# Introduction of Relational Data Bases Within a Cost-Estimating System

TUNG AU, CHRIS T. HENDRICKSON, and ANGELO F. PASQUALE

ABSTRACT

The concept of a relational data base for data storage and retrieval in a cost-estimating system is introduced. The unique characteristics of the relational data base are discussed in contrast with hierarchical and network data base models. Various components of data bases and their uses are described in the context of cost estimation, with an example of a relational data base for bridge work. An application computer program for estimating final design costs is discussed. Advantages and disadvantages of different data base designs for project planning and management are briefly considered.

The planning and control of construction projects involve the creation, manipulation, and access of a great deal of numerical data and records on project activities including activity definitions, material and labor inputs, and unit prices. This paper will focus only on the cost-estimating aspect of the planning and control process for the purpose of introducing the concept of relational data bases.

There are two main advantages of a data base. First, a data base contains only operational data and thus provides data independence from individual applications. Certain associations or relationships link the data together in some fashion depending on the meaning of the data in the data base. The relationships themselves may become part of the operational data stored. Second, a data base can provide more effective centralized control of the data. This allows stored data to be concurrently accessed (i.e., simultaneously shared) by many users with different applications. The amount of redundancy can consequently be reduced, thereby saving computer resources and reducing the chance of inconsistency in the data. Even on a personal computer, the central control of data in a data base may be advantageous for different applications or uses.

There are two reasons to develop data independence in data bases. One is that each application may need its own distinct view of common data. For example, a computer program written by the cost estimation division of an organization may use all of the data entities, but other departments of the same organization may only use specific segments of the data, and upper management may need only limited information for long-range planning. Data independence makes any view of the data possible. The second reason for providing data independence is that if the storage structure and its accompanying access strategy have to be changed because of changing requirements of the organization, the existing programs and the access and storage of data will be unaffected. Thus data independence allows the data base to grow and respond to the organization's needs without making all previous applications obsolete.

The centralized control of data generally facilitates the maintenance of data integrity. Not only inconsistent but even grossly incorrect data entries can be checked through validation procedures. Data bases provide an excellent means by which the standards of an organization can be enforced because centralized control ensures that the form of all data must be exactly as specified. Furthermore, centralized control enhances data security for mainframe computers because the privilege of retrieving, updating, and generally manipulating data must be authorized. As the use of microcomputers becomes more widespread, centralized control can be effected by limiting authorized access to the data base.

## COMPONENTS OF DATA BASE SYSTEMS

An overview of the components of a data base system and their interrelationships is shown in Figure 1. The functions of these components are as follows:

- The data structure or conceptual data base model is the layout of the storage structure and access strategy without the data. The overall configuration of the data base is referred to as the conceptual schema. The storage structure refers to how the data are physically recorded in the data base. The access strategy pertains to how the data are accessed by the user. Data independence is defined by Date (1) as the immunity of applications to changes in storage structure and access strategy. The applications may be computer programs or other user interfaces. In other words, no knowledge of the organization of data or any access techniques need be built into the logic of the application.
- The data base management system (DBMS) is the software that stores, maintains, and retrieves data. The DBMS is involved in every aspect of the data base system's operation because it alone accesses and stores data. The DBMS works in the following way: The user issues a request through an external model (explained later) using a language understandable to the DBMS. The DBMS interprets and determines if it can fulfill the request. If it can, the DBMS performs the required operations on the data base. The objective is to free the user from the detail of exactly how data are stored and manipulated. Desirable properties (2) of a data base management system include the ability to provide the user with ready access to the stored data and to maintain the integrity and security of the data.
- The data dictionary contains the definitions of the data in the data base. It stores the characteristics and relationships of the data entities. It is the information source for anything dealing with the data base system. It contains all of the design

