APPLICATION OF A DATA BASE MANAGEMENT SYSTEM TO A FINITE ELEMENT MODEL

James L. Rogers, Jr.
Langley Research Center

INTRODUCTION

In today's software market, much effort is being expended on the development of data base management systems (DBMS). Most commercially available DBMS were designed for business use. However, the need for such systems within the engineering and scientific communities is becoming apparent.

A potential DBMS application that appears attractive is the handling of data for finite element engineering models. The purpose of this paper is to explore the application of a commercially available, business-oriented DBMS to a structural engineering, finite element model. The model, DBMS, an approach to using the DBMS, advantages and disadvantages are explored in detail; and plans for research on a scientific and engineering DBMS are discussed.

THE FINITE ELEMENT MODEL

Many organizations use more than one finite element computer program for analyzing structural models. For instance NASTRAN might be used because of its generality, standardization, and universal acceptance, or SPAR might be used to gain quick turnaround and interactive capability. Input into these programs usually consists of data representing a finite element model. Typically, the model has a single physical representation which can easily be pictured as a hierarchial structure composed of the complete model, substructures, elements, and grid points (see fig. 1). This single representation, however, can take the form of input to any one or more of the many finite element programs available today.

If an engineer wishes to use more than one application program for analysis, a separate input deck must be maintained for each program, although in truth only one model is being analyzed. This proliferation of input decks can lead to a loss of integrity in the data, because of the difficulty in keeping all related input decks at the same level of modification -- especially when many changes are being made to the model. To solve the proliferation problem and maintain integrity of the data, the finite element model input data can be stored in a data base.

THE DATA BASE MANAGEMENT SYSTEM[1]

Storing the model input data in a data base independent of all the application programs that are to make use of it as input can solve the problem of proliferation of input decks and aid in maintaining the integrity of the model. The data base is loaded into the DBMS in the steps shown in figure 2. Step 1 simply defines a data base name for the user to work with. The model input data must now be stored in data elements, the basic components of a data base. By using the data base definition the hierarchial structure of the finite element model is easily adapted to the hierarchial structure of the data base. A typical data base definition is shown in figure 3. Particular attention should be paid to its generality and its hierarchial arrangement. Most any finite element model, large or small, can be input with this definition. The hierarchial structure can be seen by scanning the indentation of the definition. Data element 1 allows input of a model name. Data element 2 is a repeating group (RG) of substructures. A repeating group allows more than one substructure per model to be input. Data element 3 allows input of a substructure name. Data element 4 is a repeating group of finite elements within a substructure. Data elements 5-10 provide for input of element information. Data element 11 is a repeating group of grid points within a finite element. Data elements 12-16 provide for input of grid point information. Data elements 20-49 define supporting information (not necessarily hierarchial) about the model such as element properties, single point constraints, omitted coordinates, and eigenvalue information. Step 3 (fig. 2) inputs the finite element model into the data base through the data base definition.

After the data has been loaded into the data base, it cannot be used in an application program without undergoing some sort of conversion. The easiest approach when working with already existing programs, is to use FORTRAN pre-processor programs to convert the data from the independent data base format to an input format for a particular application program such as NASTRAN or SPAR (see fig. 4). Pre-processor programs for converting data base input into NASTRAN and SPAR input decks are shown in Appendices I and II, respectively. A different pre-processor is needed for each application program that will use the model data as input. A data manipulation language (DML) which can be embedded within the pre-processor programs allows the user to write a FORTRAN program that will interact with the data base. The DML statements of the programs in the appendices have *PL in columns 1-3.

---

[1] SYSTEM 2000, a commercially available data base management system developed by MRI Systems Corporation, was used for all research done for this paper. However, almost any DBMS with a FORTRAN procedural language and query language could have been used. Use of commercial products and names of manufacturers in this report does not constitute an official endorsement of such products or manufacturers either expressed or implied, by the National Aeronautics and Space Administration.

