RELATIONAL DATABASE MODELING OF BUILDING DESIGN DATA

By William J. Rasdorf, P. E.*

ABSTRACT

This paper originates from research related to the development of an information representation system for the building design process. The paper presents a portion of a general conceptual schema for an integrated database that can support structural engineering design of buildings. The topological and geometric modeling of a variety of structural systems is discussed, and the relational database implementation of a particular system is presented.

INTRODUCTION

In the broad arena of computer-aided design, relational database management systems are establishing themselves as useful engineering tools for several reasons. Among those reasons are that the relational database model can accurately represent a wide variety of engineering design data and that the data contained in such a model is directly and readily available to all members of the design team needing it, allowing them to work concurrently.

Because engineers and architects are moving toward centralized, integrated repositories of data, the significant promise of the relational model warrants investigation. The purpose of this paper is to discuss the use of a relational database for representing the data gathered and derived during the building design process. In doing so, the paper considers the complexity of the relationships among topology, geometry, and attributes, and the critical requirements they impose on a building design database.

BUILDING STRUCTURE

Before a building can be represented accurately in a database, it must be described precisely. The description must reflect the hierarchical nature of the building and must include the topology and geometry of its form as well as the attributes of its component elements. This section briefly describes topology, geometry, and attributes.

*Assistant Professor of Civil Engineering and Computer Science, North Carolina State University at Raleigh, P.O. Box 7908, Raleigh, NC 27695.

Topology and Geometry

Two types of information are necessary to represent spatially the basic physical elements of a building. Geometry defines the dimensions and spatial location of each element, while topology describes the connections between the elements. For the representation of a structure to be complete, its topological definition must be accompanied by a geometric definition that fixes its position in space.

Attributes

Each physical element in a building may be described by attributes. Attributes define the physical composition of an element and describe its functional characteristics. Each attribute has a name and, for each element, a value. Examples of different types of attributes include:

- physical properties such as area, depth, width, section modulus, and moment of inertia;

- response properties consisting of forces and moments; and

- design properties such as yield stress and modulus of elasticity.

Attributes combine the functional and physical composition of the structure with its topological and geometrical representation, thereby converting a conceptual representation to a material representation. Without attribute information, the database can be used only for computer-aided drafting applications. Design applications mandate that attributes be associated with topological and geometric information to represent a building comprehensively.

DATABASE STRUCTURE

The overall, logical structure of a database, a schema, is composed of relations. A relation is a two-dimensional table whose rows contain instances of a building element and whose columns describe the attributes of the element. A row-column intersection contains an attribute value for a particular element (e.g.: element - W shapes; relation - section properties; instance - W12x50; attribute - area; value - 14.70). After carefully determining the elements and element properties that are needed for design, the schema is defined by specifying the corresponding relations and attributes using standard relational database design practices. A subset of the total schema is referred to as a subschema.

A subschema allows the user to access a limited amount of data and provides a unique, restricted perspective of the database. Such a view allows each design discipline to concentrate its efforts on the relations and attribute values it needs. For example, architects will access relations containing data about space function, location, use, etc., while structural engineers will access relations containing data like that shown above for section properties.

In an integrated database, subschemas can be used to limit access to data that is clearly of use to a single discipline, thus helping to reduce potential database integrity violations - situations in which data that violates a design constraint resides in the database, thus limiting its reliability. At the same time, unrestricted access to data permits it to be used as needed by different design disciplines. Location data, for example, is an important link between the design disciplines.

RELATIONAL DATA STRUCTURE FOR BUILDINGS

A database forms the internal representation of the physical elements of a building. In order to create such a representation, there must exist a set of basic database components corresponding to the elements of the physical structure. The representation of these components, their characteristics, and their relationships to other components are described in this section.

Grid Representation

In a database, connections (topology) between the physical elements of a building frame are represented by nodes. In addition to representing connections, nodes provide spatially well-defined reference points that can be located (geometry) by x, y, and z coordinates. A two-dimensional collection of nodes is a grid. A grid is the conceptual framework to which all building components are related. Nodes connect components of the data structure, providing a means of access between them. Beams and columns, for example, are defined by two adjacent nodes, while wall and floor panels are defined by four adjacent nodes. The physical significance of nodes is that they represent connections in the structure. The connections are one of three kinds: pinned, rigid, or continuous.

A structure is divided into three primary grids called bays, aisles, and floors, corresponding respectively to the x, y, and z coordinate axes. A structure may be further divided into subgrids, where each subgrid represents an offset between the primary grids. The distance between grids and the building origin defines the geometry of the frame represented by each grid and, consequently, of their lower level internal components.

This nodal-based representation is both general and flexible. It provides precise access to any node, and it allows any number of subnode divisions between grids in any direction. The grid provides a uniform representation of the regular portions of a structure while the subgrids allow for irregularities in plan and elevation.

