To: Distribution

From: R. D. Lackey

Date: 07/28/78

Subject: Multics Relational Database Restructuring

Introduction

This document provides an overview of the proposed Restructuring Facility for the Multics Relational Database Manager. It describes the functions to be provided along with the user interface.

Scope

The MDBM Restructuring Facility provides the data base administrator with a method to:

1) Define and undefine files, relations, domains, attributes, indexes and foreign keys in a populated MRDS data base.

2) Redefine domains, attributes, relations, files and foreign keys in populated MRDS data bases.

Description

Restructuring is defined as the modification of the logical definition of the data base. In the populated data base environment this means changing the data model's definition of the *data* base and the contents of the data base to coincide with the new data model.

Restructuring may include the deletion of elements from the data base, the addition of elements to the data base or the redefinition of existing data base elements including reordering elements.

Undefinition of Elements

Elements that may be undefined are: domains, attributes, indexes, relations, files and foreign keys. The undefinition restructuring process not only involves the removal of the elements definition from the existing data model, but also includes the deletion of the respective data items. A non-existent element cannot be undefined.

The undefinition of a domain may affect attributes in more then one relation. An attribute will not be undefined until all its foreign key references have been undefined. When an attribute is undefined any associated indexes will also be removed. A relation will not be undefined until all of its attributes have been undefined. A file will not be undefined until all of its contained relations have been undefined.

Definition of Elements

Elements that may be defined are: domains, attributes, indexes, relations, files and foreign keys. An element cannot be defined if an element of the same type and name exists. Element definitions for restructuring have the same form as element definitions for the create_mrds_db command.

Redefinition of Elements

Elements that may be redefined are: domains, attributes, relations, files and foreign keys. Only elements that exist in the data model being restructured can be redefined. Redefinition is defined as redefining an element that exists in the data base being restructured. The redefinition process will be done without loss of specified data items in a populated data base.

For the redefinition of a relation, the relation must exist in the data base being restructured. All attributes associated with the redefined relation must be defined and any attributes

that are not carried over to the redefined relation must be explicitly deleted with an undefine statement. The types of redefinition allowed for relations include: changing the relation name, adding an attribute, deleting a nonkey attribute, reordering of attributes.

For the redefinition of a file, the file must exist in the data base being restructured and all relations to be associated with the file must have existed in the data base being restructured.

User Interface Description

Name: restructure_mrds_db, rmdb

This command restructures an existing populated MRDS data base using the directives supplied in the restructuring source segment. The existing data base is restructured in place and the resulting data base is ready for use. (It should be noted that programs that ran against the old data based and have not been modified to coincide with the new data model may experience errors.) This command is intended for use primarily by data base administrators but may be used by individuals restructuring their private data bases. In any case, this command should be used with great discretion and only by someone with an intimate knowledge of the data base being restructured and its associated programs.

Usage

restructure_mrds_db source_path database_path {-control_args}

where:

1. source_path
   is the pathname of a restructuring source segment. If source_path does not have the suffix of rmdb, then one is assumed. However, the rmdb suffix must be the last component of the name of the source segment.

2. database_path
   is the pathname of the data base to be restructured.

3. control_args
   may be chosen from the following:

   -list, -ls
   produces a segment containing a listing of the restructuring source segment, followed by detailed information about the results of the restructured data base. This segment is created in the working directory and has the same name as the source_path with a list (rather than rmdb) suffix.

   -check, -ck
   prevents any changes made to the data base but verifies that the restructuring directives have been correctly specified.

-intermediate_storage_sw <switch_name>, -iss <switch_name>
specifies the switch name of a predefined switch to be used for intermediate storage during the restructuring process. If not specified then the users process directory is used for intermediate storage.

-brief, -bf
inhibits the output of information about the progress of the restructuring progress.

Notes

Error messages are written to error_output I/O switch as they occur. They are also included in the listing segment if one is produced.

If any errors are detected during the analysis of the source segment, no restructuring is done.

If no errors are detected during the analysis of the source segment the physical restructuring of the data base is done to make it comply with the new model definitions. Information about the progress of the restructuring will be written to user_output I/O switch as the intermediate phase of the restructuring is completed. A successful execution of this command leaves the data base in usable state, however, relations that have had attributes added may contain "null" attrubutes and any new relations or files added to the data base will be unpopulated.

Restructuring Source

The restructuring source segment is a text segment containing the restructuring directives. The valid directives are define, undefine and redefine and must appear in this order. Comments appear in the source text in the same manner that they appear in PL/I programs (delimited by /* and */). Each directive section in the source segment begins with a directive name followed by a colon and at least one blank followed by one or more directive statements. Each directive statement begins with a directive statement name and terminates with a semicolon. The directive section is terminated by either the beginning of a new directive section or then end of segment.

UNDEFINE DIRECTIVE

The undefine directive specifies items that exits in the data base to be undefined from the data base being restructured. The items being undefined must be supplied in the order specified.

