A GLOBAL QUERY PROCESSING IN COMPOSITE INFORMATION SYSTEMS

Y. RICHARD WANG
STUART E. MADNICK

May 1989 WP # 3015-89-MS

Global Query Processing in Composite Information Systems

Y. Richard Wang
Stuart E. Madnick

E53-317, MIT
Sloan School of Management
Cambridge, MA 02139

(617) 253-0442
rwang@sloan

March 1989

Center for Information Systems Research
Sloan School of Management
Massachusetts Institute of Technology

Global Query Processing in Composite Information Systems

ABSTRACT Many important applications in the 1990's will require access to and integration of multiple disparate databases both within and across organizational boundaries because of the push-pull effect between the significant advances in information technologies and the ever increasing competition in the global market. This type of application system has been referred to as a Composite Information Systems (CIS). In order to facilitate the increased connectivity for CIS, many strategic, organizational, and technical connectivity issues need to be addressed. This paper examines the critical issues involved in global query processing in CIS -- a key aspect of technical connectivity. A query processing mechanism is developed to transform a CIS query into an equivalent Global Syntax Construct (GSC) table. Semantic action routines are invoked, in turn, to translate the GSC table into an Equivalent Internal Construct (EIC) table. The EIC table can be further augmented with the required information for retrieving information and formulating composite answers from multiple disparate databases. In addition, optimization is performed along the way for reducing the amount of information to be retrieved from the remote local databases.

1. INTRODUCTION

Advances in computer and communication technologies have provided significant opportunities for, and successful examples of, dramatically increased connectivity among information systems [6, 12, 20, 25]. These opportunities, in turn, have enabled new forms of intra- and inter-organizational connectivity [2, 5, 16, 24]. Meanwhile globalization, whereby the scope and presence of organizations expand beyond their traditional geographic boundaries, has propelled many organizations to capitalize on these increased connectivity opportunities. As a result of the interplay between the increased connectivity and globalization, many important applications in the 1990's will require access to and integration of multiple disparate distributed databases both within and across organizational boundaries. This type of application system has been referred to as a Composite Information System (CIS) [21, 27, 28, 29].

In a major international bank, for example, issues involved in global security trading have become increasingly critical. The bank acts as an intermediary in the handling of securities for many other financial institutions around the world. It is becoming increasingly important to know almost on a minute by minute basis what is the current status of a security transaction: Is the security in transit? Has the payment been cleared? This affects whether a given security can be sold at that moment or later days. More than two thirds of the security transactions handled by this bank occur between countries. For instance, a security being held in Germany is to be sold in Hong Kong by someone in the USA. Thus it is necessary to make movements between the separate country-based security systems. Currently much of the global security exchange processing must be handled through human operators. If an agent in the USA wishes to sell securities that are held in Germany to someone in Hong Kong, someone would actually call Germany to find out the current status, and contact Hong Kong to track its processing there. To provide some degree of "global" view, a "shadow database" is being introduced. That is, a copy of some aspects of the security status information from all of the centers is transferred to a database held in the U.S. once a day. This gives a snapshot of global status as of some moment in the past 24 hours. The bank is very concerned that this twenty four hour lag is not adequate to deal with the increasing amount of almost real time security handling requirements. Developing a CIS that will provide worldwide status information in less than five minutes as needed has become a high priority.

Many strategic, organizational, and technical connectivity issues involved in CIS have been addressed previously. This paper focuses on CIS global query processing. It provides a mechanism for a user[1] to specify a CIS global query with the "appearance" of a single global database that combines all of the relevant disparate and geographically distributed systems (e.g., global security systems). Section 2 discusses issues involved in accessing multiple disparate databases. A CIS global query processing example is presented in section 3. Section 4 presents the algorithms for translating a CIS global query into GSC and EIC which has the necessary information for further optimization and query execution. Finally, concluding remarks are made in section 5.

2. CONNECTIVITY ISSUES

Technical connectivity issues can be categorized into first- and second-order issues. The first-order issues are encountered immediately when attempting to provide access to and integration of

----

1. By "user" we mean, throughout the paper, a person who has knowledge of how to formulate queries against a database schema. This person could be a database administrator, a view administrator, an application programmer, or an end-user who has the knowledge.

multiple information resources [8, 10, 13, 18, 19], such as machines from multiple vendors (IBM, DEC, AT&T, etc.), multiple physical connections (local net, wide-area net, etc.), and different database accesses (ORACLE/SQL, IBM's SQL/DS, flat files, menu-driven systems).

Suppose that one is able to resolve the above first-order issues, the challenging information composition task still abounds with second-order issues [3, 11, 27] such as attribute naming (*company* attribute vs. *comp_name* attribute), domain and attribute value mapping ($ vs. ¥ or £), query formulation (where is the data for alumni *position*, base *salary*, *Ford* sales, *Fiat* sales,etc.), and inter-database instance identification (*IBM Corp* in one database vs. *IBM* in another database).

The capability to resolve these problems is essential in accessing autonomous information systems in concert because of the ubiquitous syntactical and semantic inconsistencies and incompatibilities inherent among systems. Without some kind of facilities to help resolve the syntactical and semantic differences, the user would have to be knowledge-rich about the multiple information systems to be accessed, and hardwire the necessary transformation in his code.

In MULTIBASE[2], for example, a view administrator (VA) is trained and authorized by a database administrator (DBA) to interpret the DBA's database. The VA is authorized by multiple DBAs; therefore, can interpret the syntactical and semantic differences among these databases. Although an auxiliary database is also constructed to help reconcile some of the differences among disparate databases, the VA needs to specify the necessary mappings in DAPLEX using his knowledge of the underlying databases. It would be very useful if this knowledge could be codified into the system. State-of-the-art database theories have started to deal with problems such as schema integration and heterogeneous database systems. However, a comprehensive mechanism for accessing autonomous information systems is still critically desired [29].

----

2. MULTIBASE, developed by the Computer Corporation of America [13], is one of the leading research prototypes for demonstrating the feasibility of heterogenous database systems.

2.1 Tasks for Connectivity

From a straight forward consideration of the ways in which disparate information systems can be accessed in concert for composite information, schema integration [3] and its output codification need to be performed. The schema integration activity is critical in the CIS environment. The reason is that in the process of schema integration, all the local databases intensions are examined and can be codified. In the MULTIBASE example, the VA is trained to understand the intensions of the multiple local databases. Therefore, there is no need for schema integration because the intensions are already integrated and stored in the VA's head. Whereas, in the CIS environment, this knowledge needs to be explicitly codified in order for the system (not a human) to process a CIS query.

The schema integration task requires the DBAs who are knowledgeable about the disparate databases to work together. The input of the task is the knowledge about the database intensions. The output of the task is the intension of the global schema and a body of information about how to reconcile the syntactic and semantic differences among the disparate databases.

Furthermore, a CIS query language needs to be developed. A mechanism is also needed to transform a CIS query into some kind of internal queries which, when executed, would obtain data from the remote local databases, resolve the inconsistencies and incompatibilities, and return the information as specified in the CIS query.

2.2 CIS Query Processor Architecture

A CIS query processor architecture, shown in Figure 1, has been proposed [29] to partition the connectivity problems. This architecture encapsulates the first-order issues into the local query processors (LQPs). Each LQP is responsible for the physical connection to a particular remote machine and the required database accesses.