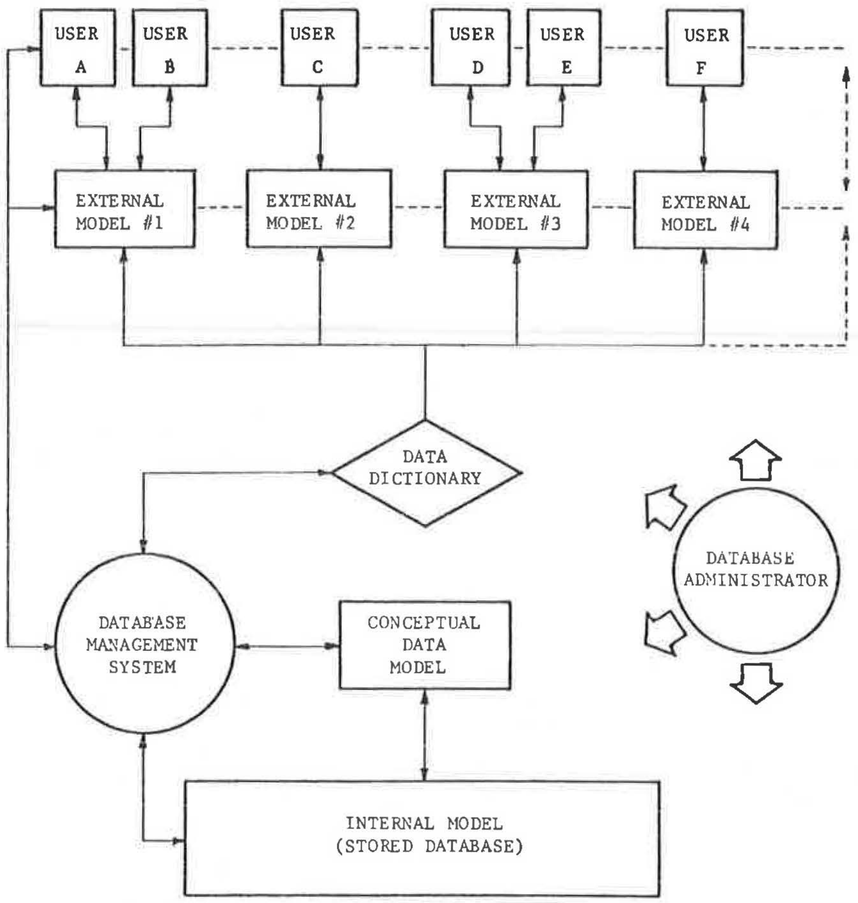

**FIGURE 1 Data base system architecture.**

or layout of the data base (e.g., what data are stored, how they are conceptually related). The data dictionary is the most important tool of the data base administrator. The data dictionary may also contain user authorization (i.e., who may have access to what data). The data dictionary is an essential part of the system because the data base may be unintelligible without it.

• External models are the means by which the users view the data base. Any particular user's view may be of just a subset of the total data in the data base. For example, the external model for the payroll department may consider the data base to consist solely of engineer and salary entities. As far as that department is concerned, no other information exists. Another case would be one in which a user views only certain types of certain entities, such as the payroll department viewing all engineers from a certain branch office with salaries of more than $40,000. External models provide the format by which any specific information needed is retrieved. The user can interface the data base either directly through a sublanguage or indirectly through an application program.

• Finally, the data base administrator (DBA) is the person or persons responsible for managing the data base system. The assignment of the DBA by an enterprise should not be taken lightly. Especially in large organizations with many users, the DBA is vital to the success of the data base system.

## ORGANIZATION OF INFORMATION IN A DATA BASE

A conceptual data base model is characterized by its data content and the relationships among data items. It includes the conceptual schema and any authorization check and validation procedures. The three most popular models are hierarchical, network, and relational.

### Hierarchical Data Base Model

In the hierarchical model, data can be represented in a tree structure. Nodes of the tree are data items. The branches of the tree represent one-to-many relationships among the data items. Any type of data entry can be the root of the tree (the highest level of the tree). The root may have any number of lower level dependents, which may also have any number of lower level dependents, and so on. With hierarchical models, no piece of data--with the exception of the root--can be accessed without the presence of its superior in the tree. Hierarchical models do not lend themselves readily to the structuring of data that are not hierarchical in actuality. The hierarchical model is a special case of the network model.

### Network Data Base Model

The network model uses branches to indicate the relationships among the data items, which are represented as the nodes. The network model is a more general structure than the hierarchical model because any piece of data may have any number of immediate superiors and dependents. With the network model, an additional piece of data called a connector is introduced. Efficiency is usually the reason for choosing the network data model over the hierarchical or other models. A data item in a network may be accessed through a number of routes, whereas only one path is available to any datum in the hierarchical model. The major disadvantage of the network model is its complexity.

### Relational Data Base Model

The relational model is based on an explicit algebra of relations that can be visualized as sets of two-dimensional data files or "tables." The data are stored in tables that conform to certain constraints that allow the application of elementary mathematical relation theory (3). The method that ensures that the organization of the data base has certain desirable properties is referred to as normalization (4). It is an effective way of determining the best possible arrangement of the data in the data base. Because the relational data base model is a relatively recent development, its applications to engineering design and construction have been heretofore limited. Examples of application to structural engineering (5) illustrate the potential of this model.