Another powerful tool of some DBMS is the query language. This language allows the user to interactively query the data base. Using this language, he can list the grid points associated with a particular finite element or vice-versa he can list the finite elements associated with a particular grid point. Many other associations can be listed depending upon the relationships defined in the data base definition.

This has only been a brief introduction into the world of DBMS. More details concerning the data base definition, data base input, DML, query language, and creation of a data base, can be obtained from manuals available from the developer of a specific DBMS.

APPLILCATION OF A DBMS TO A FINITE ELEMENT CANTILEVER BEAM

The data base input of a cantilever beam is shown in figure 5. Close attention should be paid to the correlation of numbers between figure 3 (data base definition) and figure 5 (data base input). To demonstrate the process shown in figure 4, a cantilever beam finite element model is created (see fig. 6). The data base definition and data base input described above and shown in figures 3 and 5 are used in this demonstration. After the data is stored in the data base, it is input into the pre-processor programs shown in appendices. Other input that does not pertain to the model but is necessary input to NASTRAN and SPAR (e.g., Executive and Case Control decks of NASTRAN) are read in from another file and inserted into the proper place in the output from the pre-processor programs. The pre-processor program for NASTRAN is not set up to handle PARAM and CNGRNT Bulk Data cards, thus these cards are also read from the same file as the Executive and Case Control cards. The pre-processor programs could be expanded and made sufficiently general to handle almost any input requirement. The pre-processor programs output a file ready for input into either NASTRAN (fig. 7) or SPAR (fig. 8). Thus it is shown that a finite element model data can be stored in a data base, independent of any application program which would use the model data as input. With the aid of pre-processor FORTRAN programs, the model data can subsequently be converted to a format for direct input into one of several application programs.

ADVANTAGES AND DISADVANTAGES

The advantages in using a data base to store a finite element model have been discussed in detail in previous sections so only a summary appears in this section. The data base requires only one copy of the model to be stored even though the data may be used as input to more than one application program. This situation aids in maintaining the integrity of the model, especially a model undergoing frequent changes. The stored data are independent of the application programs that use it, thus changes can be made to the programs and/or the data without having cross effects. A query language in the DBMS gives the user easy interactive access to the data, allowing listings of many types of data elements and cross-references between data elements.

The use of a DBMS is not without disadvantages. Of primary importance is the computer storage overhead (disk space) required to store a model. The small cantilever beam problem described earlier took almost $11{,}000_{10}$ words of storage. Changing the NON-KEY values to KEY in the data base definition results in the data base requiring $23{,}500_{10}$ words of storage. (A KEY value permits the user to make sorts on that particular data element.) The user must judiciously choose which data elements are KEY to minimize storage overhead. Another disadvantage is that most business-oriented DBMS do not permit "E" formats (scientific notation). Since many structural terms involve very large or very small numbers, the "E" format is essential. One final disadvantage is the lack of support for matrix or vector input to the data base. Although this capability is not required to store finite element model data, it would be particularly useful when storing and querying intermediate or final results from the application programs.

These are the major advantages and disadvantages revealed during the course of the research. The user will have to make trade-offs in determining the most optimal way to use a DBMS with finite element models and computer programs.

PLANS FOR DBMS WORK WITH FINITE ELEMENT MODELS

New data base management systems (DBMS) are now being designed and developed for engineering and scientific applications. For this reason, as well as the previously discussed disadvantages, no future research is planned using a commercial business-oriented DBMS with a finite element model. Future research is planned using E/S DMS, a scientific and engineering oriented DBMS being developed under a NASA grant at the University of Texas. E/S DMS is scheduled to be delivered to NASA Langley in August 1979. It executes on CDC CYBER computers with a NOS operating system. It has been proposed to implement E/S DMS on a minicomputer allowing users to distribute the data base as well as the processing between a mainframe and a minicomputer. This arrangement would give the user a very powerful tool; because the design and modeling could then be done on the faster (line speed) interactive graphics terminal, while the "number crunching" applications programs such as NASTRAN and SPAR could be executed on the more powerful mainframe. Future research is planned along these lines.