An important design choice for the CIS query processor architecture is the data model to be used for the representation of the data in the LQPs, the Global Query Processor (GQP), and the Application Query Processors (AQPs)[3]. Although many semantic data models and schema integration methodologies have been proposed and discussed in depth in the literature [1, 3, 4, 8, 10, 11, 14, 17, 18, 19, 22, 23, 26], our experiences in developing CIS [29] have led us to a combination of the Entity-Relation (ER) [7] and relational models: The relational model is theoretically mature, information-structure oriented, and commercially successful. The ER diagram is better suited for representing the semantics of the underlying model. It specifies the relationship between entities explicitly[4]. We have, therefore, chosen to extend the relational model in our work, and to incorporate the useful constructs developed in the semantic data models, such as the ER diagram, where relevant. Section 3 exemplifies global query processing in CIS.

3. GLOBAL QUERY PROCESSING EXAMPLE IN CIS

An implication of the relational/ER approach is that a relational interface is required for each LQP[5]. In representing the local database, the relationship between entities is explicitly defined as link[6] entities. By requiring a relational interface for each LQP as well as for the GQP, a coherent CIS query processing mechanism based on the relational theory can be developed. The mechanism is exemplified below using a simplified scenario of a placement assistant system.

3.1 Schema Integration

Suppose that an alumni database has a relational/ER model interface as shown in Figure 2. The

----

3. Building on top of the global schema, an AQP can provide many additional functionalities, including natural language, graphics interfaces, and any productivity tools. We focus on the GQP in the rest of the paper.
4. In the relational model, the relationship between two entities is defined implicitly through common attributes.
5. The issue of data model mapping to a relational interface have been discussed extensively in the literature , e.g. Hwang and Dayal [15], thus is beyond the scope of this paper.
6. We use "link" instead of "relationship" to avoid the confusion between a relation and a relationship.

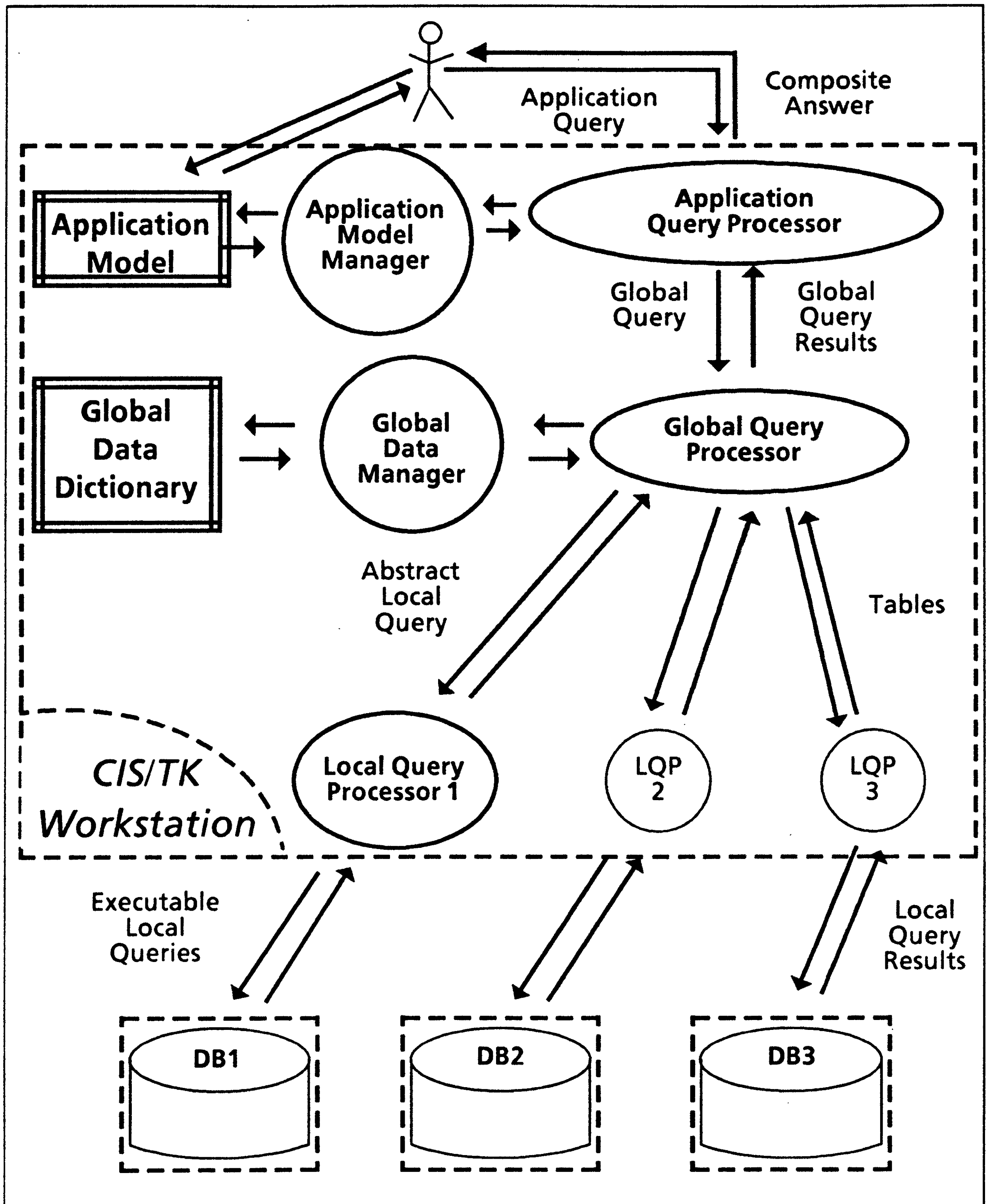
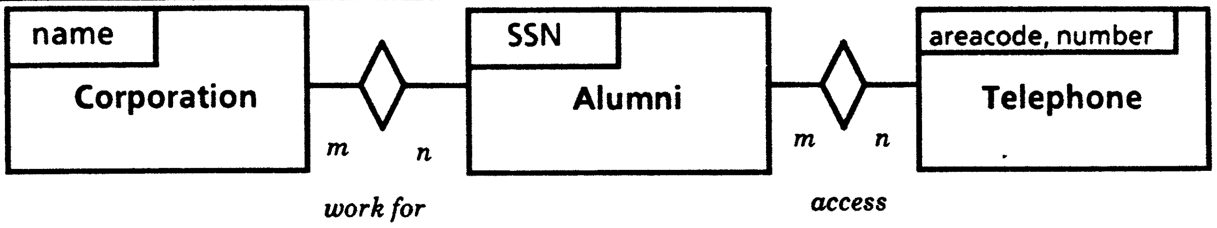


Figure 1 The CIS/TK Query Processor Architecture



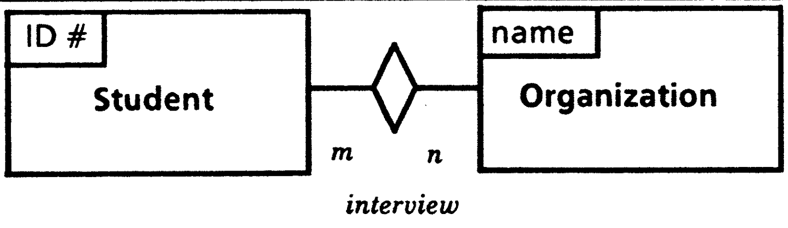
Alumni:	(<u>SSN</u>, name, major, degree)
Access:	(<u>SSN, areacode, number</u>)
Telephone:	(<u>areacode, number</u>)
Workfor:	(<u>SSN, name</u>, department, position)
Corporation:	(<u>name</u>, industry, CEO)

Figure 2 Alumni Database in Relational Model with an ER Diagram

model stipulates that each alumnus is uniquely identified through a social security number (SSN). Associated with each alumnus is a a name, a major, and a degree. An alumnus can be accessed through multiple telephone numbers. An alumnus may work for many corporations, each with a position in a department. Finally, a corporation is associated with an industry and has a CEO. It may employ multiple alumni.