## RELATIONAL DATA BASES FOR COST ESTIMATING

The basic terms in the relational data base model will be described using cost estimating as an example. It is informative to compare this model with a generalized hierarchical and network model that was developed for a cost-estimating system by O'Connor and Lemberger (6).

A relational data base stores information in a set of relations that are basically two-dimensional tables, each identified by a relation name. The description of an individual relation is called the relation schema. Each column of the table representing a relation describes the characteristic of a data entry and is called an attribute; each row of the table indicates some unique entity of the data entry and is called a tuple. The ordering of columns or rows is arbitrary. The domain of an attribute refers to the set of all allowable values for that attribute. A constraint is a tool used to control the integrity of a relation. For example, it may specify nonnegative values for the attributes or prescribe the dependency of attributes. When all relations have been described, the data base schema is completely defined.

For example, a possible relation in a data base for cost estimating is given in Table 1. This RESOURCE RATES relation lists the cost (in dollars per crew-hour), task productivity (in task unit per crew-hour), and unit cost (in dollars per task unit) for labor, material, and equipment needed for different activities. Given in Table 1 is a set of data for repaving asphalt roadway surfaces where the task unit refers to a square yard of surface. A constraint for each attribute except the resource type and task definition is that entries must be numerical and greater than 0. Also, if one attribute is dependent on another, a constraint can be specified to ensure this dependency. Thus, for RESOURCE RATES relation,

**TABLE 1 Example of a Relation for Cost Estimation: RESOURCE RATES**

| Task Definition | Resource Type | Cost ($/crew-hour) | Task Productivity (task unit/ crew-hour) | Unit Cost ($/task unit) |
|---|---|---|---|---|
| APAVING | Labor | 66.24 | 13.00 | 5.10 |
| APAVING | Material | 68.90 | 13.00 | 5.30 |
| APAVING | Equipment | 73.24 | 13.00 | 5.63 |

Cost = Task productivity x Unit cost

Each relation should possess a key that is a set of one or more attributes that uniquely identify a particular tuple or row. The key is used to access or manipulate particular data items as explained hereafter. In Table 1, the task and resource type attributes together uniquely identify each tuple in the RESOURCE RATES relation. In contrast, the task definition and task productivity attributes have the same entries for more than one tuple and thus do not form a key.

## DATA RETRIEVAL

Basic relational operators can be used individually or in any combination to retrieve data from the data base. Some basic operators useful for cost estimating include

1. SELECT particular rows that meet a desired criterion from a relation,
2. JOIN to produce a new relation from selected rows of two or more other relations, and
3. PROJECT to produce a new relation consisting of a desired subset of attributes (columns) from a relation.

Thus the data base management system (DMBS) software can take advantage of algebraic operations for combining or manipulating information (3). For example, consider the situation of an agency wishing to construct a bridge. The agency has had considerable experience managing the construction of bridges of all types. Over the years, performance data on the agency's experience had been accumulated. The data were then entered into a relational data base. Table 2 gives the relation called BRIDGE WORK and some sample, but hypothetical, data entries.

The attributes of this relation are

- PROJECT NUMBER--a six-digit code identifying the particular project;
- TYPE OF BRIDGE--the type of bridge constructed;
- LOCATION--the location of the project;
- CROSSING--what the bridge crosses over (e.g., a river);
- SITE CONDITIONS--a brief description of site peculiarities;
- ERECTION TIME--time taken to erect the bridge;
- SPAN--span of the bridge in feet; and
- ESTIMATED - ACTUAL COST--difference of actual and estimated cost.

Assume that the conditions of the present project are that the span is 700 ft and that the bridge is to cross a river with limestone substrata and will be located in Pittsburgh, Pennsylvania. In initial or preliminary planning, the planner might query the data base four separate times as follows:

- SELECT from BRIDGEWORK where SPAN > 600 ft, SPAN < 800 ft, and CROSSING = "River";
- SELECT from BRIDGEWORK where SPAN > 600 ft, SPAN < 800 ft, and SITE CONDITIONS = "Limestone";
- SELECT from BRIDGEWORK where TYPE OF BRIDGE = "Steel Suspension" and LOCATION = "Pittsburgh, Pennsylvania"; and
- SELECT from BRIDGEWORK where SPAN < 700 ft and ESTIMATED - ACTUAL COST > 0.