The format of the undefine directive is:

undefine: [domain_stmt] [attr_stmt] [relation_stmt] [file_stmt] [index_stmt] [foreign_key_stmt]

where:

1. domain_stmt
   specifies the domains to be undefined from the data base and has the format:

   domain: domain_name1 . . . domain_namen;

2. attr_stmt
   specifies the attributes to be undefined from the data base and has the format:

   attribute: attr_name1 . . . attr_namen;

3. relation_stmt
   specifies the relations to be undefined from the data base and has the format:

   relation: rel_name1 . . . rel_namen;

4. file_stmt
   specifies the files to be undefined from the data base and has the format:

   file: file_name1 . . . file_namen;

5. foreign_key_stmt
   specifies the foreign keys to be defined to the data base and has the format:

   foreign_key: fk_spec1[, ..., fk_specn];

   where fk_spec has the format:

   link_name prel_name (pattr_name [pattr_name ...]) crel_name (cattr_name [cattr_name ...]) [-cluster]

6. index_stmt
specifies the indexes to be defined to the data base and has the format:

index: index_spec1 . . . index_specn;

where index_spec has the format:

relation_name (attr_name1 . . . attr_namen)

DEFINE DIRECTIVE

The define directive specifies new items to be defined for data base being restructured. The items being defined must be supplied in the order specified.

The format of the define directive is:

define: [domain_stmt] [attr_stmt] [rel_stmt] [file_stmt] [index_stmt] [foreign_key_stmt]

where:

1. domain_stmt
   specifies the domains to be defined to the data base and has the format:

   domain: domain_name1 declaration1[, ..., domain_namen declarationn];

2. attr_stmt
   specifies the attribute to be defined to the data base and has the format:

   attribute: attr_name1 domain_name1[,...., attr_namen domain_namen];

3. rel_stmt
   specifies the relations to be defined to the data base and has the format:

   relation: rel_exp1[, ..., rel_expn];

   and rel_expi has the format:

   relation_name (attribute1* attribute2* ... attributen)

4. file_stmt
specifies the files to be defined to the data base and has the format:

file: file_spec1[, ..., file_specn];

where file_spec has the format:

file_name (rel_name [rel_name....]) [-blocked [n] [h/b] [-unblocked]

where file_name is the name of the file being defined, rel_name is the name of a relation residing in the file. If -block is specified, n is the number of Multics pages per block, and h is the number of hash bucket headers per block b. It is possible to specify multiple headers per block (h > 1, b = 1) or one header for several blocks (h = 1, b > 1). The default is h = b = 1. The default for block size is n = 1. If -unblocked is specified, then the file is defined to be an unblocked file. In this case, only one rel_name may be designated. The -blocked and -unblocked arguments are mutually exclusive. If either is specified, the default file type is determined as follows. If multiple rel_names are present, the file will be blocked with the default values for n, h and b. If only one rel_name is present, the file will be unblocked.

5. foreign_key_stmt
specifies the foreign keys to be defined to the data base and has the format:

foreign_key: fk_spec1[, ..., fk_specn];

where fk_spec has the format:

link_name prel_name (pattr_name [pattr_name ...]) crel_name (cattr_name [cattr_name ...]) [-cluster]

6. index_stmt
specifies the indexes to be defined to the data base and has the format:

index: index_spec1 . . . index_specn;

where index_spec has the format:

relation_name (attr_name1 . . . attr_namen)

Note:
The domain, attr, index, relation, file, foreign_key statements in the define directive are identical to the associated statements in the create_mrds_db command.

REDEFINE DIRECTIVE

The redefine directive specifies items to be redefined to the data base being restructured. The item must exist in the data base being restructured with the specified name. The items being redefined must be supplied in the order specified.

The format of the redefine directive is:

```
redefine:    [domain_stmt]    [attr_stmt]    [index_stmt]
       [rel_stmt]  [file_stmt]  [foreign_key_stmt]
```

where:

1. domain_stmt
   specifies the domains to be redefined to the data base and has the format:

   ```
   domain:  domain_name1  declaration1[,  ....,
   domain_namen declarationn];
   ```

2. attr_stmt
   specifies the attribute to be redefined to the data base and has the format:

   ```
   attribute:   attr_name1    domain_name1[,....,
   attr_namen domain_namen];
   ```

3. rel_stmt
   specifies the relations to be redefined to the data base and has the format:

   ```
   relation:  rel_exp1[, ..., rel_expn];
   ```

   and rel_expi has the format:

   ```
   relation_name   (attribute1*   attribute2*   ...
   attributen)
   ```

4. file_stmt
specifies the files to be redefined to the data base and has the format:

file: file_spec1[, ..., file_specn];

where file_spec has the format:

file_name (relation_name [relation_name....]) [-blocked [n] [h/b] [-unblocked]

5. foreign_key_stmt
specifies the foreign keys to be defined to the data base and has the format:

foreign_key: fk_spec1[, ..., fk_specn];

where fk_spec has the format:

link_name prel_name (pattr_name [pattr_name ...]) crel_name (cattr_name [cattr_name ...]) [-cluster]

where file_name is the name of the file being defined, rel_name is the name of a relation residing in the file. If -block is specified, n is the number of Multics pages per block, and h is the number of hash bucket headers per block b. It is possible to specify multiple headers per block (h > 1, b = 1) or one header for several blocks (h = 1, b > 1). The default is h = b = 1. The default for block size is n = 1. If -unblocked is specified, then the file is defined to be an unblocked file. In this case, only one rel_name may be designated. The -blocked and -unblocked arguments are mutually exclusive. If either is specified, the default file type is determined as follows. If multiple rel_names are present, the file will be blocked with the default values for n, h and b. If only one rel_name is present, the file will be unblocked.

Note:
The domain, attr, relation, file, foreign_key statements in the redefine directive are identical to the associated statements in the create_mrds_db command.