Providing Multidatabase Access – an Association Approach

Paolo Missier, Marek Rusinkiewicz*
Department of Computer Science
University of Houston
Houston, TX 77204-3475

Avi Silberschatz†
Department of Computer Science
The University of Texas at Austin
Austin, TX 78712

April 10, 1995

Abstract

One of the major tasks in the design of a multidatabase system (MDBS) is the definition and maintenance of the global schema. Traditionally, this is accomplished by requiring the local databases participating in the MDBS to provide "export schemas" that are merged into a global schema. Resolution of schema and data incompatibilities, and mapping between local and global schemas are, in general, very difficult tasks that must be performed at the multidatabase level. We believe that a solution to this formidable problem may lie in the shifting of responsibility for these tasks to the local level. We propose a model in which the MDBS administrator defines the global schema as a view that is to be maintained by each of the participating databases. The MDBS layer supports submission and processing of (global) queries expressed over a union of such views. Each participating database must provide a view of its database that conforms to the global specification and must promise to respond to queries formulated over this view. We discuss the architecture of such systems and the problems involved in the processing of global queries.

*The research of Marek Rusinkiewicz was supported in part by the Texas Advanced Research Program under Grant No. ARP-003652-008, and grants from the MCC and Bellcore corporations.

†The research of Avi Silberschatz was supported in part by the Texas Advanced Technology Program under Grant No. ATP-024, the National Science Foundation under Grant Nos. IRI-9003341 and IRI-9106450, and grants from the IBM and Hewlett-Packard corporations.

1 Introduction

A multidatabase system (MDBS) is a collection of autonomous local database systems (DBMSs) that are logically integrated to provide access to data located at multiple sites. Development of such systems is made feasible by the standardization of network interfaces providing interconnectivity among heterogeneous machines.

One major obstacle in building an MDBS is the problem of how to define a global schema. Current proposals for dealing with this problem are based on an architectural model with the following assumptions:

- Each local DBMS wishing to join the MDBS must provide an *export schema* – a view of those parts of its database that it wants to make available to the MDBS. If this export schema is provided using a canonical data model, no model translation is needed on the multidatabase level. However, many discrepancies between the export schemas may exist [SK92]. For example, equivalent attributes may have different data types, or may be subject to different constraints. Also, similar entities may be described at different abstraction levels, may assume different default values, etc.
- The export schemas of the various participating local DBMSs are merged into a global schema, which allows the multidatabase users to formulate queries ranging over the entire MDBS.

The process of merging can be performed either directly by the multidatabase users [L+90] or by the multidatabase administrator (MDBA) [HM85]. In the first case, a multidatabase manipulation language (such as MSQL) provides the user with facilities to directly refer to objects in multiple databases and to resolve schema and data incompatibilities. In the second case, an "integrated schema" is built by the MDBA, who establishes the mapping between the global object (used to formulate global queries) and local objects (made visible in the export schemas).

In both cases, the MDBS users face a formidable (or sometimes impossible) task of resolving interdatabase incompatibilities (either on-the-fly, by the MSQL users, or off-line by the MDBA). This requires the users to be aware of all details of a very heterogeneous environment, which is subject to change, as export schemas are being added, dropped and modified. As a result, no successful heterogeneous MDBS exists that spans more than several databases and is "open" (permits dynamic alteration of membership status).

What is a solution to this problem? If it is too difficult to resolve matters at the global level, the answer may lie in simply shifting the responsibility for providing the mappings between local schemas and global views from the global level to the local level. Hence, the model that we propose

in this paper is based on the assumption that the database systems wishing to collaborate, form an *association* with an appointed administrator, whose primary responsibility is to provide a view that is to be maintained by each local participating DBMS. This view is well defined and each local DBMS must allow submission and processing of queries expressed over such views. We will argue that this model fits the multidatabase paradigm better, and opens the way to a practical solution to the multiple access problem.