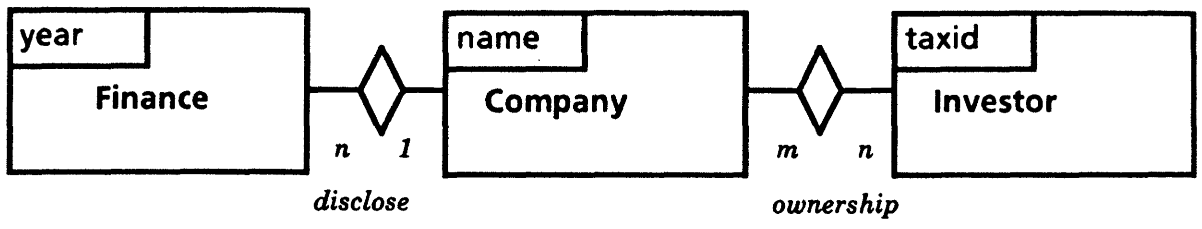
A placement office database is depicted in Figure 3. A student is uniquely identified by an ID#, and associated with a name, a GPA, and a major. A student may interview many organizations for positions at the specified job locations by interviewers at the visit-days. Finally, an organization is associated with a business, and is based in a state.

A company database is depicted in Figure 4. A company has a name, a CEO, belongs to an



Student:	(<u>ID#</u>, name, GPA, major)
Interview:	(<u>ID#</u>, <u>name</u>, position, job-location, interviewer, visit-day)
Organization:	(<u>name</u>, business, base-state)

Figure 3 Placement Office Database in Relational Model with an ER Diagram



Company:	(name, CEO, industry, headquarter)
Disclose:	(name, year)
Finance:	(name, year, asset, revenue, profit)
Ownership:	(name, taxid, percentage)
Investor:	(taxid, type)

Figure 4 Company Database in Relational Model with an ER Diagram

industry, and its headquarter is located in a city. The company discloses financial information each year on asset, revenue, and profit. It is owned by many investors, each of them can be identified through a tax identifier. There are different types of investors, e.g., institutional or private.

From the knowledge about the three databases, an integrated global schema can be constructed as shown in Figure 5. As mentioned earlier, an important task after the schema integration activity is to encode the mappings between the entities, links, and attribute values as well as other pertinent information. Ideally, all the intensions about the databases should be codified explicitly. The entity, link, entity attribute, and link attribute mappings will be shown later in Table 1, 2, 3, and 4 respectively. They are used in the query transformation process as discussed below.

3.2 Global Query Parsing

Consider the query "Find all the names and degrees of the alumni who are CEOs of corporations" for the query to the global schema in Figure 5. An SQL expression and the corresponding relational algebra for the *CIS query* are shown in Figure 6(a). The parse tree for the relational algebra is shown in Figure 6(b) where π denotes project, σ denotes select, and $\bowtie$ denotes join. The GQP applies the schema integration output encoded in Table 1 through Table 4 to transform the original global parse tree into an equivalent tree as follows: First the *join* operation at the lower left corner of Figure 6(b)

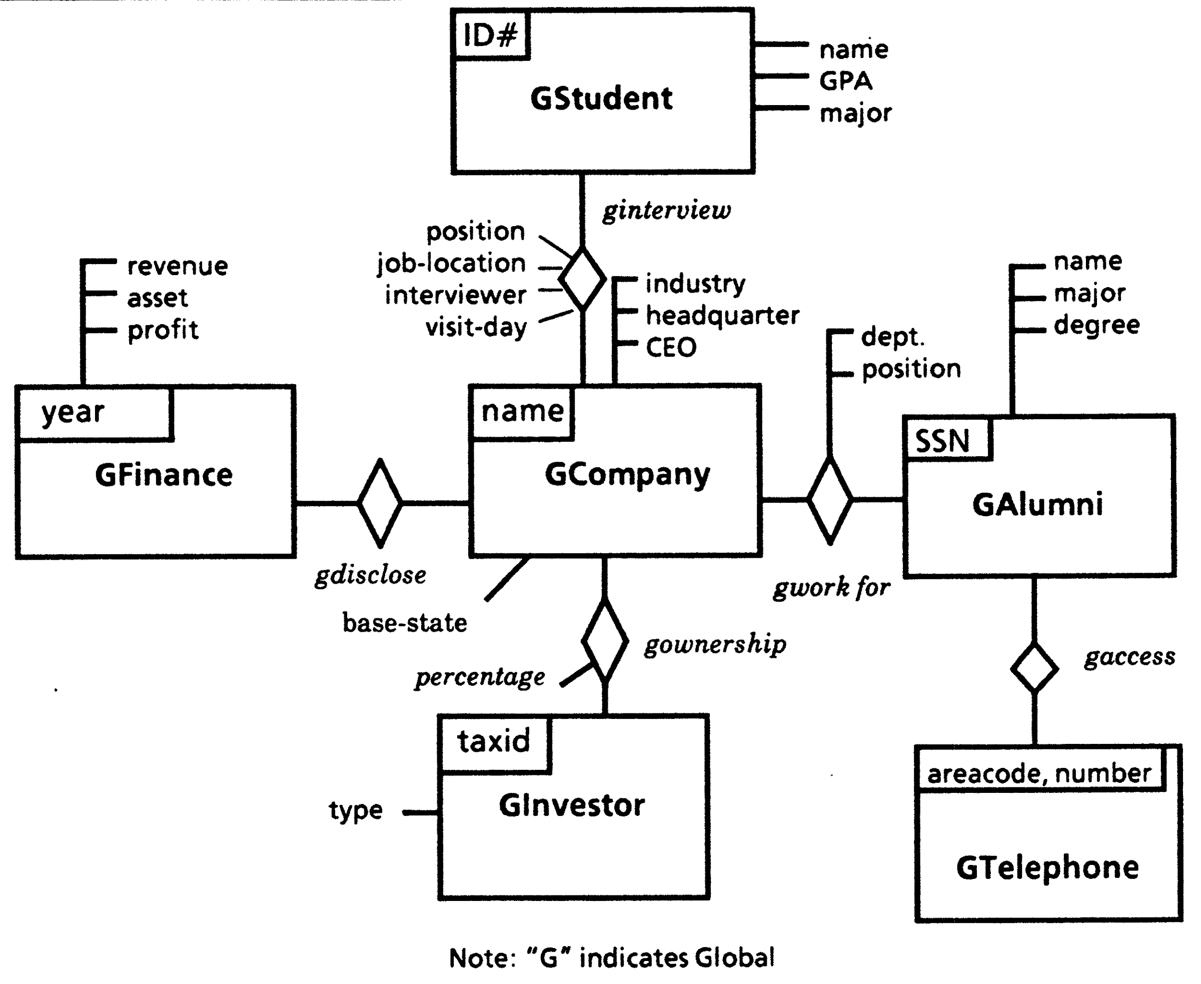


Figure 5 A Global Schema for the Alumni, Company and Placement Office Data Bases

Query: Find all the names and degrees of the alumni who have the same names as the CEOs of the companies they work for.