TABLE 2 Example of Bridge Work Relation

| Project No. | Type of Bridge | Location | Crossing | Site Conditions | Erection Time (yr) | Span (ft) | Estimated – Actual Cost ($) |
|---|---|---|---|---|---|---|---|
| 170145 | Steel plate girder | Portland, Maine | Railroad tracks | 200-ft valley limestone | 1 | 650 | −500,000 |
| 169137 | Concrete arch | San Diego, California | Highway | 250-ft-high sandy loam | 2 | 478 | −27,500 |
| 197108 | Cable-stayed steel girder | St. Louis, Missouri | River | Urban area congestion | 5 | 956 | +480,000 |
| 140079 | Steel suspension | Norfolk, Virginia | Ocean bay | 135-ft-deep pile foundation | 4 | 3,820 | −760,800 |

Each SELECT operation would yield the bridge examples in the data base that correspond to the desired criteria. In practice, an input-output interpreter program should be available to translate these inquiries to and from a more general problem-oriented language (7).

These four queries may directly model what goes through a project manager's mind when faced with these project conditions. He or she may first ask: "What experience have we had with bridges of this span over rivers?" (the first query). "What experience have we had with bridges of this span with these site conditions?" (the second query). "What is our experience with suspension bridges in Pittsburgh?" (the third query). "For bridges of this span or less, how many and which were erected without a cost overrun?" (the fourth query). Many more similar questions can be modeled using only the relatively small data base proposed.

The versatility of data views provided by relational data bases and commercially available data base management systems (which handle the details of data storage, manipulation, and retrieval automatically) can give all participants efficient information storage and retrieval abilities. The potential for data base management systems in cost estimation alone is great, as illustrated later.

## DOMAIN-KEY NORMAL FORM

Although the relational data base organization provides great flexibility in organizing data, care should be taken to reduce redundancy (i.e., storing multiple instances of the same information) and to avoid dependencies among attributes in the same relation. In particular, using normalized relations is desirable to avoid dependencies.

At the most basic level, an unnormalized relation might have more than one instance of an attribute for a particular row. For example, some bridge might cross both a river and railroad tracks in Table 2. Such multiple occurrences of data items create significant problems in manipulating and storing data. Because computer implementations of relational data bases generally preclude such multiple entries, such unnormalized relations are not possible in practice.

More generally, dependencies among data items should be removed. For example, "Task Productivity" in Table 1 is equal in each row corresponding to "Asphalt Paving" and the three different resource types. Altering "Task Productivity" in one row and not all three would result in an inconsistent and potentially misleading data base. To avoid this problem, a new relation can be defined with attributes "Task Definition" and "Task Productivity" (in which one row is "Asphalt Paving" and "13.00") and eliminate the "Task Productivity" attribute in RESOURCE RATES (Table 1).

To assure that all the undesirable properties are eliminated in each relation, Fagin (4) has defined a domain-key normal form (DK/NF) that encompasses the removal of all insertion and deletion anomalies by whatever means are necessary. According to Fagin (4), an insertion anomaly occurs when a seemingly legal insertion of a single tuple into a valid instance of the relation schema causes the resulting relation to violate one of its constraints; a deletion anomaly occurs when the deletion of a single tuple from a valid instance of the relation schema causes a constraint to be violated. Thus the removal of these anomalies leads to a domain-key normal form.

## APPLICATION TO FINAL DESIGN ESTIMATION

A FORTRAN computer program called ESTA (8), which uses a relational data base to compute final design cost estimates for building projects, has been developed at Carnegie-Mellon University. ESTA uses a relational data base management system called the Relational Information Management (RIM) system (9) developed by the Boeing Commercial Airplane Company. This system frees the user from the concern for data storage and recall, while ESTA relieves the estimator from the tedious task of calculating final design estimates. A program listing and detailed description of ESTA appears in Pasquale (8).

RIM is a general purpose data base management system that has been used successfully in both engineering and business applications. It is just one of several commercially available relational data base management systems; others include INGRES from Relational Technology, Inc., ORACLE from Relational Software, Inc., and IBM's System SOL. RIM provides a user with a relational algebraic sublanguage for reporting and other functions. RIM can intersect, select, join, subtract, and project relations, giving the user considerable capability for data manipulation; explicit descriptions of these operations are given elsewhere (9). The user can create, update, or query the data base through two modes. One is the stand-alone system that allows interactive queries using the RIM command sublanguage. The other is the host language interface system that enables RIM subroutines to be called from a FORTRAN program like ESTA.