The remainder of the paper is organized as follows. In section 2 we describe the data and software architecture of systems based on the approach outlined above. In section 3 we discuss how the meta-data characterizing the informational content of each database can be used to decide whether a global query is pertinent to the local system. In section 4 we present a detailed example of schema definition for a university type environment. In section 5 we describe how query processing can be handled in our architecture, using our university running example. Partial answers are discussed in section 6. A summary of the proposed architecture and its advantages are presented in section 7.

2 The Architecture of a Multidatabase Association

As stated in the introduction, we require that the association administrator define a global view, and that this view be maintained by each of the participating local DBMSs. Thus, there is a contractual agreement between the local DBMSs and the association that this view is to be maintained at all times.

A local system wishing to join the MDBS must provide a local view of its database that conforms to the global view, and must guarantee to respond to queries expressed in some standard form over this view. Each local system is responsible for maintaining its local view, and should react to local database changes in order to restore consistency with the accepted global view.

Furthermore, global schemas should be restricted to specific domains. In the examples proposed in this paper, we consider local databases that hold information about Computer Science Departments, their faculty members, relevant research topics, and the set of courses offered. The intended users of the MDBS would be potential Computer Science students as well as researchers. The purpose of this domain restriction is to make it easier for the global query processor to evaluate the correctness of a query and the soundness of the corresponding answers, based on domain-specific knowledge.

The proposed approach offers several advantages. Instead of the MDBA's fighting the losing battle of maintaining all the mappings from export to global schemas (or requiring the end-user to

do it on-the-fly), the association administrator dictates the conditions that the local DBMSs must satisfy if they wish to participate in the MDBS. Furthermore, each local system can join those associations in which it is interested, being responsible for meeting the conditions dictated by the global authority. The cost of ensuring limited conformance should be evaluated by each potential member, against the benefit of sharing information with the other members of the association.

Under this approach, the multidatabase system is able to maintain itself in the presence of new clients' joining or dropping out. A centralized development of view definition, communication and data transfer software becomes possible.

Several phases can be identified in the data manipulation protocol based on the above association architecture. First, a set of domain-specific (i.e., contextual) external schemas are defined, possibly in a centralized way. These schemas effectively constitute views from the point of view of the application designer (user). Second, each local system administrator is responsible for the development of the *mappings* from the local schemas to any of the selected export schemas [BLN86]. Finally, queries can be issued against any of the views defined by the export schemas, using a SQL-based relational language.

In this scenario, the MDBS query processor is responsible for deciding to which local system a given query is pertinent, and for issuing a set of elementary queries, accordingly.

Several types of domain-specific information could be used by the query processor to carry out this task, and recognize which systems should actually be involved in the retrieval process. Let us consider for instance a query about courses and professors in the various Computer Science departments of universities located in Texas. We would expect this query to be addressed only to those databases that are known to hold specific information about Texas institutions (e.g., NYU database should probably not be consulted). For this to be possible, the member DBMSs are requested to provide some kind of self-description – a *profile*, as a part of the agreement protocol. This information, an extension of the concept of self-describing databases introduced in [MR87], can be used to decide which queries can be answered by the system.

Figure 1 depicts the scenario we have outlined. The query processor accepts queries formulated with respect to one of the global schemas. It decides which local systems should be involved, by consulting the available profiles, but it does not carry out any translation. The burden of defining, maintaining, and applying the appropriate mappings to the local schemas is shifted instead from the MDBA (or the end-user) to the local administrators. Those mappings must incorporate strategies for data conflict resolution [SK92, MR93] during query evaluation.

Several points emerge from this proposed architecture. First, each system is responsible for the definition of a sound mapping of the global view into the local schema. If the schemas are