CONCLUDING REMARKS

A structural engineering, finite element model can be represented as a hierarchial structure within a business-oriented data base management system (DBMS). Use of the DBMS and conversion pre-processor computer programs, allow the data to be stored independent of application programs that use the data as input. This arrangement has the advantage that only one copy of the data needs to be stored and maintained even though more than one application program can use the data as input. The DBMS may also allow the user to query the data base as an aid in his modeling. The data manipulation language of the DBMS gives the application programmer the power and flexibility of FORTRAN combined

with the data storage capabilities of the data base. A small example problem demonstrates the use of a data base with one specific finite element model. Output of the conversion programs is shown in the form of input decks to both the NASTRAN and SPAR structural analysis application programs. Advantages and disadvantages are described to show that while although a DBMS created for business-oriented data can be useful for handling a finite element engineering model, it does not have all the capabilities required for engineering and scientific computing.

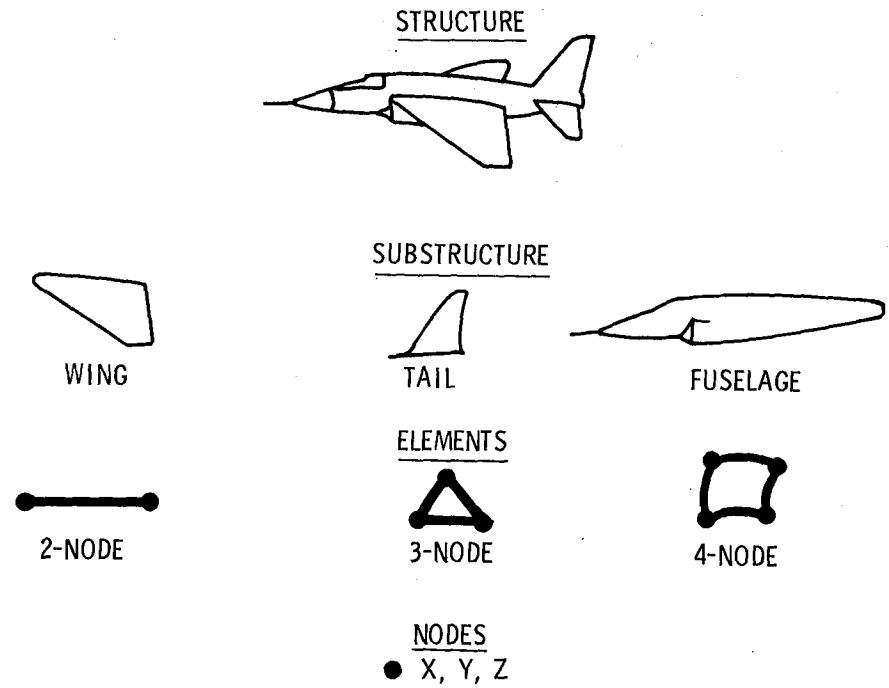

Figure 1.- Hierarchial structure of a finite element model.

