Chapter 8
An Extended Relational Model & SQL for Fuzzy Multidatabases

Awadhesh Kumar Sharma
M.M.M. Engg College, India

A. Goswami
IIT Kharagpur, India

D.K. Gupta
IIT Kharagpur, India

ABSTRACT

This chapter investigates the problems in integration of fuzzy relational databases and extends the relational data model to support fuzzy multidatabases of type-2 that contain integrated fuzzy relational databases. The extended model is given the name fuzzy tuple source (FTS) relational data model which is provided with a set of FTS relational operations to manipulate the global relations, called FTS relations, from such fuzzy multidatabases. The chapter proposes and implements a full set of FTS relational algebraic operations capable of manipulating an extensive set of fuzzy relational multidatabases of type-2 that include fuzzy data values in their instances. To facilitate formulation of global fuzzy query over FTS relations in such fuzzy multidatabases, an appropriate extension to SQL can be done so as to get fuzzy tuple source structured query language (FTS-SQL).

Many real world problems involve imprecise and ambiguous information rather than crisp information. Recent trends in the database paradigm are to incorporate fuzzy sets to tackle imprecise and ambiguous information of real world problems. Fuzzy query processing in multidatabases have been extensively studied, however, the same has rarely been addressed for fuzzy multidatabases. This chapter attempts to extend the SQL to formulate a global fuzzy query on a fuzzy multidatabase under FTS relational model discussed earlier. The chapter provides architecture for distributed fuzzy query processing with a strategy for fuzzy query decomposition and optimization. Proofs of consistent global fuzzy operations and some of algebraic properties of FTS Relational Model are also supplemented.

DOI: 10.4018/978-1-60960-475-2.ch008
INTRODUCTION

Databases hold data that represent properties of real-world objects. Ideally, a set of real-world objects can be described by the constructs of a single data model and stored in one and only one database. Nevertheless, in reality, one can usually find two or more databases storing information about the same real-world objects. There are several reasons that result in the overlapping representations. These include:

- Different roles played by the same real-world objects in different applications. For example, a company can be the customer as well as the supplier for a firm. Hence, the company’s information can be found in both the customers’ database and supplier’s database.
- For performance reasons, a piece of information may be fully or partially duplicated and stored in databases at different geographical locations. For example, the customers’ information may be stored in both the branches and headquarter.
- Different ownership of information can also lead to information stored in different databases. For example, the information of a raw material item may be stored in different production databases because each production line wants to own a copy of the information and to exercise control over the information.

When two or more databases represent overlapping sets of real world objects, there is a strong need to integrate these databases in order to support applications of cross-functional information systems. It is therefore important to examine strategies for database integration. An important aspect of database integration is the definition of a global schema that captures the description of the combined (or integrated) database. Here, we define schema integration to be the process of merging schemas of databases, and instance integration to be the process of integrating the database instances. Schema integration is a problem well studied by database researchers (Batini, Lenzerini, and Navade, 1986; Hayne and Ram, 1990; Kaul, Drosten, and Neuhold, 1990; Larson, Navade and Elmasari, 1989; Spaccapietra, Parent and Dupont, 1992). The solution approaches identify the correspondences between schema constructs (e.g. entity types, attributes, etc.) from different databases and resolve their differences. The end result is a global schema which describes the integrated database. In contrast, instance integration focuses on merging the actual values found in instances from different databases. There are two major problems in instance integration:

a. entity identification; and
b. attribute value conflict resolution

The entity identification problem involves matching data instances that represent the same real-world objects. The attribute value conflict resolution problem involves merging the values of matching data instances. These two problems have been studied in (Chatterjee and Segev, 1991; Lim, Srivastava, Prabhakar and Richardson, 1993; Wang and Madnick, 1989) and (DeMichiel 1989; Lim, Srivastava, Prabhakar and Richardson, 1993; Lim, Srivastava and Shekhar, 1994; Tasi and Chen, 1993) respectively. It is not possible to have attribute value conflicts resolved without entity identification because attribute value conflict resolution can only be done for matching data instances. In defining the integrated database, one has to choose a global data model so that the global schema can be described by the constructs provided by the data model. The queries that can be formulated against the integrated database also depend on
A Measurement Ontology Generalizable for Emerging Domain Applications on the Semantic Web


www.igi-global.com/article/measurement-ontology-generalizable-emerging-domain/3365?camid=4v1a