INEL-95/0455

User's and Reference Guide to the INEL RML/Analytical Radiochemistry Sample Tracking Database Version 1.00

D. A. Femec

Published September 1995

Idaho National Engineering Laboratory
Nuclear Engineering Department
Lockheed Idaho Technologies Company
Idaho Falls, Idaho 83415-7111

Prepared for the
U.S. Department of Energy
Under DOE Idaho Field Office
Contract DE-AC07-94ID13223

ABSTRACT

This report discusses the sample tracking database in use at the Idaho National Engineering Laboratory (INEL) by the Radiation Measurements Laboratory (RML) and Analytical Radiochemistry. The database was designed in-house to meet the specific needs of the RML and Analytical Radiochemistry. The report consists of two parts, a user's guide and a reference guide. The user's guide presents some of the fundamentals needed by anyone who will be using the database via its user interface. The reference guide describes the design of both the database and the user interface. Briefly mentioned in the reference guide are the code-generating tools, CREATE_SCHEMA and BUILD_SCREEN, written to automatically generate code for the database and its user interface. The appendices contain the input files used by the these tools to create code for the sample tracking database. The output files generated by these tools are also included in the appendices.

CONTENTS

FIGURES

TABLES

1. INTRODUCTION

1.1 Purpose

The need for secure and auditable tracking of samples submitted to the INEL Radiation Measurements Laboratory (RML) and Analytical Radiochemistry for analysis is clear. This need has led to the development of the database and user interface described herein.

1.2 Scope

Version 1.00 of the database represents the essential capabilities necessary for such a database:

- The sample coordinator can track the whereabouts of samples within the RML and Analytical Radiochemistry.

- The sample coordinator can monitor the progress of requested analyses.

- Submitters can provide information on how the requested analyses are to be performed.

- Analysts and operators can obtain basic descriptive information on a sample and the types of analyses to be performed on it.

- Analysts and operators can document the completion of their analyses and the promulgation of the results.

1.3 Future Major Revisions

Major revisions could provide the following additional capabilities:

- The sample coordinator will be able to generate printed versions of any of the screens, including expanded listings such as all possessors of a given sample.

- Submitters will be able to provide more detailed information on how the requested analyses are to be performed.

- Analysts and operators will be able to retrieve and enter pertinent analysis parameters.

- Certain programs used in preparing VAXGAP gamma-ray spectral data files, such as HEAD (see Reference 1), will talk directly with the database.

- RML operators will be able to submit gamma-ray spectra to the appropriate analysis program directly from a user interface.

- Operator-approved results from these analysis programs will be stored in this or a related database.

- Analyst-approved batch analysis results from gamma-ray spectra will be stored in this or a related database.

- Other analysts will be able to enter their analysis results.

- RML Data Analysis and Management QA/QC programs will retrieve results from this or a related database.

These modifications will extend the security and audit safety nets inherent in a database over the entire process of sample-received to analysis-performed to sample-returned-and-results-reported. Other modifications will be implemented as the needs arise and are identified.

1.4 Definitions, Acronyms, and Abbreviations

Terminology related to Digital Equipment Corporation's VMS operating system, DECnet communication protocol, DECwindows Motif graphical user interface, FORTRAN compiler, GKS (Graphical Kernel System) plotting routines, Rdb relational database software, and SQL Fortran precompiler, is desirable. Otherwise, this document explicitly defines technical terms as they are presented and avoids unnecessary technical jargon for the sake of the general reader.

1.5 Registered Trademarks and Copyrights

2. USER'S GUIDE

2.1 Data Entry

The following subsections describe how data is accepted by the user interface before being entered into the database. Some special function keys for simplifying data entry are also presented.

2.1.1 Use Of Uppercase Letters

The user interface converts to uppercase all letters of the alphabet entered by the user. (A query could be a data field to be filled, or a request or question to be answered.) Searches of the database are case-sensitive; this means that searches distinguish between "a" and "A," for example. Forcing all letters to uppercase eliminates the need for a user to remember how a name was entered; for example, "Was that 'Jones' or 'JONES' I entered?"

2.1.2 Striking The Return Key

Striking the return key is the ubiquitous key stroke for terminating an interactive entry of data into a requesting program. Within the user interface, the effect of striking the return key depends on the application. Consider the options request screen, presented upon entering the user interface. After the user chooses an option and enters the corresponding highlighted letter, striking the return key begins execution of that option. Within a data entry screen, striking the return key can move the cursor to the next accessible data field. (An accessible data field is one into which data can be entered; that is, the field is not fixed. See Screen Legend, section 2.2.) During a listing of sample entries retrieved from the database, striking the return key tells the user interface to present the next screen of entries. All of these instances will be demonstrated in more detail in subsequent sections.

2.1.3 Other Keys Of Use During Data Entry

Within a data entry screen, there are other keys that can be used to move the cursor from field to field. These include the arrow keys and some keypad and special function keys. For example, when it is time to move to the next screen during data entry, strike the "Do" special function key (or control-Z). Or, to print an additional bar code label for the sample entry currently displayed, strike the F17 special function key. Figure 1 displays the functional mapping of the arrow and keypad keys and of the special function keys located above them. This figure can be obtained during data entry by striking the "Help" special function key (or keypad key PF2).

The arrow and keypad key mappings are similar to those mappings used in most VMS editors. Note, though, that all of the delete keys (keypad comma for "delete character," keypad dash for "delete word," . . .) erase the entire contents of the current data field; there is no editing-of-an-entry capability. Also, all of the "move-to-the-next" keys (keypad 1 for "next word," keypad 3 for "next character," . . .) move the cursor to the next accessible data field. "Prev Screen" and "Next Screen" move the cursor to the top-most and bottom-most accessible data field of the current screen, respectively.

During cursor movement, any data entry screen acts like a sphere. Suppose the cursor is in the last accessible field of a screen and the user strikes the return key. The cursor can move to the first accessible field of that screen. Or suppose the cursor is on the top-most accessible row of a screen and the user strikes the up-arrow key. The cursor can move to the bottom-most accessible row of that screen.

Help	Next Screen

Print Bar code			

	Ditto Entry	Delete Entry
	Top of Screen	End of Screen
	Move Up	
Move Left	Move Down	Move Right

Abort Program	Help		Delete Entry
Top/End Screen			Delete Entry
Advance	Backup	Delete Entry	Delete Entr
Next Entry	End of Line	Next Entry	Nex Entr
Next Line			

Figure 1. Mapping of Arrow and Keypad Keys and of Some Special Function Keys.

2.2 Screen Legend

The cursor, on most monitors, will appear as a blinking, upright rectangle (■). The cursor tells the user where the next key stroke will be read.

The rendition (or appearance) of a field depends on whether an entry into that data field is optional (regular text) or required (bold text). Fixed fields cannot be changed (reverse video text; presented here as double-underlined). This will be demonstrated in the figures of subsequent subsections. An illegal entry causes the field's regular or bold text to blink (that is, alternate between normal and reverse video) while the monitor beeps.

2.3 User Interface Options

The user starts the user interface by typing STDB (for sample tracking database) at the DCL prompt and then striking the return key. Figure 2 displays the options request screen presented to the user upon entering the user interface. (This screen also appears upon completion of any of the options, except for exiting from the database session.)

The user interface presents six choices:

- Add a Sample Entry

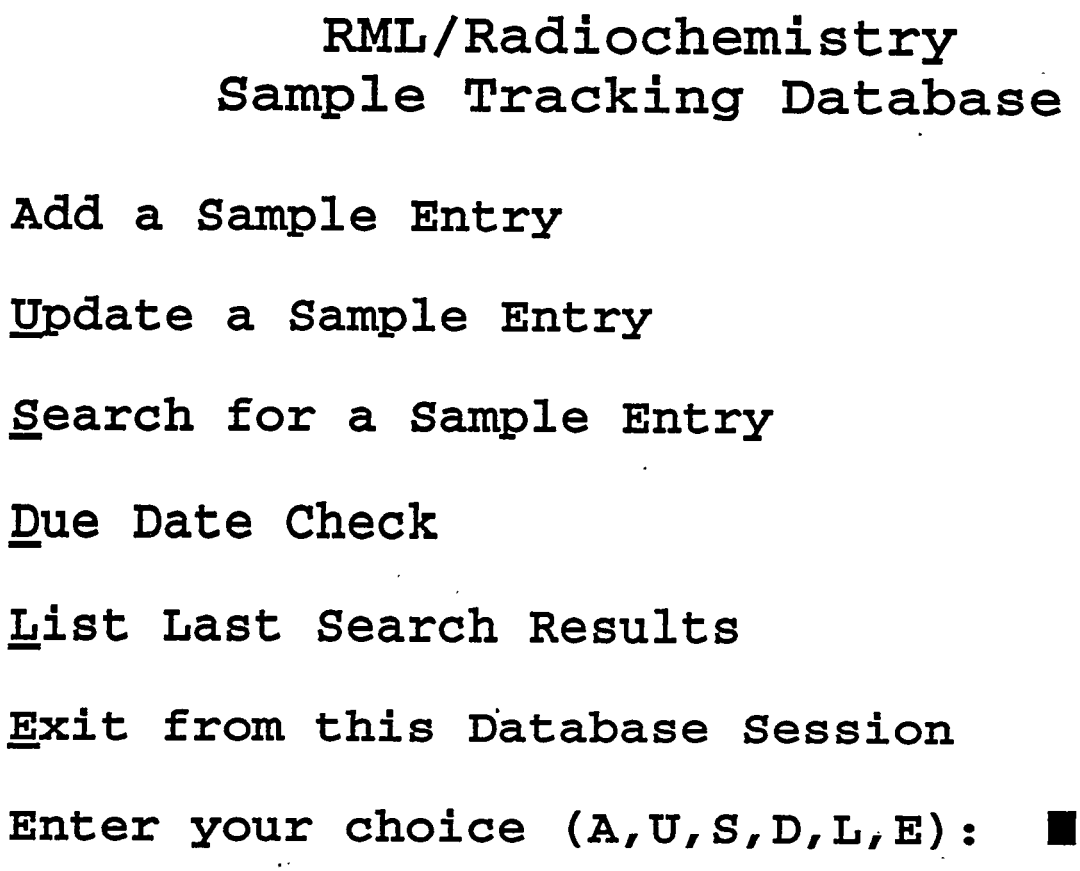

Figure 2. Options Request Screen of the Sample Tracking Database User Interface.

- Update a Sample Entry
- Search for a Sample Entry
- Due Date Check
- List Last Search Results and
- Exit from this Database Session.

The following subsections describe these options.

2.3.1 Add a Sample Entry

The tracking of samples begins with their entry into the database. The sample coordinator should make only one entry for each sample and should enter samples as they arrive. Prior to adding a sample, the following information needs to be at hand:

- who is submitting the sample for analysis
- is there a chain-of-custody form associated with the sample
- what identifier (name or identification code?) does the submitter use to distinguish this sample from any other
- in what form is the sample (soil, water, . . .)
- how large is the sample
- does the sample contain hazardous chemicals
- has the sample been surveyed
- what analyses are to be performed
- how are the analyses to be performed
- who should be contacted in case there are any technical questions regarding the sample or the analyses to be performed
- when are the results due
- to whom are the results to be reported
- what special requirements does the submitter have for the analyses and handling of the sample
- to whom is the sample to be returned and
- which charge number is to be used to pay for the analyses.

The user interface automatically generates a unique sample tracking identification code for each sample entry added to the database. The identification code is twelve digits long, constructed by concatenation of the six-digit integer representations of the current date (mmddyy) and time (hhmmss). Upon successful addition of an entry to the database, the bar code printer prints labels containing the identification code. The code appears literally and in the CODE 39 symbology on the labels. This way the code can be read even if a functioning bar code scanner is not available.

This unique identification code, and the date the sample was entered in the database, cannot be changed by the sample coordinator. The user interface supplies both. These two fields will appear in reverse video on the monitor screen (and double-underlined here). Figure 3 is a depiction of a fresh main screen for adding a sample entry. Note that the cursor is in the data field "Completed" and not in the data field "Tracking ID." Since the "Tracking ID" and "Date Entered" fields cannot be changed, the cursor cannot be moved into them.

Some of the fields in the figure appear in bold text ("Desired Completion Date," "Sample Name," . . .). Recall that these fields must be filled before the user interface will go to the next screen.

The user interface assigns the sample coordinator as the current possessor; the sample coordinator

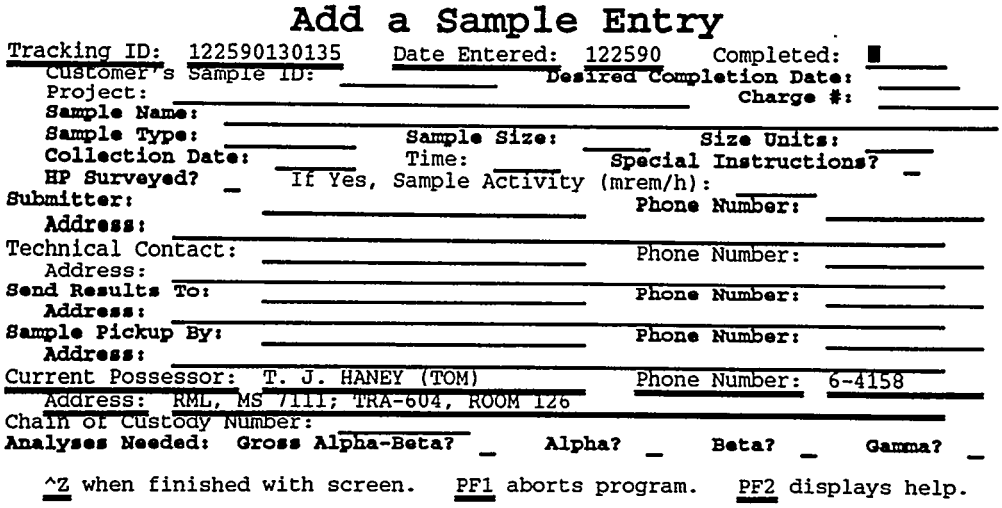

Figure 3. A Fresh Main Screen for Adding a Sample Entry.

is the first possessor of any sample received for analysis. The current possessor cannot be updated in the main screen; it can be updated in the possessor tracking screen, as described in Possessor Tracking, section 2.3.1.2.

The main data screen presents a prime situation to use the "ditto" key. For example, "Customer's Sample ID" dittos to "Sample Name." With the cursor in the "Sample Name" field, depressing the "ditto" key (see Figure 1) copies the contents of the "Customer's Sample ID" field to the "Sample Name" field. Also, the set of "Submitter" fields ditto to the "Technical Contact" fields. These in turn ditto to the "Send Results To" fields, which ditto to the "Sample Pickup By" fields.

If, after adding one sample, the sample coordinator asks to add another, the screens displayed will contain most of the same information entered for the prior sample. In this way the user interface simplifies the entry of a series of related samples. The sample tracking identification code will be different and the date entered may be different. The rest of the information can be changed or left unaltered. What appears on the screens will go into the database.

2.3.1.1 Special Instructions. If the user enters a "Y" into the "Special Instructions?" field, the interface presents the special instructions screen (see Figure 4). The special instructions screen can be thought of as a sub-screen of the main screen. The information contained in it is part of the main screen. The special instruction screen also serves as sub-screens for the specific analysis screens, as will be demonstrated later (Special Instructions, section 2.3.1.3.2).

At the top of the special instructions screen are two fixed fields, "Tracking ID" and "Sample Name." These two fields will appear at the top of all other screens needed for this sample entry. This serves to remind the sample coordinator which sample is being entered.

At this point, the user interface fills all of the "From" fields in the special instruction screen with "Main Screen." The user interface assigns the contents of the available "From" fields based on which

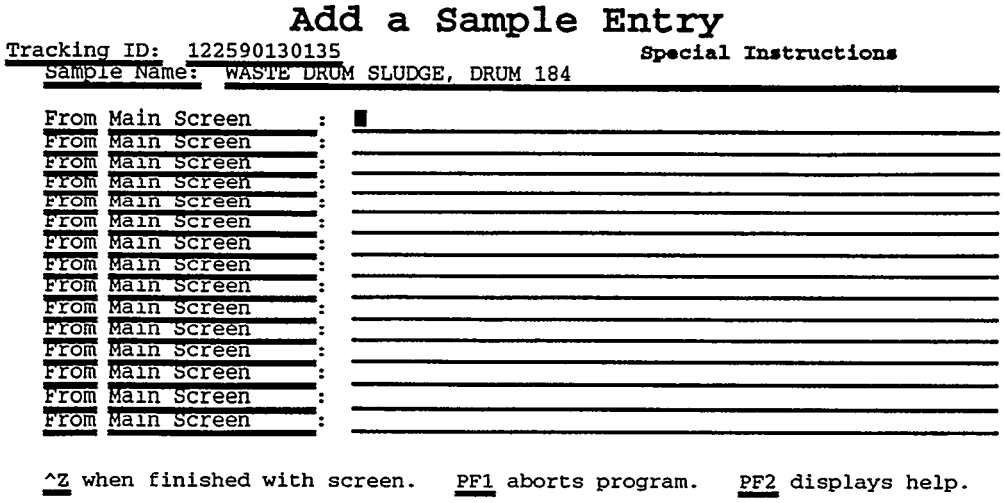

Figure 4. A Fresh Special Instructions Screen.

screen calls for the special instructions screen. The interface also displays all previously entered special instructions; this will be demonstrated later in Special Instructions, section 2.3.1.3.2. The sample coordinator fills the instruction fields as desired. Blank lines between instructions should be avoided. Also, since duplicate lines are undesirable, there are no "ditto" links in the special instructions screen.

2.3.1.2 Possessor Tracking. After the main screen (and the special instructions screen, if needed), the user interface presents the possessor tracking screen. The sample coordinator uses this screen to update the current possessor of the sample. (Please note that this is not a chain of custody. A chain of custody is not required to be maintained within a laboratory performing analyses. The chain of custody must include the sample's arriving at and departing from the laboratory, though.) The user interface presents the current possessor and blanks for two additional possessors (see Figure 5). For samples just received for analyses, the sample coordinator is the first possessor.

The possessors are numbered sequentially. Note that none of the current possessor fields can be changed, except the "Date sample relinquished" field. After the sample coordinator fills this field, the cursor moves to the next set of possessor fields. Since this set of fields is blank, a new possessor needs to be entered. The user interface presents a list of possible possessors (all the analysts/operators within the RML and Analytical Radiochemistry) for the sample coordinator to choose from. This list appears as an overlay to the possessor tracking screen (see Figure 6).

This overlaid list screen is simpler than other data entry screens. The only fields in which entries can be made are the underlined blanks, one following each name. The sample coordinator chooses a name in the list by typing any character in the appropriate blank. If the sample coordinator puts characters in more than one blank, the interface recognizes only the first one. Cursor movement between fields and deletion of entries made in fields operate the same way as with a data entry screen.

If the sample coordinator chooses a listed possessor, the user interface automatically fills the appropriate fields. The interface presents a best guess in the "Reason for possession" field; the sample

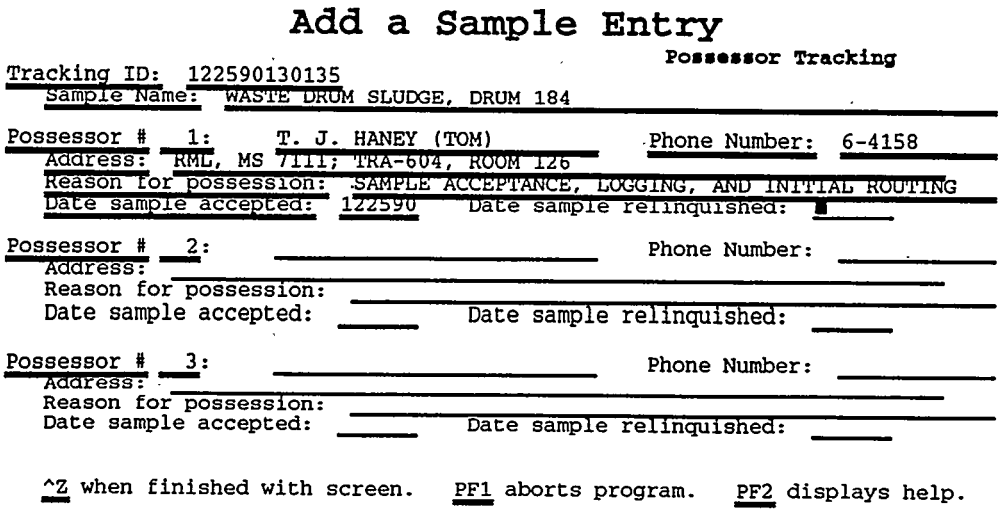

Figure 5. A Fresh Possessor Tracking Screen.

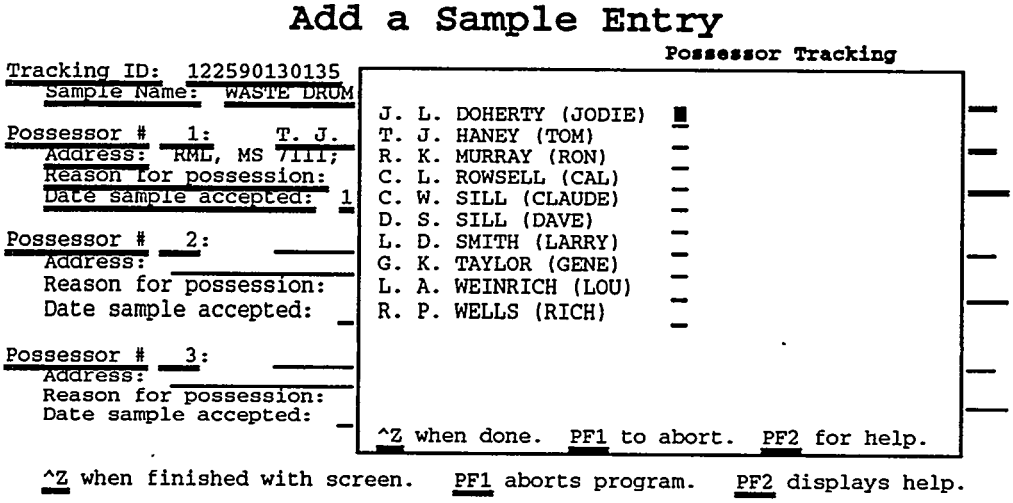

Figure 6. A Possessor Tracking Screen with the Possessor List Overlaid.

coordinator made need to change this. The sample coordinator does not need to choose an item from the list before finishing with the list screen and can instead fill the possessor fields manually.

As already noted, the interface presents two blanks for additional possessors. This allows the sample coordinator to accept the sample from the prior possessor and pass it on to someone else. Only the "Date sample relinquished" fields are "ditto" linked.

2.3.1.3 Analyses Request Screens. The user interface then presents the desired analyses request screens. Here the sample coordinator specifies which analyses are to be performed and any data acquisition requirements. Corresponding fields within an analyses request screen (for example, all "Results report citation" fields) are "ditto" linked.

2.3.1.3.1 Alpha Analyses - Figure 7 presents the alpha analyses request screen. The following alpha analyses are currently available:

- uranium isotopes
- thorium isotopes
- plutonium isotopes
- americium-241 (Am-241) separate from plutonium-238 (Pu-238)
- americium-241 combined with plutonium-238
- total spectrometric alpha and
- other (i.e., special requests).

```
                    Add a Sample Entry
                                                    Alpha Analyses
Tracking ID:  122590130135
  Sample Name:  WASTE DRUM SLUDGE, DRUM 184

Uranium isotopes? _          Needed by: _____     Completed: _____
  Results report citation: _____________________ Special Instructions? _
Thorium isotopes? _          Needed by: _____     Completed: _____
  Results report citation: _____________________ Special Instructions? _
Plutonium isotopes? _        Needed by: _____     Completed: _____
  Results report citation: _____________________ Special Instructions? _
Am-241 separate from Pu-238? _ Needed by: _____   Completed: _____
  Results report citation: _____________________ Special Instructions? _
Am-241 combined with Pu-238? _ Needed by: _____   Completed: _____
  Results report citation: _____________________ Special Instructions? _
Total Spectrometric Alpha? _ Needed by: _____     Completed: _____
  Results report citation: _____________________ Special Instructions? _
Other? _                     Needed by: _____     Completed: _____
  Results report citation: _____________________ Special Instructions? _

  ^Z when finished with screen.   PF1 aborts program.   PF2 displays help.
```

Figure 7. The Alpha Analyses Request Screen.

For these analyses the submitter can specify when the results are needed and any special instructions. The analyst can document the completion of an analysis and cite the appropriate report.

2.3.1.3.2 Special Instruction Note that there is a "Special Instruction?" field associated with each type of analysis. Suppose the submitter has some special instructions to go with a requested analysis. The user interface presents a special instructions screen (see Figure 8) after the sample

coordinator finishes with the alpha analyses request screen. The special instructions entered from the main screen appear first. The interface places the cursor at the first data field for special instructions from the alpha analyses request screen. New special instructions can be entered and any instruction on the screen can be changed.