In SQL:

```
Select  name, degree  From   GAlumni   Where SSN =
        (Select SSN    From   Gworkfor  Where Name =
              (Select name   From   GCompany     Where CEO = GAlumni.name))
```

In Relational Algebra:

Let A denote GAlumni, W denote Gworkfor, and C denote **GCompany**,

(((A [A.SSN = W.SSN] W) [W.name = C.name] C) [C.CEO = A.name]) [A.name A,degree]

Figure 6(a) Example Qeury for the CIS Global Schema

is examined. From Table 2 the GQP knows that the *Gworkfor* link has a counterpart, i.e. *workfor*, in the alumni database. Therefore, it knows that the *join* will be done with an (alumni) entity in the alumni database, as confirmed by Table 1. As a result, GAlumni is translated into Alumni.

Furthermore, GAlumni.SSN is translated into Alumni.SSN, and *Gworkfor*.SSN is translated into *workfor*.SSN from Table 3 and Table 4 respectively.

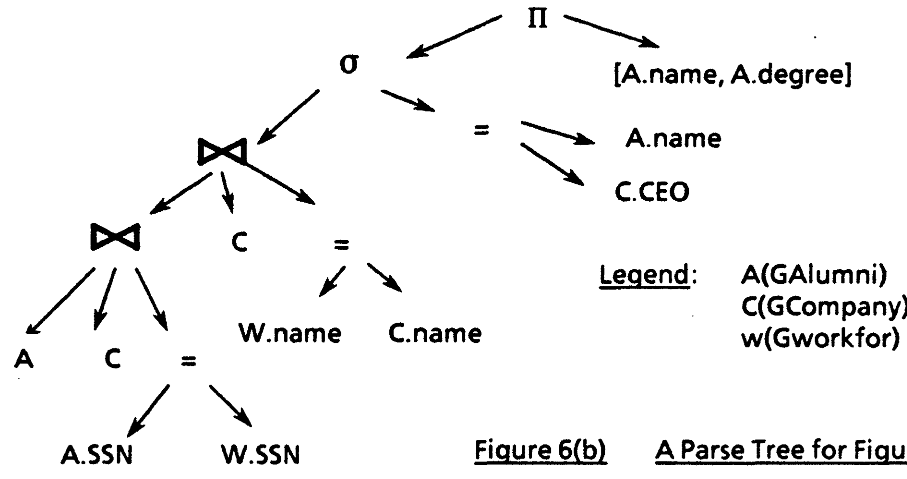


Figure 6(b) A Parse Tree for Figure 6(a)

Table 1: Entity Integration of Alumni, Comapany, and Placement Databases

Global Schema	Alumni Database	Placement Database	Company Database
GCompany	Corporation	Organization	Company
GStudent	---	Student	---
GFinance	---	---	Finance
GInvestor	---	---	Investor
GAlumni	Alumni	---	---
GTelephone	Telephone	---	---

Table 2: Link Integration of Alumni, Comapany, and Placement Databases

Global Schema	Alumni Database	Placement Database	Company Database
ginterview	---	*interview*	---
gdisclose	---	---	*disclose*
gownership	---	---	*ownership*
gworkfor	*workfor*	---	---
gaccess	*access*	---	---

Table 3: Entity Attribute Integration of Alumni, Comapany, and Placement Databases

Global Schema	Alumni Database	Placement Database	Company Database
GCompany.name	Corporation.name	Organization.name	Company. name
GCompany.Industry	Corporation.industry	Organization.business	Company.industry
GCompany,headquarter	---	---	Company.headquarter
GCompany.CEO	Corporation.CEO	---	Company.CEO
GCompany. base-state	---	Organization, base-state	---
GStudent. ID#	---	Student .ID#	---
GStudent.name	---	Student.name	---
GStudent.GPA	---	Student.GPA	---
GStudent.major	---	Student.major	---
GFinance.year	---	--	Finance.year
GFinance.asset	---	---	Finance.asset
GFinance.revenue	---	---	Finance.revenue
GFinance.profit	---	---	Finance.profit
GInvestor.taxid	--	---	Investor.taxid
GInvestor.type	---	---	Investor.type
GAlumni.SSN	Alumni.SSN	---	---
GAlumni.name	Alumni.name	---	---
GAlumni.major	Alumni.major	--	---
GAlumni.degree	Alumni.degree	---	---
GTelephone.areacode	Telephone.areacode	---	---
GTelephone.number	Telephone.number	---	---

Next the second *join* operation in Figure 6(b) is parsed. From Table 2, the GQP knows that the *Gworkfor* link has a counterpart, i.e., *workfor* in the alumni database. It uses this information to translate GCompany into "Corporation" although from Table 1 the GQP actually finds that GCompany has three counterparts in the local databases: "Corporation" in the alumni database, "Organization" in the student database, and "Company" in the company database. In addition, the GQP translates GCompany.name into Corporation.name, and *Gworkfor*.name into workfor.name

Table 4: Link Attribute Integration of Alumni, Comapany, and Placement Databases

Global Schema	Alumni Database	Placement Database	Company Database
ginterview.ID#	---	Student.ID#	---
ginterview.name	---	Organization.name	---
ginterview.position	---	*interview*. position	---
ginterview.job-location	---	*interview*. job-location	---
ginterview.interviewer	---	*interview*. interviewer	---
ginterview.visit-day	---	*interview*.visit-day	---
gdisclose.name	---	---	Company.name
gdisclose.year	---	---	Finance.year
gownership.name	---	---	Company.name
gownership.taxid	---	---	Investor.taxid
gownership.percentage	---	---	*ownership*.percentage
gworkfor.SSN	*workfor*.SSN	---	---
gworkfor.name	*workfor*.name	---	---
gworkfor.department	*workfor*.department	---	---
gworkfor.position	*workfor*.position	---	---
gaccess.SSN	Alumni.SSN	---	---
gaccess.areacode	Telephone.areacode	---	---
gaccess.number	Telephone.number	---	---

from Table 3 and Table 4 respectively, knowing that the mappings should be made to the Alumni database.

Next the GQP parses the select (σ) operation. From Table 3 and the relational scheme of the intermediate result, the GQP recognizes that the GCompany.CEO information is stored under the attribute Corporation.CEO; similarly GAlumni.name under Alumni.name. Finally, the GQP parses the project (π) operation. It recognizes Alumni.name and Alumni.degree from Table 3. The equivalent parse tree is shown in Figure 6(c). Since no further optimization can be made on the tree, the GQP sends it to the LQP responsible for the alumni database.