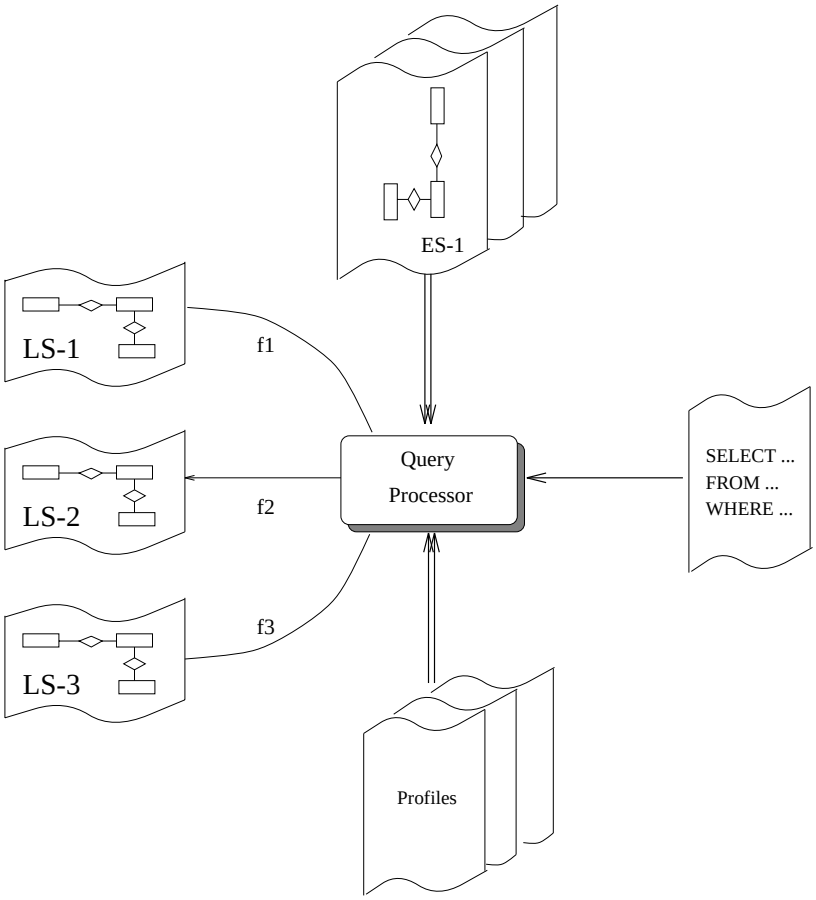

Figure 1: Local schemas, global views and multiple query processing

incompatible (no mapping is possible), then the local system is considered not eligible to join the association.

Second, the mappings should take care of both structural and semantic mismatches between the schemas [SK92]. The success of the resolution process depends on the local availability of a set of "tools" (i.e., external applications, filters, data conversion programs, etc.). These tools should support the implementation of a set of translating processors [SL90]. Depending on the nature of the local mappings, the association may allow limited updates in some cases, but the normal mode of operation assumes that the global queries are read-only and that the updates are carried out locally.

Finally, this architecture is based on the assumption that the MDBS users would mainly retrieve *unions* of relations coming from the local sites. However, overlaps in domain values among the schemas are possible. In this case, interdatabase joins – or some other form of data

fusion/aggregation, based on value matching – can in principle be performed. We believe that the specific issue of how domain disjointness can be detected by the global query processor, and exploited to setup interdatabase connections automatically, goes beyond the scope of this paper.

3 Domain and Control Data

In the proposed model, users formulate their queries with respect to any of a limited number of global schemas defined for the whole association. The integration process is transparent to the end user. For purpose of illustration, we use in this paper the relational data model and SQL. We note, however, that the architecture discussed here is independent of the data model.

The global relational schema and the collection of local profile information are accessible at the MDBS level. We will assume that the local profile information, which is actually meta-data at the global level, is also stored in relational form. A global query is validated against the global schema by the global query processor, and meta-data is consulted to decide whether a query is pertinent to a given local database. The meta-data can be represented simply as a set of control attributes that are not part of the relational global schema proper, but may appear in selection predicates. A selection on any of these attributes would allow the query processor to limit the scope of the query.

In our running university example, profile information about the local department databases may include the following control attributes:

- Department location;
- Number of professors;
- Number of enrolled students;
- Number of graduate students (if applicable);
- Annual in-state tuition and fee estimate;
- Annual out-of-state tuition and fee estimate;
- Does the Department offer a M.Sc. degree program;
- Does the Department offer a Ph.D. degree program;

Depending on the nature of the local schemas, some of these attributes can be obtained by aggregation from the local data; for example, the total number of instances for a given entity. Some others, like the Department location, should be supplied explicitly as a part of the agreement between the system and the association.