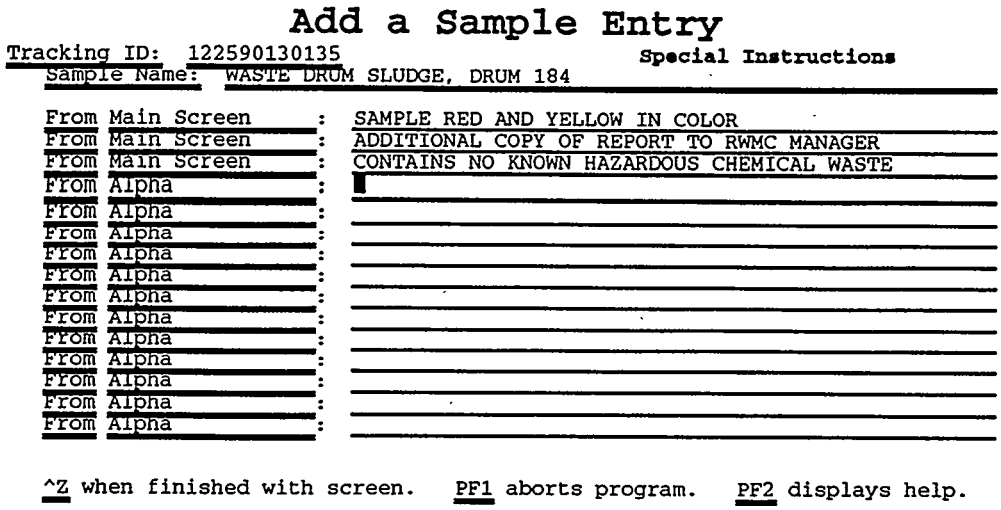

Figure 8. A Special Instructions Screen Following an Alpha Analyses Request Screen.

The following figures, 9-11, contain the three remaining analyses request screens.

2.3.1.3.3 Gross Alpha-Beta Analyses - The following gross alpha-beta analyses are currently available:

- for air filters and
- other (i.e., special requests).

For these analyses the submitter can specify the duration of the count, when the results are needed, and any special instructions. The analyst can document the completion of an analysis and cite the appropriate report.

2.3.1.3.4 Beta Analyses - The following beta analyses are currently available:

- strontium-90 (Sr-90)
- strontium-89 and -90
- total strontium
- tritium (H-3) and

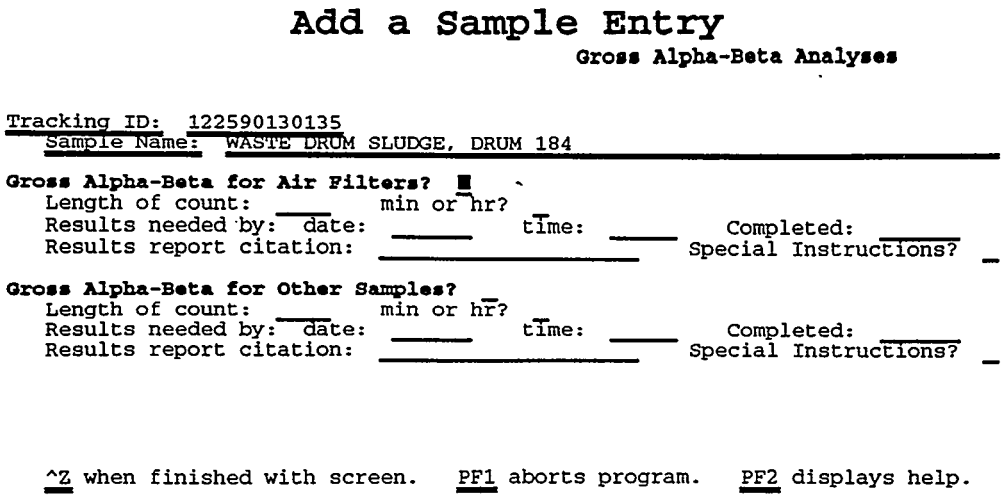

Figure 9. The Gross Alpha-Beta Analyses Request Screen.

• other: nickel-63 (Ni-63), iron-55 (Fe-55), sulfur-35 (S-35), plutonium-241 (Pu-241), or any other special requests.

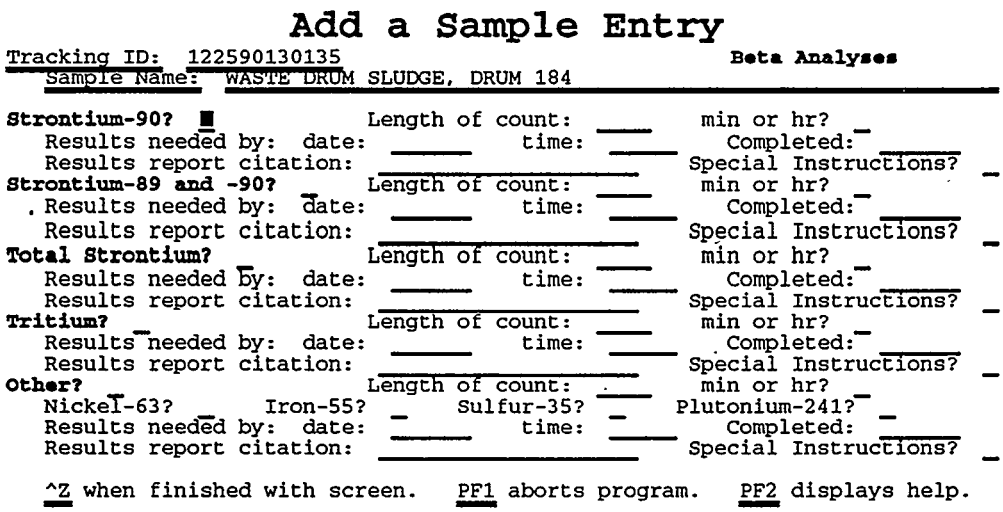

Figure 10. The Beta Analyses Request Screen.

For these analyses the submitter can specify the duration of the count, when the results are needed, and any special instructions. The analyst can document the completion of an analysis and cite the appropriate report.

2.3.1.3.5 Gamma Analyses - The following gamma analyses are currently available:

- screening/shipping counts (for deciding whether a sample has an activity above two nanocuries per gram - above this level the sample must be classified as radioactive for shipping as per U.S. Department of Transportation orders)
- full isotopic and
- other (i.e., special requests).

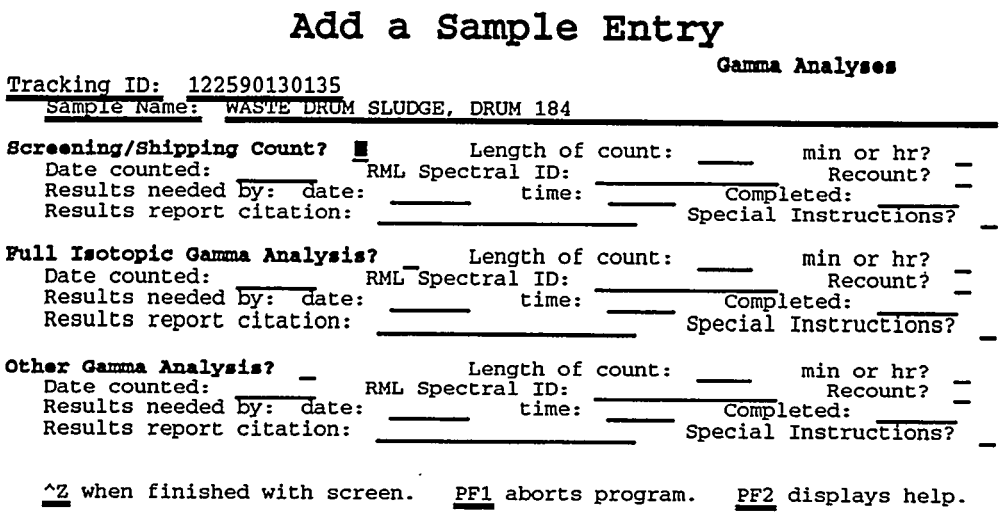

Figure 11. The Gamma Analyses Request Screen.

For these analyses the submitter can specify the duration of the count, when the results are needed, and any special instructions. The operator can record the spectrum acquisition date, the RML identification code for the spectrum, and whether the spectrum is from a recount. The analyst can document the completion of an analysis and cite the appropriate report.

2.3.1.4 Execution Sequence Tracking. Once all of the appropriate data screens are properly filled in, the user interface inserts the new entry into the database. As this is occurring, the user interface clears the monitor and presents an execution sequence tracking display. A sequence of messages such as the following will appear in the display:

```
Execution sequence tracking:
Constructing transaction settings . . .
 . . . transaction setting construction successful.
Beginning the transaction . . .
 . . . transaction successfully begun.
Inserting into the database files . . .
 . . . insertion into the database files successful.
Committing the insert transaction . . .
 . . . committing the insert transaction successful.
```

Any time the user interface accesses the database, it will keep the user informed as to the progress of the database transaction in just this way. Execution sequence tracking serves two functions. First, it lets the user know that the user interface is still working. Second, it provides the database programmer with some additional information in case the user interface in some way fails while attempting to perform a transaction.

As soon as the database transaction is successfully completed, the execution sequence tracking display disappears and the options request screen reappears on the monitor.

2.3.2 Update a Sample Entry

The interface asks the sample coordinator to " . . . enter the number of the entry to update or the tracking id of the entry to be retrieved." After any successful search, the user interface displays the number of entries retrieved on the options request screen (see Figure 12). The first way in which the sample coordinator can specify the sample entry to be updated makes use of the list of retrieved samples. Here the sample coordinator enters the ordinal number of the desired entry in the list (see Figure 13). The list is explained in more detail below.

The second way allows the sample coordinator to obtain directly the entry for a specific sample tracking identification code without first constructing a search for the entry. The sample coordinator simply enters the sample tracking identification code. (Remember that all sample tracking identification codes are twelve digits long and that any leading or trailing zeros must be included.) The user interface then performs a search for the entry.

Once the sample coordinator specifies the desired entry, the user interface attempts to retrieve it. First the interface searches for the entry. Upon finding the entry, the interface then retrieves it. Finally, the interface presents a screen akin to that presented in Figure 14.

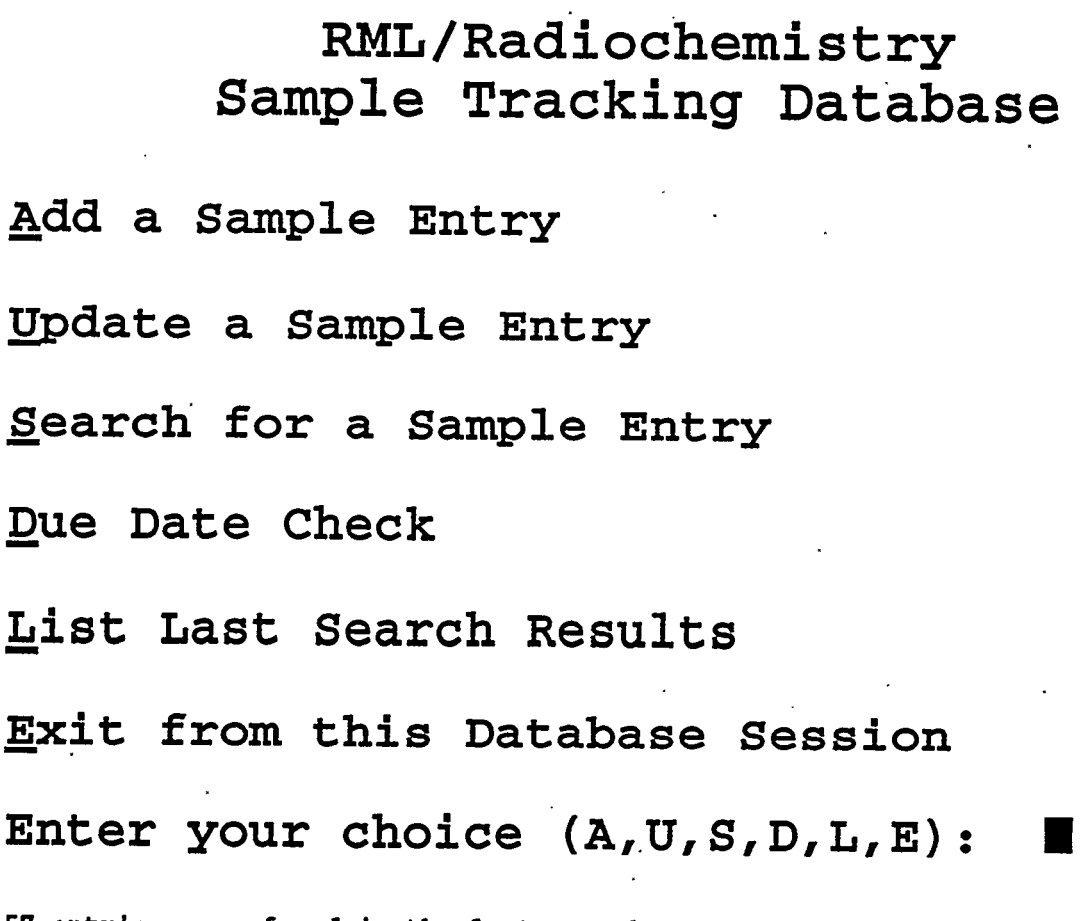

Figure 12. Options Request Screen After a Successful Search.

If the entry does not exist (suppose the sample coordinator has entered a bogus sample tracking identification code such as 123456879012), two things happen. First, the following appears at the end of the execution sequence tracking display:

```
Rolling back the transaction because of an error . . .
. . . roll back successful.
```

Second, because the user interface recognizes that no entry was found, "No entry exists for this tracking id!" appears in big, blinking text at the bottom of the options request screen.

For updating a sample entry, the user interface behaves much as it does for adding a sample entry. The screens will have the same appearance. The screen heading will be "Update a Sample Entry" instead of "Add a Sample Entry." Some fields will be fixed (displayed in reverse video) and therefore inaccessible with the cursor. For example, the cursor begins in the data field "Completed," not in the data field "Tracking ID." Others will be presented in bold text, and so must be filled before the sample coordinator can go to the next screen.

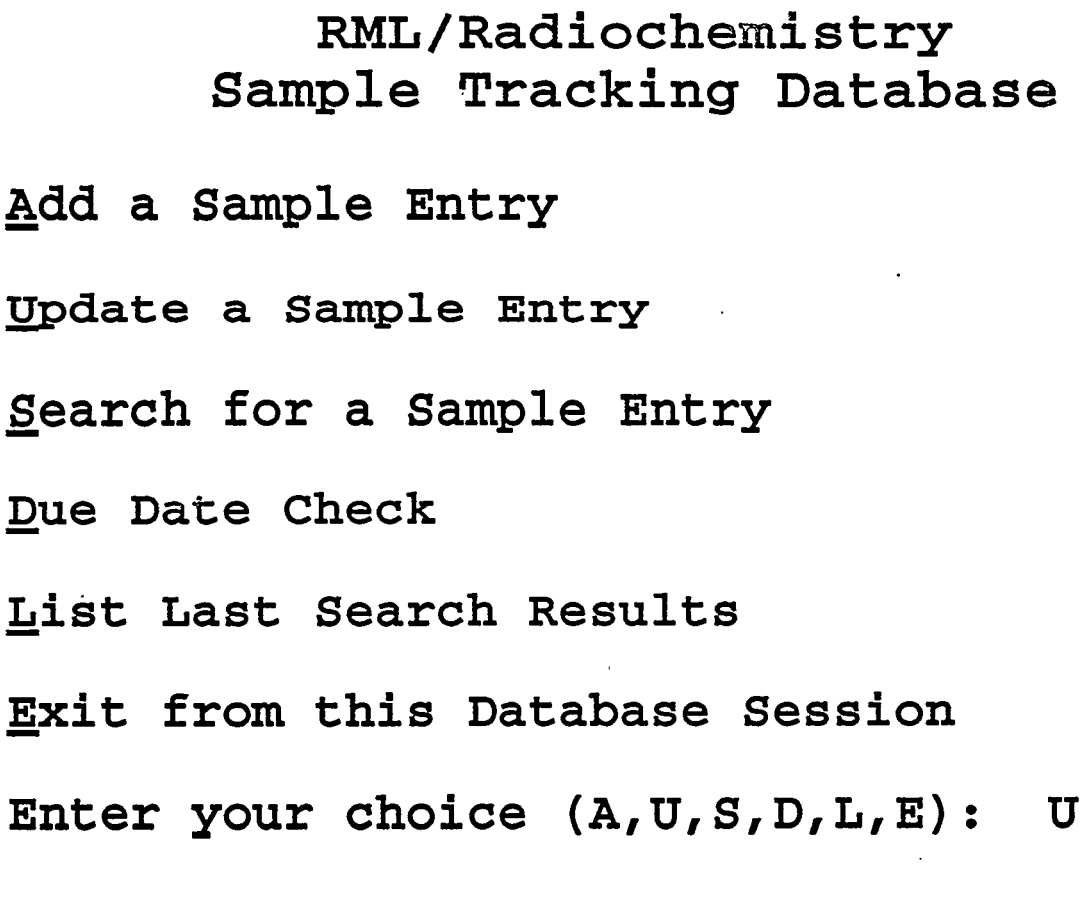

Figure 13. Options Request Screen During Initiation of the Update Option.

2.3.3 Search for a Sample Entry (User-Specified Searches)

If the user does not know the sample tracking identification code for a sample to be updated, a search for the desired sample needs to be performed. Searches also prove useful when the user wants to collect several related sample entries. For example, the user may want to know how many samples have been submitted under a given charge number.

The user interface first presents the main screen displayed in Figure 15 when constructing a search. None of the fields in any of these data entry screens are fixed and none are required. The user can construct a simple or complex search. The user needs to keep in mind, though, that the search, in effect, retrieves the intersection of several individual retrievals. The search performs an "and" union, not an "or" set union. This can be viewed as follows:

(i) for the first filled data field, the search finds all entries that contain a match ...us field's

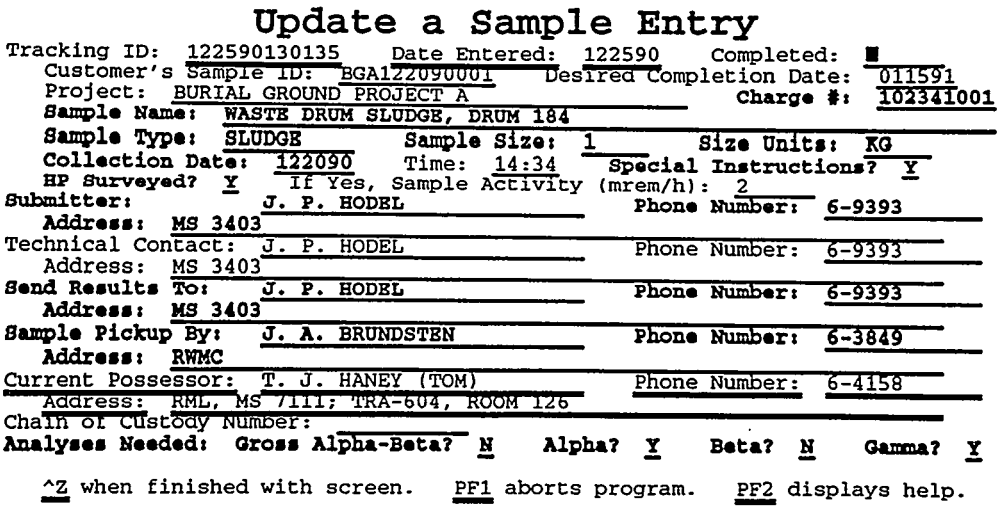

Figure 14. A Main Screen for Updating a Sample Entry.

value; call these entries set number one

(ii) for the next filled data field, the search finds all entries that contain a match for this next field's value; call these entries set number two

(iii) those entries found in both sets become the new set number one

(iv) repeat steps ii and iii for the remaining filled data fields, and

(v) the final set number one contains the search results.

(In contrast, step iii for a search using an "or" set union would read "all the unique entries found in either set number one or set number two become set number one.")

There are times when providing less information is desirable. Be sure to consider if data entered is truly common to all the desired entries. It is generally advisable, though, to provide as much information as possible. This improves the accuracy of the search. While making the additional comparisons increases the response time, fewer extraneous sample entries will be retrieved. This makes it easier for the user to pick out the desired entry from the list.

The user interface presents the following information for each entry that it finds:

- the sample tracking identification code (SID)
- the customer's identification code (CID)
- the customer's sample name (CSN) and

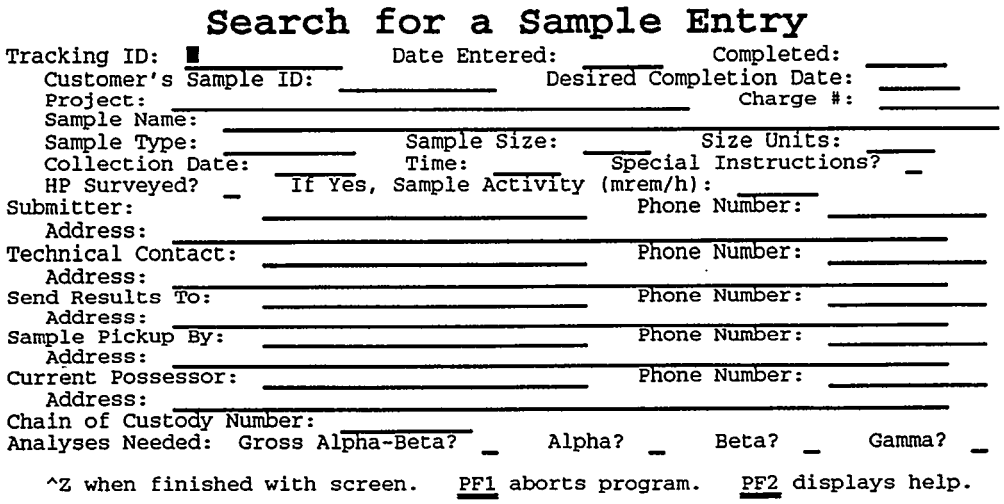

Figure 15. A Fresh Main Screen for Searching for a Sample Entry.

- the respective ordinal number within the list.

The entries are displayed in a fashion similar to the execution sequence tracking display. After the last data entry screen is completed, the user interface erases the display and begins listing the entries as they are found:

```
Starting the due date search . . .
Entries found:

   Entry   1, SID=010190080000, CID=AAABBBCCCDDD,
      CSN=PIECE OF ROCK

                          . . . .
```

At the end of a full screen of entries, the user interface asks the user to strike the return key before it continues displaying entries. This gives the user time to review the entries displayed. The user interface also waits until the user strikes any key after displaying the last entry found. The interface then returns to the options request screen.

The interface saves the information that is displayed. This information can be redisplayed by requesting a list of the last search results (option L at the options request screen; see List Last Search Results, section 2.3.5). Entries from the list can be updated using the option "Update a Sample Entry" by entering the entry's ordinal number (see Update a Sample Entry, section 2.3.2).

2.3.4 Due Date Check

A specialized search provided by the user interface scans for analyses that are soon to be due (or are overdue). In this option the interface asks "How many days forward from today should be checked?" The user should enter a positive integer number. The interface then calculates the latest "Results needed by" date (or due date) to be scanned for. The target of the interface's search is all uncompleted analysis

having a due date on or before this latest due date. The user interface presents the same information it does for a user-specified search (see Search for a Sample Entry, section 2.3.3) for each soon-to-be-due or overdue analysis it finds, along with the analysis that is due:

```
Starting the due date search . . .
Entries found:

   Alpha uranium analysis for 010191080000 due 030190
     (Entry   1, CID=AAABBBCCCDDD,
      CSN=PIECE OF ROCK                                    ).

                          . . . .
```

The user interface saves the same information that it does from a user-specified search; the analysis due is not saved. The listing generated can be used in the same ways a list from a user-specified search can be used.

2.3.5 List Last Search Results

The listing referred to here are the results from either a due date check or a user-specified search. The user interface saves the first one hundred entries from the most recent search. The interface displays the number of entries in the list, <n>, at the bottom of the options request screen: "<n> entries were found in the last search." If the list is empty, the user interface declares that "No search results are available!" when the user selects this option.

For a due date check, the list may not contain <n> unique sample entries. More than one analysis may be due for a given entry. Recall that the search results list from a due date check does not contain which analysis is due for a given entry.

2.3.6 Exit from this Database Session

To stop execution of the user interface, the user enters an "E" and then strikes the return key. The user interface clears the monitor display, the message "FORTRAN STOP" appears on the monitor, and the terminal session returns to the DCL prompt.

2.4 Miscellaneous Notes

2.4.1 "When the nightmare comes to pass . . . "

There are two potentially unnerving occurrences for any user of the database. The following subsections cover these.

2.4.1.1 System Crashes. The nightmare: in the middle of a database session, the VAX mainframe computer crashes or is purposely shutdown. What is the user to do? Not a lot! If there are any warnings that a shutdown is imminent, the user should heed them and finish using the user interface as quickly as possible.

How worried should the user be? Not too much! The user interface minimizes the amount of time spent in communication with the database. Therefore, the likelihood of the computer going down while the interface is accessing the database is small. Also, there are some database self-repair mechanisms for dealing with improper shutdowns like a system crash. These execute automatically run

when the computer comes back up. If the user is unable to access the database after a system crash, the database programmer should be notified.

What does the user stands to lose? Input to the user interface will be lost, such as data input to a screen when adding, updating, or searching for an entry. The list of sample tracking identification codes found in the last search of the database will be lost. The weekly user disk backup includes data already in the database. In the event of a catastrophic disk failure, at most the past week's additions and updates could be lost.

2.4.1.2 Database-Related Errors. The second nightmare: at some point the user receives this series of messages from the user interface:

```
Rolling back the transaction because of an error . . .
. . . roll back successful.
Execution terminating . . .
Please see error log file for error encountered.
Please bring this error to a programmer's attention.
```

What happened? A database-related error occurred that the user interface does not know how to handle. Recall that, for example, the first two messages appear whenever the user asks to update a sample entry that does not exist. In this instance the user interface knows how to handle the error. It tells the user that an entry with the specified sample tracking identification code does not exist.

Whenever the user interface cannot handle a database error without terminating execution, it first rolls back the outstanding transaction. This undoes the outstanding (and probably incomplete or otherwise corrupt) transaction and returns the database to the pristine state it existed in before the transaction began. In contrast, when a transaction is committed, it has successfully completed without corrupting the database. Committed transactions cannot be subsequently rolled back, since committing a transaction ends the transaction, just as rolling it back does.