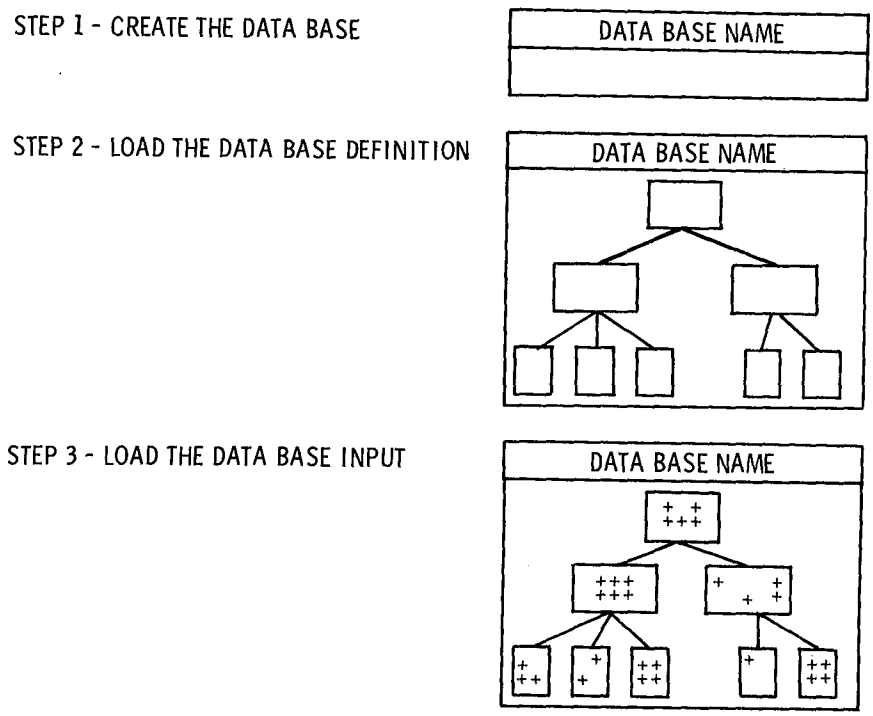

Figure 2.- Steps for loading a data base.

```
1* MODEL (NAME X(21));
2* SUBSTRUCTURES(RG);
  3* SUBSTRUCTURE NAME (NAME X(10) IN 2);
  4* SUBSTRUCTURE ELEMENTS(RG IN 2);
    5* ELEMENT ID (INTEGER NUMBER 9(7) IN 4);
    6* ELEMENT NAME (NON-KEY NAME X(7) IN 4);
    7* X1 VECTOR COMPONENT (NON-KEY DECIMAL NUMBER IN 4);
    8* X2 VECTOR COMPONENT (NON-KEY DECIMAL NUMBER IN 4);
    9* X3 VECTOR COMPONENT (NON-KEY DECIMAL NUMBER IN 4);
    10* ELEMENT PROPERTY ID (INTEGER NUMBER 9(7) IN 4);
    11* ELEMENT GRID POINTS (RG IN 4);
       12* GRID POINT ID (INTEGER NUMBER 9(7) IN 11);
       13* X COORDINATE (NON-KEY DECIMAL NUMBER IN 11);
       14* Y COORDINATE (NON-KEY DECIMAL NUMBER IN 11);
       15* Z COORDINATE (NON-KEY DECIMAL NUMBER IN 11);
       16* PERMANENT SPC (NON-KEY INTEGER NUMBER 9(7) IN 11);
  20* ELEMENT PROPERTIES (RG IN 2);
    21* PROPERTY ID (INTEGER NUMBER 9(7) IN 20);
    22* MATERIAL ID (NON-KEY INTEGER NUMBER 9(7) IN 20);
    23* AREA (NON-KEY NAME X(7) IN 20);
    24* MOMENTS OF INERTIA (NON-KEY NAME X(7) IN 20);
