?"

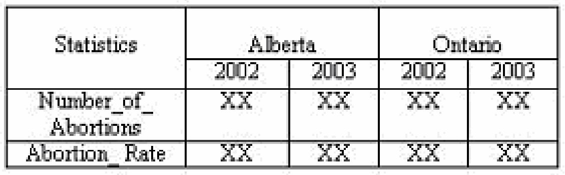
Table VI. Query Q5



Region	Median_Total_Income		Infant_Mortality_Rate	
	2002	2003	2002	2003
Alberta	XX	XX	XX	XX
Ontario	XX	XX	XX	XX
Manitoba	XX	XX	XX	XX

Query Q6:
"For the provinces Alberta, Ontario, how do the abortion-related statistics vary in the years 2002 and 2003?"

Table VII. Query Q6



Statistics	Alberta		Ontario	
	2002	2003	2002	2003
Number_of_Abortions	XX	XX	XX	XX
Abortion_ Rate	XX	XX	XX	XX

5. Summary and Conclusions

We have demonstrated the retrieval of values from a simple dimensional database by means of a query table. The database is constructed by agglomerating information from web tables, but a query may require data from several tables. We are currently scaling up the database with many more tables. The challenge is to identify the dimensions and facts correctly and efficiently. The queries we can answer so far require only Select and Join operations. We are now developing methods to answer more complicated queries including aggregation operations like summing averaging, and evaluating maxima, minima and conditional queries. The mild constraints imposed on the query tables are readily accepted by naïve users. Through our ongoing research in the TANGO project, we also expect to make query tables even more intuitive and more widely accessible.

Acknowledgment: This work was supported by the National Science Foundation under Grant # 04414854

References:

[1] Yuri A. Tijerino, D.W. Embley, Deryle W. Lonsdale, and G. Nagy, "Towards ontology generation from tables," World Wide Web Journal, vol. 6, #3, Springer-Verlag, September 2005.
[2] Wang, X.: Tabular abstraction, editing, and formatting. Ph.D. thesis,University of Waterloo (1996)
[3] Hu, J., Kashi, R., Lopresti, D., Nagy, G., Wilfong, G.: "Why table ground-truthing is hard." In: Proceedings of the Sixth International Conference on Document Analysis and Recognition, pp. 129–133.Seattle, WA (2001).
[4] Jha. P. Wang Notation Tool: "Layout Independent Representation of Tables,"ICPR 08, submitted.
[5] D. Lopresti, D.W. Embley, M. Hurst, and G. Nagy, "Table Processing Paradigms: A Research Survey," International Journal of Document Analysis and Recognition, vol 8, no. 2-3, pp. 66-86, Springer, June 2006.

Preprint submitted to 19th International Conference on Pattern Recognition.
Received April 5, 2008.