A multidatabase system allows applications and users to access heterogeneous information through a common interface. Information is shared and exchanged in a transparent way within a federation of autonomous database systems. In a multi-temporal database, the connected systems may be standard (snapshot) databases, temporal databases (either valid-time, transaction-time or bitemporal) or even single multitemporal systems in which data with different temporal formats may coexist.

In this paper we consider semantic interoperability problems arising from the interaction of temporally heterogeneous data in a multi-temporal environment. In particular, we focus on the evaluation of multitemporal algebraic expressions for the support of an SQL-like query interface. The basic requirement of a multitemporal algebra is that all the operations involved must be carefully designed in order to retrieve the largest amount of certain stored information and avoid the creation of spurious information during the interaction of heterogeneous data. The main difficulties underlying the definition of such operations are presented in the paper, together with solutions to overcome them. They include conversion of data from one temporal format into another, definition of lossless operations on temporally incomplete information, and semantic optimization of multitemporal algebraic expressions.

The interaction of different database systems (Elmagarmid & Pu, 1990; Sheth, 1991) is one of the requirements of today’s advanced applications. The purpose of dealing with semantic heterogeneity is to overcome the problems connected with the management of possibly related data in a federation of systems. A multidatabase system allows applications and users to access heterogeneous information through a common interface in a transparent way. In this work, we only consider relational databases; this assumption is not overly restrictive since different database systems often provide a relational interface towards the external world (Litwin & Abdellativ, 1986). Hence, the user/application interface provided by a multidatabase system can be based on the standard SQL relational data definition and manipulation language (SQL–92, 1992).

Another important issue recognized by database research in recent years is the representation of time and the capability of managing versioned data (Kline, 1993; Soo, 1991; Tansel et al., 1993). This requirement arises from many application areas, such as CAD/CAM/CIM, management of legal and medical records, financial information systems, and from the re-engineering of legacy applications. Two orthogonal time dimensions are usually considered in the current literature (Jensen et al. 1993; Tansel et al.): transaction time, which indicates when an event is recorded in a database, and valid time, which represents when an event occurs, occurred or is expected to occur in the real world. According to this taxonomy, snapshot databases without time support, transaction- or valid-time databases supporting one time dimension (monotemporal) or bitemporal databases supporting both time dimensions can be defined (Jensen et al.).

The problem of temporal semantic interoperability dealt with in this paper consists of the sharing and exchanging of information between relational databases of different temporal types or even between relations of different temporal types within the same database system. Other kinds of temporal interoperability (e.g. between different time representations (Jajodia & Wang, 1993)) are not taken into account for
simplicity. However, different scenarios can be imagined in this context, as sketched in the following.

In a temporal multidatabase, different database sites or autonomous systems can hold data based on different temporal models. For instance, most of the existing databases are snapshot, but should be made operative in a distributed temporal environment, as required in many application areas.

In a multitemporal database, a single system may allow the coexistence of relations with different temporal structures (Snodgrass, 1987; Snodgrass et al., 1995). For instance, selective versioning techniques can be used to partition attributes of a relation into groups with different update frequencies. Thus, storing them separately reduces the amount of storage space and can improve the overall database efficiency. Different groups can divide constant from time-varying attributes. Hence, owing to data semantics or application aims, one can design some relations as snapshot, some as mono- and some as bi-temporal. It should be noticed that without selective versioning, constant attribute values would be duplicated in all the versions of a tuple with time-varying attributes. Therefore, selective versioning can be considered as a further normalization step which can be applied to relations during the logical design of a temporal database (e.g., enforcement of Time Normal Form (Navathe & Ahmed, 1989)). However, the reconstruction of unnormalized data from the resulting relations via join operations requires a sort of internal interoperability. It can also be noticed that the TSQL2 query language (Snodgrass et al., 1994; Snodgrass et al., 1995), designed as the standard temporal SQL extension, actually allows the definition of relations with different temporal formats within the same database although the semantics of “multitemporal” queries has not been formally specified.

In the most general case, we can think of a multi-temporal database (see Fig. 1), in which a federation of independent temporal or multitemporal systems is connected to interoperate and provide the functionality of a single multitemporal multidatabase.

The common task of temporal interoperability problems is to provide correct interaction between relations containing information of a different temporal nature. To this purpose, an algebra working on relations of different temporal types must be defined, on which true multitemporal query languages can be based. The main problem connected with the definition of a multitemporal algebra is the transformation of data based on a given temporal format into another. Once all the relations involved in an algebraic expression, corresponding to a given query, are translated into a common temporal format, the expression can be evaluated by means of operators defined to work on data with that format. To this purpose, temporal conversion functions have been introduced in (De Castro, Grandi & Scalas, 1994) and will be briefly surveyed in Sec. 2.

Since the bitemporal model is the most general representation of data evolving both in the real world and in the database, and the monotemporal and snapshot models can be seen as an approximate representation of a bitemporal reality, the common format into which data are to be translated before executing algebraic operations is the bitemporal one. In this way, the largest amount of information contained in the original relations is guaranteed to be preserved. If the result is required in a temporal format differing from the bitemporal one, a final conversion step must be executed.

Additional issues arising from the presence of temporal

![Figure 1: Internal and external interoperability in a multi-temporal database](image-url)
Related Content

Document SQL (DSQL): A Conservative Extension to SQL as an Ad-hoc Querying Frontend for XQuery
www.igi-global.com/chapter/document-sql-dsql/52961?camid=4v1a

Towards User-Oriented Enterprise Modeling for Interoperability
www.igi-global.com/chapter/towards-user-oriented-enterprise-modeling/4371?camid=4v1a

Integrating Web Data and Geographic Knowledge into Spatial Databases
www.igi-global.com/chapter/integrating-web-data-geographic-knowledge/29658?camid=4v1a

INDUSTRY AND PRACTICE: The Database Designer is a Person Too!
www.igi-global.com/article/industry-practice-database-designer-person/51137?camid=4v1a