The user interface then writes the text of the error message to a file named SAMPLE_TRACKING_ERRORS.LOG. This file exists in the directory from which the user invoked the interface and is created anew each time the user interface starts. After successfully completing a database session, the user interface deletes the error log file.

If a session terminates because of an error, the given instructions should be followed. The user should print the screen containing the termination message. Also, it is preferable, though not required, that the user not invoke the interface until the condition that caused the error is resolved.

2.4.2 Bar Code Labels

A few points about the bar code labels are worth mentioning.

2.4.2.1 Affixing Bar Code Labels to Sample Containers. The bar code labels currently in use are self-adhesive. The label, the adhesive, and the printing on the label should prove long-lasting with little fading or discoloration and are resilient to most mild chemical reagents. To apply a label, peel it from the backing paper and press it onto the surface of the sample container.

Most of the containers used by the RML for making gamma-ray spectral measurements are cylindrical in shape. For these containers, affix the self-adhesive bar code label with its long edge parallel to the axis of rotation of the cylinder (see Figure 16). This will minimize the amount of curvature across

the bar code label, allowing for easier reading by the scanner. The laser-diode bar code scanners can read bar codes through the yellow secondary-containment bags used in the RML, as long as the bag is smoothed over the bar code.

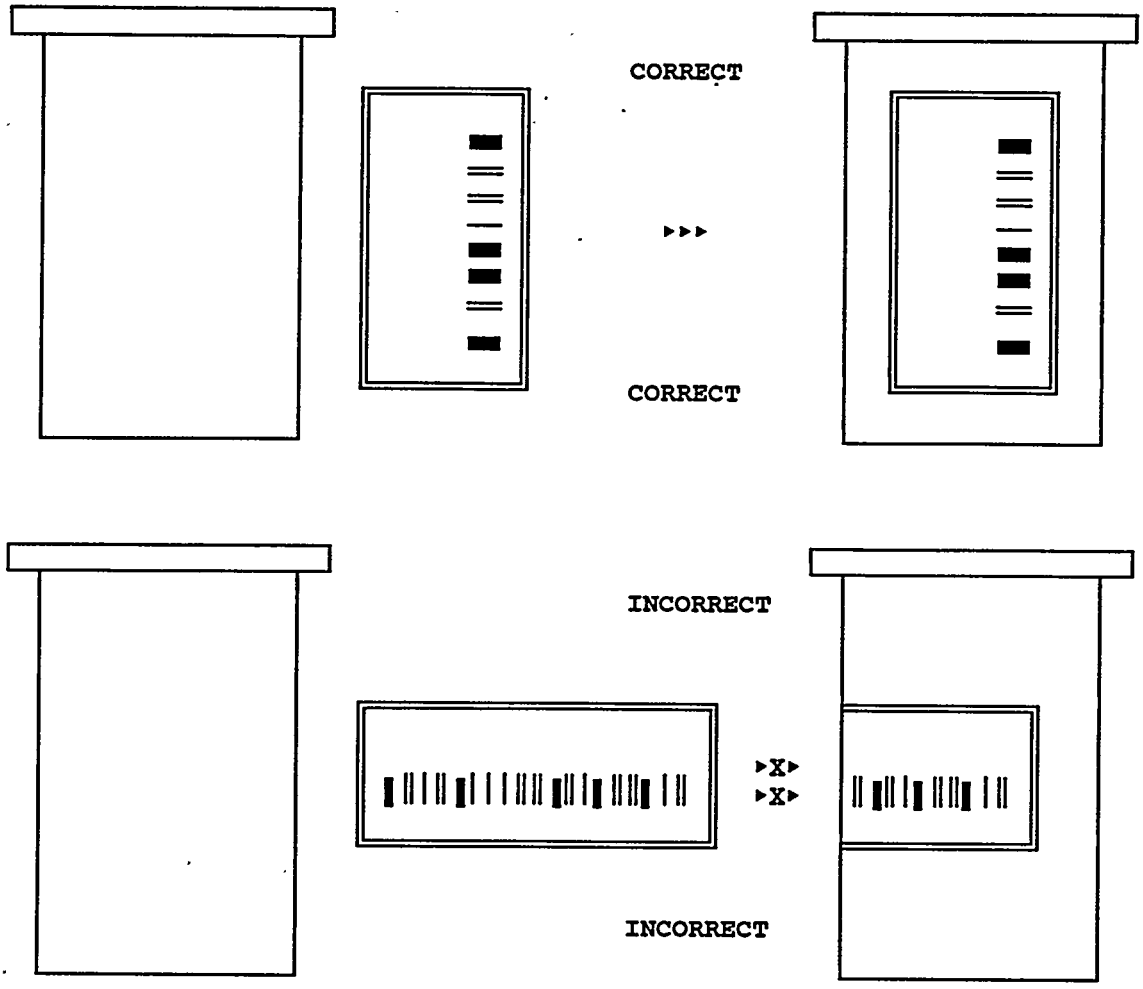

Figure 16. How to Affix Bar Code Labels to Cylindrical Sample Containers.

The remainder of the samples received by the RML are irregular in shape. For irregularly shaped sample containers, affix the bar code label to as smooth and as flat a portion of the sample container as is possible. A useful rule-of-thumb is that if the user cannot read the entire bar code label without moving the sample container, neither can the bar code scanner. For example, bar code labels should not be applied around the edge of an envelope. In this instance the user would have to turn the envelope over to see the other part of the label.

2.4.2.2 Replacing Printer Ribbons and Label Rolls. The printer ribbons on the bar code printer should last through one roll of labels. Although the ribbon may be printing well, replace it each

time an empty roll of labels is replaced. This will help to ensure that the printer produces a high-contrast, easily read label.

2.4.2.3 What if the bar code printer stops printing labels? The pattern for the bar code is stored in the bar code printer in a non-volatile RAM chip. This chip contains a battery which allows it to retain what is stored in its memory even if power to the printer is turned off. Nonetheless, this chip can sometimes lose the bar code pattern. (Note that this may indicate that the chip should be replaced; its internal battery may be dead.)

If the pattern is lost, it needs to be down-loaded into the chip from a personal computer. An INTERMEC® Model 3000A bar code label printer is currently in use by the RML, and so the compatible LabelWorks™ software is used to prepare the bar code pattern and down-load it to the printer.[a] Please refer to the appropriate manuals for how to prepare and down-load bar code patterns.

2.4.3 What to do with Suggestions or Comments on the Database or the User Interface

Are the database and user interface doing what they need to do? Could either be improved, whether to upgrade performance, simplify use, or enhance presentation? Please feel free to present any suggestions or comments to the database programmer. The users are the primary resources in making the database and its user interface the best tools for the job.

a. This does not constitute an endorsement of INTERMEC® or LabelWorks™. Other bar code printers and software could also be used to generate the needed bar codes.

3. DESIGN OF THE DATABASE

3.1 Basic Description

This database is a relational, multifile database. It is created and accessed using the Digital Equipment Corporation's Rdb/VMS implementation of the industry standard Structured Query Language (SQL). There is a separate file for each type of analyses and for basic sample information, tracking information, and special instructions. The directory [RDB_DATABASES.SAMPLE_TRACKING] on the user disk of the RML's VAX 6000 mainframe computer contains the database files. The weekly user disk backup includes the contents of these files (see Specifics, section 2.2).

The following subsections present a brief overview of the design of the database. The corresponding subsections under Specifics (section 2.2) present more detail. The presentation assumes the reader has some familiarity with relational database terminology.

3.1.1 Tables and their Fields and Indexes for Sorting

A table (or entity) contains a number of named fields for data storage. The names of the tables are as descriptive as possible while keeping the length of the name within the SQL-standard thirty-character limit.

Each field maps to only one data type (or domain). Appropriate fields serve as primary keys (for uniquely identifying each entry in a table) and foreign keys (for connecting entries between tables). Fields can have their own programmer-specified constraints (in addition to those of the corresponding domain) and comments. For example, null entries can be forbidden for a given field, such as the field that serves as the primary key.

A single field or a collection of fields serves as the primary key index for sorting each table. The name of this index is of the form <storage area name>_PRIMARY_KEY_INDEX. Its size depends on the fields which compose it. The comment for this index usually states the table for which it is the sorting key.

3.1.2 Domains and Attributes

A minimal set of data types, or domains, describes the space requirements for data in the database. Associated with each data type are a default value, constraints on the acceptable values, and a comment, all of which are programmer-specified.

Attributes are the data-containing fields within the database tables. Their names are as descriptive as possible while keeping the length of the name within the SQL-standard thirty-character limit. Each attribute's data type maps to only one domain.

3.1.3 Storage Areas and Snapshot Files

A separate storage area file contains each table and its definitions and constraints. The name of a storage area file is of the form <storage area name>.RDA. Its initial size is determined by the amount of data to be contained in a single entry to the table times an estimate of the number of entries that need to be available in the working database. The storage areas (and snapshot) files are automatically increased

in size as needed by Rdb/VMS. (<storage area name> is a concatenation of the first two letters of each underscore-separated part of the table's name (and so can be computer-generated). For example, <storage area name> for the table SAMPLE_INFORMATION is SAIN. The name of the table's snapshot file is of the form <storage area name>.SNP.)

3.2 Specifics

A listing of the SQL instruction set for creating this multifile database appears in Appendix A. The CREATE_SCHEMA tool generated this instruction set. (The software tools used to design the database are described in Reference 2. The examples presented in that report make use of this database.) The Lotus® 1-2-3® print files,[b] containing the entity- (table-) and screen-sorted attribute description lists, that serve as input to CREATE_SCHEMA appear in Appendix B. A recording of the terminal session from the execution of CREATE_SCHEMA, specifying the programmer's input for this database, appears in Appendix C.

The schema for this database is SAMPLE_TRACKING and so the name of the root file is SAMPLE_TRACKING.RDB. The root file contains the complete set of domain and table definitions.

3.2.1 Names of Tables, Storage Areas, and Snapshot Files

The following are the table names, with the respective storage area names in parentheses:

- SAMPLE_INFORMATION (SAIN)

- for alpha analyses: ALPHA_URANIUM (ALUR), ALPHA_THORIUM (ALTH), ALPHA_PLUTONIUM (ALPL), ALPHA_AM241_SANS_PU239 (ALAMSAPU), ALPHA_AM241_WITH_PU239 (ALAMWIPU), ALPHA_TOTAL_SPECTROMETRIC (ALTOSP), and ALPHA_OTHER (ALOT)

- for gross alpha-beta analyses: GROSS_ALPHA_BETA_AIR_FILTERS (GRALBEAIFI) and GROSS_ALPHA_BETA_AIR_OTHER (GRALBEOT)

- for beta analyses: BETA_STRONTIUM_90 (BEST90), BETA_STRONTIUM_89_AND_90 (BEST89AN90), BETA_TOTAL_STRONTIUM (BETOST), BETA_TRITIUM (BETR), and BETA_OTHER (BEOT)

- for gamma analyses: GAMMA_SCREEN (GASC), GAMMA_FULL_ISOTOPIC (GAFUIS), and GAMMA_OTHER (GAOT)

- SPECIAL_INSTRUCTIONS (SPIN) and

- for possessors: TRACKING (TR).

3.2.2 Names and Descriptions of Domains

The following lists the domains and their respective descriptions. Many of the names fu

b. The use of Lotus® 1-2-3® in constructing this database does not constitute an endorsement of Lotus® 1-2-3®. Other spreadsheet programs could also be used to generate these output files.

describe the data type of the domain. For example, read "at most X alphanumerics" as the description when "X_CHARACTERS" is the name of the domain.

- CHARACTER_TIME_HH_MM (four or five numbers-plus-colon long and of the form hh:mm; for recording times-of-day)

- CHARGE_NUMBERS (exactly nine alphanumerics long)

- FIFTY_CHARACTERS

- FLAGS (one character long; for Yes/No data)

- FORTY_CHARACTERS

- IDENTIFICATION_CODES (exactly twelve alphanumerics long and of the numerical form mmddyyhhmmss; for the sample tracking database identification code)

- INTEGER_DATE_COUNT (date count starting at January 1, 1990; for use in "Due Date Check")

- INTEGER_NUMBERS

- NAMES (at most twenty-five alphanumerics long)

- PHONE_NUMBERS (at most twelve alphanumerics long)

- REAL_NUMBERS

- RML_SPECTRAL_ID_CODES (exactly fourteen alphanumerics long and of the form <save area abbreviation of two characters in length>$<spectral identification code of eleven alphanumerics in length>, for the location and file name of spectral data; see Reference 1)

- SIXTEEN_CHARACTERS

- SIXTY_CHARACTERS

- TEN_CHARACTERS

- TWENTY_CHARACTERS

- UNITS (at most five alphanumerics long; for abbreviated units of measure such as H for hour, M for minute, G for gram, and ML for milliliter; note the exclusive use of uppercase letters, consistent with the user interface forcing all letters entered to uppercase)

- USER_NAME (at most thirty alphanumerics long; for tracking who last modified an entry) and

- VMS_DATE_TIME (seventeen or eighteen alphanumerics long and of the form "dd-mmm-yy hh:mm:ss"; for noting when a record was last modified).

3.2.3 Backing Up the Database

DCL BACKUP does not work correctly on the database files (the root file and storage area and snapshot files), owing to their internal structure. These files are all marked for no backup using the DCL command SET FILE /NOBACKUP. This way the weekly backup of the user disk on the VAX 6000 mainframe computer does not include these files.

The weekly backup includes the contents of the database storage area files, though. The routine used by the RML operators to perform the backup contains code to do this (see Appendix D). The routine

(i) verifies the integrity of the database

(ii) performs an RMU BACKUP on the database (this a creates file than can be correctly saved and restored using DCL BACKUP

(iii) performs the DCL BACKUP

(iv) purges all but the two most recent versions of the file generated by RMU BACKUP and

(v) marks these files for no backup.

3.2.4 Removal of Inactive Entries from the Database

An inactive entry is one for which all work has been completed, say, at least six months ago. The inactive entries could be moved to a second, nearly identical version of the database. At a minimum, the inactive database would differ from the active database in that it would contain the date the entry was moved. The goal of removing inactive entries is to keep the active database as small as it needs to be; unneeded entries slow down searches and retrievals. The actual moving of entries can be performed automatically or by hand. It could be included as part of the backup procedure (see Backing Up the Database). Inactive entries could be made accessible to the user interface by allowing the user to specify the active or inactive database as the one to use. This capability does not exist with version 1.00 of the sample tracking database.

4. DESIGN OF THE USER INTERFACE

4.1 Basic Description

Except for informational messages, the interface makes relating to the database transparent to the user. After selecting the desired option (Add a Sample Entry, Update a Sample Entry, . . .), the user fills in screen-displayed forms. The interface then constructs and executes the appropriate SQL calls. Screen management system routines are used for constructing the screen-displayed forms, reading data from them, and flagging entry errors and missing-yet-required information.

Owing to the case-sensitive nature of SQL searches, all alphabetic input to the interface program is forced to uppercase. Numerical input to data fields can be in the form of integer, real, or exponential numbers; the interface chooses the most complete and concise representation for displaying numerical data retrieved from the database.

4.2 Specifics

The software tools described in Reference 2 were used in preparing portions of code, such as declaration statements. They were also used to generate most of the dynamic SQL calls and related FORTRAN declaration and data-type conversion statements for the user interface. The examples presented in that report make use of this database.

The following subsections provide more detail on how the user interface stores data, converts data types, and performs error checking on entered data. It also describes the types of SQL calls the user interface uses and how, using SQL calls, it communicates with the database in performing insertions (for adding a sample entry), cursor declarations and fetches (for searching for a sample entry and due date checks), and modifications (for updating a sample entry). The routines and functions mentioned in the following subsections are described in more retail in Reference 2.

4.2.1 Data Storage Variables and Arrays

The array and scalar declaration statements (and some source code) related to the screen management routines are contained in the include file FLDDEF.INC (see Appendix E). These statements were generated using the tools BUILD_SCREEN and CREATE_SCHEMA; the screen templates submitted to BUILD_SCREEN are presented in Appendix F (see Reference 2).

4.2.1.1 Storage of Data. For ease of use with screen management routines, all information for a given screen is saved in a character array (composed of eighty-character-long elements) assigned to that screen. Since SQL calls cannot make use of arrays (and variable names over six characters long are discouraged in Rdb/VMS), the arrays are equivalenced to a set of scalar variables that are of the required length for the appropriate domains. A character array's name is of the form fidat<first letter of the screen's name>; for example, for the Main screen the array's name is fidatM.

BUILD_SCREEN creates a set of scalar variables using a generic naming scheme that only identifies the screen. For example, scalar variables for the main screen's data fields are named fldM<n>, where <n> is the sequential number of the field within the screen. For clarity, the names of the scalar variables were changed to include some relation to the appropriate attribute.

The new names of the scalar variables begin with f<first letter of the screen's name>. The

remaining four letters, at most, of each name are constructed in a fashion similar to how storage area names are constructed (see Storage Areas and Snapshot Files, section 2.1.3) but using only the first letter of each underscore-separated part of the respective attribute. For example, for the main screen's sample tracking identification code (attribute name SAMPLE_TRACKING_ID) the scalar variable is named fMSTI (Main screen, then SAMPLE, then TRACKING, then ID). (Some hand modifications were required to eliminate duplicate names).

4.2.1.2 Conversion of Character Data to Numerical Data. If the database requires a number for a given attribute, the appropriate translation of character-to-integer or character-to-real is performed. (The result is stored in a scalar variable the name of which begins with an "i" instead of an "f" for integer data and with an "r" instead of an "f" for real data). The reverse translation is performed for numerical data retrieved from the database and to be displayed via screen management routines. These conversions are handled by the FORTRAN functions RVALUE (for character string-to-real number conversions) and CVALUE (for real number-to-character string conversions). The codes for these functions, again, are presented in Reference 2.

The arrays containing labels for the fields have names of the form field<first letter of the screen's name>. The arrays containing the names of the entities and attribute corresponding to the screen fields have names of the form atrbt<first letter of the screen's name>. There are no scalar equivalents for these arrays. Other arrays and scalars used in the screen management calls are similarly constructed; Table 1 lists all the arrays associated with any screen. (Note that from here on and in the table "<first letter of the screen's name>" is abbreviated "<>.")

Some of these arrays will be discussed in more detail later, in the following subsection on error checking for example. The implementation of these arrays, and the coding for the error checking, is found in the FORTRAN routine DISPLY (see Reference 2).

4.2.1.3 Error Checking. The user interface checks for three types of errors. First, is the data entered short enough to fit into the appropriate field? Each data field has a maximum number of alphanumerics that can be entered into it, mxnoc<>. Second, is the data required? If the field is rendered in bold print, an entry must be made into the data field before the user can leave that screen. Also, an entry can be required in a field because an entry has been made in a related field: if an entry appears in the "Date Completed" field, for example, an entry would be required in the "Results report citation" field. Third, is the data entered of the type specified by the fityp<> array? While data entered into a field is read into a FORTRAN CHARACTER variable, a real number may be needed for the corresponding attribute in the database. The following data types, as implemented in the FORTRAN routine DISPLY, are presently available.

4.2.1.3.1 "a" for Anything - Any combination of alphanumerics, spaces included and up to the specified maximum length, is allowed.

4.2.1.3.2 "d" for Date - A date in the integer form mmddyy is allowed. The entry will be converted into an integer date count (that is, how many days have elapsed between the specified date and December 31, 1989) for storage in the database. Integer date counts make due date checks easier to perform, for instance. Conversions are accomplished using the FORTRAN functions DAYCNT (character mmddyy-to-integer date count) and CHRDAT (integer date count-to-character mmddyy).

An entry of this type must be composed of exactly six integers (leading and trailing zeros must be included). The first two digits, mm, represent the month and must fall in the range 01 to 12 (inclusive). The second two digits, dd, represent the day of the month. The last two digits, yy, represent the

Table 1. Arrays and Scalars Used in Coding the User Interface.

Array Name	Description	Scalar Equivalent?	Letter for Equivalent
field<>	contents of the description field	N	-
fidat<>	contains error-free data (or is null) from the data field	Y	f
atrbt<>	table (entity) and attribute names corresponding to the screen field	N	-
varbl<>	contains the names of scalar variables equivalenced to fidat<> and to the respective if-null indicator	N	-
fityp<>	allowed data type for the data field ("a" for alphanumeric, e.g.)	N	-
fdlen<>	length of the description field	N	-
fdrow<>	row number for display of the description and data fields	N	-
fdcol<>	starting column number for display of the description field	N	-
ficol<>	starting column number for display of the data field	N	-
mxnoc<>	maximum number of alphanumerics allowed in the data field (length of data field)	Y	l (for length)
fdrnd<>	rendition of the description field (normal if optional, bold if required, reverse video if fixed)	N	-
firnd<>	rendition of the data field	N	-
ifnul<>	if-null indicator: -1 if the data field is empty (null), 0 otherwise	Y	n (for null)
ifrqd<>	entry into the current data field required if data field number ifrqd<> is not empty	N	-
idito<>	"ditto" transfers contents of data field number idito<> into the current data field	N	-
ifmod<>	if-modified indicator: true if the data field's value has been changed (during Update), false otherwise	Y	m (for modified)

year and are interpreted as follows: if yy is between 90 and 99, the year is 19yy; if yy is between 00 and

79, the year is 20yy. Entries between 80 and 89 are not allowed as they should be interpreted as 20yy, yet the resulting integer date count would be too large to fit into a FORTRAN INTEGER*4 variable and hence into the respective database attribute.

4.2.1.3.3 "l" for Length - Here the entry must be exactly mxnoc<> alphanumerics long. There are no restriction as to the alphanumerics used. One instance in which "l" is used is the charge number to be billed for performing the analyses. All charge numbers are exactly nine alphanumerics long, so a shorter or longer entry is invalid.

4.2.1.3.4 "t" for Time - The format of the entry allowed here is either "h:mm" or "hh:mm." h or hh represents the hour as an integer. A leading zero is not necessary because the colon can be used to delineate the hour portion of the entry from the minute portion. mm represents the number of minutes.

4.2.1.3.5 "u" for Units of Time - In instances such as the units for the duration of the spectral acquisition, an entry of either H for hours or M for minutes is allowed. "u" covers only units of time (duration); it does not cover sample size units, for example.

4.2.1.3.6 "?" for Questions needing a Yes-or-No Answer - The only allowed entries here are "Y" for yes and "N" for no. This data type is used for flags such as whether a certain type of analysis is to be performed.

4.2.2 Communication between the User Interface and the Database

SQL calls are used to transfer data between the user interface and the database. These calls are either hard-coded into the user interface (precompiled SQL) or are constructed by it (dynamic SQL).

4.2.2.1 Forms of SQL Used. Both precompiled SQL and dynamic SQL calls are used. Precompiled SQL calls are completely known by the user interface before they are executed. Such calls include inserting all attributes of an entry into the database. An EXEC SQL preface marks these calls as precompiled SQL statements rather than as statements in the host program's language.

Dynamic SQL calls can only be composed during execution of the user interface. Instances of these include requesting the tables needed for the current transaction. The instruction set for a dynamic SQL call is constructed as a FORTRAN CHARACTER variable and is executed from the main routine using EXEC SQL EXECUTE IMMEDIATE. To minimize the length of the character string, a FORTRAN routine APSTRG (APpend character STRinG) fills the character variable with each additional portion of the SQL call passed to it.

Modular SQL, in which calls are made from the main routine to SQL routines constructed in a single and separate file, is not used. The main routine of the user interface is larger than might seem prudent for ease of code development because of the Rdb/VMS requirement that all precompiled SQL statements reside in the same routine.

4.2.2.2 Commencing a Database Transaction. Communications with the database occur within transactions. The transaction definition (SET TRANSACTION) specifies the tables of the database the user needs and the type of access required. For example, for adding an entry to the database, the sample coordinator needs exclusive write access to all of the tables for the requested analyses. No one else should be able to access these tables in any fashion until the addition is completed. The construction of the SET TRANSACTION call depends on the application (for example, see Add a Sample Entry,

section 3.2.2.3).

At present, all SET TRANSACTION calls will wait at most sixty seconds for the requested tables to become available. To minimize conflicts (lock conditions resulting from two users wanting the same table at the same time and in incompatible fashions), database transactions last only as long as necessary. Any database transaction starts only as soon as all the requisite information is available. Again using the example of adding an entry, the database transaction does not begin until all of the appropriate screens are filled by the user. And, accordingly, each transaction is closed - either committed, that is, saved, or rolled back, that is, undone - upon completion.