Element Representation

Not all physical elements of a structure need to be directly represented in a database. Although intermediate level abstractions, such as cores, walls, bays, frames, etc., are conceptually useful in describing the hierarchy of building components and are used in entering the representation of a building into a database, they are not necessary database components. The database can alternatively represent a rigid frame as a bay or aisle composed of beams and columns. In effect, a frame is computed rather than stored. As a result a frame is an abstract model of a portion of the conceptual database: it is a view or subschema.

Building Representation

The following discussion shows how topological, geometrical, and attribute data can be represented in a relational database. It does so by giving the relations that define these three different classes of building design data for a two-dimensional, concrete, building frame.

In the following relations, the suffix ATT indicates a relation containing attribute data, the suffix GEO indicates a relation containing geometric location data, and the suffix TOPO indicates a relation containing topological data. In a graphical form of the relationships embodied in the representation, the TOPO relations represent the links between the nodes of the graph and the ATT relations represent the nodes themselves.

Two relations model the building. These are:

BUILDINGATT1(buildingID, name, location, area, stories, ...);

BUILDINGATT2(buildingID, main steel grade, stirrup grade, concrete strength, ...); and

BUILDINGTOPO (buildingID, X gridID, YgridID).

The BUILDINGATT1 relation contains general descriptive information about the building, including an identification number, designated by the suffix ID, the name of the building, its address, the amount of rentable floor area, the number of stories it contains, etc. The BUILDINGATT2 relation contains design data that has been determined to be common for all components of the building. The BUILDINGTOPO relation identifies the nodes that it contains, thereby defining the building's coordinate system and establishing the frame as a reference datum.

Information about nodes and grids is contained in the relations:

NODETOPO (nodeID, xgridID, ygridID);

XGRIDGEO (xgridID, xdistance); and

NODEATT (nodeID, xdisp, ydisp, xrot, yrot).

The NODETOPO relation defines the position of each node in the structure by relating it to the grid coordinate system. Its precise location in space is stored in the GEO relation in terms of a distance from the origin for each of the x, y, and z grids. Attribute data about these nodes is given in the NODEATT relation, where joint displacements and rotations are stored.

Connecting nodes to other components, in this case beams, is a relation called BEAMTOPO:

BEAMTOPO (<u>beamID</u>, nodeIDend1, nodeIDend2).

This relation defines the relationship between beams and their two end nodes, which in turn defines the relationship between beams and other components of the structure.

Many relations can be defined to supply attribute information about beams. Among these are:

BEAMLOADS (<u>beamID</u>, load type, load magnitude, ...)

BEAMFORCES (<u>beamID</u>, Xforce1, Yforce1, Xmoment1, Ymoment1, Xforce2, Yforce2, Xmoment2, Ymoment2);

BEAMATTREQ (<u>beamID</u>, A, B, D, Dc, Ast, Asc, Asv, ...); and

BEAMATTPROV (<u>beamID</u>, B, D, Tbarsize, Cbarsize, Vbarsize,...).

These relations store data supplied to and generated by beam analysis and design procedures. The BEAMLOADS relation holds data about uniformly distributed loads imposed on the beam. This data is used during analysis to determine member forces and moments, which are stored in the BEAMFORCES relation and which in turn are used during design to determine the required properties of each beam for the relation BEAMATTREQ. The required properties are compared to actual member properties to select a section or they are determined by the designer and stored in the relation BEAMATTPROV. In many cases actual member properties can be determined from protected <u>project independent</u> relations that contain standard information. Two such relations could be stored as follows:

REBARATT1(<u>barsize</u>, diameter, area, perimeter, weight); and

REBARATT2(<u>ASTMID</u>, <u>barsize</u>, <u>grade</u>, Fy, Ft).

Accessing all of the properties of the first relation requires only the bar size. Therefore, the BEAMATTPROV relation need only store the bar size of the chosen shape. The properties of the bar can be determined from the protected library relations REBARATT.

The relations presented above represent those that define the topology, geometry, and attributes of the components of concrete framed buildings. Although the representation is by no means complete it is clear that all of the relationships and data of the engineering design process can be modelled relationally. This example demonstrates the ease with which a relational representation can be defined.

<u>Database Use</u>

One of the most important characteristics of a relational database management system is that it permits ready access to any information stored in the database by providing convenient and powerful facilities for locating and gathering data, modifying it to the desired form, and exporting it to a designer or an application program. Interaction with a relational database for retrieving and manipulating data may be divided into two, distinct types: user interaction and application-program interaction.

Relational algebra and relational calculus languages provide a rich set of data retrieval and manipulation operators that allow a user to rigorously query the database. These languages apply a combination of special relational operators (SELECT, PROJECT, JOIN, etc.) and traditional set operators (INTERSECT, UNION, DIFFERENCE, etc.) to relations, enabling the user to restructure existing relations into new ones to satisfy information needs. Built-in functions (SUM, AVERAGE, MAX, etc.) provide yet another means of extracting information from the database.