    25* YOUNGS MODULUS (NON-KEY NAME X(7) IN 20);
    26* SHEAR MODULUS (NON-KEY DECIMAL NUMBER IN 20);
    27* POISSONS RATIO (NON-KEY DECIMAL NUMBER IN 20);
    28* MASS DENSITY (NON-KEY DECIMAL NUMBER IN 20);
    29* PROPERTY NAME (NON-KEY NAME X(7) IN 20);
  35* SINGLE POINT CONSTRAINTS(RG IN 2);
    36* SPC SET ID (NON-KEY INTEGER NUMBER 9(7) IN 35);
    37* SPC GRID ID (NON-KEY INTEGER NUMBER 9(7) IN 35);
    38* SPC COMPONENT NUMBER (NON-KEY INTEGER NUMBER 9(7) IN 35);
  39* OMITTED COORDINATES(RG IN 2);
    40* OMIT GRID ID (NON-KEY INTEGER NUMBER 9(7) IN 39);
    41* OMIT COMPONENT NUMBER (NON-KEY INTEGER NUMBER 9(7) IN 39);
  42* EIGENVALUE INFORMATION(RG IN 2);
    43* EIGENVALUE SET ID (NON-KEY INTEGER NUMBER 9(7) IN 42);
    44* METHOD (NON-KEY NAME X(7) IN 42);
    45* LO FREQ (NON-KEY DECIMAL NUMBER IN 42);
    46* HI FREQ (NON-KEY DECIMAL NUMBER IN 42);
    47* ESTIMATE OF ROOTS (NON-KEY INTEGER NUMBER 9(7) IN 42);
    48* DESIRED ROOTS (NON-KEY INTEGER NUMBER 9(7) IN 42);
    49* NORMALIZING METHOD (NON-KEY NAME X(7) IN 42);
```