4.2.2.3 Add a Sample Entry. The addition (insertion) of a sample entry to the database occurs via the SQL INSERT command. Here the main routine constructs the SET TRANSACTION command (using the FORTRAN routine APSTRG mentioned above in Forms of SQL Used), and then executes it as a dynamic SQL call (EXECUTE IMMEDIATE). Again, dynamic SQL is used since all of the tables that will be needed are generally not known until it is time for the addition to occur. All tables needed are included in a single SET TRANSACTION command because violations of constraints are better managed within a single transaction than over a series of transactions. A constraint, for example, may require that an entry for sample identification code exists in the sample information table before an entry for that same identification code can be made into an analysis table. A typical sequence of calls to APSTRG, followed by execution of the SET TRANSACTION call, would be

```
      call apstrg(comnd1,lcmnd1,('set transaction read write wait 60'//
     . ' reserving SAMPLE_INFORMATION for exclusive write,'),.true.)
      if(ifgros(1))then
         if(ifgros(2))call apstrg(comnd1,lcmnd1,
     .    ' GROSS_ALPHA_BETA_AIR_FILTERS for exclusive write,',.false.)

                                  . . .

         endif

                                  . . .

      call apstrg(comnd1,lcmnd1,
     . (' SPECIAL_INSTRUCTIONS for exclusive write,'//
     . ' TRACKING for exclusive write'),.false.)

                                  . . .

      exec sql execute immediate :comnd1

                                  . . . .
```

The next SQL calls are precompiled SQL calls and insert the data provided by the sample coordinator into the database:

```
      exec sql insert into SAMPLE_INFORMATION values(
     .    :fMSTI:nMSTI,:iMDSE:nMDSE,:iMDAWC:nMDAWC,:fMCSI:nMCSI,
     .    :iMRNBD:nMRNBD,:fMCPN:nMCPN,:fMWCN:nMWCN,:fMCSN:nMCSN,
     .    :fMST:nMST,:rMSS:nMSS,:fMSSU:nMSSU,:iMSCD:nMSCD,
     .    :fMSCT:nMSCT,:fMASI:nMASI,:fMSHS:nMSHS,:rMSHSA:nMSHSA,
     .    :fMSSN:nMSSN,:fMSSPN:nMSSPN,:fMSSA:nMSSA,:fMTCN:nMTCN,
     .    :fMTCPN:nMTCPN,:fMTCA:nMTCA,:fMSRTN:nMSRTN,:fMSRTP:nMSRTP,
     .    :fMSRTA:nMSRTA,:fMSPBN:nMSPBN,:fMSPBP:nMSPBP,:fMSPBA:nMSPBA,
     .    :fMNGAB:nMNGAB,:fMNA:nMNA,:fMNB:nMNB,:fMNG:nMNG,
```

```
.    :nxinst:ifnnxi,:nxpssr:ifnnxp,:fGNGS:nGNGS,:fGNGFI:nGNGFI,
.    :fGNGO:nGNGO,:fCNGAF:nCNGAF,:fCNGAO:nCNGAO,:fBNBSr:nBNBSr,
.    :fBNBSC:nBNBSC,:fBNBST:nBNBST,:fBNBT:nBNBT,:fBNBO:nBNBO,
.    :fANAU:nANAU,:fANATh:nANATh,:fANAPu:nANAPu,:fANASP:nANASP,
.    :fANAWP:nANAWP,:fANATS:nANATS,:fANAO:nANAO,USER,:today:notnul,
.    :fMCOCN:nMCOCN)
```

The values are passed in the form :<scalar equivalent of data field>:<scalar equivalent of if-null indicator>. The colon preceding each variable name tells the SQL precompiler that what follows is a variable in the user interface's native language, not an SQL variable or key word.

What about fields into which no entry was made? Recall that there is an if-null indicator array. If a field is left blank, that field's if-null indicator is set (to a value of -1). When this if-null indicator is passed to the database (with a value of -1), the attribute in the database entry is marked NULL. (The if-null indicator could be used to specify which attributes are to be passed to a table. Dynamic SQL would be used in place of precompiled SQL in this case. Pointers to variables are required in place of variables for dynamic SQL, though.)

4.2.2.4 Search for a Sample Entry. This is another instance where dynamic SQL calls could be used; this time they are, in conjunction with the APSTRG routine and the atrbt<> and fidat<> arrays. To perform a search, the user fills in all the fields necessary to describe the entry or entries desired. The if-null indicator is used to indicate the fields to be included in the search command. The user is to leave blank those fields for which no information is known or for which the information may vary among the desired entries.

Recall that the atrbt<> arrays contain the name of the table (entity) and attribute that correspond to a given screen field. The search command is constructed using the atrbt<> arrays and the values contained in the fidat<> arrays. If a data field contains an entry (and so its if-null indicator is not set), the appropriate elements from atrbt<> and the entry are used in constructing the dynamic SQL search call via APSTRG. These portions of the search command are constructed in the FORTRAN routine SRCBLD (SEARCH BUILD).

Searches are conducted using cursors and fetches from cursors. A cursor can be thought of as a virtual table containing only the specified entries (EXEC SQL DECLARE . . . CURSOR for <statement name>; EXEC SQL PREPARE <statement name> from <search command>). Data from a cursor can be retrieved into native language variables (EXEC SQL FETCH . . . INTO . . .). A typical portion of the main routine's code for constructing the search command and executing the search and retrieval is

```
 call apstrg(comnd1,lcmnd1,
. ('select distinct SAMPLE_INFORMATION.SAMPLE_TRACKING_ID,'//
. ' SAMPLE_INFORMATION.CUSTOMER_SAMPLE_ID,'//
. ' SAMPLE_INFORMATION.CUSTOMER_SAMPLE_NAME'//
. ' from SAMPLE_INFORMATION'),.true.)
 call apstrg(comnd2,lcmnd2,' where',.true.)

                          . . .

 call srcbld(comnd1,lcmnd1,comnd2,lcmnd2,numfdM,fidatM,mxnocM,
. fitypM,atrbtM)
 if(ifgama(1))call srcbld(comnd1,lcmnd1,comnd2,lcmnd2,numfdG,
. fidatG,mxnocG,fitypG,atrbtG)

                          . . .
```

```
      call apstrg(comnd1,lcmnd1,(comnd2(1:lcmnd2)//
     . ' order by SAMPLE_INFORMATION.SAMPLE_TRACKING_ID asc'),.false.)

                                  . . .

      exec sql declare SEARCH_cursor cursor for SEARCH_statement
      exec sql prepare SEARCH_statement from comnd1
      exec sql open SEARCH_cursor
      exec sql whenever not found goto 704

                                  . . .

  702 . . .
      exec sql fetch SEARCH_cursor into :smplid,:custid,:csname

                                  . . .

      goto 702
  704 . . .
      exec sql close SEARCH_cursor

                                  . . . .
```

Note that two character strings are input to the routine APSTRG. For each non-null data field, a call to SRCBLD transfers the name of the respective entity alone to the first string (if it does not already appear there) and, to the second string, the entity (table) name, the attribute name, and the value contained in the data field. The second string is appended to the first string prior to the search. Finally, the search is performed as the cursor is declared (defined), opened (much as a file is opened), and read from (via FETCH) until the last entry is found (watched for by WHENEVER NOT FOUND . . .). The data retrieved can be displayed as it is found and stored for future reference. After all the data is retrieved from the cursor, the cursor is closed.

4.2.2.5 Due Date Checks. Due date checks are a form of search where the same, specific attributes in the various tables are always interrogated, namely the date-completed fields. The sample coordinator states how many days ahead should be checked for analyses and reports which will be due. The user interface calculates the integer date count for the current day, and adds to it the number of days ahead to be checked. A search is then preformed for all entries having due dates the integer date count for which is less than or equal to the above sum (IDUCNT) and for which the specified work has not been completed (that is, for which the DATE_ALL_WORK_COMPLETED field is null). The coding for one such search is

```
      exec sql declare BEST90_due_cursor cursor for
     .   select
     .     BETA_STRONTIUM_90.SAMPLE_TRACKING_ID,
     .     SAMPLE_INFORMATION.CUSTOMER_SAMPLE_ID,
     .     SAMPLE_INFORMATION.CUSTOMER_SAMPLE_NAME,
     .     BETA_STRONTIUM_90.RESULTS_NEEDED_BY_DATE
     .   from BETA_STRONTIUM_90, SAMPLE_INFORMATION
     .   where BETA_STRONTIUM_90.RESULTS_NEEDED_BY_DATE<=:iducnt and
     .     BETA_STRONTIUM_90.DATE_ALL_WORK_COMPLETED is NULL and
     .     BETA_STRONTIUM_90.SAMPLE_TRACKING_ID=
     .      SAMPLE_INFORMATION.SAMPLE_TRACKING_ID
     .   order by BETA_STRONTIUM_90.SAMPLE_TRACKING_ID asc

                                  . . .
```

```
    exec sql open BEST90_due_cursor

                                . . .

    exec sql whenever not found goto 836
834 exec sql fetch BEST90_due_cursor into :smplid,:custid,:csname,
  . :duedat

                                . . .

    goto 834
836 . . .
    exec sql close BEST90_due_cursor

                                . . . .
```

4.2.2.6 Update a Sample Entry. This final mode of communication uses the techniques described above. First, data for the specified entry is fetched from the database via a cursor:

```
        exec sql declare ALUR_cursor cursor for
.          select
.             ALPHA_URANIUM.SAMPLE_TRACKING_ID,
.             SAMPLE_INFORMATION.CUSTOMER_SAMPLE_NAME,
.             ALPHA_URANIUM.RESULTS_NEEDED_BY_DATE,
.             ALPHA_URANIUM.DATE_ALL_WORK_COMPLETED,
.             ALPHA_URANIUM.RESULTS_REPORT_CITATION,
.             ALPHA_URANIUM.ANY_SPECIAL_INSTRUCTIONS
.          from ALPHA_URANIUM, SAMPLE_INFORMATION
.          where ALPHA_URANIUM.SAMPLE_TRACKING_ID=:smplid and
.             ALPHA_URANIUM.SAMPLE_TRACKING_ID=
.              SAMPLE_INFORMATION.SAMPLE_TRACKING_ID
.          order by ALPHA_URANIUM.SAMPLE_TRACKING_ID desc

                                . . .

        exec sql open ALUR_cursor

                                . . .

        exec sql fetch ALUR_cursor into
.          :fASTI:nASTI,:fACSN:nACSN,:iARNB1:nARNB1,:iADAW1:nADAW1,
.          :fARRC1:nARRC1,:fAASI1:nAASI1

                                . . .

        exec sql close ALUR_cursor

                                . . . .
```

The retrieved data is then displayed for the sample coordinator to update. Once the updating of the screens is completed, the entire database entry is updated (modified) if any changes have been made within the screens.

The if-modified indicator plays a role different from that the if-null indicator plays in searching for a sample entry. If any of the if-modified indicators for a table's fields are true, the entire entry is updated if it previously existed in the database. If the entry is new to the database, the entry is added to the database:

```
if(ifalph(2).and.(mARNB1.or.mADAW1.or.mARRC1.or.mAASI1))then

                                . . .

   if(ifalpO(2))then   !Use update to change the entry.
         exec sql update ALPHA_URANIUM set
.                RESULTS_NEEDED_BY_DATE=:iARNB1:nARNB1,
.                DATE_ALL_WORK_COMPLETED=:iADAW1:nADAW1,
.                RESULTS_REPORT_CITATION=:fARRC1:nARRC1,
.                ANY_SPECIAL_INSTRUCTIONS=:fAASI1:nAASI1,
.                USER_CREATING_OR_MODIFYING=USER,
.                DATE_CREATED_OR_MODIFIED=:today:notnul
.             where SAMPLE_TRACKING_ID=:smplid
      else   !Use insert since it was not there before.
         exec sql insert into ALPHA_URANIUM values(
.           :fASTI:nASTI,:iARNB1:nARNB1,:iADAW1:nADAW1,
.           :fARRC1:nARRC1,:fAASI1:nAASI1,USER,:today:notnul)
      endif
   endif

                               . . . .
```

5. REFERENCES

1. E. W. Killian and D. A. Femec, *Operator's Guide for VAXGAP, a Gamma-Ray Spectrum Analysis Package*, EGG-2672, August 1992.

2. D. A. Femec, *Version 1.00 Programmer's Tools Used in Constructing the INEL RML/Analytical Radiochemistry Sample Tracking Database and its User Interface*, INEL-95/0454, September 1995.