ESTA is an interactive program that offers four basic function options to the user. The user has the capability of (a) writing ESTA user documentation, (b) creating new final design files, (c) computing final design cost estimates, and (d) manipulating information in the data base.

The input to ESTA consists of final design files specifying the types and quantities of items in the project design. ESTA output is the final design cost estimate for each project based on the information loaded into the data base by the user. The user loads the data base with unit price and location cost index information. ESTA helps the user construct the design files based on some given final design for a project.

For each design item, ESTA searches the data base for the item's material and installation unit costs and finds the material and installation cost indexes that correspond to the project location. The unit costs are multiplied by their respective location cost adjustments and then added to yield the total unit cost. Total estimated project cost is found by multiplying the total unit price of each item by its quantity and then summing all of the items in the design.

To supply unit price information, two relations are required: a unit price relation and a location cost index relation. The unit price relation contains attributes dealing with the unit cost of items that may be used in any given building design. The location cost index relation contains attributes dealing with the effect of geographic location on project costs. The attributes and their domains for both relations are given in Tables 3 and 4.

**TABLE 3 Unit Price Relation (relation name: UNITCOST, relation description: unit price information)**

| Attribute Name | Attribute Description | Type | Length | Key |
|---|---|---|---|---|
| ITEMCODE | Item code number | Text | 11 | Yes |
| DESCRIPT | Item description | Text | 50 | No |
| ITEMUNIT | Item units | Text | 6 | No |
| MATLCOST | Material unit cost | Real | 1 | No |
| INSTCOST | Installation unit cost | Real | 1 | No |
| DATEMCOS | Date for material unit cost | Text | 9 | No |
| DATEICOS | Date for installation unit cost | Text | 9 | No |

**TABLE 4 Location Index Relation (relation name: LOCALITY, relation description: location index relation)**

| Attribute Name | Attribute Description | Type | Length | Key |
|---|---|---|---|---|
| CITY | City | Text | 25 | No |
| STATE | State | Text | 5 | No |
| MATLINDX | Material cost index | Real | 1 | No |
| INSTINDX | Installation cost index | Real | 1 | No |
| DATEMIND | Date for material index | Text | 9 | No |
| DATEIIND | Date for installation index | Text | 9 | No |

Note: City and state attributes together are considered "key" in this relation.

The unit price relation has seven attributes: item code number, item description, item units, material and installation unit costs, and dates for the unit costs. The item code number can be any 11-character text string that identifies one and only one item of all the items possible in a design. This restriction is necessary for this attribute to be considered "key," that is identifying a unique combination of other attributes. In practice, each separate item will have its own individual item code number. Any combination of up to 11 numbers, letters, blanks, or other characters can be used as item code numbers.

The item description can be any text string up to 50 characters long. This attribute gives the estimator some idea of the type of item and its qualities. An example description might be "concrete block, foundation walls, 12 in. thick." With a record of the item description, the estimate reviewer need not consult an item code number dictionary to determine what type of item a given code number represents.

The item unit attribute is the desired units for the item in text form of up to five characters. Examples of units are "S.F.," "C.Y.," "Each," and "Acre."

The unit cost of each item is decomposed into two parts--material unit cost and installation unit cost. The reason for this is that costs are usually known more accurately and readily on this basis. Also, estimators and other members of the organization may prefer this disaggregation for cost control purposes. The material unit cost is strictly the cost of the material component of the item cost in the appropriate units. Likewise, the installation unit cost is strictly the cost of the labor and equipment component of item cost. The domains of these values are the positive real numbers.

Because the dates for the material and installation unit costs are important, they too are stored in the data base. A separate date appears for each unit cost. Dates are entered automatically when a user loads a unit cost in the data base. The last date of modification automatically replaces the load-in date if the user modifies the unit prices in any way. Thus the accuracy of unit price information can be judged from its date. Dates are stored as text of nine characters, for example "25-Dec-82."

The location index relation contains cost indexes that reflect variations in material and installation costs due to geographic effects. The attributes of this relation are city, state, material and installation cost indexes, and their accompanying dates.

The city can be a text string of up to 25 characters. The state refers to the state abbreviation and can only be five characters long. For foreign country indexes the state attribute can be used for the abbreviation of the country.

The material and location cost indexes are based on a scale of 100.0. For instance, a city with installation costs lower than the national average might have an installation index of 89.7, and a city with material costs higher than the national average might have a material index of 107.9. The dates for each of these indexes are stored and maintained automatically just as they are for costs in the unit price relation.