The profile of each local database may be represented by attributes in the meta-schema at the global level. Since these attributes may appear in the query, the processor must extract and

analyze those query fragments in which they are mentioned, and eventually define the query scope by direct retrieval of the corresponding profile values. "Guard predicates" can be defined on profile attributes of each database and used to restrict the potential query targets. For instance, for all local systems which hold information about departments in Texas, we may have a guard condition defined by the predicate:

Department location = "Texas"

which will be evaluated whenever LOCATION is mentioned in a user's query. Thus, a potential foreign student searching for information about graduate programs in USA may limit the search by imposing a condition in the form of selection predicates on LOCATION ("location = Texas" or "location = Minnesota").

This scenario suggests that guards may be represented in a uniform way in the query language itself (i.e., with no need for extensions), by conditions expressed using standard relational operators. At the same time, it seems that guard evaluation gives a complete operational definition of the scope of the query. While this may be sufficient in general, the examples presented below will illustrate the need for an extension of this simple notion of guard condition.

4 Example of Schema Definitions

In this section, as well as the next two sections, we describe the process of query evaluation in our architecture. We illustrate our ideas with a running example of a university environment. The association will consist of two local database systems A and B. We will first describe the global schema that is defined by the association administrator. Following this, we show how the two local schemas can be defined and mapped into the global schema.

At the local level, each system is responsible for establishing the necessary correspondences between the local and the global schema, by defining the mappings both at the relation and the attribute levels. The correspondences are defined as part of the agreement between the local members and the association. Traditionally, they represent the result of a pre-integration phase, carried out on each schema separately (one at a time). We sketch these correspondences, and the necessary auxiliary mappings. We show how the notion of meta-data can be refined to include explicitly the established correspondences. Since we cannot expect to have a perfect mapping to each local schema, we will also define a relaxed notion of "partial correct answers".

4.1 The Global Schema

In our example, the global schema defined by the association administrator, covers the following application domain. Faculty members are assigned to departments (more generally, they belong to "academic areas"), possibly to more than one, and may have a set of research interest topics associated with them. They teach courses, which are organized into sections and are also related to a set of topics, taken from the same relation of research interests.

The global schema consistes of a number of different relations (tables), as shown below. The primary key attributes are underlined in each relation.

FACULTY(ssn, rank, fname, lname, e_mail, b_date, is_chairman, is_advisor)
TOPICS(Tcode, descr)
DEPTS(Dname, Dph, addr)
COURSES(code, Cname, Cdesc, level)
SECTIONS(yr, sem, fssn, Ccode)
FAC_DEPT(Fssn, Dname)
RES_INT(Fssn, Tcode)
COU_TOP(Ccode, Tcode)

Figure 2 shows the database graph for the global schema. In the graph, edges represent the referential integrity constraints defined together with the schema.

4.2 The Local Schemas

We now consider the two local schemas of two database systems that wish to participate in the association. The schema of the local database A captures the relations between FACULTY members (teachers), the courses they teach, and the set of departments (areas) they belong to. As can be noticed, however, there is no entity to describe research interests.

TEACHER(id, rank, name, e_mail, Dcode)
COURSES(code, Tid, name, hours, level, yr, sem)
DEPT(Dcode, addr, name)

The database graph for the schema of system A is shown in Figure 3. Correspondences at the relation level can be drawn by comparing this with the global graph (Figure 2).