Figure 3.- Data base definition for a finite element model.

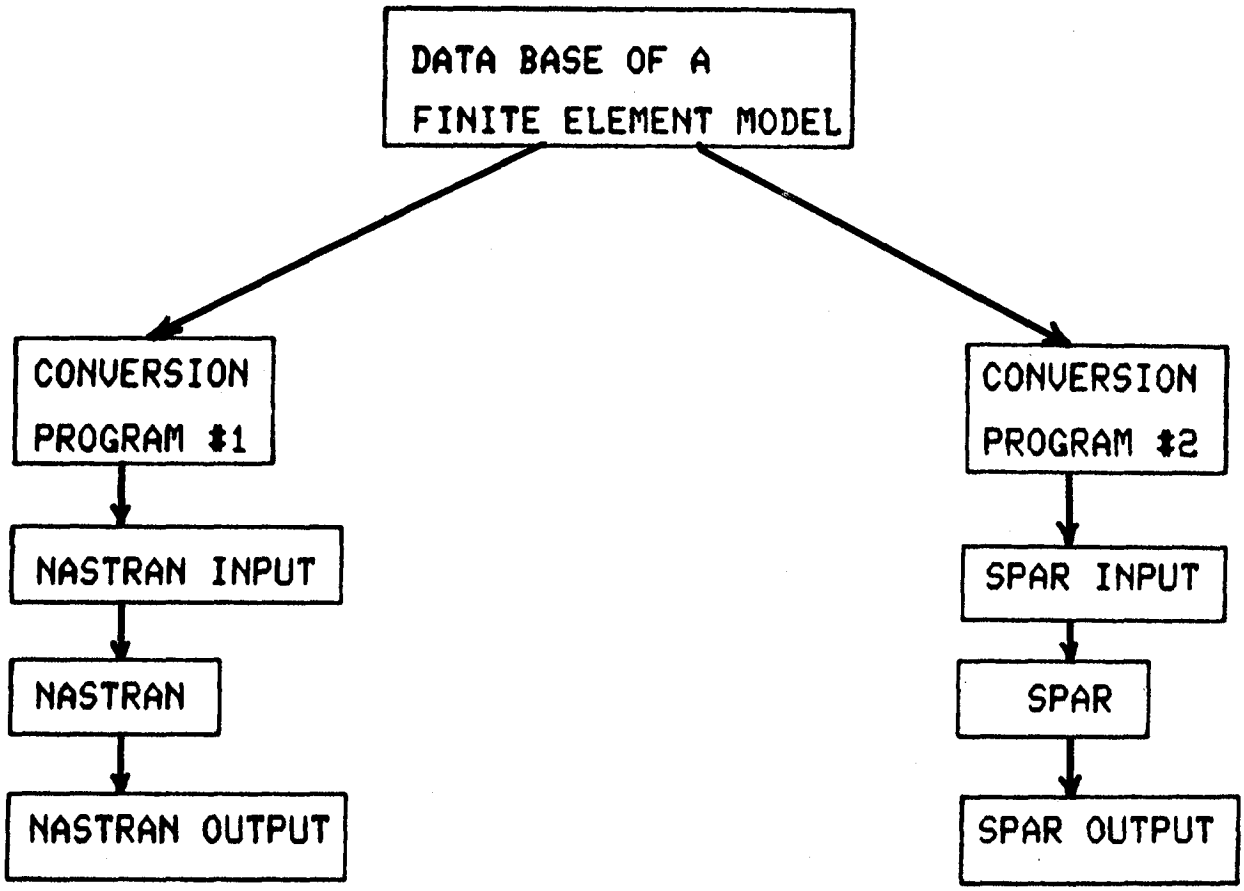

Figure 4.- Flowchart of finite element data base process.

```
1*DATA BASE TEST*
2*3*CANT BEAM*
 4*5*11*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*1*    13*0.0*  16*1345*
 11*12*2*
4*5*12*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*2*  13*0.1*  16*1345*
 11*12*3*
4*5*13*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*3*    13*0.2*  16*1345*
 11*12*4*
4*5*14*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*4*     13*0.3*  16*1345*
 11*12*5*
4*5*15*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*5*    13*0.4*  16*1345*
 11*12*6*
4*5*16*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*6*    13*0.5*  16*1345*
 11*12*7*
4*5*17*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*7*    13*0.6*  16*1345*
 11*12*8*
4*5*18*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*8*    13*0.7*  16*1345*
 11*12*9*
4*5*19*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*9*    13*0.8*  16*1345*
 11*12*10*
4*5*20*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*10*  13*0.9*  16*1345*
 11*12*11*
4*5*21*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*11*  13*1.0*  16*1345*
 11*12*12*
4*5*22*  6*CBAR*  7*0.*  8*1.*  9*0.*  10*201*
 11*12*12*  13*1.1*  16*1345*
 11*12*13*  13*1.2*  16*1345*
20*21*201*
   22*6*
   23*6.0-4*
   24*5.0-9*
   25*7.0+10*
   27*.3*
   28*86*
   29*PBAR*
35*36*20*
   37*1*
   38*26*
39*40*13*     41*6*
39*40*2*      41*6*
39*40*3*      41*6*
39*40*4*      41*6*
39*40*5*      41*6*
39*40*6*      41*6*
39*40*7*      41*6*
39*40*8*      41*6*
39*40*9*      41*6*
39*40*10*     41*6*
39*40*11*     41*6*
39*40*12*     41*6*
42*43*12*
   44*INV*
   45*0.0*
   46*500.*
   47*3*
   48*3*
   49*MAX*
END**
```

Figure 5.- Data base input for a cantilever beam.