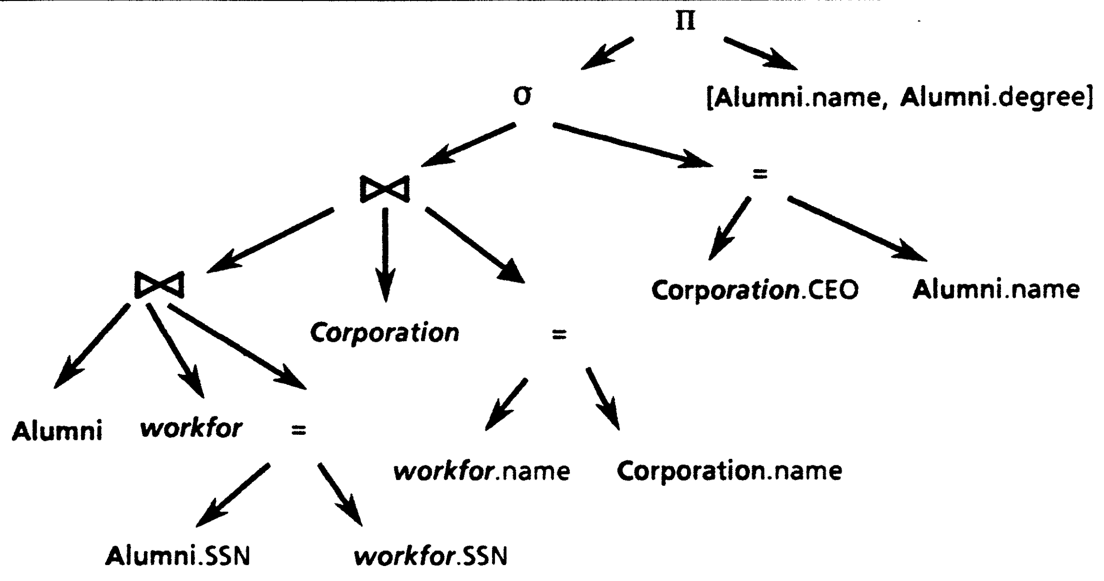


Figure 6(c) An Equivalent Parse Tree for the CIS Query in Figure 6(b)

We have exemplified the basic mapping mechanism for a CIS query. Interesting issues and approaches involved in multiple databases global query will be discussed in section 4.4. Section 4 presents the algorithms for translating a CIS global query into an Equivalent Internal Construct (EIC) table which has the necessary information for further query optimization, routing, and execution.

4 CIS GQP

A simplified BNF grammar[7] for the GQP is shown in Figure 7. It enables us to analyze the syntax of a global CIS query. For example, the CIS global query in Figure 6(a) can be analyzed, as shown in Figure 8, and decomposed into the Global Syntax Construct (GSC) tuples in Table 5.

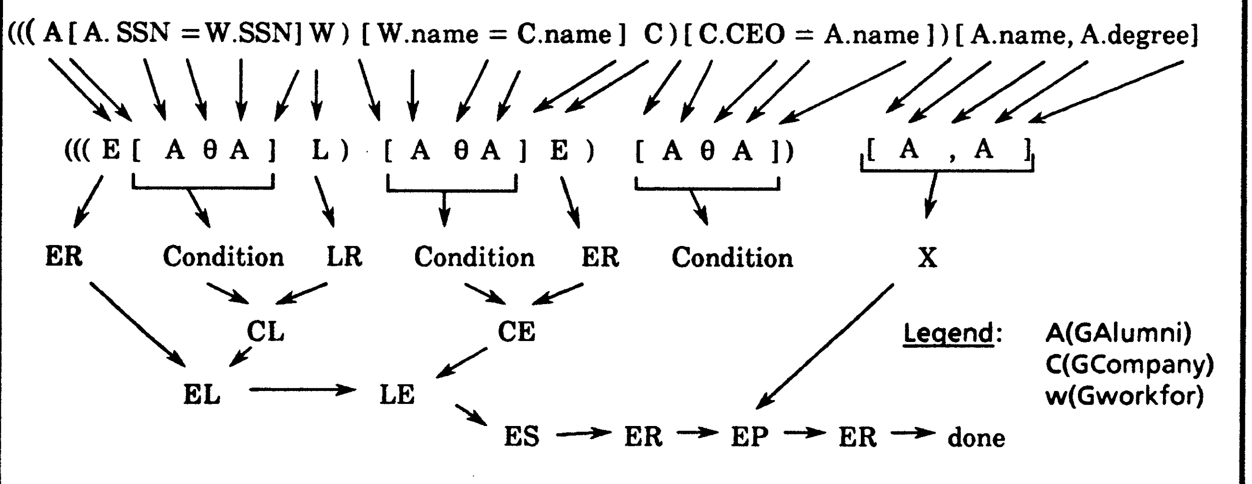
Corresponding to the GSC table are many semantic action routines that need to be executed, depending on the GSC tuples, in order to generate an Equivalent Internal Construct (EIC) table. Specifically, a GSC driver can be developed. Given a GSC table as input, the driver will invoke the

7. Only the select, project, and join operators are illustrated in the grammar. Conditions and other operators such as union, intersection, difference, divide, and Cartesian product should also be implemented.

BNF for the CIS Global Query Processor

```
Terms:      Entity | Link | Attribute | Space | ( | ) | [ | ] | , | ≤ | ≥ | ≠ | = | > | <
Entity:     an entity relation defined in the entity table.
Link:       a relationship relation defined in the link table.
Attribute:  an attribute in a relation defined in the entity.attribute or link.attribute table.
Note:       Space is ignored by the lexical analyzer, ( ) takes precedence over left to right parsing.
<θ> ::=  ≤ | ≥ | ≠ | = | > | <
<E> ::= <entity>
<L> ::= <link>
<A> ::= <attribute>
<X> ::= <A> | <X>,<A>                   /* X is a subset of a relational scheme */
<condition> ::= [<A><θ><A>]
<ER>::= <E> | <EP> | <ES> | <LE>        /* Entity Relation */
<LR>::= <L> | <LP> | <LS>               /* Link Relation */
<CL> ::= <condition> <LR>               /* Condition followed by an LR */
<CE> ::= <condition> <ER>               /* Condition followed by an ER */
<EP> ::= <ER> [<X>]                     /* Entity projection */
<LP> ::= <LR> [<X>]                     /* Link projection */
<ES> ::= <ER> <condition>               /* Entity selection (restriction) */
<LS> ::= <LR> <condition>               /* Link selection (restriction) */
<EL> ::= <ER> <CL> | <EL> <condition> | <EL> [<X>]     /* Entity joins a Link */
<LE> ::= <EL> <CE>                                     /* Link joins an Entity */
<path> ::= <ER> | <path> <CL> <CE>                     /* A legal CIS global query */
<done> ::= <path> | <LR> | <EL>
```

Figure 7   A Simplified Grammar for the CIS GQP



(((A [A. SSN = W.SSN] W) [W.name = C.name] C) [C.CEO = A.name]) [A.name, A.degree]
(((E [A θ A] L) [A θ A] E) [A θ A]) [A , A]
ER
Condition
LR
Condition
ER
Condition
X
CL
CE
EL
LE
ES
ER
EP
ER
done
Legend:
A(GAlumni)
C(GCompany)
w(Gworkfor)

Figure 8   Parsing the CIS Query in Figure 6(a)

Table 5 The Global Syntax Construct (GSC) Table for the CIS Query in Figure 6(a)

R#	Type	Left-hand Relational Scheme	Construct Unit	Right-hand Relational Scheme
R1	EL	GAlumni	GAlumni[GAlumni.SSN = Gworkfor.SSN]Gworkfor	Gworkfor
R2	LE	GAlumni U Gworkfor	R1 [Gworkfor.name = GCompany.name] GCompany	GCompany
R3	ES	GAlumni U Gworkfor U GCompany	R2 [GCompany.CEO = GAlumni.name]	Not Applicable
R4	EP	GAlumni U Gworkfor U GCompany	R3 [GAlumni.name, GAlumni.degree]	Not Applicable

appropriate action routines for translating the GSC table into an EIC table. The EIC table is used to determine how to decompose a global query into local queries and how to process them so composite information can be formulated. The algorithms of action routines for σ, π, and join are described below.