The local TEACHER relation corresponds to the global FACULTY relation. Similarly, local DEPT corresponds to global DEPT, and local COURSES to global COURSES. However, these correspondences

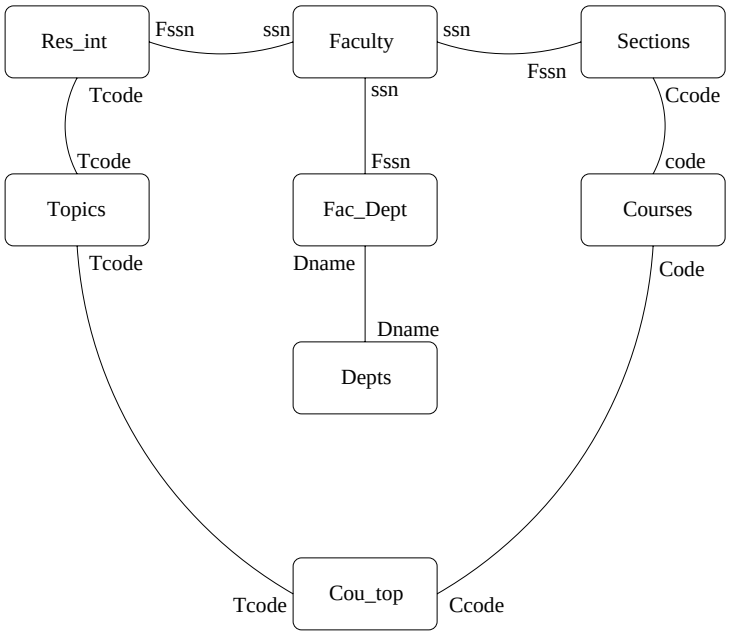

Figure 2: Database graph for the global schema

do not carry over entirely at the attribute level. Since locally there is no table to hold topics, the COURSES.NAME attribute may be used to indicate the (single) topic of a course. This corresponds to the assumption that the FACULTY and TOPICS global tables are linked by a path through COURSES in the global graph (not through RES_INT). Attribute mappings are discussed further along with the example queries.

The schema of local database B holds information about all department members (faculty, researchers, assistants, visiting professors), their topics of interest, and courses to which they are assigned. As can be noticed, there is no provision for department information.

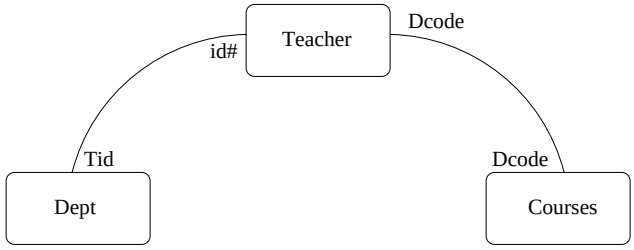

Figure 3: Database graph for the first local schema

DEPT_MEMBER(ssn, e_mail, fname, lname, Ophone)
WORKS_ON(Rssn, Tcode)
TOPICS(Tcode, Tdescr)
TEACHES(Rssn, Ccode)
COURSES(Ccode, yr, sem, level, title)
COURSE_ON(Ccode, Tcode)

The database graph for the schema of B is shown in Figure 4. Again, correspondences at the relation level can be drawn by comparing this with the global graph (Figure 2).

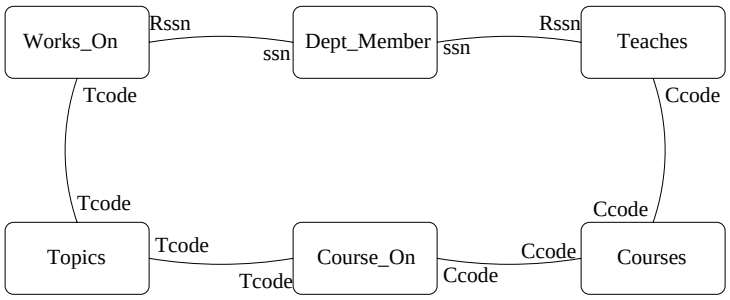

Figure 4: Database graph for the second local schema

5 Example of Query Processing

In this section we illustrate the query evaluation process by means of detailed examples, using the university environment we have discussed in the previous section.

As a first example, suppose that we want to retrieve information about all faculty members in Texas (name, rank, e-mail) who were teaching graduate level database courses in 1993.