APPENDIXES

A. SQL INSTRUCTION SET FOR CREATION OF SAMPLE TRACKING DATABASE

Note the ellipses in the two "define/system" lines below; these would be replaced with the appropriate root directory specifier ($USER:[DATABASES.], for example).

```
! SQL commands for creating sample_tracking schema.
!
! Define the data and snapshot file location logical names.
!
$ define/system/nolog/translation_attributes=concealed rdb_data . . .
$ define/system/nolog/translation_attributes=concealed rdb_snap . . .
!
! Create the schema.
!
create schema
   filename "sample_tracking"
   buffer size is 18

!
! Create the storage areas.
!
create storage area SAIN
   filename "rdb_data:[sample_tracking]SAIN"
   allocation is 4400 pages
   page size is 2
   page format is uniform
   extent is 441
   snapshot filename "rdb_snap:[sample_tracking]SAIN.snp"
   snapshot allocation is 10 pages
   snapshot extent is 2 pages

create storage area ALUR
   filename "rdb_data:[sample_tracking]ALUR"
   allocation is 1100 pages
   page size is 1
   page format is uniform
   extent is 111
   snapshot filename "rdb_snap:[sample_tracking]ALUR.snp"
   snapshot allocation is 10 pages
   snapshot extent is 2 pages

create storage area ALTH
   filename "rdb_data:[sample_tracking]ALTH"
   allocation is 1100 pages
   page size is 1
   page format is uniform
   extent is 111
   snapshot filename "rdb_snap:[sample_tracking]ALTH.snp"
   snapshot allocation is 10 pages
   snapshot extent is 2 pages

create storage area ALPL
   filename "rdb_data:[sample_tracking]ALPL"
   allocation is 1100 pages
```

```
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]ALPL.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area ALAMSAPU
    filename "rdb_data:[sample_tracking]ALAMSAPU"
    allocation is 1100 pages
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]ALAMSAPU.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area ALAMWIPU
    filename "rdb_data:[sample_tracking]ALAMWIPU"
    allocation is 1100 pages
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]ALAMWIPU.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area ALTOSP
    filename "rdb_data:[sample_tracking]ALTOSP"
    allocation is 1100 pages
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]ALTOSP.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area ALOT
    filename "rdb_data:[sample_tracking]ALOT"
    allocation is 1100 pages
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]ALOT.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area BEST90
    filename "rdb_data:[sample_tracking]BEST90"
    allocation is 1100 pages
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]BEST90.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area BEST89AN90
    filename "rdb_data:[sample_tracking]BEST89AN90"
    allocation is 1100 pages
```

```
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]BEST89AN90.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area BETOST
    filename "rdb_data:[sample_tracking]BETOST"
    allocation is 1100 pages
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]BETOST.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area BETR
    filename "rdb_data:[sample_tracking]BETR"
    allocation is 1100 pages
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]BETR.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area BEOT
    filename "rdb_data:[sample_tracking]BEOT"
    allocation is 1100 pages
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]BEOT.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area GASC
    filename "rdb_data:[sample_tracking]GASC"
    allocation is 4400 pages
    page size is 1
    page format is uniform
    extent is 441
    snapshot filename "rdb_snap:[sample_tracking]GASC.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area GAFUIS
    filename "rdb_data:[sample_tracking]GAFUIS"
    allocation is 4400 pages
    page size is 1
    page format is uniform
    extent is 441
    snapshot filename "rdb_snap:[sample_tracking]GAFUIS.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area GAOT
    filename "rdb_data:[sample_tracking]GAOT"
    allocation is 4400 pages
```

```
    page size is 1
    page format is uniform
    extent is 441
    snapshot filename "rdb_snap:[sample_tracking]GAOT.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area GRALBEAIFI
    filename "rdb_data:[sample_tracking]GRALBEAIFI"
    allocation is 1100 pages
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]GRALBEAIFI.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area GRALBEOT
    filename "rdb_data:[sample_tracking]GRALBEOT"
    allocation is 1100 pages
    page size is 1
    page format is uniform
    extent is 111
    snapshot filename "rdb_snap:[sample_tracking]GRALBEOT.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area SPIN
    filename "rdb_data:[sample_tracking]SPIN"
    allocation is 1466 pages
    page size is 6
    page format is uniform
    extent is 147
    snapshot filename "rdb_snap:[sample_tracking]SPIN.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages

create storage area TR
    filename "rdb_data:[sample_tracking]TR"
    allocation is 4400 pages
    page size is 3
    page format is uniform
    extent is 441
    snapshot filename "rdb_snap:[sample_tracking]TR.snp"
    snapshot allocation is 10 pages
    snapshot extent is 2 pages
    ;

!
! Create the domains, including defaults, constraints, and comments.
!
create domain USER_NAME char(30)
    default NULL;

comment on domain USER_NAME
    is 'Domain for user name.';

create domain VMS_DATE_TIME char(18)
    default NULL;
```

```
comment on domain VMS_DATE_TIME
   is 'Domain for VMS date (dd-mmm-yy) and time (hh:mm:ss).';

create domain CHARACTER_TIME_HH_MM char(5)
   default NULL;

comment on domain CHARACTER_TIME_HH_MM
   is 'Domain for time in the form hh:mm.';

create domain CHARGE_NUMBERS char(9)
   default 'PF7400000';

comment on domain CHARGE_NUMBERS
   is 'Domain for nine-alphanumeric-long charge numbers.';

create domain FIFTY_CHARACTERS char(50)
   default NULL;

comment on domain FIFTY_CHARACTERS
   is 'Domain for fifty-alphanumeric-long fields.';

create domain FLAGS char(1)
   default 'N';

comment on domain FLAGS
   is 'Domain for one-character-long yes/no flags.';

create domain FORTY_CHARACTERS char(40)
   default NULL;

comment on domain FORTY_CHARACTERS
   is 'Domain for forty-alphanumeric-long fields.';

create domain IDENTIFICATION_CODES char(12)
   default NULL;

comment on domain IDENTIFICATION_CODES
   is 'Domain for twelve-character-long identification codes.';

create domain INTEGER_DATE_COUNT integer
   default NULL;

comment on domain INTEGER_DATE_COUNT
   is 'Domain for integer*4 date count starting at January 1, 1990.';

create domain INTEGER_NUMBERS integer
   default NULL;

comment on domain INTEGER_NUMBERS
   is 'Domain for integer*4 numbers.';

create domain NAMES char(25)
   default NULL;

comment on domain NAMES
   is 'Domain for twenty-five-alphanumeric-long names.';

create domain PHONE_NUMBERS char(12)
   default NULL;
```

```
comment on domain PHONE_NUMBERS
   is 'Domain for twelve-alphanumeric-long phone numbers.';

create domain REAL_NUMBERS real
   default NULL;

comment on domain REAL_NUMBERS
   is 'Domain for real*4 numbers.';

create domain RML_SPECTRAL_ID_CODES char(14)
   default NULL;

comment on domain RML_SPECTRAL_ID_CODES
   is 'Domain for fourteen-character-long RML spectral identification
codes (<save area>$<spectral id>).';

create domain SIXTEEN_CHARACTERS char(16)
   default NULL;

comment on domain SIXTEEN_CHARACTERS
   is 'Domain for sixteen-alphanumeric-long fields.';

create domain SIXTY_CHARACTERS char(60)
   default NULL;

comment on domain SIXTY_CHARACTERS
   is 'Domain for sixty-alphanumeric-long fields.';

create domain TEN_CHARACTERS char(10)
   default NULL;

comment on domain TEN_CHARACTERS
   is 'Domain for ten-alphanumeric-long fields.';

create domain TWENTY_CHARACTERS char(20)
   default NULL;

comment on domain TWENTY_CHARACTERS
   is 'Domain for twenty-alphanumeric-long fields.';

create domain UNITS char(5)
   default NULL;

comment on domain UNITS
   is 'Domain for five-alphanumeric-long units.';

!
! Create tables.
!
create table SAMPLE_INFORMATION (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      DATE_SAMPLE_ENTERED              INTEGER_DATE_COUNT,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      CUSTOMER_SAMPLE_ID               IDENTIFICATION_CODES,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      CUSTOMER_PROJECT_NAME            FORTY_CHARACTERS,
      WORK_CHARGE_NUMBER               CHARGE_NUMBERS,
      CUSTOMER_SAMPLE_NAME             SIXTY_CHARACTERS,
      SAMPLE_TYPE                      TEN_CHARACTERS,
      SAMPLE_SIZE                      REAL_NUMBERS,
```

```
        SAMPLE_SIZE_UNITS                UNITS,
        SAMPLE_COLLECTION_DATE           INTEGER_DATE_COUNT,
        SAMPLE_COLLECTION_TIME           CHARACTER_TIME_HH_MM,
        ANY_SPECIAL_INSTRUCTIONS         FLAGS,
        SAMPLE_HP_SURVEYED               FLAGS,
        SAMPLE_HP_SURVEY_ACTIVITY        REAL_NUMBERS,
        SAMPLE_SUBMITTER_NAME            NAMES,
        SAMPLE_SUBMITTER_PHONE_NUMBER    PHONE_NUMBERS,
        SAMPLE_SUBMITTER_ADDRESS         SIXTY_CHARACTERS,
        TECHNICAL_CONTACT_NAME           NAMES,
        TECHNICAL_CONTACT_PHONE_NUMBER   PHONE_NUMBERS,
        TECHNICAL_CONTACT_ADDRESS        SIXTY_CHARACTERS,
        SEND_RESULTS_TO_NAME             NAMES,
        SEND_RESULTS_TO_PHONE_NUMBER     PHONE_NUMBERS,
        SEND_RESULTS_TO_ADDRESS          SIXTY_CHARACTERS,
        SAMPLE_PICKUP_BY_NAME            NAMES,
        SAMPLE_PICKUP_BY_PHONE_NUMBER    PHONE_NUMBERS,
        SAMPLE_PICKUP_BY_ADDRESS         SIXTY_CHARACTERS,
        NEED_GROSS_ALPHA_BETA            FLAGS,
        NEED_ALPHA                       FLAGS,
        NEED_BETA                        FLAGS,
        NEED_GAMMA                       FLAGS,
        NEXT_SPECIAL_INSTRUCTION_LINE    INTEGER_NUMBERS,
        NEXT_POSSESSOR_SEQUENCE_NUMBER   INTEGER_NUMBERS,
        NEED_GAMMA_SCREEN                FLAGS,
        NEED_GAMMA_FULL_ISOTOPIC         FLAGS,
        NEED_GAMMA_OTHER                 FLAGS,
        NEED_GROSS_ALPHA_BETA_FILTERS    FLAGS,
        NEED_GROSS_ALPHA_BETA_OTHER      FLAGS,
        NEED_BETA_STRONTIUM_90           FLAGS,
        NEED_BETA_STRONTIUM_89_AND_90    FLAGS,
        NEED_BETA_TOTAL_STRONTIUM        FLAGS,
        NEED_BETA_TRITIUM                FLAGS,
        NEED_BETA_OTHER                  FLAGS,
        NEED_ALPHA_URANIUM               FLAGS,
        NEED_ALPHA_THORIUM               FLAGS,
        NEED_ALPHA_PLUTONIUM             FLAGS,
        NEED_ALPHA_AM241_SANS_PU238      FLAGS,
        NEED_ALPHA_AM241_WITH_PU238      FLAGS,
        NEED_ALPHA_TOTAL_SPECTROMETRIC   FLAGS,
        NEED_ALPHA_OTHER                 FLAGS,
        USER_CREATING_OR_MODIFYING       USER_NAME,
        DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
        CHAIN_OF_CUSTODY_NUMBER          TEN_CHARACTERS,
    primary key (SAMPLE_TRACKING_ID) );

comment on table SAMPLE_INFORMATION
    is 'Table for sample information.';

create unique index SAIN_PRIMARY_KEY_INDEX
    on SAMPLE_INFORMATION (SAMPLE_TRACKING_ID)
    type is sorted
    node size 104
    store in SAIN;

comment on index SAIN_PRIMARY_KEY_INDEX
    is 'Primary key index for table SAMPLE_INFORMATION.';

create storage map SAIN_map for SAMPLE_INFORMATION
    store in SAIN
```

```
   placement via index SAIN_PRIMARY_KEY_INDEX
   disable compression;

create table ALPHA_URANIUM (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table ALPHA_URANIUM
   is 'Table for requested alpha analyses for Uranium radionuclides.';

create unique index ALUR_PRIMARY_KEY_INDEX
   on ALPHA_URANIUM (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in ALUR;

comment on index ALUR_PRIMARY_KEY_INDEX
   is 'Primary key index for table ALPHA_URANIUM.';

create storage map ALUR_map for ALPHA_URANIUM
   store in ALUR
   placement via index ALUR_PRIMARY_KEY_INDEX
   disable compression;

create table ALPHA_THORIUM (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table ALPHA_THORIUM
   is 'Table for requested alpha analyses for Thorium radionuclides.';

create unique index ALTH_PRIMARY_KEY_INDEX
   on ALPHA_THORIUM (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in ALTH;

comment on index ALTH_PRIMARY_KEY_INDEX
   is 'Primary key index for table ALPHA_THORIUM.';

create storage map ALTH_map for ALPHA_THORIUM
   store in ALTH
   placement via index ALTH_PRIMARY_KEY_INDEX
   disable compression;
```

```
create table ALPHA_PLUTONIUM (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table ALPHA_PLUTONIUM
   is 'Table for requested alpha analyses for Plutonium radionuclides.';

create unique index ALPL_PRIMARY_KEY_INDEX
   on ALPHA_PLUTONIUM (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in ALPL;

comment on index ALPL_PRIMARY_KEY_INDEX
   is 'Primary key index for table ALPHA_PLUTONIUM.';

create storage map ALPL_map for ALPHA_PLUTONIUM
   store in ALPL
   placement via index ALPL_PRIMARY_KEY_INDEX
   disable compression;

create table ALPHA_AM241_SANS_PU238 (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table ALPHA_AM241_SANS_PU238
   is 'Table for requested alpha analyses for Americium-241 separate from
Plutonium-238.';

create unique index ALAMSAPU_PRIMARY_KEY_INDEX
   on ALPHA_AM241_SANS_PU238 (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in ALAMSAPU;

comment on index ALAMSAPU_PRIMARY_KEY_INDEX
   is 'Primary key index for table ALPHA_AM241_SANS_PU238.';

create storage map ALAMSAPU_map for ALPHA_AM241_SANS_PU238
   store in ALAMSAPU
   placement via index ALAMSAPU_PRIMARY_KEY_INDEX
   disable compression;
```

```
create table ALPHA_AM241_WITH_PU238 (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table ALPHA_AM241_WITH_PU238
   is 'Table for requested alpha analyses for Americium-241 combined with
Plutonium-238.';

create unique index ALAMWIPU_PRIMARY_KEY_INDEX
   on ALPHA_AM241_WITH_PU238 (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in ALAMWIPU;

comment on index ALAMWIPU_PRIMARY_KEY_INDEX
   is 'Primary key index for table ALPHA_AM241_WITH_PU238.';

create storage map ALAMWIPU_map for ALPHA_AM241_WITH_PU238
   store in ALAMWIPU
   placement via index ALAMWIPU_PRIMARY_KEY_INDEX
   disable compression;

create table ALPHA_TOTAL_SPECTROMETRIC (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table ALPHA_TOTAL_SPECTROMETRIC
   is 'Table for requested total spectrometric alpha analyses.';

create unique index ALTOSP_PRIMARY_KEY_INDEX
   on ALPHA_TOTAL_SPECTROMETRIC (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in ALTOSP;

comment on index ALTOSP_PRIMARY_KEY_INDEX
   is 'Primary key index for table ALPHA_TOTAL_SPECTROMETRIC.';

create storage map ALTOSP_map for ALPHA_TOTAL_SPECTROMETRIC
   store in ALTOSP
   placement via index ALTOSP_PRIMARY_KEY_INDEX
   disable compression;

create table ALPHA_OTHER (
```

```
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table ALPHA_OTHER
   is 'Table for requested alpha analyses for other, unlisted
radionuclides.';

create unique index ALOT_PRIMARY_KEY_INDEX
   on ALPHA_OTHER (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in ALOT;

comment on index ALOT_PRIMARY_KEY_INDEX
   is 'Primary key index for table ALPHA_OTHER.';

create storage map ALOT_map for ALPHA_OTHER
   store in ALOT
   placement via index ALOT_PRIMARY_KEY_INDEX
   disable compression;

create table BETA_STRONTIUM_90 (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      DETECTOR_COUNT_LENGTH            REAL_NUMBERS,
      DETECTOR_COUNT_LENGTH_UNITS      UNITS,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      RESULTS_NEEDED_BY_TIME           CHARACTER_TIME_HH_MM,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table BETA_STRONTIUM_90
   is 'Table for requested beta analyses for Strontium-90.';

create unique index BEST90_PRIMARY_KEY_INDEX
   on BETA_STRONTIUM_90 (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in BEST90;

comment on index BEST90_PRIMARY_KEY_INDEX
   is 'Primary key index for table BETA_STRONTIUM_90.';

create storage map BEST90_map for BETA_STRONTIUM_90
   store in BEST90
   placement via index BEST90_PRIMARY_KEY_INDEX
   disable compression;
```

```
create table BETA_STRONTIUM_89_AND_90 (
      SAMPLE_TRACKING_ID          IDENTIFICATION_CODES      not null,
      DETECTOR_COUNT_LENGTH        REAL_NUMBERS,
      DETECTOR_COUNT_LENGTH_UNITS  UNITS,
      RESULTS_NEEDED_BY_DATE       INTEGER_DATE_COUNT,
      RESULTS_NEEDED_BY_TIME       CHARACTER_TIME_HH_MM,
      DATE_ALL_WORK_COMPLETED      INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION      TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS     FLAGS,
      USER_CREATING_OR_MODIFYING   USER_NAME,
      DATE_CREATED_OR_MODIFIED     VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table BETA_STRONTIUM_89_AND_90
   is 'Table for requested beta analyses for Strontium-89 and -90.';

create unique index BEST89AN90_PRIMARY_KEY_INDEX
   on BETA_STRONTIUM_89_AND_90 (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in BEST89AN90;

comment on index BEST89AN90_PRIMARY_KEY_INDEX
   is 'Primary key index for table BETA_STRONTIUM_89_AND_90.';

create storage map BEST89AN90_map for BETA_STRONTIUM_89_AND_90
   store in BEST89AN90
   placement via index BEST89AN90_PRIMARY_KEY_INDEX
   disable compression;

create table BETA_TOTAL_STRONTIUM (
      SAMPLE_TRACKING_ID          IDENTIFICATION_CODES      not null,
      DETECTOR_COUNT_LENGTH        REAL_NUMBERS,
      DETECTOR_COUNT_LENGTH_UNITS  UNITS,
      RESULTS_NEEDED_BY_DATE       INTEGER_DATE_COUNT,
      RESULTS_NEEDED_BY_TIME       CHARACTER_TIME_HH_MM,
      DATE_ALL_WORK_COMPLETED      INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION      TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS     FLAGS,
      USER_CREATING_OR_MODIFYING   USER_NAME,
      DATE_CREATED_OR_MODIFIED     VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table BETA_TOTAL_STRONTIUM
   is 'Table for requested beta analyses for total Strontium.';

create unique index BETOST_PRIMARY_KEY_INDEX
   on BETA_TOTAL_STRONTIUM (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in BETOST;

comment on index BETOST_PRIMARY_KEY_INDEX
   is 'Primary key index for table BETA_TOTAL_STRONTIUM.';
```

```
create storage map BETOST_map for BETA_TOTAL_STRONTIUM
   store in BETOST
   placement via index BETOST_PRIMARY_KEY_INDEX
   disable compression;

create table BETA_TRITIUM (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      DETECTOR_COUNT_LENGTH            REAL_NUMBERS,
      DETECTOR_COUNT_LENGTH_UNITS      UNITS,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      RESULTS_NEEDED_BY_TIME           CHARACTER_TIME_HH_MM,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table BETA_TRITIUM
   is 'Table for requested beta analyses for Tritium.';

create unique index BETR_PRIMARY_KEY_INDEX
   on BETA_TRITIUM (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in BETR;

comment on index BETR_PRIMARY_KEY_INDEX
   is 'Primary key index for table BETA_TRITIUM.';

create storage map BETR_map for BETA_TRITIUM
   store in BETR
   placement via index BETR_PRIMARY_KEY_INDEX
   disable compression;

create table BETA_OTHER (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      DETECTOR_COUNT_LENGTH            REAL_NUMBERS,
      DETECTOR_COUNT_LENGTH_UNITS      UNITS,
      NEED_BETA_NICKEL_63              FLAGS,
      NEED_BETA_IRON_55                FLAGS,
      NEED_BETA_SULFUR_35              FLAGS,
      NEED_BETA_PLUTONIUM_241          FLAGS,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      RESULTS_NEEDED_BY_TIME           CHARACTER_TIME_HH_MM,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table BETA_OTHER
   is 'Table for requested beta analyses for other, unlisted
radionuclides.';
```

```
create unique index BEOT_PRIMARY_KEY_INDEX
   on BETA_OTHER (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in BEOT;

comment on index BEOT_PRIMARY_KEY_INDEX
   is 'Primary key index for table BETA_OTHER.';

create storage map BEOT_map for BETA_OTHER
   store in BEOT
   placement via index BEOT_PRIMARY_KEY_INDEX
   disable compression;

create table GAMMA_SCREEN (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      DETECTOR_COUNT_LENGTH            REAL_NUMBERS,
      DETECTOR_COUNT_LENGTH_UNITS      UNITS,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      RESULTS_NEEDED_BY_TIME           CHARACTER_TIME_HH_MM,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      DATE_SAMPLE_COUNTED              INTEGER_DATE_COUNT,
      RML_SPECTRAL_ID                  RML_SPECTRAL_ID_CODES,
      IS_THIS_A_SPECTRUM_RECOUNT       FLAGS,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table GAMMA_SCREEN
   is 'Table for requested screening and shipping gamma analyses.';

create unique index GASC_PRIMARY_KEY_INDEX
   on GAMMA_SCREEN (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in GASC;

comment on index GASC_PRIMARY_KEY_INDEX
   is 'Primary key index for table GAMMA_SCREEN.';

create storage map GASC_map for GAMMA_SCREEN
   store in GASC
   placement via index GASC_PRIMARY_KEY_INDEX
   disable compression;

create table GAMMA_FULL_ISOTOPIC (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      DETECTOR_COUNT_LENGTH            REAL_NUMBERS,
      DETECTOR_COUNT_LENGTH_UNITS      UNITS,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      RESULTS_NEEDED_BY_TIME           CHARACTER_TIME_HH_MM,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      DATE_SAMPLE_COUNTED              INTEGER_DATE_COUNT,
      RML_SPECTRAL_ID                  RML_SPECTRAL_ID_CODES,
      IS_THIS_A_SPECTRUM_RECOUNT       FLAGS,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
```

```
      ANY_SPECIAL_INSTRUCTIONS          FLAGS,
      USER_CREATING_OR_MODIFYING        USER_NAME,
      DATE_CREATED_OR_MODIFIED          VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table GAMMA_FULL_ISOTOPIC
   is 'Table for requested full isotopic gamma analyses.';

create unique index GAFUIS_PRIMARY_KEY_INDEX
   on GAMMA_FULL_ISOTOPIC (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in GAFUIS;

comment on index GAFUIS_PRIMARY_KEY_INDEX
   is 'Primary key index for table GAMMA_FULL_ISOTOPIC.';

create storage map GAFUIS_map for GAMMA_FULL_ISOTOPIC
   store in GAFUIS
   placement via index GAFUIS_PRIMARY_KEY_INDEX
   disable compression;

create table GAMMA_OTHER (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      DETECTOR_COUNT_LENGTH            REAL_NUMBERS,
      DETECTOR_COUNT_LENGTH_UNITS      UNITS,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      RESULTS_NEEDED_BY_TIME           CHARACTER_TIME_HH_MM,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      DATE_SAMPLE_COUNTED              INTEGER_DATE_COUNT,
      RML_SPECTRAL_ID                  RML_SPECTRAL_ID_CODES,
      IS_THIS_A_SPECTRUM_RECOUNT       FLAGS,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table GAMMA_OTHER
   is 'Table for requested gamma analyses for other, unlisted
radionuclides.';

create unique index GAOT_PRIMARY_KEY_INDEX
   on GAMMA_OTHER (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in GAOT;

comment on index GAOT_PRIMARY_KEY_INDEX
   is 'Primary key index for table GAMMA_OTHER.';

create storage map GAOT_map for GAMMA_OTHER
   store in GAOT
   placement via index GAOT_PRIMARY_KEY_INDEX
   disable compression;
```

```
create table GROSS_ALPHA_BETA_AIR_FILTERS (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not nu_ ,
      DETECTOR_COUNT_LENGTH            REAL_NUMBERS,
      DETECTOR_COUNT_LENGTH_UNITS      UNITS,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      RESULTS_NEEDED_BY_TIME           CHARACTER_TIME_HH_MM,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table GROSS_ALPHA_BETA_AIR_FILTERS
   is 'Table for requested gross alpha-beta analyses of air filters.';

create unique index GRALBEAIFI_PRIMARY_KEY_INDEX
   on GROSS_ALPHA_BETA_AIR_FILTERS (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in GRALBEAIFI;

comment on index GRALBEAIFI_PRIMARY_KEY_INDEX
   is 'Primary key index for table GROSS_ALPHA_BETA_AIR_FILTERS.';

create storage map GRALBEAIFI_map for GROSS_ALPHA_BETA_AIR_FILTERS
   store in GRALBEAIFI
   placement via index GRALBEAIFI_PRIMARY_KEY_INDEX
   disable compression;

create table GROSS_ALPHA_BETA_OTHER (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      DETECTOR_COUNT_LENGTH            REAL_NUMBERS,
      DETECTOR_COUNT_LENGTH_UNITS      UNITS,
      RESULTS_NEEDED_BY_DATE           INTEGER_DATE_COUNT,
      RESULTS_NEEDED_BY_TIME           CHARACTER_TIME_HH_MM,
      DATE_ALL_WORK_COMPLETED          INTEGER_DATE_COUNT,
      RESULTS_REPORT_CITATION          TWENTY_CHARACTERS,
      ANY_SPECIAL_INSTRUCTIONS         FLAGS,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table GROSS_ALPHA_BETA_OTHER
   is 'Table for requested gross alpha-beta analyses of other, unlisted
samples.';

create unique index GRALBEOT_PRIMARY_KEY_INDEX
   on GROSS_ALPHA_BETA_OTHER (SAMPLE_TRACKING_ID)
   type is sorted
   node size 104
   store in GRALBEOT;

comment on index GRALBEOT_PRIMARY_KEY_INDEX
   is 'Primary key index for table GROSS_ALPHA_BETA_OTHER.';
```

```
create storage map GRALBEOT_map for GROSS_ALPHA_BETA_OTHER
   store in GRALBEOT
   placement via index GRALBEOT_PRIMARY_KEY_INDEX
   disable compression;

create table SPECIAL_INSTRUCTIONS (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      ORIGIN_OF_SPECIAL_INSTRUCTION   SIXTEEN_CHARACTERS       not null,
      SPECIAL_INSTRUCTION              FIFTY_CHARACTERS,
      SPECIAL_INSTRUCTION_LINE        INTEGER_NUMBERS          not null,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID, ORIGIN_OF_SPECIAL_INSTRUCTION,
                SPECIAL_INSTRUCTION_LINE),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table SPECIAL_INSTRUCTIONS
   is 'Table for special instructions.';

create unique index SPIN_PRIMARY_KEY_INDEX
    on    SPECIAL_INSTRUCTIONS    (SAMPLE_TRACKING_ID,
ORIGIN_OF_SPECIAL_INSTRUCTION, SPECIAL_INSTRUCTION_LINE)
   type is sorted
   node size 170
   store in SPIN;

comment on index SPIN_PRIMARY_KEY_INDEX
   is 'Primary key index for table SPECIAL_INSTRUCTIONS.';

create storage map SPIN_map for SPECIAL_INSTRUCTIONS
   store in SPIN
   placement via index SPIN_PRIMARY_KEY_INDEX
   disable compression;

create table TRACKING (
      SAMPLE_TRACKING_ID              IDENTIFICATION_CODES      not null,
      POSSESSOR_SEQUENCE_NUMBER       INTEGER_NUMBERS          not null,
      POSSESSOR_NAME                   NAMES,
      POSSESSOR_PHONE_NUMBER           PHONE_NUMBERS,
      POSSESSOR_ADDRESS                SIXTY_CHARACTERS,
      POSSESSOR_POSSESSION_REASON      FIFTY_CHARACTERS,
      DATE_ACCEPTED_BY_POSSESSOR       INTEGER_DATE_COUNT,
      DATE_RELINQUISHED_BY_POSSESSOR   INTEGER_DATE_COUNT,
      USER_CREATING_OR_MODIFYING       USER_NAME,
      DATE_CREATED_OR_MODIFIED         VMS_DATE_TIME,
   primary key (SAMPLE_TRACKING_ID, POSSESSOR_SEQUENCE_NUMBER),
   foreign key (SAMPLE_TRACKING_ID)
      references SAMPLE_INFORMATION (SAMPLE_TRACKING_ID) );

comment on table TRACKING
   is 'Table for tracking samples.';

create unique index TR_PRIMARY_KEY_INDEX
   on TRACKING (SAMPLE_TRACKING_ID, POSSESSOR_SEQUENCE_NUMBER)
   type is sorted
   node size 119
   store in TR;

comment on index TR_PRIMARY_KEY_INDEX
```

```
   is 'Primary key index for table TRACKING.';

create storage map TR_map for TRACKING
   store in TR
   placement via index TR_PRIMARY_KEY_INDEX
   disable compression;

commit;

exit;
```

B. LOTUS® 1-2-3® PRINT FILES CONTAINING THE ENTITY- AND SCREEN-SORTED ATTRIBUTE DESCRIPTION LISTS

The series of blank lines which appear regularly throughout these two files are a result of these being Lotus® 1-2-3® print files; the series of blank lines correspond to page throws. The tool CREATE_SCHEMA ignores these page throws.

B.1 Entity-Sorted List

Rdb Entity-Sorted Worksheet - duplicate entries retained for cursor generation - dashes demarcate entities.

Entity	Attribute	Key	Screens	#	Domain
sample_information	.sample_tracking_id	p	m	1	identification_codes
sample_information	.date_sample_entered		m	2	integer_date_count
sample_information	.date_all_work_completed		m	3	integer_date_count
sample_information	.customer_sample_id		m	4	identification_codes
sample_information	.results_needed_by_date		m	5	integer_date_count
sample_information	.customer_project_name		m	6	forty_characters
sample_information	.work_charge_number		m	7	charge_numbers
sample_information	.customer_sample_name		m	8	sixty_characters
sample_information	.sample_type		m	9	ten_characters
sample_information	.sample_size		m	10	real_numbers
sample_information	.sample_size_units		m	11	units
sample_information	.sample_collection_date		m	12	integer_date_count
sample_information	.sample_collection_time		m	13	character_time_hh_mm
sample_information	.any_special_instructions		m	14	flags
sample_information	.sample_hp_surveyed		m	15	flags
sample_information	.sample_hp_survey_activity		m	16	real_numbers
sample_information	.sample_submitter_name		m	17	names
sample_information	.sample_submitter_phone_number		m	18	phone_numbers
sample_information	.sample_submitter_address		m	19	sixty_characters
sample_information	.technical_contact_name		m	20	names
sample_information	.technical_contact_phone_number		m	21	phone_numbers
sample_information	.technical_contact_address		m	22	sixty_characters
sample_information	.send_results_to_name		m	23	names
sample_information	.send_results_to_phone_number		m	24	phone_numbers
sample_information	.send_results_to_address		m	25	sixty_characters
sample_information	.sample_pickup_by_name		m	26	names
sample_information	.sample_pickup_by_phone_number		m	27	phone_numbers
sample_information	.sample_pickup_by_address		m	28	sixty_characters
tracking	.possessor_name	d	m	29	names
tracking	.possessor_phone_number	d	m	30	phone_numbers
tracking	.possessor_address	d	m	31	sixty_characters
sample_information	.chain_of_custody_number		m	32	ten_characters
sample_information	.need_gross_alpha_beta		m	33	flags
sample_information	.need_alpha		m	34	flags
sample_information	.need_beta		m	35	flags
sample_information	.need_gamma		m	36	flags
sample_information	.next_special_instruction_line		m	nd	integer_numbers
sample_information	.next_possessor_sequence_number		m	nd	integer_numbers
sample_information	.need_gamma_screen		g	3	flags
sample_information	.need_gamma_full_isotopic		g	11	flags
sample_information	.need_gamma_other		g	19	flags
sample_information	.need_gross_alpha_beta_filters		c	3	flags
sample_information	.need_gross_alpha_beta_other		c	11	flags
sample_information	.need_beta_strontium_90		b	3	flags

Domain
*
Domain
character_time_hh_mm
charge_numbers
fifty_characters
flags
forty_characters
identification_codes
integer_date_count
integer_numbers
names
phone_numbers
real_numbers
rml_spectral_id_codes
sixteen_characters
sixty_characters
ten_characters
twenty_characters
units

```
sample_information          .need_beta_strontium_89_and_90      b       11 flags
sample_information          .need_beta_total_strontium          b       19 flags
sample_information          .need_beta_tritium                  b       27 flags
sample_information          .need_beta_other                    b       35 flags
sample_information          .need_alpha_uranium                 a        3 flags
sample_information          .need_alpha_thorium                 a        8 flags
sample_information          .need_alpha_plutonium               a       13 flags
sample_information          .need_alpha_am241_sans_pu238        a       18 flags
sample_information          .need_alpha_am241_with_pu238        a       23 flags
sample_information          .need_alpha_total_spectrometric     a       28 flags
sample_information          .need_alpha_other                   a       33 flags
-----------------------------
alpha_uranium               .sample_tracking_id          f/p    a        1 identification_codes



sample_information          .customer_sample_name        d      a        2 sixty_characters
alpha_uranium               .results_needed_by_date             a        4 integer_date_count
alpha_uranium               .date_all_work_completed            a        5 integer_date_count
alpha_uranium               .results_report_citation            a        6 twenty_characters
alpha_uranium               .any_special_instructions           a        7 flags
-----------------------------
alpha_thorium               .sample_tracking_id          f/p    a        1 identification_codes
sample_information          .customer_sample_name        d      a        2 sixty_characters
alpha_thorium               .results_needed_by_date             a        9 integer_date_count
alpha_thorium               .date_all_work_completed            a       10 integer_date_count
alpha_thorium               .results_report_citation            a       11 twenty_characters
alpha_thorium               .any_special_instructions           a       12 flags
-----------------------------
alpha_plutonium             .sample_tracking_id          f/p    a        1 identification_codes
sample_information          .customer_sample_name        d      a        2 sixty_characters
alpha_plutonium             .results_needed_by_date             a       14 integer_date_count
alpha_plutonium             .date_all_work_completed            a       15 integer_date_count
alpha_plutonium             .results_report_citation            a       16 twenty_characters
alpha_plutonium             .any_special_instructions           a       17 flags
-----------------------------
alpha_am241_sans_pu238      .sample_tracking_id          f/p    a        1 identification_codes
sample_information          .customer_sample_name        d      a        2 sixty_characters
alpha_am241_sans_pu238      .results_needed_by_date             a       19 integer_date_count
alpha_am241_sans_pu238      .date_all_work_completed            a       20 integer_date_count
alpha_am241_sans_pu238      .results_report_citation            a       21 twenty_characters
alpha_am241_sans_pu238      .any_special_instructions           a       22 flags
-----------------------------
alpha_am241_with_pu238      .sample_tracking_id          f/p    a        1 identification_codes
sample_information          .customer_sample_name        d      a        2 sixty_characters
alpha_am241_with_pu238      .results_needed_by_date             a       24 integer_date_count
alpha_am241_with_pu238      .date_all_work_completed            a       25 integer_date_count
alpha_am241_with_pu238      .results_report_citation            a       26 twenty_characters
alpha_am241_with_pu238      .any_special_instructions           a       27 flags
-----------------------------
alpha_total_spectrometric   .sample_tracking_id          f/p    a        1 identification_codes
sample_information          .customer_sample_name        d      a        2 sixty_characters
alpha_total_spectrometric   .results_needed_by_date             a       29 integer_date_count
alpha_total_spectrometric   .date_all_work_completed            a       30 integer_date_count
alpha_total_spectrometric   .results_report_citation            a       31 twenty_characters
alpha_total_spectrometric   .any_special_instructions           a       32 flags
-----------------------------
alpha_other                 .sample_tracking_id          f/p    a        1 identification_codes
sample_information          .customer_sample_name        d      a        2 sixty_characters
alpha_other                 .results_needed_by_date             a       34 integer_date_count
alpha_other                 .date_all_work_completed            a       35 integer_date_count
alpha_other                 .results_report_citation            a       36 twenty_characters
alpha_other                 .any_special_instructions           a       37 flags
```

beta_strontium_90	.sample_tracking_id	f/p	b	1	identification_codes
sample_information	.customer_sample_name	d	b	2	sixty_characters
beta_strontium_90	.detector_count_length		b	4	real_numbers
beta_strontium_90	.detector_count_length_units		b	5	units
beta_strontium_90	.results_needed_by_date		b	6	integer_date_count
beta_strontium_90	.results_needed_by_time		b	7	character_time_hh_mm
beta_strontium_90	.date_all_work_completed		b	8	integer_date_count
beta_strontium_90	.results_report_citation		b	9	twenty_characters
beta_strontium_90	.any_special_instructions		b	10	flags
beta_strontium_89_and_90	.sample_tracking_id	f/p	b	1	identification_codes
sample_information	.customer_sample_name	d	b	2	sixty_characters
beta_strontium_89_and_90	.detector_count_length		b	12	real_numbers
beta_strontium_89_and_90	.detector_count_length_units		b	13	units
beta_strontium_89_and_90	.results_needed_by_date		b	14	integer_date_count
beta_strontium_89_and_90	.results_needed_by_time		b	15	character_time_hh_mm
beta_strontium_89_and_90	.date_all_work_completed		b	16	integer_date_count
beta_strontium_89_and_90	.results_report_citation		b	17	twenty_characters
beta_strontium_89_and_90	.any_special_instructions		b	18	flags
beta_total_strontium	.sample_tracking_id	f/p	b	1	identification_codes
sample_information	.customer_sample_name	d	b	2	sixty_characters
beta_total_strontium	.detector_count_length		b	20	real_numbers
beta_total_strontium	.detector_count_length_units		b	21	units
beta_total_strontium	.results_needed_by_date		b	22	integer_date_count
beta_total_strontium	.results_needed_by_time		b	23	character_time_hh_mm
beta_total_strontium	.date_all_work_completed		b	24	integer_date_count
beta_total_strontium	.results_report_citation		b	25	twenty_characters
beta_total_strontium	.any_special_instructions		b	26	flags
beta_tritium	.sample_tracking_id	f/p	b	1	identification_codes
sample_information	.customer_sample_name	d	b	2	sixty_characters
beta_tritium	.detector_count_length		b	28	real_numbers
beta_tritium	.detector_count_length_units		b	29	units
beta_tritium	.results_needed_by_date		b	30	integer_date_count
beta_tritium	.results_needed_by_time		b	31	character_time_hh_mm
beta_tritium	.date_all_work_completed		b	32	integer_date_count
beta_tritium	.results_report_citation		b	33	twenty_characters
beta_tritium	.any_special_instructions		b	34	flags
beta_other	.sample_tracking_id	f/p	b	1	identification_codes
sample_information	.customer_sample_name	d	b	2	sixty_characters
beta_other	.detector_count_length		b	36	real_numbers
beta_other	.detector_count_length_units		b	37	units
beta_other	.need_beta_nickel_63		b	38	flags
beta_other	.need_beta_iron_55		b	39	flags
beta_other	.need_beta_sulfur_35		b	40	flags
beta_other	.need_beta_plutonium_241		b	41	flags
beta_other	.results_needed_by_date		b	42	integer_date_count
beta_other	.results_needed_by_time		b	43	character_time_hh_mm
beta_other	.date_all_work_completed		b	44	integer_date_count
beta_other	.results_report_citation		b	45	twenty_characters
beta_other	.any_special_instructions		b	46	flags
gamma_screen	.sample_tracking_id	f/p	g	1	identification_codes
sample_information	.customer_sample_name	d	g	2	sixty_characters
gamma_screen	.detector_count_length		g	4	real_numbers
gamma_screen	.detector_count_length_units		g	5	units
gamma_screen	.results_needed_by_date		g	6	integer_date_count