4.1 The σ Algorithms

Consider a σ operation specified by a GSC tuple (ES or LS in Figure 7). The operation may be defined on a single global relation (either an entity or a link) with certain attribute restrictions. There are two possible cases of the relation and attributes in the local databases:

(σ1) Only one local database has a relation corresponding to the global relation, and all the global attributes have a counterpart in the local relation.

(σ2) Multiple local databases have a relation corresponding to the global relation, and all the global attributes have a counterpart in at least one of these local relations.

Alternatively, a σ operation may be defined on an intermediate relation, denoted by R# in a GSC as a result of other operations (e.g., R2 in Table 5). There are two possible cases of the relation and attributes:

(σ3) All the attributes exist in the R# relational scheme.

(σ4) Some of the attributes do not exist in the R# relational scheme, but have their counterparts in some local databases.

The action routine algorithms for σ1 through σ4 are described below.

The σ1 Algorithm

Step 1: Substitute the relation and attributes of the GSC tuple by the local relation and attributes.

Step 2: Insert the result from step 1 as an EIC tuple, and register the local database as the query execution location, as shown in Table 6.

Table 6 An Equivalent Internal Construct (EIC) Relation For the CIS Query in Figure 6(a)

R#	Type	Left-hand Relational Scheme	Construct Unit	Right-hand Relational Scheme	Query Execution Location
R1	EL (J1)	Alumni	Alumni[Alumni.SSN = workfor.SSN]workfor	workfor	Alumni
R2	LE (J2)	Alumni U workfor	R1 [workfor.name = Corporation.name] Corporation	Corporation	Alumni
R3	ES (σ3)	Alumni U workfor U Corporation	R2 [Corporation.CEO = Alumni.name]	Not Applicable	GQP
R4	EP (π3)	Alumni U workfor U Corporation	R3 [Alumni.name, Alumni.degree]	Not Applicable	GQP

The σ2 Algorithm

Step 1: For each of the local databases participating in the σ operation, do Step 1.1 to Step 1.2

Step 1.1 Substitute the relation and attributes of the GSC tuple by the local relation and attributes.

Step 1.2: If all the global attributes in the restriction also exist in a local relation, then insert an EIC tuple including the restriction; otherwise, insert an EIC tuple without the restriction. In either case, register the local database as the query execution location.

Step 2: Insert an equi-join operation on the primary key of the global relation for the EIC tuples created from step 1. Register the GQP as the query execution location.

Step 3: Insert a σ operation for the EIC tuple created in step 2 with the restriction. Register the GQP as the query execution location.

The σ3 Algorithm

Step 1: Substitute the global attributes of the GSC tuple by the attributes in the R# relational scheme.

Step 2: Store the result from step 1 as an EIC tuple, and register the GQP as the query execution location.

The σ4 Algorithm

Step 1: For each of the local databases participating in the operation, do Step 1.1 to Step 1.3

Step 1.1 Substitute the relation and attributes of the GSC tuple by the local relation and attributes.

Step 1.2: If all the global attributes in the restriction also exist in a local relation, then insert an EIC tuple including the restrictions; otherwise, insert an EIC tuple without the restrictions. In either case, register the local database as the query execution location.

Step 1.3: Insert an equi-join operation on the primary key of the global relation for the EIC tuples created from step 1.2 with the beginning intermediate relation. Register the GQP as the query execution location.

Step 2: Insert a σ operation for the EIC tuple created in step 1 on the condition. Register the GQP as the query execution location.

Note that some optimization is embedded in σ2 and σ4: when an operation can be performed locally so that only the useful data will be returned, the information is sent to a local database. For example, in σ4 step 1.2 if all the global attributes in the restriction also exist in a local relation, this information is included in an EIC tuple which is sent to the corresponding local database. The local database in turn makes use of the information to return only the relevant information to the GQP. We now turn our attention to the π algorithms.

The π Algorithms

The π algorithms are very similar to σ because both of them are unary operators. Corresponding to the σ algorithms, there are four π algorithms, denoted as π1 through π4. The π1 and π3 algorithms are identical to σ1 and σ3. The π2 and π4 algorithms are described below.

The π2 Algorithm

Step 1: For each of the local databases participating in the π operation, do Step 1.1 to Step 1.2

Step 1.1 substitute the relation and attributes of the GSC tuple by the local relation and attributes.

Step 1.2: Insert an EIC tuple with the primary key and as many of the local attributes corresponding to the global attributes to be projected; register the local database as the query execution location.

Step 2: Insert an outer equi-join operation on the primary key of the global relation for the EIC tuples created in step 1.2. Register the GQP as the query execution location.

The π4 Algorithm

Step 1: For each of the local databases participating in the operation, do Step 1.1 to Step 1.3

Step 1.1 Substitute the relation and attributes of the GSC tuple by the local relation and attributes.

Step 1.2: Insert an EIC tuple with the primary key and as many of the local attributes corresponding to the global attributes to be projected; register the local database as the query execution location.

Step 1.3: Insert an outer equi-join operation on the primary key of the local relation for the EIC tuples created from step 1.2 with the beginning intermediate relation. Register the GQP as the query execution location.

Step 2: Insert a π operation on the the R# created from step 1. Register the GQP as the query execution location.

We now turn our attention to the join operator.

4.3 The Join Algorithms

There are two possible local links: (1) only one local database has the link relation corresponding to the global link, and (2) multiple local databases have a link relation corresponding to the global link. Further, the join can be between two relations or between an intermediate result (R#) and a relation. Thus, we have four join algorithms as described below.

(J1) Between two global relations, and only one local database has the link relation corresponding to the global link.

(J2) Between an R# and a relation, and only one local database has the link relation corresponding to the global link.

(J3) Between two global relations, and multiple local databases have a link relation corresponding to the global link.

(J4) Between an R# and a relation, and multiple local databases have a link relation corresponding to the global link.

The J1 and J2 algorithms are identical, as described below.

Step 1: Find the local entity, next to the local link, which corresponds to the global entity to be joined.

Step 2: Substitute the global relations and attributes by the local relations and attributes.

Step 3: Insert the result from step 2 as an EIC tuple, and register the local database as the query execution location.

The J3 Algorithm

Step 1: For each of the local databases which has a link corresponding to the global link, do Step 1.1 to Step 1.3

Step 1.1: Find the local entity, next to the local link, which corresponds to the global entity to be joined.

Step 1.2: Substitute the global relations and attributes with the local relations and attributes.

Step 1.3: Insert an EIC tuple which joins the two local relations; register the local database as the query execution location.

Step 2: Insert an EIC tuple which performs an outer-union of the results from Step 1.