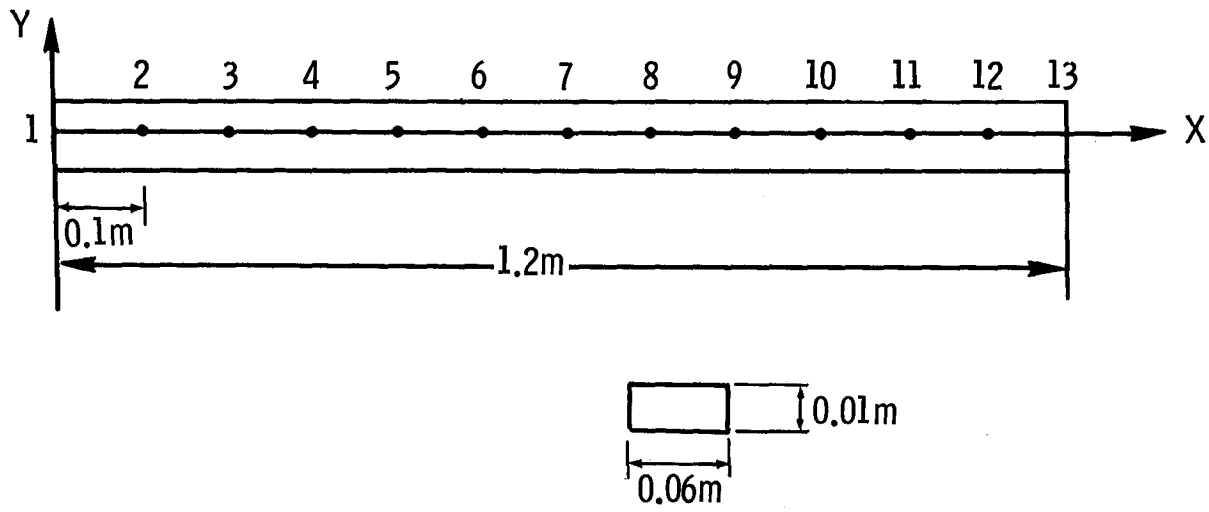

Figure 6.- Cantilever beam finite element model.

```
ID DATABASE,TEST
APP DISP
SOL 3,0
TIME 4
CEND
TITLE = TEST PROBLEM FOR DATA BASE(NASTRAN CONVERSION)
SUBTITLE = VIBRATION ANALYSIS OF A CANTILEVER BEAM
  LINE = 40
  ECHO = BOTH
  METHOD = 12
  SPC = 20
OUTPUT
  DISP = ALL
BEGIN BULK
CBAR            11      201       1       2      0.0     1.0     0.0     1
CBAR            12      201       2       3      0.0     1.0     0.0     1
CBAR            13      201       3       4      0.0     1.0     0.0     1
CBAR            14      201       4       5      0.0     1.0     0.0     1
CBAR            15      201       5       6      0.0     1.0     0.0     1
CBAR            16      201       6       7      0.0     1.0     0.0     1
CBAR            17      201       7       8      0.0     1.0     0.0     1
CBAR            18      201       8       9      0.0     1.0     0.0     1
CBAR            19      201       9      10      0.0     1.0     0.0     1
CBAR            20      201      10      11      0.0     1.0     0.0     1
CBAR            21      201      11      12      0.0     1.0     0.0     1
CBAR            22      201      12      13      0.0     1.0     0.0     1
GRID             1              0.00                                1345
GRID             2              .10                                 1345
GRID             3              .20                                 1345
GRID             4              .30                                 1345
GRID             5              .40                                 1345
GRID             6              .50                                 1345
GRID             7              .60                                 1345
GRID             8              .70                                 1345
GRID             9              .80                                 1345
GRID            10              .90                                 1345
GRID            11             1.00                                 1345
GRID            12             1.10                                 1345
GRID            13             1.20                                 1345
PBAR           201        6    6.0-4   5.0-9
MAT1             6  7.0+10              0.3      .86
OMIT            13        6
OMIT             2        6
OMIT             3        6
OMIT             4        6
OMIT             5        6
OMIT             6        6
OMIT             7        6
OMIT             8        6
OMIT             9        6
OMIT            10        6
OMIT            11        6
OMIT            12        6
SPC             20        1      26
EIGR            121NV            0.0      500.0      3       3                +BC
+BC     MAX
PARAM   COUPMASS        1
CNGRNT          11      12THRU           22
ENDDATA
```

Figure 7.- NASTRAN input deck output from pre-processor program NTRNS2K.

```
[XQT TAB
START 13,2 6
TITLE  " VIBRATION ANALYSIS OF A CANTILEVER BEAM
TEXT
" TEST PROBLEM FOR DATA BASE(SPAR CONVERSION)
MATERIAL CONSTANT
 1 7.0+10   .30   .86
JOINT LOCATIONS
 1    0.0
 2     .1
 3     .2
 4     .3
 5     .4
 6     .5
 7     .6
 8     .7
 9     .8
 10    .9
 11   1.0
 12   1.1
 13   1.2
CONSTRAINT CASE 1
 ZERO 1 3 4 5
[XQT ELD
E23
NSECT=1
 1    2
 2    3
 3    4
 4    5
 5    6
 6    7
 7    8
 8    9
 9    10
 10   11
 11   12
 12   13
[XQT EXIT
```

Figure 8.- SPAR input deck output from pre-processor program SPARS2K.