gamma_screen	.results_needed_by_time		g	7 character_time_hh_mm
gamma_screen	.date_all_work_completed		g	8 integer_date_count
gamma_screen	.date_sample_counted		g	9 integer_date_count
gamma_screen	.rml_spectral_id		g	10 rml_spectral_id_codes
gamma_screen	.is_this_a_spectrum_recount		g	11 flags
gamma_screen	.results_report_citation		g	12 twenty_characters
gamma_screen	.any_special_instructions		g	13 flags
---	---	---	---	---
gamma_full_isotopic	.sample_tracking_id	f/p	g	1 identification_codes
sample_information	.customer_sample_name	d	g	2 sixty_characters
gamma_full_isotopic	.detector_count_length		g	15 real_numbers
gamma_full_isotopic	.detector_count_length_units		g	16 units
gamma_full_isotopic	.results_needed_by_date		g	17 integer_date_count
gamma_full_isotopic	.results_needed_by_time		g	18 character_time_hh_mm
gamma_full_isotopic	.date_all_work_completed		g	19 integer_date_count
gamma_full_isotopic	.date_sample_counted		g	20 integer_date_count
gamma_full_isotopic	.rml_spectral_id		g	21 rml_spectral_id_codes
gamma_full_isotopic	.is_this_a_spectrum_recount		g	22 flags
gamma_full_isotopic	.results_report_citation		g	23 twenty_characters
gamma_full_isotopic	.any_special_instructions		g	24 flags
---	---	---	---	---
gamma_other	.sample_tracking_id	f/p	g	1 identification_codes
sample_information	.customer_sample_name	d	g	2 sixty_characters
gamma_other	.detector_count_length		g	26 real_numbers
gamma_other	.detector_count_length_units		g	27 units
gamma_other	.results_needed_by_date		g	28 integer_date_count
gamma_other	.results_needed_by_time		g	29 character_time_hh_mm
gamma_other	.date_all_work_completed		g	30 integer_date_count
gamma_other	.date_sample_counted		g	31 integer_date_count
gamma_other	.rml_spectral_id		g	32 rml_spectral_id_codes
gamma_other	.is_this_a_spectrum_recount		g	33 flags
gamma_other	.results_report_citation		g	34 twenty_characters
gamma_other	.any_special_instructions		g	35 flags
---	---	---	---	---
gross_alpha_beta_air_filters	.sample_tracking_id	f/p	c	1 identification_codes
sample_information	.customer_sample_name	d	c	2 sixty_characters
gross_alpha_beta_air_filters	.detector_count_length		c	4 real_numbers
gross_alpha_beta_air_filters	.detector_count_length_units		c	5 units
gross_alpha_beta_air_filters	.results_needed_by_date		c	6 integer_date_count
gross_alpha_beta_air_filters	.results_needed_by_time		c	7 character_time_hh_mm
gross_alpha_beta_air_filters	.date_all_work_completed		c	8 integer_date_count
gross_alpha_beta_air_filters	.results_report_citation		c	9 twenty_characters
gross_alpha_beta_air_filters	.any_special_instructions		c	10 flags
---	---	---	---	---
gross_alpha_beta_other	.sample_tracking_id	f/p	c	1 identification_codes
sample_information	.customer_sample_name	d	c	2 sixty_characters
gross_alpha_beta_other	.detector_count_length		c	12 real_numbers
gross_alpha_beta_other	.detector_count_length_units		c	13 units
gross_alpha_beta_other	.results_needed_by_date		c	14 integer_date_count
gross_alpha_beta_other	.results_needed_by_time		c	15 character_time_hh_mm
gross_alpha_beta_other	.date_all_work_completed		c	16 integer_date_count
gross_alpha_beta_other	.results_report_citation		c	17 twenty_characters
gross_alpha_beta_other	.any_special_instructions		c	18 flags
---	---	---	---	---
special_instructions	.sample_tracking_id	f/p1	i	1 identification_codes
sample_information	.customer_sample_name	d	i	2 sixty_characters
special_instructions	.origin_of_special_instruction	p2	i	3 sixteen_characters
special_instructions	.special_instruction		i	4 fifty_characters
special_instructions	.special_instruction_line	p3	i	nd integer_numbers
---	---	---	---	---

tracking	.sample_tracking_id	f/p1	t	1	identification_codes
sample_information	.customer_sample_name	d	t	2	sixty_characters
tracking	.possessor_name		t	3	names
tracking	.possessor_phone_number		t	4	phone_numbers
tracking	.possessor_address		t	5	sixty_characters
tracking	.possessor_possession_reason		t	6	fifty_characters
tracking	.date_accepted_by_possessor		t	7	integer_date_count
tracking	.date_relinquished_by_possessor		t	8	integer_date_count
tracking	.possessor_sequence_number	p2	t	nd	integer_numbers

B.2 Screen-Sorted List

Worksheet for Sorting Entities and Attributes and Assigning Keys and Relationships.

Base Worksheet - Sorted by Screen

Entity	Attribute	Key	Screens	#	Domain
sample_information	.sample_tracking_id	p	m	1	identification_codes
sample_information	.date_sample_entered		m	2	integer_date_count
sample_information	.date_all_work_completed		m	3	integer_date_count
sample_information	.customer_sample_id		m	4	identification_codes
sample_information	.results_needed_by_date		m	5	integer_date_count
sample_information	.customer_project_name		m	6	forty_characters
sample_information	.work_charge_number		m	7	charge_numbers
sample_information	.customer_sample_name		m	8	sixty_characters
sample_information	.sample_type		m	9	ten_characters
sample_information	.sample_size		m	10	real_numbers
sample_information	.sample_size_units		m	11	units
sample_information	.sample_collection_date		m	12	integer_date_count
sample_information	.sample_collection_time		m	13	character_time_hh_mm
sample_information	.any_special_instructions		m	14	flags
sample_information	.sample_hp_surveyed		m	15	flags
sample_information	.sample_hp_survey_activity		m	16	real_numbers
sample_information	.sample_submitter_name		m	17	names
sample_information	.sample_submitter_phone_number		m	18	phone_numbers
sample_information	.sample_submitter_address		m	19	sixty_characters
sample_information	.technical_contact_name		m	20	names
sample_information	.technical_contact_phone_number		m	21	phone_numbers
sample_information	.technical_contact_address		m	22	sixty_characters
sample_information	.send_results_to_name		m	23	names
sample_information	.send_results_to_phone_number		m	24	phone_numbers
sample_information	.send_results_to_address		m	25	sixty_characters
sample_information	.sample_pickup_by_name		m	26	names
sample_information	.sample_pickup_by_phone_number		m	27	phone_numbers
sample_information	.sample_pickup_by_address		m	28	sixty_characters
tracking	.possessor_name	d	m	29	names
tracking	.possessor_phone_number	d	m	30	phone_numbers
tracking	.possessor_address	d	m	31	sixty_characters
sample_information	.chain_of_custody_number		m	32	ten_characters
sample_information	.need_gross_alpha_beta		m	33	flags
sample_information	.need_alpha		m	34	flags
sample_information	.need_beta		m	35	flags
sample_information	.need_gamma		m	36	flags
sample_information	.next_possessor_sequence_number		m	nd	integer_numbers
sample_information	.next_special_instruction_line		m	nd	integer_numbers
alpha_uranium	.sample_tracking_id	f/p	a	1	identification_codes
sample_information	.customer_sample_name	d	a	2	sixty_characters
sample_information	.need_alpha_uranium		a	3	flags
alpha_uranium	.results_needed_by_date		a	4	integer_date_count
alpha_uranium	.date_all_work_completed		a	5	integer_date_count
alpha_uranium	.results_report_citation		a	6	twenty_characters
alpha_uranium	.any_special_instructions		a	7	flags
sample_information	.need_alpha_thorium		a	8	flags
alpha_thorium	.results_needed_by_date		a	9	integer_date_count
alpha_thorium	.date_all_work_completed		a	10	integer_date_count
alpha_thorium	.results_report_citation		a	11	twenty_characters
alpha_thorium	.any_special_instructions		a	12	flags
sample_information	.need_alpha_plutonium		a	13	flags
alpha_plutonium	.results_needed_by_date		a	14	integer_date_count
alpha_plutonium	.date_all_work_completed		a	15	integer_date_count
alpha_plutonium	.results_report_citation		a	16	twenty_characters
alpha_plutonium	.any_special_instructions		a	17	flags
sample_information	.need_alpha_am241_sans_pu238		a	18	flags
alpha_am241_sans_pu238	.results_needed_by_date		a	19	integer_date_count
alpha_am241_sans_pu238	.date_all_work_completed		a	20	integer_date_count
alpha_am241_sans_pu238	.results_report_citation		a	21	twenty_characters
alpha_am241_sans_pu238	.any_special_instructions		a	22	flags
sample_information	.need_alpha_am241_with_pu238		a	23	flags
alpha_am241_with_pu238	.results_needed_by_date		a	24	integer_date_count
alpha_am241_with_pu238	.date_all_work_completed		a	25	integer_date_count
alpha_am241_with_pu238	.results_report_citation		a	26	twenty_characters
alpha_am241_with_pu238	.any_special_instructions		a	27	flags
sample_information	.need_alpha_total_spectrometric		a	28	flags
alpha_total_spectrometric	.results_needed_by_date		a	29	integer_date_count
alpha_total_spectrometric	.date_all_work_completed		a	30	integer_date_count
alpha_total_spectrometric	.results_report_citation		a	31	twenty_characters

alpha_total_spectrometric	.any_special_instructions		a	32	flags
sample_information	.need_alpha_other		a	33	flags
alpha_other	.results_needed_by_date		a	34	integer_date_count
alpha_other	.date_all_work_completed		a	35	integer_date_count
alpha_other	.results_report_citation		a	36	twenty_characters
alpha_other	.any_special_instructions		a	37	flags
beta_strontium_90	.sample_tracking_id	f/p	b	1	identification_codes
sample_information	.customer_sample_name	d	b	2	sixty_characters
sample_information	.need_beta_strontium_90		b	3	flags
beta_strontium_90	.detector_count_length		b	4	real_numbers
beta_strontium_90	.detector_count_length_units		b	5	units
beta_strontium_90	.results_needed_by_date		b	6	integer_date_count
beta_strontium_90	.results_needed_by_time		b	7	character_time_hh_mm
beta_strontium_90	.date_all_work_completed		b	8	integer_date_count
beta_strontium_90	.results_report_citation		b	9	twenty_characters
beta_strontium_90	.any_special_instructions		b	10	flags
sample_information	.need_beta_strontium_89_and_90		b	11	flags
beta_strontium_89_and_90	.detector_count_length		b	12	real_numbers
beta_strontium_89_and_90	.detector_count_length_units		b	13	units
beta_strontium_89_and_90	.results_needed_by_date		b	14	integer_date_count
beta_strontium_89_and_90	.results_needed_by_time		b	15	character_time_hh_mm
beta_strontium_89_and_90	.date_all_work_completed		b	16	integer_date_count
beta_strontium_89_and_90	.results_report_citation		b	17	twenty_characters
beta_strontium_89_and_90	.any_special_instructions		b	18	flags
sample_information	.need_beta_total_strontium		b	19	flags
beta_total_strontium	.detector_count_length		b	20	real_numbers
beta_total_strontium	.detector_count_length_units		b	21	units
beta_total_strontium	.results_needed_by_date		b	22	integer_date_count
beta_total_strontium	.results_needed_by_time		b	23	character_time_hh_mm
beta_total_strontium	.date_all_work_completed		b	24	integer_date_count
beta_total_strontium	.results_report_citation		b	25	twenty_characters
beta_total_strontium	.any_special_instructions		b	26	flags
sample_information	.need_beta_tritium		b	27	flags
beta_tritium	.detector_count_length		b	28	real_numbers
beta_tritium	.detector_count_length_units		b	29	units
beta_tritium	.results_needed_by_date		b	30	integer_date_count
beta_tritium	.results_needed_by_time		b	31	character_time_hh_mm
beta_tritium	.date_all_work_completed		b	32	integer_date_count
beta_tritium	.results_report_citation		b	33	twenty_characters
beta_tritium	.any_special_instructions		b	34	flags
sample_information	.need_beta_other		b	35	flags
beta_other	.detector_count_length		b	36	real_numbers
beta_other	.detector_count_length_units		b	37	units
beta_other	.need_beta_nickel_63		b	38	flags
beta_other	.need_beta_iron_55		b	39	flags
beta_other	.need_beta_sulfur_35		b	40	flags
beta_other	.need_beta_plutonium_241		b	41	flags
beta_other	.results_needed_by_date		b	42	integer_date_count
beta_other	.results_needed_by_time		b	43	character_time_hh_mm
beta_other	.date_all_work_completed		b	44	integer_date_count
beta_other	.results_report_citation		b	45	twenty_characters
beta_other	.any_special_instructions		b	46	flags
gamma_screen	.sample_tracking_id	f/p	g	1	identification_codes
sample_information	.customer_sample_name	d	g	2	sixty_characters
sample_information	.need_gamma_screen		g	3	flags
gamma_screen	.detector_count_length		g	4	real_numbers
gamma_screen	.detector_count_length_units		g	5	units
gamma_screen	.results_needed_by_date		g	6	integer_date_count
gamma_screen	.results_needed_by_time		g	7	character_time_hh_mm
gamma_screen	.date_all_work_completed		g	8	integer_date_count
gamma_screen	.date_sample_counted		g	9	integer_date_count
gamma_screen	.rml_spectral_id		g	10	rml_spectral_id_codes
gamma_screen	.is_this_a_spectrum_recount		g	11	flags
gamma_screen	.results_report_citation		g	12	twenty_characters
gamma_screen	.any_special_instructions		g	13	flags
sample_information	.need_gamma_full_isotopic		g	14	flags
gamma_full_isotopic	.detector_count_length		g	15	real_numbers
gamma_full_isotopic	.detector_count_length_units		g	16	units
gamma_full_isotopic	.results_needed_by_date		g	17	integer_date_count
gamma_full_isotopic	.results_needed_by_time		g	18	character_time_hh_mm
gamma_full_isotopic	.date_all_work_completed		g	19	integer_date_count
gamma_full_isotopic	.date_sample_counted		g	20	integer_date_count
gamma_full_isotopic	.rml_spectral_id		g	21	rml_spectral_id_codes
gamma_full_isotopic	.is_this_a_spectrum_recount		g	22	flags
gamma_full_isotopic	.results_report_citation		g	23	twenty_characters
gamma_full_isotopic	.any_special_instructions		g	24	flags
sample_information	.need_gamma_other		g	25	flags

gamma_other	.detector_count_length		g	26	real_numbers
gamma_other	.detector_count_length_units		g	27	units
gamma_other	.results_needed_by_date		g	28	integer_date_count
gamma_other	.results_needed_by_time		g	29	character_time_hh_mm
gamma_other	.date_all_work_completed		g	30	integer_date_count
gamma_other	.date_sample_counted		g	31	integer_date_count
gamma_other	.rml_spectral_id		g	32	rml_spectral_id_codes
gamma_other	.is_this_a_spectrum_recount		g	33	flags
gamma_other	.results_report_citation		g	34	twenty_characters
gamma_other	.any_special_instructions		g	35	flags
gross_alpha_beta_air_filters	.sample_tracking_id	f/p	c	1	identification_codes
sample_information	.customer_sample_name	d	c	2	sixty_characters
sample_information	.need_gross_alpha_beta_filters		c	3	flags
gross_alpha_beta_air_filters	.detector_count_length		c	4	real_numbers
gross_alpha_beta_air_filters	.detector_count_length_units		c	5	units
gross_alpha_beta_air_filters	.results_needed_by_date		c	6	integer_date_count
gross_alpha_beta_air_filters	.results_needed_by_time		c	7	character_time_hh_mm
gross_alpha_beta_air_filters	.date_all_work_completed		c	8	integer_date_count
gross_alpha_beta_air_filters	.results_report_citation		c	9	twenty_characters
gross_alpha_beta_air_filters	.any_special_instructions		c	10	flags
sample_information	.need_gross_alpha_beta_other		c	11	flags
gross_alpha_beta_other	.detector_count_length		c	12	real_numbers
gross_alpha_beta_other	.detector_count_length_units		c	13	units
gross_alpha_beta_other	.results_needed_by_date		c	14	integer_date_count
gross_alpha_beta_other	.results_needed_by_time		c	15	character_time_hh_mm
gross_alpha_beta_other	.date_all_work_completed		c	16	integer_date_count
gross_alpha_beta_other	.results_report_citation		c	17	twenty_characters
gross_alpha_beta_other	.any_special_instructions		c	18	flags
special_instructions	.sample_tracking_id	f/p1	i	1	identification_codes
sample_information	.customer_sample_name	d	i	2	sixty_characters
special_instructions	.origin_of_special_instruction	p2	i	3	sixteen_characters
special_instructions	.special_instruction		i	4	fifty_characters
special_instructions	.special_instruction_line	p3	i	nd	integer_numbers
tracking	.sample_tracking_id	f/p1	t	1	identification_codes
sample_information	.customer_sample_name	d	t	2	sixty_characters
tracking	.possessor_name		t	3	names
tracking	.possessor_phone_number		t	4	phone_numbers
tracking	.possessor_address		t	5	sixty_characters
tracking	.possessor_possession_reason		t	6	fifty_characters
tracking	.date_accepted_by_possessor		t	7	integer_date_count
tracking	.date_relinquished_by_possessor		t	8	integer_date_count
tracking	.possessor_sequence_number	p2	t	nd	integer_numbers

C. EXECUTION OF CREATE_SCHEMA - A SAMPLE TERMINAL SESSION

What the user enters appears in bold text in the following.

```
$ @create_schema
Enter the name of the schema to be created:  sample_tracking
For domain CHARACTER_TIME_HH_MM:
   Enter the data type for this domain (char/date/int/real):  c
      Enter the length of the character domain:  5
   Enter the default value (return says NULL):
   Enter any desired comment:  Domain for time in the form hh:mm.
For domain CHARGE_NUMBERS:
   Enter the data type for this domain (char/date/int/real):  c
      Enter the length of the character domain:  9
   Enter the default value (return says NULL):  PF7400000
   Enter any desired comment:  Domain for nine-alphanumeric-long charge
numbers.
For domain FIFTY_CHARACTERS:
   Enter the data type for this domain (char/date/int/real):  c
      Enter the length of the character domain:  50
   Enter the default value (return says NULL):
   Enter any desired comment:  Domain for fifty-alphanumeric-long fields.
For domain FLAGS:
   Enter the data type for this domain (char/date/int/real):  c
      Enter the length of the character domain:  1
   Enter the default value (return says NULL):  N
   Enter any desired comment:  Domain for one-character-long yes/no
flags.
For domain FORTY_CHARACTERS:
   Enter the data type for this domain (char/date/int/real):  c
      Enter the length of the character domain:  40
   Enter the default value (return says NULL):
```

Enter any desired comment: **Domain for forty-alphanumeric-long fields.**

For domain IDENTIFICATION_CODES:

Enter the data type for this domain (char/date/int/real): **c**

Enter the length of the character domain: **12**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for twelve-character-long identification codes.**

For domain INTEGER_DATE_COUNT:

Enter the data type for this domain (char/date/int/real): **i**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for integer*4 date count starting at January 1, 1990.**

For domain INTEGER_NUMBERS:

Enter the data type for this domain (char/date/int/real): **i**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for integer*4 numbers.**

For domain NAMES:

Enter the data type for this domain (char/date/int/real): **c**

Enter the length of the character domain: **25**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for twenty-five-alphanumeric-long names.**

For domain PHONE_NUMBERS:

Enter the data type for this domain (char/date/int/real): **c**

Enter the length of the character domain: **12**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for twelve-alphanumeric-long phone numbers.**

For domain REAL_NUMBERS:

Enter the data type for this domain (char/date/int/real): **r**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for real*4 numbers.**

For domain RML_SPECTRAL_ID_CODES:

Enter the data type for this domain (char/date/int/real): **c**

Enter the length of the character domain: **14**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for fourteen-character-long RML spectral identification codes (<save area>$<spectral id>).**

For domain SIXTEEN_CHARACTERS:

Enter the data type for this domain (char/date/int/real): **c**

Enter the length of the character domain: **16**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for sixteen-alphanumeric-long fields.**

For domain SIXTY_CHARACTERS:

Enter the data type for this domain (char/date/int/real): **c**

Enter the length of the character domain: **60**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for sixty-alphanumeric-long fields.**

For domain TEN_CHARACTERS:

Enter the data type for this domain (char/date/int/real): **c**

Enter the length of the character domain: **10**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for ten-alphanumeric-long fields.**

For domain TWENTY_CHARACTERS:

Enter the data type for this domain (char/date/int/real): **c**

Enter the length of the character domain: **20**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for twenty-alphanumeric-long fields.**

For domain UNITS:

Enter the data type for this domain (char/date/int/real): **c**

Enter the length of the character domain: **5**

Enter the default value (return says NULL):

Enter any desired comment: **Domain for five-alphanumeric-long units.**

For table SAMPLE_INFORMATION:

Enter any desired comment: **Table for sample information.**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **4000**

For table ALPHA_URANIUM:

Enter any desired comment: **Table for requested alpha analyses for Uranium radionuclides.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table ALPHA_THORIUM:

Enter any desired comment: **Table for requested alpha analyses for Thorium radionuclides.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table ALPHA_PLUTONIUM:

Enter any desired comment: **Table for requested alpha analyses for Plutonium radionuclides.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table ALPHA_AM241_SANS_PU238:

Enter any desired comment: **Table for requested alpha analyses for**

Americium-241 separate from Plutonium-238.

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table ALPHA_AM241_WITH_PU238:

Enter any desired comment: **Table for requested alpha analyses for Americium-241 combined with Plutonium-238.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table ALPHA_TOTAL_SPECTROMETRIC:

Enter any desired comment: **Table for requested total spectrometric alpha analyses.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table ALPHA_OTHER:

Enter any desired comment: **Table for requested alpha analyses for other, unlisted radionuclides.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table BETA_STRONTIUM_90:

Enter any desired comment: **Table for requested beta analyses for**

Strontium-90.

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table BETA_STRONTIUM_89_AND_90:

Enter any desired comment: **Table for requested beta analyses for Strontium-89 and -90.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table BETA_TOTAL_STRONTIUM:

Enter any desired comment: **Table for requested beta analyses for total Strontium.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table BETA_TRITIUM:

Enter any desired comment: **Table for requested beta analyses for Tritium.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table BETA_OTHER:

Enter any desired comment: **Table for requested beta analyses for**

other, unlisted radionuclides.

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table GAMMA_SCREEN:

Enter any desired comment: **Table for requested screening and shipping gamma analyses.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **4000**

For table GAMMA_FULL_ISOTOPIC:

Enter any desired comment: **Table for requested full isotopic gamma analyses.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **4000**

For table GAMMA_OTHER:

Enter any desired comment: **Table for requested gamma analyses for other, unlisted radionuclides.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **4000**

For table GROSS_ALPHA_BETA_AIR_FILTERS:

Enter any desired comment: **Table for requested gross alpha-beta**

analyses of air filters.

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table GROSS_ALPHA_BETA_OTHER:

Enter any desired comment: **Table for requested gross alpha-beta analyses of other, unlisted samples.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **1**

Enter the maximum number of rows for this table: **1000**

For table SPECIAL_INSTRUCTIONS:

Enter any desired comment: **Table for special instructions.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **15**

Enter the maximum number of rows for this table: **20000**

For table TRACKING:

Enter any desired comment: **Table for tracking samples.**

For foreign key SAMPLE_TRACKING_ID:

Enter the reference table: **SAMPLE_INFORMATION**

Enter the reference column: **SAMPLE_TRACKING_ID**

Enter the number of rows per page to be clustered together: **5**

Enter the maximum number of rows for this table: **20000**

Execution has successfully completed. The create_schema.sql and cursor and insert files will be closed and retained. The entity and screen files will also be closed and retained.