The J4 Algorithm

Step 1: For each of the local databases which has a link corresponding to the global link, do Step 1.1 to Step 1.2

Step 1.1: Find the local entity, next to the local link, which corresponds to the global entity to be joined.

Step 1.2: Insert an EIC tuple which retrieves the information in the local relation that we are interested in (if EL, then take link, if LE, then take entity); register the local database as the query execution location.

Step 2: Insert an EIC tuple which performs an outer-union of the results from Step 1.

Step 3: Insert EIC tuples which join the results from Step 2 with the left hand-side of the join.

We have presented the algorithms of the action routines for σ, π, and join. With the semantic action routines and the GSC driver, Table 5 can be translated into an EIC relation shown in Table 6.

4.4 Issues and Approaches

We now revisit the global query exemplified in Section 3. Suppose that CEO is not an attribute of the Corporation entity in the alumni database. In this case, although the GSC table remains to be the same, the corresponding EIC table will be different. The GSC driver recognizes that the "CEO" attribute does not exist in R2 anymore, but does exist in the company database. Therefore, the σ4 action routine is invoked instead of σ3, as shown in Table 7. It first inserts the R3.1 tuple which retrieves the company entity from the company database without any restriction. In addition, register the company database as the location for query execution. Secondly, it inserts R3.2 which joins company (R3.1) with R2 on the primary key of GCompany, i.e., join Corporation.name with Company.name. In addition, it registers the GQP as the location for query execution. Now that R3.2 has all the attributes required for the σ operation, the restriction is made in the GQP, and the result is retained in R3.3. Figure 9 depicts the equivalent parse tree for the case.

Now suppose that not only that the Corporation entity does not have the CEO attribute, but also the names in the Corporation column are spelled differently from the names in the Company column

Table 7 Alumni Databsse without the CEO attribute

R#	Type	Left-hand Relational Scheme	Construct Unit	Right-hand Relational Scheme	Query Execution Location
R1	EL (J1)	Alumni	Alumni [Alumni.SSN = workfor.SSN] workfor	workfor	Alumni
R2	LE (J2)	Alumni U workfor	R1 [workfor.name = Corporation.name] Corporation	Corporation	Alumni
R3.1	ES (σ4)	Company	Company	Not Applicable	Company
R3.2	(σ4)	Alumni U workfor U Corporation	R2 [Corporation.name = Company.name] R3.1	Company	GQP
R3.3	(σ4)	Alumni U workfor U CorporationU Company	R3.2 [Company.CEO = Alumni.name]	Not Applicable	GQP
R4	EP (π3)	Alumni U workfor U CorporationU Company	R3.3 [Alumni.name, Alumni.degree]	Not Applicable	GQP

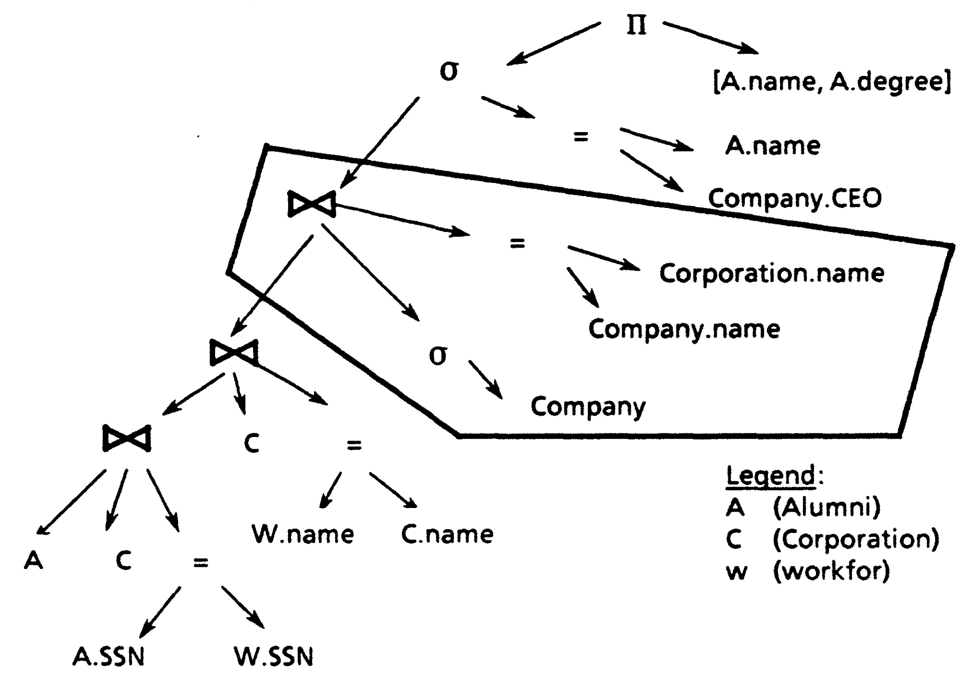


Figure 9 Equivalent Parse Tree for the Alumni Database w/o the CEO Attriobute

in the company database (e.g., IBM vs. International Business Machines, Inc.). The reader may have noticed that the action routines described earlier assumed that the instances in different databases have the identical format (e.g., all spelled as IBM). In order to handle these inter-database instance problem and other incompatibilities among disparate databases, further enhancements are needed in order to produce an even more detailed table which will resolve inconsistencies and incompatibilities among systems. Figure 10 exemplifies the equivalent parse tree in this case.

With the detailed EIC table, addition optimization can be performed before remote local databases are accessed, and composite information formulated. The approach presented in this paper also allows us to perform operations over mismatched domains [9] and to handle queries with missing information [26] systematically.

5. CONCLUDING REMARKS

Recent business changes are both enabled by and are the driving forces towards *increased connectivity*. Some of the increasingly evident changes include *globalization* of markets requiring