At the global level, the query, referred to as Q1, may be defined as follows:

select fname, lname, e-mail, rank
from FACULTY, SECTIONS, COURSES, COU_TOP, TOPICS
where FACULTY.ssn = SECTIONS.fssn
and SECTIONS.Ccode = COURSES.code
and COU_TOP.Ccode= COURSES.code
and COU_TOP.Tcode = TOPICS.Tcode
and TOPICS.descr **like** "%database%"

and SECTIONS.yr = 1993
and COURSES.level > 1
and LOCATION **like** "Texas"

The query processor first identifies control attributes. In this case, LOCATION is immediately retrieved from the global profile tables, to define the subset of relevant DBMS which satisfy the corresponding condition. Then, assuming that both our local schemas are pertinent with respect to the guard conditions, the correspondences are analyzed to determine the appropriateness of the domain information available locally, that is, the *extent* to which local systems can respond to the query.

In particular, the database system A will answer by mapping SECTIONS and COURSES simply into COURSES, so that SECTION.yr becomes COURSES.yr, and "COURSES.level > 1" would be translated into the corresponding course level: "COURSES.level **in** ("M", "P')" for "Master's or Ph.D." This mapping clearly requires domain knowledge about the semantics of entities in the global schema.

Then, the set of joins in the global query is translated into the single connection between TEACHER and COURSES. Finally, discrepancies on the number and types of attributes used to express equivalent entities (TEACHER.name is made to correspond to FACULTY.lname) are solved by inserting appropriate type conversion functions, and by padding the final projected relation with NULL attributes where appropriate, to preserve union compatibility.

The resulting local query for system A is as follows:

select NULL_{fname}, name, e-mail, rank
from TEACHER, COURSES
where TEACHER.id = COURSES.Tid
and COURSES.level = "G"
and COURSES.name **like** "%database%"
and COURSES.yr = 1993

Notice that the two joins from FACULTY to COURSES through SECTIONS are mapped into the single join from TEACHER to COURSES, and that the reference to relation TOPICS is converted into a condition on COURSES.name.

Local system B, on the other hand, does have tables both for courses and topics. Its database will answer the query by retrieving department members' names, rather than FACULTY names, and the courses they teach. However, again SECTIONS and COURSES are mapped into COURSES, and the corresponding attributes are translated accordingly. Given the relation correspondences, the

path from FACULTY to TOPICS in the global graph is translated into a path from DEPT_MEMBER to TOPICS, which goes through the TEACHES, COURSES and COURSE_ON tables. As a consequence of the discrepancy between the "FACULTY" and "department members" entities, the rank attribute in the final projection is padded with a constant like 'unknown' (or is left NULL) for each tuple.

The resulting local query for system B is the following:

select fname, lname, e-mail, 'unknown'
from DEPT_MEMBER, COURSES, COURSE_ON, TEACHES, TOPICS
where DEPT_MEMBER.ssn = TEACHES.Rssn
and TEACHES.Ccode = COURSES.Ccode
and COURSE_ON.Ccode = COURSES.Ccode
and COURSE_ON.Tcode = TOPICS.Tcode
and TOPICS.Tdescr **like** "%database%"
and COURSES.level = "G"
and COURSES.yr = 1993

Several issues are raised by the scenario depicted in the example above. For instance, since the implicit domain relation: DEPT_MEMBER $\supseteq$ FACULTY holds, we can expect that non-faculty members will belong to the answer relation as well, although we have no way, from the schema, to identify the extra tuples.

Furthermore, sometimes partial answers are yielded by the local DBMS. In the following examples, two global queries are issued separately, to retrieve data which is pertinent to only one of two databases. A problem arises when the two queries are combined: none of the two answers is complete, but each can supply only partial data.

Suppose that we want to retrieve all faculty members (name, rank, e-mail) who have a research interest in databases.

At the global level, the query, referred to as Q2, may be issued as follows:

select fname, lname, e-mail, rank
from FACULTY, RES_INT, TOPICS
where FACULTY.ssn = RES_INT.fssn
and RES_INT.Tcode = TOPICS.t_code
and TOPICS.descr **like** "%database%"

By examining the correspondences between local and global schemas, the query processor must recognize that system A cannot answer queries about "research interests". This conclusion is

derived from the lack of correspondence between local and global schemas, for the TOPICS table through the RES_INT table. Therefore, the query is not issued to local system A.