Please wait while certain sections of the cursor and insert files are

```
rearranged; this may take a few minutes.
   Cursor file rearranged.
   Insert file rearranged.
$
```

D. DATABASE BACKUP - AN EXCERPT FROM THE RML OPERATOR'S BACKUP ROUTINE

Note the closed sets of ellipses in the two "set default" lines below; these would be replaced with the appropriate directory ($USER:[DATABASES.FILES], for example).

```
. . .

$ ! Prior to performing the DCL BACKUP, backup the contents of the
$ ! sample tracking database.
$ !
$ write sys$output " "
$ write sys$output " Verifying the integrity of the database . . ."
$ write sys$output " "
$ set default ...
$ rmu/verify sample_tracking   !Verify the integrity of the database.
$ write sys$output " "
$ write sys$output " Preparing a version of the database which can be backed up . . ."
$ write sys$output " "
$ rmu/backup sample_tracking sample_tracking   !Backup the database.
$ write sys$output " "
$ write sys$output " Database successfully verified and prepared."
$ write sys$output " "

. . .

$ !
$ ! Purge most prior versions of the database backup file, and mark
$ ! those that remain for future "no backup."
$ !
$ write sys$output " "
$ write sys$output " Purging old backup versions of the database . . ."
$ write sys$output " "
$ set default ...
$ purge/keep=2 sample_tracking.rbf
$ set file sample_tracking.rbf;* /nobackup

. . . .
```