Estimators must load the ESTA data base before any estimates can be computed. The number of tuples (rows) entered will depend on the number of possible items in any given design and on the number of possible locations for any project. The data base structure is fixed and only the number of tuples may be varied by the user. The actual source of the data may be the organization itself or nationally averaged data.

With these relations, the RIM management program, and the ESTA interface software program, a variety of query, edit, and data manipulations can be undertaken efficiently. "Viewing" data from different perspectives, as illustrated in the previous section, can be accomplished for a wide variety of queries. Moreover, other application programs could be designed to access the same data set by means of the RIM system. These other applications might include accounting, preliminary cost estimation, project monitoring, or other project management endeavors. For example, prototype design files can be developed and preliminary cost estimates using current unit rates can be quickly obtained from ESTA.

## CONCLUSION

The concept and use of a relational data base information system in project management have been introduced. Relational data base systems have a number of general advantages such as (a) freeing the user from concerns about data storage and retrieval, (b) facilitating data manipulation, (c) allowing any "view" of data, (d) providing immunity to changes in storage structure or access, (e) providing centralized control of data, (f) reducing data inconsistency and promoting data integrity, and (g) providing general capability, adaptability, and stability.

All of these advantages can be realized with the implementation and use of a relational data base system. Some of the more intangible advantages include the benefits from thought processes generated by devising new data designs and improved ways of thinking of data entities in terms of their relationships to one another. The revolution of data base storage provided by computer programmers may lead to further revolutions in applications provided by engineers.

The application to final design cost estimation illustrates a strategy for adopting data bases for project management. A general purpose, available software package (RIM) was used to control the actual data base, serve administrative functions, and provide limited query capabilities. An application program (ESTA) was designed to communicate with RIM for the purpose of final design cost estimation. A user would not be aware of the interaction between the two programs (ESTA and RIM) or of the actual structure of data in the data base. Additional application programs can be designed to access the RIM system and data base for other purposes.

The ESTA final design cost estimator is just one application. Relational data base systems can be beneficial in all phases of project development. Future work to pursue the goals of more complete computerized decision support might include

- The development of computerized design software that synthesizes the design and cost estimation processes,
- The development of "knowledge-based" systems to incorporate local and global constraints and design rules, and
- The identification of ways in which organization-wide knowledge can be increased through the use of such systems.

Surely these objectives transcend the realm of cost estimation, but the independence of the data stored in relational data bases enables a myriad of uses. It is a natural extension to use the operational data for purposes above and beyond cost estimation alone. In making decisions regarding in-house data base designs, organizations would be well advised to consider systems that might grow to accommodate such expanded uses. Relational data bases are among the best means of obtaining such flexibility.

## REFERENCES

1. C.J. Date. An Introduction to Database Systems, 2nd ed. Addison-Wesley Publishing Co., Inc., Reading, Mass., 1977.
2. M.E. Walsh. Information Management Systems/Virtual Storage. Reston Publishing Co., Inc., Reston, Va., 1979.
3. E.F. Codd. Courant Computer Science Symposia Series, Vol. 6: Relational Completeness of Data Base Sublanguages. Prentice-Hall, Inc., Englewood Cliffs, N.J., 1972.
4. R. Fagin. A Normal Form for Relational Databases That is Based on Domains and Keys. ACM Transaction on Database Systems, Vol. 6, No. 3, Sept. 1981.
5. M.J. Schaefer, D.R. Rehak, and S.J. Fenves. Introduction to Relational Databases Using Structural Engineering Examples. Journal of Division of Technical Councils, ASCE, Vol. 110, No. TC1, May 1984, pp. 1-18.
6. M.J. O'Connor and E.S. Lemberger. The Use of FOCUS DBMS in a Cost Estimating System. Proc., ASCE Specialty Conference on Computing in Civil Engineering, June 1978, pp. 465-476.
7. D.R. Rehak and L.A. Lopez. A Tool for Translating Problem Oriented Languages. Journal of Division of Technical Councils, ASCE, Vol. 105, No. TC1, April 1979, pp. 31-42.
8. A.F. Pasquale. Construction Cost Estimation Using Relational Databases. Master's thesis. Carnegie-Mellon University, Pittsburgh, Pa., 1982.
9. Relational Information Management RIM System, User Guide: Rim 5.0. Boeing Commercial Airplane Company, Seattle, Wash., 1981.