In system B, however, there exists an established correspondence between department members and their interests, given by the link through the WORKS_ON table. Following the same steps as in the first example, the corresponding elementary query can be formulated:

select fname, lname, e-mail
from DEPT_MEMBER, TOPICS, WORKS_ON
where DEPT_MEMBER.ssn = WORKS_ON.Rssn
and WORKS_ON.Tcode = TOPICS.Tcode
and TOPICS.Tdescr **like** "%database%"

As an example of a query pertinent to A but not to B, assume that we want to retrieve the information about all faculty members (name, rank, e-mail) who belong to some specific department (let it be "EE").

At the global level, the query, referred to as Q3, may be issued as follows:

select fname, lname, e-mail, rank
from FACULTY, FAC_DEPT, DEPTS
where FACULTY.ssn = FAC_DEPT.Fssn
and FAC_DEPT.Dname = DEPTS.Dname
and DEPTS.Dname **like** "%EE%"

Database B cannot provide an answer to this query, since it has no notion of departments. Hence, no selection can be made on the set of department members.

In database A, a correspondence between departments and faculty members is established through the direct link "TEACHER.Dcode = DEPT.Dcode". Again, a local query can be formulated by following the guidelines for attributes translation given above:

select NULL_{fname}, name, e-mail, rank
from TEACHER, COURSES
where TEACHER.Dcode = DEPT.Dcode
and DEPT.name **like** "%EE"

6 Partial Answers

What happens if we combine queries Q2 and Q3; that is, if the query involves both interests and departments of faculty members? Neither schema is able to answer in full; however, each of them may supply at least a partial answer. It seems appropriate to find some form of relaxation for the rules used to decide which query is pertinent to which system. These rules will be used to decide whether a local answer to a global query can still be considered correct, given that not all the conditions specified in the global query can be evaluated in the local schema.

Suppose we want to select information about faculty members, subject to conditions both on departments and on research topics. In the first schema, the condition about topics cannot be evaluated. Therefore, if the query is actually issued to that system, it can only return a relation which satisfies the condition on the departments, regardless of their research interest. A symmetrical situation is faced in the second database.

We may look at this problem from a different perspective. Instead of redefining correctness *a priori*, we may *delegate* the decision whether the partial results are acceptable, to a post-processor at the global level. In a sense, we change the question from that of correctness to one of *acceptability*.

To this end, we require that all the attributes for which the global query defines selection conditions, appear in the resulting relation, that is, are *visible* in the result. We will consider a global query *valid* if it satisfies this condition[1]. This would permit application of more flexible acceptance criteria, possibly heuristic and context-dependent.

Of course, since we expect that a schema may not be able to evaluate all conditions, we also expect the final relations not to be union compatible. A final outer union [ECR87] may be needed to produce the final result.

This, in turn, would permit application of more flexible acceptance criteria, possibly heuristic and context-dependent.

To illustrate these concepts, we present a valid query that summarizes queries Q2 and Q3 above. Suppose that we want to retrieve the information about all faculty members (name, rank, e-mail) who have a research interest in databases *and* belong to a "EE" department.

[1] Notice that a invalid query can always be expanded into a valid query by the processor by analyzing the selection conditions.

The global query, referred to as Q4, would be:

select fname, lname, e-mail, rank, depts.Dname, topics.descr
from FACULTY, RES_INT, TOPICS, FAC_DEPT, DEPTS
where FACULTY.ssn = RES_INT.fssn
and RES_INT.Tcode = TOPICS.T_code
and FACULTY.ssn = FAC_DEPT.Fssn
and FAC_DEPT.Dname = DEPTS.Dname
and DEPTS.Dname **like** "EE"
and TOPICS.descr **like** "%database%"

In this query the attributes DEPTS.Dname and TOPICS.descr are added to the projection list.

