Chapter XIV
How to Achieve Fuzzy Relational Databases Managing Fuzzy Data and Metadata

Mohamed Ali Ben Hassine
Tunis El Manar University, Tunisia

Amel Grissa Touzi
Tunis El Manar University, Tunisia

José Galindo
University of Málaga, Spain

Habib Ounelli
Tunis El Manar University, Tunisia

ABSTRACT

Fuzzy relational databases have been introduced to deal with uncertain or incomplete information demonstrating the efficiency of processing fuzzy queries. For these reasons, many organizations aim to integrate flexible querying to handle imprecise data or to use fuzzy data mining tools, minimizing the transformation costs. The best solution is to offer a smooth migration towards this technology. This chapter presents a migration approach from relational databases towards fuzzy relational databases. This migration is divided into three strategies. The first one, named “partial migration,” is useful basically to include fuzzy queries in classic databases without changing existing data. It needs some definitions (fuzzy metaknowledge) in order to treat fuzzy queries written in FSQL language (Fuzzy SQL). The second one, named “total migration,” offers in addition to the flexible querying, a real fuzzy database, with the possibility to store imprecise data. This strategy requires a modification of schemas, data, and eventually programs. The third strategy is a mixture of the previous strategies, generally as a temporary step, easier and faster than the total migration.

INTRODUCTION

New enterprise information systems are requested to be flexible and efficient in order to cope with rapidly changing business environments and advancement of services. An information system that develops its structure and functionality in a continuous, self-organized, adaptive, and interactive way can use many sources of incoming information and can perform intelligent tasks such as language
learning, reasoning with uncertainty, decision making, and more. According to Bellman and Zadeh (1970), “much of the decision making in the real world takes place in an environment in which the goals, the constraints, and the consequences of possible actions are not known precisely.” Management often makes decisions based on incomplete, vague, or uncertain information. In our context, the data which are processed by the application system and accumulated over the lifetime of the system may be inconsistent and may not express the reality. In fact, one of the features of human reasoning is that it may use imprecise or incomplete information and in the real world, there exists a lot of this kind of fuzzy information. Hence, we can assert that in our everyday life we use several linguistic labels to express abstract concepts such as young, old, cold, hot, cheap, and so forth. Therefore, human–computer interfaces should be able to understand fuzzy information, which is very usual in many human applications. However, the majority of existing information systems deal with crisp data through crisp database systems (Elmasri & Navathe, 2006; Silberschatz, Korth, & Sudarshan, 2006). In this scenario, fuzzy techniques have proven to be successful principles for modeling such imprecise data and also for effective data retrieval. Accordingly, fuzzy databases (FDBs) have been introduced to deal with uncertain or incomplete information in many applications demonstrating the efficiency of processing fuzzy queries even in classical or regular databases. Besides, FDBs allow storing fuzzy values, and of course, they should allow fuzzy queries using fuzzy or nonfuzzy data (Bosc, 1999; De Caluwe & De Tré, 2007; Galindo, Urrutia, & Piattini, 2006; Petry, 1996).

Facing this situation, many organizations aim to integrate flexible querying to handle imprecise data or to use fuzzy data mining tools, minimizing the transformation costs. A solution of the existing (old) systems is the migration, that is, moving the applications and the database to a new platform and technologies. Migration of old systems, or legacy systems, may be an expensive and complex process. It allows legacy systems to be moved to new environments with the new business requirements, while retaining functionality and data of the original legacy systems. In this context, the migration towards FDBs, which constitutes a step to introduce imprecise data in an information system, does not only constitute the adoption of a new technology, but also, and especially, the adoption of a new paradigm. Consequently, it constitutes a new culture of development of information systems, and this book is evidence of the current interest and the promising future of this paradigm and its multiple fields.

However, with important amounts invested in the development of relational systems, in the enrollment and the formation of “traditional” programmers, and so forth, enterprises appear reticent to invest important sums in the mastery of a new fuzzy paradigm. The best solution is to offer a smooth migration toward this technology, allowing them to keep the existing data, schemas, and applications, while integrating the different fuzzy concepts to benefit of the fuzzy information processing. It will lower the costs of the transformations and will encourage the enterprises to adapt the concept of fuzzy relational databases (FRDBs). Moreover, although the migration of the information systems constitutes a very important research domain, there is a limited number of migration methods between two specific systems. We mention some examples (e.g., Behm, Geppert, & Dittrich, 1997; Hennard, Hick, Thiran, & Hainaut, 2002; Menhoudj & Ou-Halima, 1996). To our knowledge, the migration of relational databases (RDB) towards FRDB is not even studied.

FDBs allow storing fuzzy values and, besides, they allow making fuzzy queries using fuzzy or nonfuzzy data. It should be noted that classic querying is qualified by “Boolean querying,” although some systems use a trivalued logic with the three values true, false, and null, where null indicates that the condition result is unknown because some data is unknown. The user formulates a query usually with a condition, for example, in SQL, which returns a list of rows, when the condition is true. This querying system constitutes a hindrance for