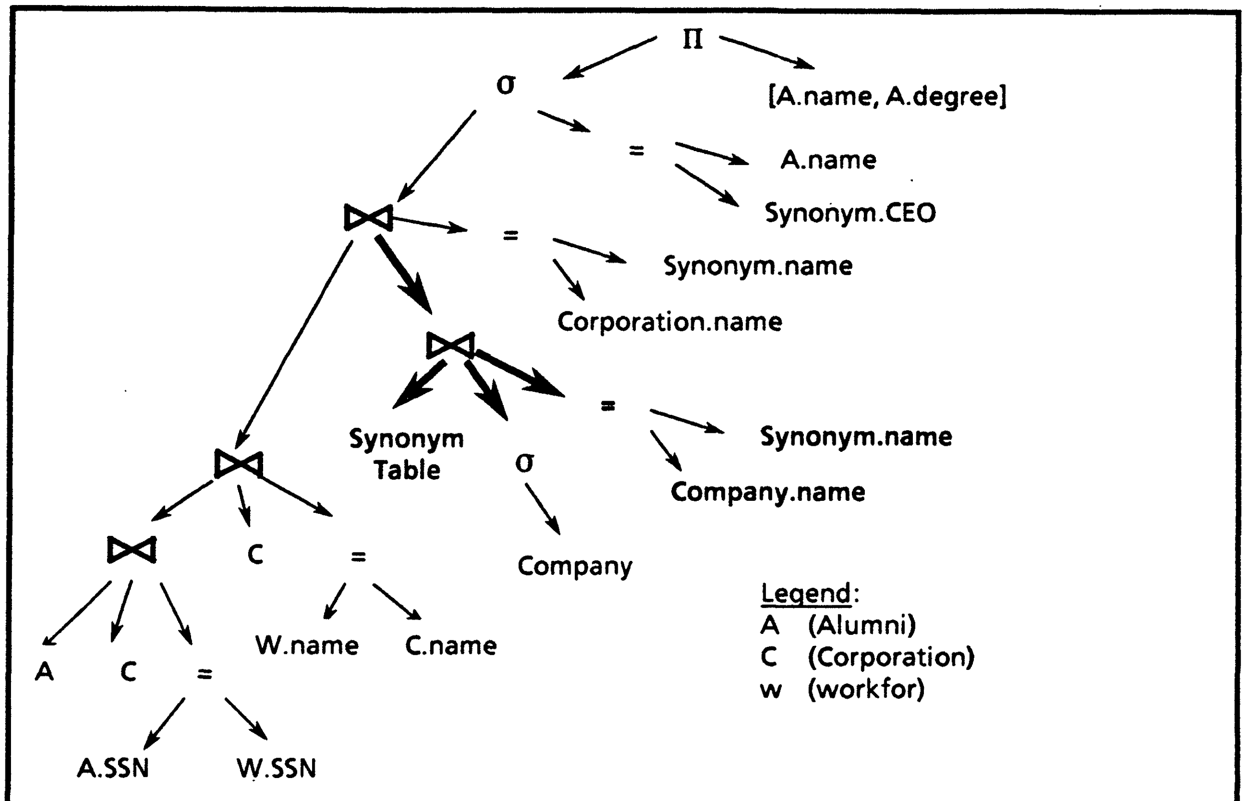


Figure 10 Parse Tree for the Alumni Database w/o the CEO Attriobute and Incompatible Names Between Corporation and Company

complex interconnectivity of systems, and innovative information systems requiring a high level of strategy, technology, and cross-functional integration such as global securities trading. Shadow databases (or subject databases) have been the predominant approach practiced in the industry today. The human intervention required in the approach is similar to a "batch-oriented" operating system. The query processing capability presented in this paper is a step towards an "on-line, interactive" CIS for increased connectivity. A research prototype is currently being developed to demonstrate the feasibility.

References

1. Abiteboul, S. & Hull, R.. (1987) . IFO : A Formal Semantic Database Model. ACM Transactions on Database Systems, 12 (4), 525-565.
2. Barrett S. "*Strategic Alternatives and Inter-Organizational Systems Implementations: An Overview,*" Journal of Management Information Systems, (Winter 1986-87), Vol. 3, No. 3, pp.3-16.
3. Batini, C. Lenzirini, M. and Navathe, S.B. "*A Comparative Analysis of Methodologies for Database Schema Integration,*" ACM Computing Surveys, Vol. 18, No. 4, (December 1986), pp. 323 - 363.
4. Brodie, M. and Mylopoulos, J. (Ed.) On Knowledge Base Management Systems, Springer-Verlag (1986).
5. Cash, J. I., and Konsynski, B.R. "*IS Redraws Competitive Boundaries,*" Harvard Business Review, (March-April 1985), 134-142.
6. Clemons, E.K. and McFarlan, F.W., "*Telecom: Hook Up or Lose Out,*" Harvard Business Review, (July-August, 1986).
7. Codd, E.F. Extending the Database Relational Model to Capture More Meaning. ACM Transactions on Database Systems. 4 (4), 397-434.
8. Dayal, U. and Hwang, K. "*View Definition and Generalization for Database Integration in Multidatabase System,*" IEEE Transactions on Software Engineering, Vol. SE-10, No. 6, (November 1984), pp. 628-644.
9. Demichiel, L. G., "Performing Operations Over Mismatched Domains," Proceedings of the Fifth International Conference on Data Engineering, February, 1989, p. 36 - 45.
10. Deen, S. M., Amin, R.R., and Taylor M.C. "*Data integration in distributed databases,*" IEEE Transactions on Software Engineering, Vol. SE-13, No. 7, (July 1987) pp. 860-864.
11. Elmasri R., Larson J. and Navathe, S. "*Schema Integration Algorithms for Federated Databases and Logical Database Design,*" Submitted for Publication, (1987).
12. Frank, Madnick, and Wang, "*A Conceptual Model for Integrated Autonomous Processing: An International Bank's Experience with Large Databases,*" Proceedings of the 8th International Conference on Information Systems (ICIS), (December, 1987).
13. Goldhirsch, D., Landers, T., Rosenberg, R., and Yedwab, L. "*MULTIBASE: System Administrator's Guide,*" Computer Corporation of America, Cambridge, MA, (November 1984).
14. Hull, R. & King, R. (1987) . Semantic Database Modeling: Survey, Applications, and Research Issues. ACM Computing Surveys. 19 (3), 201-259.
15. Hwang, H. & Dayal, U. (1981) . Using the Entity-Relationship Model for Implementing Multi-Model Database Systems. In P. Chen (Ed.) , Entity relationship approach to information modeling and analysis (pp. 237-258) . California: ER Institute .
16. Ives, B. and Learmonth, G.P., "*The Information System as a Competitive Weapon,*" Communications of the ACM, Vol. 27(12), (December 1984), pp. 1193-1201.
17. Kerschberg, L. Ed. Expert Database Systems, Proceedings from the First International Workshop. The Benjamin/Cummings Publishing Company (1986).
18. Litwin, W. and Abdellatif, A. "*Multidatabase Interoperability,*" IEEE Computer, (December 1986).
19. Lyngbaek, P. and McLeod D. "*An approach to object sharing in distributed database systems,*" The Proceedings of the 9th International Conf. on VLDB, (October, 1983).

20. Madnick (ed.) *The Strategic Use of Information Technology*. Oxford University Press, (1987).

21. Madnick and Wang, "*Integrating Disparate Databases for Composite Answers*," Proceedings of the 21st Annual Hawaii International Conference on System Sciences, (January 1988).

22. Manola, F. and Dayal, U. "*PDM: An Object-Oriented Data Model*," Proceedings of the International Workshop on Object-Oriented Database Systems. Pacific Grove, CA. (September 1986) pp. 18 - 25.

23. Peckham, J .& Maryanski, F. (1988) . Semantic Data Models. ACM Computing Surveys, 20 (3), 153-189.

24. Porter, M. and Millar, V.E., "*How information gives you competitive advantages*," Harvard Business Review (July-August 1985) p. 149-160.

25. Rockart, J. "*The Line Takes the Leadership: IS Management in a Wired Society*," Sloan Management Review, MIT Vol. 29, No. 4 (Spring 1988) p. 57-64.

26. Shin, D.G. (1988). Semantics for Handling Queries With Missing Information. Proceedings of the Ninth International Conference on Information Systems. 9, 161 - 167.

27. Wang, Y.R. and Madnick, S.E. "*Facilitating Connectivity in Composite Information Systems*," ACM Data Base, in press.

28. Wang, Y.R. and Madnick, S.E. "*Evolution Towards Strategic Applications of Databases Through Composite Information Systems*," Journal of Management Information Systems, in press.

29. Wang, Y.R. and Madnick, S.E. (ed.) Connectivity Among Information Systems: Composite Information Systems Project, Volume 1, MIT (September 1988).