Assuming that the criteria for local query evaluation are relaxed, (or that the conditions are not passed to the local queries, but evaluated by the global processor instead) the first database will return always NULL value for TOPICS.descr, similarly the second database will return NULL value for Dname. Based on the acceptance rules the global processor may decide, either to discard the unacceptable tuples or to present them as "partial" tuples. Notice that the missing attributes in the tuples thus obtained have a "maybe" semantics [Bis83, Gra80], since we want to avoid making a closed world assumption on the domain values of the final global relation. In other words, our insufficient knowledge of some relations at the local level does not translate into *negative* knowledge.

As a last computation step, the final relation to be presented to the user can be purged of the auxiliary attributes. This approach has the advantage of making the global processor *parametric* with respect to the rules for correctness of partial answers.

As a final comment we can note that if we assume that the partial results returned from the local systems may overlap (which in our example will be true if a person can work in more that one university), the final outer union may result in an interdatabase outer join. In this case we can hope to obtain a complete result from the fusion of two partial answers. In the context of maybe-operations, the union amounts to joining partial results only on *true* attributes, thereby trying to eliminate the uncertainty in the missing attributes [DeM89].

7 Summary

We presented an approach to multidatabase access based on the concept of an association. Associations are based on the basic principle of mutual cooperation for mutual advantage, with the understanding that the systems deciding to join them, accept certain "admission conditions", which

amount to assuming an obligation to support locally a view defined by the association administrator.

We believe that the advantages of this approach clearly outweigh the potential losses. Decentralizing the responsibility for the definition of the local mapping makes the goal of achieving interoperability more realistic. The association is flexible as to membership agreement, and, even more importantly, is self-maintaining. A centralized definition of a collection of global views, far from hindering local autonomy, simply provides a "supervisor" layer which represents a guarantee for the final users.

Local systems make a responsible commitment to meet the association requirements, based on a personal estimate of the cost/benefit balance. They provide and maintain a personal profile, which help clarify their role within the association. They are not requested to process data coming from external data sources, but only to retrieve their own and deliver it in the format agreed upon in the initial commitment.

References

[Bis83] J. Biskup. A Foundation of Codd's Relational Maybe-Operations. *ACM Transactions on Database Systems*, 8(4):608–636, december 1983.

[BLN86] C. Batini, M. Lenzerini, and S.B. Navathe. A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys*, 18(4), December 1986.

[DeM89] L. DeMichiel. Performing Operations over Mismatched Domains. In *Fifth International Conference on Data Engineering*, pages 36–45. IEEE Computer Society, IEEE Computer Society Press, February 1989.

[ECR87] D. Embley, B. Czejdo, and M. Rusinkiewicz. An approach to schema integration and query formulation in federated database systems. In *Proceedings of the Third International Conference on Data Engineering*, February 1987.

[Gra80] J. Grant. Incomplete Information in a Relational Database. *Fundamenta Informaticae*, 3(3):363–378, 1980.

[HM85] D. Heimbigner and D. McLeod. A federated architecture for information management. *IEEE Data Engineering*, 3(3), July 1985.

[L+90] W. Litwin et al. MSQL: A multidatabase language. *Information Sciences*, 49(1-3):59–101, October-December 1990.

[MR87] L. Mark and N. Roussopoulos. Information interchange between self-describing databases. *IEEE Data Engineering*, 10(3), September 1987.

[MR93] P. Missier and M. Rusinkiewicz. Extending a Multidatabase Manipulation Language to Resolve Schema and Data Conflicts. Technical Report UH-CS-93-10, Dept. of Computer Science, University of Houston, November 1993.

[SK92] A. Sheth and V. Kashyap. So Far (Schematically) yet so Near (Semantically). In *IFIP TC2/WG2.6 Conference on Semantics of Interoperable Database Systems*. Elsevier Scientific Publisher B.V., Nov. 1992.

[SL90] A. Sheth and J. Larson. Federated databases: Architectures and integration. *ACM Computer Surveys*, September 1990.