Data manipulation involves changing the information content of the database. This is achieved by using a set of operators (DELETE, CHANGE, etc.) that enable data to be inserted, deleted and modified. Retrieval of data by using queries is a much less common operation in engineering design databases. Most design data is communicated among design team members in forms that are not directly processable by computer. However, with the continued development of databases that accurately reflect the integration of design processes some of these means of communication can be replaced with direct query access to the database. For example, a structural engineer could determine HVAC unit roof loads from the database using a query of the form:

```
SELECT  weight  FROM  mechanical unit relation
        WHERE  location  EQ  roof.
```

Schaefer developed a tutorial of similar structural engineering examples to describe all of the principal relational operations for data retrieval.

For engineering applications, it is of even greater importance to be able to interface a wide variety of application programs to the database allowing program input data to be retrieved from the database and program output to be stored in the database. To do so, the programs must have embedded within them a set of commands to operate on the database: opening it, locating the appropriate relations, storing or retrieving data, and closing it. These commands are implemented as DBMS subroutines external to the application program, and they are invoked by subroutine calls, written in the host programming language, that are embedded directly in the application program.

SUMMARY

This paper presented the use of a relational database management system for representing building design data and showed that relationships among topology, geometry, and attributes can be readily modeled relationally. The feasability of interaction between engineering applications and the database was indicated and relational database management systems were demonstrated to be viable engineering tools. Although the paper concentrated on structural design data for buildings, it is clear that the model can also be used to represent the data of other engineering design disciplines as well as the data for other civil engineering structures.

ACKNOWLEDGEMENTS

The MicroRIM relational database management system was used to implement the database presented herein. The author expresses his gratitude to MicroRIM for their contribution to this study. MicroRIM is the microcomputer version of the RIM DBMS originally developed for NASA's Space Schuttle program. The mainframe version, BCS RIM, is currently available from Boeing Computer Services Co. Two advantages of the RIM family of databases that make them particularly useful in engineering applications are their support of real data types and their portability - MicroRIM and BCS RIM are compatible, allowing data to be transferred between micro and mainframe computers.

This work was sponsored in part by the North Carolina State University under grant 00847 entitled "Relational Database Representation for Industrialized Building Design and Construction".

REFERENCES

American Concrete Institute, Building Code Requirements for Reinforced Concrete (ACI 318-77), American Concrete Institute (ACI), Detroit, MI, 1977.

Boeing Computer Services Co., BCS RIM Relational Information Management System Version 6.0 User's Guide, Boeing Computer Services Co. (BCS), P. O. Box 24346, Seattle, WA, July, 1983.

Burner, B., Ives, F., Lixvar, J., and Shovlin, D., "The Design, Evaluation, and Implementation of the IPAD Distributed Computing System", Proceedings of the First Conference on Computing in Civil Engineering, Pages 126-144, American Society of Civil Engineers, New York, NY, June, 1978.

Date, C. J., An Introduction to Database Systems, Addison Wesley, Reading, MA, 1981.

Erickson, W. J., Gray, F. P., Limbach, G., Relational Information Management System, Version 5.0, Boeing Commercial Airplane Company, P.O. Box 3707, Seattle, WA 1981.

Fenves, S. J. and Rasdorf, W. J., "Role of Database Management Systems in Structural Design," Proceedings of the IABSE Collogium on Informatics in Structural Engineering, International Association of Bridge and Structural Engineers, October, 1982.

Fulton, R. E., "Overview of Integrated Programs for Aerospace - Vehicle Design (IPAD)," NASA Technical Memorandum 81874, National Aeronautics and Space Administration, Langley Research Center, Hampton, VA, September, 1980.

MicroRIM, <u>MicroRIM Data Base Management System User's Manual</u>, MicroRIM Inc., P. O. Box 585, Bellevue, WA, July, 1982.

Rasdorf, W. J. and Fenves, S. J., "Organization of a Structural Design Database," <u>Proceedings of the Eighth Conference on Electronic Computation</u>, Pages 559-571, American Society of Civil Engineers, February, 1983.

Rasdorf, W. J. and Kutay, A. R., "Maintenance of Integrity During Concurrent Access in a Building Design Database," <u>Computer Aided Design</u>, Volume 14, Number 4, Pages 201-207, July, 1982.

Sandberg, G., "A Primer on Relational Database Concepts," <u>IBM Systems Journal</u>, Volume 20, Number 1, Pages 23-40, 1981.

Schaefer, M. J., Rehak, D. R., Fenves, S. J., "A Tutorial on Relational Databases Using Structural Engineering Examples," Technical Report DRC-12-09-82, Design Research Center, Carnegie-Mellon University, Pittsburgh, PA, December, 1982.