E. ARRAY AND SCALAR DECLARATIONS RELATED TO THE SCREEN MANAGEMENT ROUTINES

```
C
C Declaration statements.
C
C     Component of SAMPLE_TRACKING Version 1 completed June 6, 1991 by
C     D. A. Femec.
C
C     Note that variable names are limited to six-characters in length;
C     this is a limitation imposed by SQL for variables used in SQL calls.
C
C     Field rendition descriptions.
C
      integer*4 ferror/smg$m_blink/,    !For errors in field entries.
     . ffixed/smg$m_reverse/,  !For fixed fields.
     . finput/smg$m_underline/,    !For input fields.
     . foptnl/smg$m_normal/,   !For optional fields.
     . freqd/smg$m_bold/    !For required fields.
      common /fldscr/ ferror,ffixed,finput,foptnl,freqd
C
C     Main screen's field descriptors and field sizes.
C
      character*80 fieldM(36),fidatM(36),findat,blanks
      character*30 atrbtM(2,38)
      character*14 varblM(38)
      character*1 spaces(80)/80*' '/,fitypM(36)
      integer*4 fdlenM(36),fdrowM(36),fdcolM(36),ficolM(36),mxnocM(36),
     . fdrndM(3,36),firndM(3,36),  !Field renditions (add/update/search).
     . numfdM/36/    !The number of fields.
      integer*2 ifnulM(36),   !Null indicator (-1 if null, 0 otherwise).
     . ifrqdM(36)/36*0/,    !Points to if-required dependent field.
     . iditoM(36)/36*0/    !Points to allowed field for dittoing.
      logical ifmodM(36)    !Has a field been modified in an Update?
      common /dbdatM/ fieldM,fidatM,fdlenM,fdrowM,fdcolM,ficolM,mxnocM,
     . fdrndM,firndM,numfdM,fitypM,ifnulM,atrbtM,varblM,ifrqdM,ifmodM,
     . iditoM
      equivalence (blanks(1:1),spaces(1))
C
C      Length-specified character fields for equivalences (f for field).
C
      character fMSTI*12,fMDSE*6,fMDAWC*6,fMCSI*12,fMRNBD*6,fMCPN*40,
     . fMWCN*9,fMCSN*60,fMST*10,fMSS*5,fMSSU*5,fMSCD*6,fMSCT*5,fMASI*1,
     . fMSHS*1,fMSHSA*6,fMSSN*25,fMSSPN*12,fMSSA*60,fMTCN*25,fMTCPN*12,
     . fMTCA*60,fMSRTN*25,fMSRTP*12,fMSRTA*60,fMSPBN*25,fMSPBP*12,
     . fMSPBA*60,fMPN*25,fMPPN*12,fMPA*60,fMCOCN*10,fMNGAB*1,fMNA*1,
     . fMNB*1,fMNG*1
C
C       Real and integer field storage (r for real, i for integer).
C
      real*4 rMSS,rMSHSA
      integer*4 iMDSE,iMDAWC,iMRNBD,iMSCD
C     . ,nxinst,nxcust    !Declared in the main routine.
C
C       Nonarrayed field lengths (1 for lengths).
C
      integer*4 lMSTI,lMDSE,lMDAWC,lMCSI,lMRNBD,lMCPN,lMWCN,lMCSN,lMST,
     . lMSS,lMSSU,lMSCD,lMSCT,lMASI,lMSHS,lMSHSA,lMSSN,lMSSPN,lMSSA,
```

```
     . lMTCN,lMTCPN,lMTCA,lMSRTN,lMSRTP,lMSRTA,lMSPBN,lMSPBP,lMSPBA,
     . lMPN,lMPPN,lMPA,lMCOCN,lMNGAB,lMNA,lMNB,lMNG
C
C       Nonarrayed null indicators for data transfers (n for null).
C
      integer*2 nMSTI,nMDSE,nMDAWC,nMCSI,nMRNBD,nMCPN,nMWCN,nMCSN,nMST,
     . nMSS,nMSSU,nMSCD,nMSCT,nMASI,nMSHS,nMSHSA,nMSSN,nMSSPN,nMSSA,
     . nMTCN,nMTCPN,nMTCA,nMSRTN,nMSRTP,nMSRTA,nMSPBN,nMSPBP,nMSPBA,
     . nMPN,nMPPN,nMPA,nMCOCN,nMNGAB,nMNA,nMNB,nMNG
C
C       Nonarrayed if-modified indicators (m for modified).
C
      integer*2 mMSTI,mMDSE,mMDAWC,mMCSI,mMRNBD,mMCPN,mMWCN,mMCSN,mMST,
     . mMSS,mMSSU,mMSCD,mMSCT,mMASI,mMSHS,mMSHSA,mMSSN,mMSSPN,mMSSA,
     . mMTCN,mMTCPN,mMTCA,mMSRTN,mMSRTP,mMSRTA,mMSPBN,mMSPBP,mMSPBA,
     . mMPN,mMPPN,mMPA,mMCOCN,mMNGAB,mMNA,mMNB,mMNG
C
C       Equivalence statements for data transfers.
C
      equivalence (fidatM(1)(1:12),fMSTI),(ifnulM(1),nMSTI),
     . (fidatM(2)(1:6),fMDSE),(ifnulM(2),nMDSE),
     . (fidatM(3)(1:6),fMDAWC),(ifnulM(3),nMDAWC),
     . (fidatM(4)(1:12),fMCSI),(ifnulM(4),nMCSI),
     . (fidatM(5)(1:6),fMRNBD),(ifnulM(5),nMRNBD),
     . (fidatM(6)(1:40),fMCPN),(ifnulM(6),nMCPN),
     . (fidatM(7)(1:9),fMWCN),(ifnulM(7),nMWCN),
     . (fidatM(8)(1:60),fMCSN),(ifnulM(8),nMCSN),
     . (fidatM(9)(1:10),fMST),(ifnulM(9),nMST),
     . (fidatM(10)(1:5),fMSS),(ifnulM(10),nMSS),
     . (fidatM(11)(1:5),fMSSU),(ifnulM(11),nMSSU),
     . (fidatM(12)(1:6),fMSCD),(ifnulM(12),nMSCD),
     . (fidatM(13)(1:5),fMSCT),(ifnulM(13),nMSCT),
     . (fidatM(14)(1:1),fMASI),(ifnulM(14),nMASI),
     . (fidatM(15)(1:1),fMSHS),(ifnulM(15),nMSHS),
     . (fidatM(16)(1:6),fMSHSA),(ifnulM(16),nMSHSA),
     . (fidatM(17)(1:25),fMSSN),(ifnulM(17),nMSSN),
     . (fidatM(18)(1:12),fMSSPN),(ifnulM(18),nMSSPN),
     . (fidatM(19)(1:60),fMSSA),(ifnulM(19),nMSSA),
     . (fidatM(20)(1:25),fMTCN),(ifnulM(20),nMTCN),
     . (fidatM(21)(1:12),fMTCPN),(ifnulM(21),nMTCPN),
     . (fidatM(22)(1:60),fMTCA),(ifnulM(22),nMTCA),
     . (fidatM(23)(1:25),fMSRTN),(ifnulM(23),nMSRTN),
     . (fidatM(24)(1:12),fMSRTP),(ifnulM(24),nMSRTP),
     . (fidatM(25)(1:60),fMSRTA),(ifnulM(25),nMSRTA),
     . (fidatM(26)(1:25),fMSPBN),(ifnulM(26),nMSPBN),
     . (fidatM(27)(1:12),fMSPBP),(ifnulM(27),nMSPBP),
     . (fidatM(28)(1:60),fMSPBA),(ifnulM(28),nMSPBA),
     . (fidatM(29)(1:25),fMPN),(ifnulM(29),nMPN),
     . (fidatM(30)(1:12),fMPPN),(ifnulM(30),nMPPN),
     . (fidatM(31)(1:60),fMPA),(ifnulM(31),nMPA),
     . (fidatM(32)(1:10),fMCOCN),(ifnulM(32),nMCOCN),
     . (fidatM(33)(1:1),fMNGAB),(ifnulM(33),nMNGAB),
     . (fidatM(34)(1:1),fMNA),(ifnulM(34),nMNA),
     . (fidatM(35)(1:1),fMNB),(ifnulM(35),nMNB),
     . (fidatM(36)(1:1),fMNG),(ifnulM(36),nMNG),
     . (mxnocM(1),lMSTI),(ifmodM(1),mMSTI),
     . (mxnocM(2),lMDSE),(ifmodM(2),mMDSE),
     . (mxnocM(3),lMDAWC),(ifmodM(3),mMDAWC),
     . (mxnocM(4),lMCSI),(ifmodM(4),mMCSI),
     . (mxnocM(5),lMRNBD),(ifmodM(5),mMRNBD),
```

```
     . (mxnocM(6),lMCPN),(ifmodM(6),mMCPN),
     . (mxnocM(7),lMWCN),(ifmodM(7),mMWCN),
     . (mxnocM(8),lMCSN),(ifmodM(8),mMCSN),
     . (mxnocM(9),lMST),(ifmodM(9),mMST),
     . (mxnocM(10),lMSS),(ifmodM(10),mMSS),
     . (mxnocM(11),lMSSU),(ifmodM(11),mMSSU),
     . (mxnocM(12),lMSCD),(ifmodM(12),mMSCD),
     . (mxnocM(13),lMSCT),(ifmodM(13),mMSCT),
     . (mxnocM(14),lMASI),(ifmodM(14),mMASI),
     . (mxnocM(15),lMSHS),(ifmodM(15),mMSHS),
     . (mxnocM(16),lMSHSA),(ifmodM(16),mMSHSA),
     . (mxnocM(17),lMSSN),(ifmodM(17),mMSSN),
     . (mxnocM(18),lMSSPN),(ifmodM(18),mMSSPN),
     . (mxnocM(19),lMSSA),(ifmodM(19),mMSSA),
     . (mxnocM(20),lMTCN),(ifmodM(20),mMTCN),
     . (mxnocM(21),lMTCPN),(ifmodM(21),mMTCPN),
     . (mxnocM(22),lMTCA),(ifmodM(22),mMTCA),
     . (mxnocM(23),lMSRTN),(ifmodM(23),mMSRTN),
     . (mxnocM(24),lMSRTP),(ifmodM(24),mMSRTP),
     . (mxnocM(25),lMSRTA),(ifmodM(25),mMSRTA),
     . (mxnocM(26),lMSPBN),(ifmodM(26),mMSPBN),
     . (mxnocM(27),lMSPBP),(ifmodM(27),mMSPBP),
     . (mxnocM(28),lMSPBA),(ifmodM(28),mMSPBA),
     . (mxnocM(29),lMPN),(ifmodM(29),mMPN),
     . (mxnocM(30),lMPPN),(ifmodM(30),mMPPN),
     . (mxnocM(31),lMPA),(ifmodM(31),mMPA),
     . (mxnocM(32),lMCOCN),(ifmodM(32),mMCOCN),
     . (mxnocM(33),lMNGAB),(ifmodM(33),mMNGAB),
     . (mxnocM(34),lMNA),(ifmodM(34),mMNA),
     . (mxnocM(35),lMNB),(ifmodM(35),mMNB),
     . (mxnocM(36),lMNG),(ifmodM(36),mMNG)
C
C    Database table and attribute names for screen fields.
C
      data atrbtM/
     . 'SAMPLE_INFORMATION','SAMPLE_TRACKING_ID',
     . 'SAMPLE_INFORMATION','DATE_SAMPLE_ENTERED',
     . 'SAMPLE_INFORMATION','DATE_ALL_WORK_COMPLETED',
     . 'SAMPLE_INFORMATION','CUSTOMER_SAMPLE_ID',
     . 'SAMPLE_INFORMATION','RESULTS_NEEDED_BY_DATE',
     . 'SAMPLE_INFORMATION','CUSTOMER_PROJECT_NAME',
     . 'SAMPLE_INFORMATION','WORK_CHARGE_NUMBER',
     . 'SAMPLE_INFORMATION','CUSTOMER_SAMPLE_NAME',
     . 'SAMPLE_INFORMATION','SAMPLE_TYPE',
     . 'SAMPLE_INFORMATION','SAMPLE_SIZE',
     . 'SAMPLE_INFORMATION','SAMPLE_SIZE_UNITS',
     . 'SAMPLE_INFORMATION','SAMPLE_COLLECTION_DATE',
     . 'SAMPLE_INFORMATION','SAMPLE_COLLECTION_TIME',
     . 'SAMPLE_INFORMATION','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','SAMPLE_HP_SURVEYED',
     . 'SAMPLE_INFORMATION','SAMPLE_HP_SURVEY_ACTIVITY',
     . 'SAMPLE_INFORMATION','SAMPLE_SUBMITTER_NAME',
     . 'SAMPLE_INFORMATION','SAMPLE_SUBMITTER_PHONE_NUMBER',
     . 'SAMPLE_INFORMATION','SAMPLE_SUBMITTER_ADDRESS',
     . 'SAMPLE_INFORMATION','TECHNICAL_CONTACT_NAME',
     . 'SAMPLE_INFORMATION','TECHNICAL_CONTACT_PHONE_NUMBER',
     . 'SAMPLE_INFORMATION','TECHNICAL_CONTACT_ADDRESS',
     . 'SAMPLE_INFORMATION','SEND_RESULTS_TO_NAME',
     . 'SAMPLE_INFORMATION','SEND_RESULTS_TO_PHONE_NUMBER',
     . 'SAMPLE_INFORMATION','SEND_RESULTS_TO_ADDRESS',
```

```
     . 'SAMPLE_INFORMATION','SAMPLE_PICKUP_BY_NAME',
     . 'SAMPLE_INFORMATION','SAMPLE_PICKUP_BY_PHONE_NUMBER',
     . 'SAMPLE_INFORMATION','SAMPLE_PICKUP_BY_ADDRESS',
     . 'TRACKING','POSSESSOR_NAME',
     . 'TRACKING','POSSESSOR_PHONE_NUMBER',
     . 'TRACKING','POSSESSOR_ADDRESS',
     . 'SAMPLE_INFORMATION','CHAIN_OF_CUSTODY_NUMBER',
     . 'SAMPLE_INFORMATION','NEED_GROSS_ALPHA_BETA',
     . 'SAMPLE_INFORMATION','NEED_ALPHA',
     . 'SAMPLE_INFORMATION','NEED_BETA',
     . 'SAMPLE_INFORMATION','NEED_GAMMA',
     . 'SAMPLE_INFORMATION','NEXT_SPECIAL_INSTRUCTION_LINE',
     . 'SAMPLE_INFORMATION','NEXT_POSSESSOR_SEQUENCE_NUMBER'/
C
C    Program variable names (value and null indicator) for screen fields.
C
      data varblM/':fMSTI:nMSTI',':iMDSE:nMDSE',':iMDAWC:nMDAWC',
     . ':fMCSI:nMCSI',':iMRNBD:nMRNBD',':fMCPN:nMCPN',':fMWCN:nMWCN',
     . ':fMCSN:nMCSN',':fMST:nMST',':rMSS:nMSS',':fMSSU:nMSSU',
     . ':iMSCD:nMSCD',':fMSCT:nMSCT',':fMASI:nMASI',':fMSHS:nMSHS',
     . ':rMSHSA:nMSHSA',':fMSSN:nMSSN',':fMSSPN:nMSSPN',':fMSSA:nMSSA',
     . ':fMTCN:nMTCN',':fMTCPN:nMTCPN',':fMTCA:nMTCA',':fMSRTN:nMSRTN',
     . ':fMSRTP:nMSRTP',':fMSRTA:nMSRTA',':fMSPBN:nMSPBN',
     . ':fMSPBP:nMSPBP',':fMSPBA:nMSPBA',':fMPN:nMPN',
     . ':fMPPN:nMPPN',':fMPA:nMPA',':fMCOCN:nMCOCN',':fMNGAB:nMNGAB',
     . ':fMNA:nMNA',':fMNB:nMNB',':fMNG:nMNG',':nxinst:ifnnxi',
     . ':nxcust:ifnnxc'/
C
C    Alpha screen's field descriptors and field sizes.
C
      character*80 fieldA(37),fidatA(37)
      character*30 atrbtA(2,37)
      character*14 varblA(37)
      character*1 fitypA(37)
      integer*4 fdlenA(37),fdrowA(37),fdcolA(37),ficolA(37),mxnocA(37),
     . fdrndA(3,37),firndA(3,37), !Field renditions (add/update/search).
     . numfdA/37/   !The number of fields.
      integer*2 ifnulA(37),   !Null indicator (-1 if null, 0 otherwise).
     . ifrqdA(37)/37*0/,   !Points to if-required dependent field.
     . iditoA(37)/37*0/   !Points to allowed field for dittoing.
      logical ifmodA(37)   !Has a field been modified in an Update?
      common /dbdatA/ fieldA,fidatA,fdlenA,fdrowA,fdcolA,ficolA,mxnocA,
     . fdrndA,firndA,numfdA,fitypA,ifnulA,atrbtA,varblA,ifrqdA,ifmodA,
     . iditoA
C
C      Length-specified character fields for equivalences (f for field).
C
      character fASTI*12,fACSN*60,fANAU*1,fARNB1*6,fADAW1*6,fARRC1*20,
     . fAASI1*1,fANATh*1,fARNB2*6,fADAW2*6,fARRC2*20,fAASI2*1,fANAPu*1,
     . fARNB3*6,fADAW3*6,fARRC3*20,fAASI3*1,fANASP*1,fARNB4*6,fADAW4*6,
     . fARRC4*20,fAASI4*1,fANAWP*1,fARNB5*6,fADAW5*6,fARRC5*20,fAASI5*1,
     . fANATS*1,fARNB6*6,fADAW6*6,fARRC6*20,fAASI6*1,fANAO*1,fARNB7*6,
     . fADAW7*6,fARRC7*20,fAASI7*1
C
C       Integer field storage (i for integer).
C
      integer*4 iARNB1,iADAW1,iARNB2,iADAW2,iARNB3,iADAW3,iARNB4,iADAW4,
     . iARNB5,iADAW5,iARNB6,iADAW6,iARNB7,iADAW7
C
C       Nonarrayed maximum field lengths (l for length).
```

```
C
      integer*4 lASTI,lACSN,lANAU,lARNB1,lADAW1,lARRC1,lAASI1,lANATh,
     . lARNB2,lADAW2,lARRC2,lAASI2,lANAPu,lARNB3,lADAW3,lARRC3,lAASI3,
     . lANASP,lARNB4,lADAW4,lARRC4,lAASI4,lANAWP,lARNB5,lADAW5,lARRC5,
     . lAASI5,lANATS,lARNB6,lADAW6,lARRC6,lAASI6,lANAO,lARNB7,lADAW7,
     . lARRC7,lAASI7
C
C       Nonarrayed null indicators for data transfers (n for null).
C
      integer*2 nASTI,nACSN,nANAU,nARNB1,nADAW1,nARRC1,nAASI1,nANATh,
     . nARNB2,nADAW2,nARRC2,nAASI2,nANAPu,nARNB3,nADAW3,nARRC3,nAASI3,
     . nANASP,nARNB4,nADAW4,nARRC4,nAASI4,nANAWP,nARNB5,nADAW5,nARRC5,
     . nAASI5,nANATS,nARNB6,nADAW6,nARRC6,nAASI6,nANAO,nARNB7,nADAW7,
     . nARRC7,nAASI7
C
C       Nonarrayed if-modified indicators (m for modified).
C
      integer*2 mASTI,mACSN,mANAU,mARNB1,mADAW1,mARRC1,mAASI1,mANATh,
     . mARNB2,mADAW2,mARRC2,mAASI2,mANAPu,mARNB3,mADAW3,mARRC3,mAASI3,
     . mANASP,mARNB4,mADAW4,mARRC4,mAASI4,mANAWP,mARNB5,mADAW5,mARRC5,
     . mAASI5,mANATS,mARNB6,mADAW6,mARRC6,mAASI6,mANAO,mARNB7,mADAW7,
     . mARRC7,mAASI7
C
C       Equivalence statements for data transfers.
C
      equivalence (fidatA(1)(1:12),fASTI),(ifnulA(1),nASTI),
     . (fidatA(2)(1:60),fACSN),(ifnulA(2),nACSN),
     . (fidatA(3)(1:1),fANAU),(ifnulA(3),nANAU),
     . (fidatA(4)(1:6),fARNB1),(ifnulA(4),nARNB1),
     . (fidatA(5)(1:6),fADAW1),(ifnulA(5),nADAW1),
     . (fidatA(6)(1:20),fARRC1),(ifnulA(6),nARRC1),
     . (fidatA(7)(1:1),fAASI1),(ifnulA(7),nAASI1),
     . (fidatA(8)(1:1),fANATh),(ifnulA(8),nANATh),
     . (fidatA(9)(1:6),fARNB2),(ifnulA(9),nARNB2),
     . (fidatA(10)(1:6),fADAW2),(ifnulA(10),nADAW2),
     . (fidatA(11)(1:20),fARRC2),(ifnulA(11),nARRC2),
     . (fidatA(12)(1:1),fAASI2),(ifnulA(12),nAASI2),
     . (fidatA(13)(1:1),fANAPu),(ifnulA(13),nANAPu),
     . (fidatA(14)(1:6),fARNB3),(ifnulA(14),nARNB3),
     . (fidatA(15)(1:6),fADAW3),(ifnulA(15),nADAW3),
     . (fidatA(16)(1:20),fARRC3),(ifnulA(16),nARRC3),
     . (fidatA(17)(1:1),fAASI3),(ifnulA(17),nAASI3),
     . (fidatA(18)(1:1),fANASP),(ifnulA(18),nANASP),
     . (fidatA(19)(1:6),fARNB4),(ifnulA(19),nARNB4),
     . (fidatA(20)(1:6),fADAW4),(ifnulA(20),nADAW4),
     . (fidatA(21)(1:20),fARRC4),(ifnulA(21),nARRC4),
     . (fidatA(22)(1:1),fAASI4),(ifnulA(22),nAASI4),
     . (fidatA(23)(1:1),fANAWP),(ifnulA(23),nANAWP),
     . (fidatA(24)(1:6),fARNB5),(ifnulA(24),nARNB5),
     . (fidatA(25)(1:6),fADAW5),(ifnulA(25),nADAW5),
     . (fidatA(26)(1:20),fARRC5),(ifnulA(26),nARRC5),
     . (fidatA(27)(1:1),fAASI5),(ifnulA(27),nAASI5),
     . (fidatA(28)(1:1),fANATS),(ifnulA(28),nANATS),
     . (fidatA(29)(1:6),fARNB6),(ifnulA(29),nARNB6),
     . (fidatA(30)(1:6),fADAW6),(ifnulA(30),nADAW6),
     . (fidatA(31)(1:20),fARRC6),(ifnulA(31),nARRC6),
     . (fidatA(32)(1:1),fAASI6),(ifnulA(32),nAASI6),
     . (fidatA(33)(1:1),fANAO),(ifnulA(33),nANAO),
     . (fidatA(34)(1:6),fARNB7),(ifnulA(34),nARNB7),
     . (fidatA(35)(1:6),fADAW7),(ifnulA(35),nADAW7),
```

```
     . (fidatA(36)(1:20),fARRC7),(if  lA(36),nARRC7),
     . (fidatA(37)(1:1),fAASI7),(ifm  A(37),nAASI7),
     . (mxnocA(1),lASTI),(ifmodA(1),m STI),
     . (mxnocA(2),lACSN),(ifmodA(2),mACSN),
     . (mxnocA(3),lANAU),(ifmodA(3),mANAU),
     . (mxnocA(4),lARNB1),(ifmodA(4),mARNB1),
     . (mxnocA(5),lADAW1),(ifmodA(5),mADAW1),
     . (mxnocA(6),lARRC1),(ifmodA(6),mARRC1),
     . (mxnocA(7),lAASI1),(ifmodA(7),mAASI1),
     . (mxnocA(8),lANATh),(ifmodA(8),mANATh),
     . (mxnocA(9),lARNB2),(ifmodA(9),mARNB2),
     . (mxnocA(10),lADAW2),(ifmodA(10),mADAW2),
     . (mxnocA(11),lARRC2),(ifmodA(11),mARRC2),
     . (mxnocA(12),lAASI2),(ifmodA(12),mAASI2),
     . (mxnocA(13),lANAPu),(ifmodA(13),mANAPu),
     . (mxnocA(14),lARNB3),(ifmodA(14),mARNB3),
     . (mxnocA(15),lADAW3),(ifmodA(15),mADAW3),
     . (mxnocA(16),lARRC3),(ifmodA(16),mARRC3),
     . (m nocA(17),lAASI3),(ifmodA(17),mAASI3),
     . (m nocA(18),lANASP),(ifmodA(18),mANASP),
     . (mxnocA(19),lARNB4),(ifmodA(19),mARNB4),
     . (mxnocA(20),lADAW4),(ifmodA(20),mADAW4),
     . (mxnocA(21),lARRC4),(ifmodA(21),mARRC4),
     . (mxnocA(22),lAASI4),(ifmodA(22),mAASI4),
     . (mxnocA(23),lANAWP),(ifmodA(23),mANAWP),
     . (mxnocA(24),lARNB5),(ifmodA(24),mARNB5),
     . (mxnocA(25),lADAW5),(ifmodA(25),mADAW5),
     . (mxnocA(26),lARRC5),(ifmodA(26),mARRC5),
     . (mxnocA(27),lAASI5),(ifmodA(27),mAASI5),
     . (mxnocA(28),lANATS),(ifmodA(28),mANATS),
     . (mxnocA(29),lARNB6),(ifmodA(29),mARNB6),
     . (mxnocA(30),lADAW6),(ifmodA(30),mADAW6),
     . (mxnocA(31),lARRC6),(ifmodA(31),mARRC6),
     . (mxnocA(32),lAASI6),(ifmodA(32),mAASI6),
     . (mxnocA(33),lANAO),(ifmodA(33),mANAO),
     . (mxnocA(34),lARNB7),(ifmodA(34),mARNB7),
     . (mxnocA(35),lADAW7),(ifmodA(35),mADAW7),
     . (mxnocA(36),lARRC7),(ifmodA(36),mARRC7),
     . (mxnocA(37),lAASI7),(ifmodA(37),mAASI7)
C
C    Database table and attribute names for screen fields.
C
      data atrbtA/
     . 'ALPHA_URANIUM','SAMPLE_TRACKING_ID',
     . 'SAMPLE_INFORMATION','CUSTOMER_SAMPLE_NAME',
     . 'SAMPLE_INFORMATION','NEED_ALPHA_URANIUM',
     . 'ALPHA_URANIUM','RESULTS_NEEDED_BY_DATE',
     . 'ALPHA_URANIUM','DATE_ALL_WORK_COMPLETED',
     . 'ALPHA_URANIUM','RESULTS_REPORT_CITATION',
     . 'ALPHA_URANIUM','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_ALPHA_THORIUM',
     . 'ALPHA_THORIUM','RESULTS_NEEDED_BY_DATE',
     . 'ALPHA_THORIUM','DATE_ALL_WORK_COMPLETED',
     . 'ALPHA_THORIUM','RESULTS_REPORT_CITATION',
     . 'ALPHA_THORIUM','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_ALPHA_PLUTONIUM',
     . 'ALPHA_PLUTONIUM','RESULTS_NEEDED_BY_DATE',
     . 'ALPHA_PLUTONIUM','DATE_ALL_WORK_COMPLETED',
     . 'ALPHA_PLUTONIUM','RESULTS_REPORT_CITATION',
     . 'ALPHA_PLUTONIUM','ANY_SPECIAL_INSTRUCTIONS',
```

```
     . 'SAMPLE_INFORMATION','NEED_ALPHA_AM241_SANS_PU238',
     . 'ALPHA_AM241_SANS_PU238','RESULTS_NEEDED_BY_DATE',
     . 'ALPHA_AM241_SANS_PU238','DATE_ALL_WORK_COMPLETED',
     . 'ALPHA_AM241_SANS_PU238','RESULTS_REPORT_CITATION',
     . 'ALPHA_AM241_SANS_PU238','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_ALPHA_AM241_WITH_PU238',
     . 'ALPHA_AM241_WITH_PU238','RESULTS_NEEDED_BY_DATE',
     . 'ALPHA_AM241_WITH_PU238','DATE_ALL_WORK_COMPLETED',
     . 'ALPHA_AM241_WITH_PU238','RESULTS_REPORT_CITATION',
     . 'ALPHA_AM241_WITH_PU238','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_ALPHA_TOTAL_SPECTROMETRIC',
     . 'ALPHA_TOTAL_SPECTROMETRIC','RESULTS_NEEDED_BY_DATE',
     . 'ALPHA_TOTAL_SPECTROMETRIC','DATE_ALL_WORK_COMPLETED',
     . 'ALPHA_TOTAL_SPECTROMETRIC','RESULTS_REPORT_CITATION',
     . 'ALPHA_TOTAL_SPECTROMETRIC','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_ALPHA_OTHER',
     . 'ALPHA_OTHER','RESULTS_NEEDED_BY_DATE',
     . 'ALPHA_OTHER','DATE_ALL_WORK_COMPLETED',
     . 'ALPHA_OTHER','RESULTS_REPORT_CITATION',
     . 'ALPHA_OTHER','ANY_SPECIAL_INSTRUCTIONS'/
C
C    Program variable names (value and null indicator) for screen fields.
C
      data varblA/':fASTI:nASTI',':fACSN:nACSN',':fANAU:nANAU',
     . ':iARNB1:nARNB1',':iADAW1:nADAW1',':fARRC1:nARRC1',
     . ':fAASI1:nAASI1',':fANATh:nANATh',':iARNB2:nARNB2',
     . ':iADAW2:nADAW2',':fARRC2:nARRC2',':fAASI2:nAASI2',
     . ':fANAPu:nANAPu',':iARNB3:nARNB3',':iADAW3:nADAW3',
     . ':fARRC3:nARRC3',':fAASI3:nAASI3',':fANASP:nANASP',
     . ':iARNB4:nARNB4',':iADAW4:nADAW4',':fARRC4:nARRC4',
     . ':fAASI4:nAASI4',':fANAWP:nANAWP',':iARNB5:nARNB5',
     . ':iADAW5:nADAW5',':fARRC5:nARRC5',':fAASI5:nAASI5',
     . ':fANATS:nANATS',':iARNB6:nARNB6',':iADAW6:nADAW6',
     . ':fARRC6:nARRC6',':fAASI6:nAASI6',':fANAO:nANAO',
     . ':iARNB7:nARNB7',':iADAW7:nADAW7',':fARRC7:nARRC7',
     . ':fAASI7:nAASI7'/
C
C    Beta screen's field descriptors and field sizes.
C
      character*80 fieldB(46),fidatB(46)
      character*30 atrbtB(2,46)
      character*14 varblB(46)
      character*1 fitypB(46)
      integer*4 fdlenB(46),fdrowB(46),fdcolB(46),ficolB(46),mxnocB(46),
     . fdrndB(3,46),firndB(3,46), !Field renditions (add/update/search).
     . numfdB/46/   !The number of fields.
      integer*2 ifnulB(46),   !Null indicator (-1 if null, 0 otherwise).
     . ifrqdB(46)/46*0/,   !Points to if-required dependent field.
     . iditoB(46)/46*0/   !Points to allowed field for dittoing.
      logical ifmodB(46)   !Has a field been modified in an Update?
      common /dbdatB/ fieldB,fidatB,fdlenB,fdrowB,fdcolB,ficolB,mxnocB,
     . fdrndB,firndB,numfdB,fitypB,ifnulB,atrbtB,varblB,ifrqdB,ifmodB,
     . iditoB
C
C     Length-specified character fields for equivalences (f for field).
C
      character fBSTI*12,fBCSN*60,fBNBSr*1,fBDCL1*4,fBDCU1*1,fBRND1*6,
     . fBRNT1*5,fBDAW1*6,fBRRC1*20,fBASI1*1,fBNBSC*1,fBDCL2*4,fBDCU2*1,
     . fBRND2*6,fBRNT2*5,fBDAW2*6,fBRRC2*20,fBASI2*1,fBNBST*1,fBDCL3*4,
     . fBDCU3*1,fBRND3*6,fBRNT3*5,fBDAW3*6,fBRRC3*20,fBASI3*1,fBNBT*1,
```

```
     . fBDCL4*4,fBDCU4*1,fBRND4*6,fBRNT4*5,fBDAW4*6,fBRRC4*20,fBASI4*1,
     . fBNBO*1,fBDCL5*4,fBDCU5*1,fBNBNi*1,fBNBFe*1,fBNBS*1,fBNBPu*1,
     . fBRND5*6,fBRNT5*5,fBDAW5*6,fBRRC5*20,fBASI5*1
C
C       Real and integer field storage (r for real, i for integer).
C
      real*4 rBDCL1,rBDCL2,rBDCL3,rBDCL4,rBDCL5
      integer*4 iBRND1,iBDAW1,iBRND2,iBDAW2,iBRND3,iBDAW3,iBRND4,iBDAW4,
     . iBRND5,iBDAW5
C
C       Nonarrayed maximum field lengths (l for length).
C
      integer*4 lBSTI,lBCSN,lBNBSr,lBDCL1,lBDCU1,lBRND1,lBRNT1,lBDAW1,
     . lBRRC1,lBASI1,lBNBSC,lBDCL2,lBDCU2,lBRND2,lBRNT2,lBDAW2,lBRRC2,
     . lBASI2,lBNBST,lBDCL3,lBDCU3,lBRND3,lBRNT3,lBDAW3,lBRRC3,lBASI3,
     . lBNBT,lBDCL4,lBDCU4,lBRND4,lBRNT4,lBDAW4,lBRRC4,lBASI4,lBNBO,
     . lBDCL5,lBDCU5,lBNBNi,lBNBFe,lBNBS,lBNBPu,lBRND5,lBRNT5,lBDAW5,
     . lBRRC5,lBASI5
C
C       Nonarrayed null indicators for data transfers (n for null).
C
      integer*2 nBSTI,nBCSN,nBNBSr,nBDCL1,nBDCU1,nBRND1,nBRNT1,nBDAW1,
     . nBRRC1,nBASI1,nBNBSC,nBDCL2,nBDCU2,nBRND2,nBRNT2,nBDAW2,nBRRC2,
     . nBASI2,nBNBST,nBDCL3,nBDCU3,nBRND3,nBRNT3,nBDAW3,nBRRC3,nBASI3,
     . nBNBT,nBDCL4,nBDCU4,nBRND4,nBRNT4,nBDAW4,nBRRC4,nBASI4,nBNBO,
     . nBDCL5,nBDCU5,nBNBNi,nBNBFe,nBNBS,nBNBPu,nBRND5,nBRNT5,nBDAW5,
     . nBRRC5,nBASI5
C
C       Nonarrayed if-modified indicators (m for modified).
C
      integer*2 mBSTI,mBCSN,mBNBSr,mBDCL1,mBDCU1,mBRND1,mBRNT1,mBDAW1,
     . mBRRC1,mBASI1,mBNBSC,mBDCL2,mBDCU2,mBRND2,mBRNT2,mBDAW2,mBRRC2,
     . mBASI2,mBNBST,mBDCL3,mBDCU3,mBRND3,mBRNT3,mBDAW3,mBRRC3,mBASI3,
     . mBNBT,mBDCL4,mBDCU4,mBRND4,mBRNT4,mBDAW4,mBRRC4,mBASI4,mBNBO,
     . mBDCL5,mBDCU5,mBNBNi,mBNBFe,mBNBS,mBNBPu,mBRND5,mBRNT5,mBDAW5,
     . mBRRC5,mBASI5
C
C       Equivalence statements for data transfers.
C
      equivalence (fidatB(1)(1:12),fBSTI),(ifnulB(1),nBSTI),
     . (fidatB(2)(1:60),fBCSN),(ifnulB(2),nBCSN),
     . (fidatB(3)(1:1),fBNBSr),(ifnulB(3),nBNBSr),
     . (fidatB(4)(1:4),fBDCL1),(ifnulB(4),nBDCL1),
     . (fidatB(5)(1:1),fBDCU1),(ifnulB(5),nBDCU1),
     . (fidatB(6)(1:6),fBRND1),(ifnulB(6),nBRND1),
     . (fidatB(7)(1:5),fBRNT1),(ifnulB(7),nBRNT1),
     . (fidatB(8)(1:6),fBDAW1),(ifnulB(8),nBDAW1),
     . (fidatB(9)(1:20),fBRRC1),(ifnulB(9),nBRRC1),
     . (fidatB(10)(1:1),fBASI1),(ifnulB(10),nBASI1),
     . (fidatB(11)(1:1),fBNBSC),(ifnulB(11),nBNBSC),
     . (fidatB(12)(1:4),fBDCL2),(ifnulB(12),nBDCL2),
     . (fidatB(13)(1:1),fBDCU2),(ifnulB(13),nBDCU2),
     . (fidatB(14)(1:6),fBRND2),(ifnulB(14),nBRND2),
     . (fidatB(15)(1:5),fBRNT2),(ifnulB(15),nBRNT2),
     . (fidatB(16)(1:6),fBDAW2),(ifnulB(16),nBDAW2),
     . (fidatB(17)(1:20),fBRRC2),(ifnulB(17),nBRRC2),
     . (fidatB(18)(1:1),fBASI2),(ifnulB(18),nBASI2),
     . (fidatB(19)(1:1),fBNBST),(ifnulB(19),nBNBST),
     . (fidatB(20)(1:4),fBDCL3),(ifnulB(20),nBDCL3),
     . (fidatB(21)(1:1),fBDCU3),(ifnulB(21),nBDCU3),
```

```
. (fidatB(22)(1:6),fBRND3),(ifnulB(22),nBRND3),
. (fidatB(23)(1:5),fBRNT3),(ifnulB(23),nBRNT3),
. (fidatB(24)(1:6),fBDAW3),(ifnulB(24),nBDAW3),
. (fidatB(25)(1:20),fBRRC3),(ifnulB(25),nBRRC3),
. (fidatB(26)(1:1),fBASI3),(ifnulB(26),nBASI3),
. (fidatB(27)(1:1),fBNBT),(ifnulB(27),nBNBT),
. (fidatB(28)(1:4),fBDCL4),(ifnulB(28),nBDCL4),
. (fidatB(29)(1:1),fBDCU4),(ifnulB(29),nBDCU4),
. (fidatB(30)(1:6),fBRND4),(ifnulB(30),nBRND4),
. (fidatB(31)(1:5),fBRNT4),(ifnulB(31),nBRNT4),
. (fidatB(32)(1:6),fBDAW4),(ifnulB(32),nBDAW4),
. (fidatB(33)(1:20),fBRRC4),(ifnulB(33),nBRRC4),
. (fidatB(34)(1:1),fBASI4),(ifnulB(34),nBASI4),
. (fidatB(35)(1:1),fBNBO),(ifnulB(35),nBNBO),
. (fidatB(36)(1:4),fBDCL5),(ifnulB(36),nBDCL5),
. (fidatB(37)(1:1),fBDCU5),(ifnulB(37),nBDCU5),
. (fidatB(38)(1:1),fBNBNi),(ifnulB(38),nBNBNi),
. (fidatB(39)(1:1),fBNBFe),(ifnulB(39),nBNBFe),
. (fidatB(40)(1:1),fBNBS),(ifnulB(40),nBNBS),
. (fidatB(41)(1:1),fBNBPu),(ifnulB(41),nBNBPu),
. (fidatB(42)(1:6),fBRND5),(ifnulB(42),nBRND5),
. (fidatB(43)(1:5),fBRNT5),(ifnulB(43),nBRNT5),
. (fidatB(44)(1:6),fBDAW5),(ifnulB(44),nBDAW5),
. (fidatB(45)(1:20),fBRRC5),(ifnulB(45),nBRRC5),
. (fidatB(46)(1:1),fBASI5),(ifnulB(46),nBASI5),
. (mxnocB(1),lBSTI),(ifmodB(1),mBSTI),
. (mxnocB(2),lBCSN),(ifmodB(2),mBCSN),
. (mxnocB(3),lBNBSr),(ifmodB(3),mBNBSr),
. (mxnocB(4),lBDCL1),(ifmodB(4),mBDCL1),
. (mxnocB(5),lBDCU1),(ifmodB(5),mBDCU1),
. (mxnocB(6),lBRND1),(ifmodB(6),mBRND1),
. (mxnocB(7),lBRNT1),(ifmodB(7),mBRNT1),
. (mxnocB(8),lBDAW1),(ifmodB(8),mBDAW1),
. (mxnocB(9),lBRRC1),(ifmodB(9),mBRRC1),
. (mxnocB(10),lBASI1),(ifmodB(10),mBASI1),
. (mxnocB(11),lBNBSC),(ifmodB(11),mBNBSC),
. (mxnocB(12),lBDCL2),(ifmodB(12),mBDCL2),
. (mxnocB(13),lBDCU2),(ifmodB(13),mBDCU2),
. (mxnocB(14),lBRND2),(ifmodB(14),mBRND2),
. (mxnocB(15),lBRNT2),(ifmodB(15),mBRNT2),
. (mxnocB(16),lBDAW2),(ifmodB(16),mBDAW2),
. (mxnocB(17),lBRRC2),(ifmodB(17),mBRRC2),
. (mxnocB(18),lBASI2),(ifmodB(18),mBASI2),
. (mxnocB(19),lBNBST),(ifmodB(19),mBNBST),
. (mxnocB(20),lBDCL3),(ifmodB(20),mBDCL3),
. (mxnocB(21),lBDCU3),(ifmodB(21),mBDCU3),
. (mxnocB(22),lBRND3),(ifmodB(22),mBRND3),
. (mxnocB(23),lBRNT3),(ifmodB(23),mBRNT3),
. (mxnocB(24),lBDAW3),(ifmodB(24),mBDAW3),
. (mxnocB(25),lBRRC3),(ifmodB(25),mBRRC3),
. (mxnocB(26),lBASI3),(ifmodB(26),mBASI3),
. (mxnocB(27),lBNBT),(ifmodB(27),mBNBT),
. (mxnocB(28),lBDCL4),(ifmodB(28),mBDCL4),
. (mxnocB(29),lBDCU4),(ifmodB(29),mBDCU4),
. (mxnocB(30),lBRND4),(ifmodB(30),mBRND4),
. (mxnocB(31),lBRNT4),(ifmodB(31),mBRNT4),
. (mxnocB(32),lBDAW4),(ifmodB(32),mBDAW4),
. (mxnocB(33),lBRRC4),(ifmodB(33),mBRRC4),
. (mxnocB(34),lBASI4),(ifmodB(34),mBASI4),
. (mxnocB(35),lBNBO),(ifmodB(35),mBNBO),
```

```
     . (mxnocB(36),lBDCL5),(ifmodB(36),mBDCL5),
     . (mxnocB(37),lBDCU5),(ifmodB(37),mBDCU5),
     . (mxnocB(38),lBNBNi),(ifmodB(38),mBNBNi),
     . (mxnocB(39),lBNBFe),(ifmodB(39),mBNBFe),
     . (mxnocB(40),lBNBS),(ifmodB(40),mBNBS),
     . (mxnocB(41),lBNBPu),(ifmodB(41),mBNBPu),
     . (mxnocB(42),lBRND5),(ifmodB(42),mBRND5),
     . (mxnocB(43),lBRNT5),(ifmodB(43),mBRNT5),
     . (mxnocB(44),lBDAW5),(ifmodB(44),mBDAW5),
     . (mxnocB(45),lBRRC5),(ifmodB(45),mBRRC5),
     . (mxnocB(46),lBASI5),(ifmodB(46),mBASI5)
C
C    Database table and attribute names for screen fields.
C
      data atrbtB/
     . 'BETA_STRONTIUM_90','SAMPLE_TRACKING_ID',
     . 'SAMPLE_INFORMATION','CUSTOMER_SAMPLE_NAME',
     . 'SAMPLE_INFORMATION','NEED_BETA_STRONTIUM_90',
     . 'BETA_STRONTIUM_90','DETECTOR_COUNT_LENGTH',
     . 'BETA_STRONTIUM_90','DETECTOR_COUNT_LENGTH_UNITS',
     . 'BETA_STRONTIUM_90','RESULTS_NEEDED_BY_DATE',
     . 'BETA_STRONTIUM_90','RESULTS_NEEDED_BY_TIME',
     . 'BETA_STRONTIUM_90','DATE_ALL_WORK_COMPLETED',
     . 'BETA_STRONTIUM_90','RESULTS_REPORT_CITATION',
     . 'BETA_STRONTIUM_90','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_BETA_STRONTIUM_89_AND_90',
     . 'BETA_STRONTIUM_89_AND_90','DETECTOR_COUNT_LENGTH',
     . 'BETA_STRONTIUM_89_AND_90','DETECTOR_COUNT_LENGTH_UNITS',
     . 'BETA_STRONTIUM_89_AND_90','RESULTS_NEEDED_BY_DATE',
     . 'BETA_STRONTIUM_89_AND_90','RESULTS_NEEDED_BY_TIME',
     . 'BETA_STRONTIUM_89_AND_90','DATE_ALL_WORK_COMPLETED',
     . 'BETA_STRONTIUM_89_AND_90','RESULTS_REPORT_CITATION',
     . 'BETA_STRONTIUM_89_AND_90','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_BETA_TOTAL_STRONTIUM',
     . 'BETA_TOTAL_STRONTIUM','DETECTOR_COUNT_LENGTH',
     . 'BETA_TOTAL_STRONTIUM','DETECTOR_COUNT_LENGTH_UNITS',
     . 'BETA_TOTAL_STRONTIUM','RESULTS_NEEDED_BY_DATE',
     . 'BETA_TOTAL_STRONTIUM','RESULTS_NEEDED_BY_TIME',
     . 'BETA_TOTAL_STRONTIUM','DATE_ALL_WORK_COMPLETED',
     . 'BETA_TOTAL_STRONTIUM','RESULTS_REPORT_CITATION',
     . 'BETA_TOTAL_STRONTIUM','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_BETA_TRITIUM',
     . 'BETA_TRITIUM','DETECTOR_COUNT_LENGTH',
     . 'BETA_TRITIUM','DETECTOR_COUNT_LENGTH_UNITS',
     . 'BETA_TRITIUM','RESULTS_NEEDED_BY_DATE',
     . 'BETA_TRITIUM','RESULTS_NEEDED_BY_TIME',
     . 'BETA_TRITIUM','DATE_ALL_WORK_COMPLETED',
     . 'BETA_TRITIUM','RESULTS_REPORT_CITATION',
     . 'BETA_TRITIUM','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_BETA_OTHER',
     . 'BETA_OTHER','DETECTOR_COUNT_LENGTH',
     . 'BETA_OTHER','DETECTOR_COUNT_LENGTH_UNITS',
     . 'BETA_OTHER','NEED_BETA_NICKEL_63',
     . 'BETA_OTHER','NEED_BETA_IRON_55',
     . 'BETA_OTHER','NEED_BETA_SULFUR_35',
     . 'BETA_OTHER','NEED_BETA_PLUTONIUM_241',
     . 'BETA_OTHER','RESULTS_NEEDED_BY_DATE',
     . 'BETA_OTHER','RESULTS_NEEDED_BY_TIME',
     . 'BETA_OTHER','DATE_ALL_WORK_COMPLETED',
     . 'BETA_OTHER','RESULTS_REPORT_CITATION',
```

```
     . 'BETA_OTHER','ANY_SPECIAL_INSTRUCTIONS'/
C
C    Program variable names (value and null indicator) for screen fields.
C
      data varblB/':fBSTI:nBSTI',':fBCSN:nBCSN',':fBNBSr:nBNBSr',
     . ':rBDCL1:nBDCL1',':fBDCU1:nBDCU1',':iBRND1:nBRND1',
     . ':fBRNT1:nBRNT1',':iBDAW1:nBDAW1',':fBRRC1:nBRRC1',
     . ':fBASI1:nBASI1',':fBNBSC:nBNBSC',':rBDCL2:nBDCL2',
     . ':fBDCU2:nBDCU2',':iBRND2:nBRND2',':fBRNT2:nBRNT2',
     . ':iBDAW2:nBDAW2',':fBRRC2:nBRRC2',':fBASI2:nBASI2',
     . ':fBNBST:nBNBST',':rBDCL3:nBDCL3',':fBDCU3:nBDCU3',
     . ':iBRND3:nBRND3',':fBRNT3:nBRNT3',':iBDAW3:nBDAW3',
     . ':fBRRC3:nBRRC3',':fBASI3:nBASI3',':fBNBT:nBNBT',
     . ':rBDCL4:nBDCL4',':fBDCU4:nBDCU4',':iBRND4:nBRND4',
     . ':fBRNT4:nBRNT4',':iBDAW4:nBDAW4',':fBRRC4:nBRRC4',
     . ':fBASI4:nBASI4',':fBNBO:nBNBO',':rBDCL5:nBDCL5',
     . ':fBDCU5:nBDCU5',':fBNBNi:nBNBNi',':fBNBFe:nBNBFe',
     . ':fBNBS:nBNBS',':fBNBPu:nBNBPu',':iBRND5:nBRND5',
     . ':fBRNT5:nBRNT5',':iBDAW5:nBDAW5',':fBRRC5:nBRRC5',
     . ':fBASI5:nBASI5'/
C
C    Gamma screen's field descriptors and field sizes.
C
      character*80 fieldG(35),fidatG(35)
      character*30 atrbtG(2,35)
      character*14 varblG(35)
      character*1 fitypG(35)
      integer*4 fdlenG(35),fdrowG(35),fdcolG(35),ficolG(35),mxnocG(35),
     . fdrndG(3,35),firndG(3,35), !Field renditions (add/update/search).
     . numfdG/35/   !The number of fields.
      integer*2 ifnulG(35),   !Null indicator (-1 if null, 0 otherwise).
     . ifrqdG(35)/35*0/,   !Points to if-required dependent field.
     . iditoG(35)/35*0/   !Points to allowed field for dittoing.
      logical ifmodG(35)   !Has a field been modified in an Update?
      common /dbdatG/ fieldG,fidatG,fdlenG,fdrowG,fdcolG,ficolG,mxnocG,
     . fdrndG,firndG,numfdG,fitypG,ifnulG,atrbtG,varblG,ifrqdG,ifmodG,
     . iditoG
C
C      Length-specified character fields for equivalences (f for field).
C
      character fGSTI*12,fGCSN*60,fGNGS*1,fGDCL1*4,fGDCU1*1,fGRND1*6,
     . fGRNT1*5,fGDAW1*6,fGDSC1*6,fGRSI1*14,fGITA1*1,fGRRC1*20,fGASI1*1,
     . fGNGFI*1,fGDCL2*4,fGDCU2*1,fGRND2*6,fGRNT2*5,fGDAW2*6,fGDSC2*6,
     . fGRSI2*14,fGITA2*1,fGRRC2*20,fGASI2*1,fGNGO*1,fGDCL3*4,fGDCU3*1,
     . fGRND3*6,fGRNT3*5,fGDAW3*6,fGDSC3*6,fGRSI3*14,fGITA3*1,fGRRC3*20,
     . fGASI3*1
C
C       Real and integer field storage (r for real, i for integer).
C
      real*4 rGDCL1,rGDCL2,rGDCL3
      integer*4 iGRND1,iGDAW1,iGDSC1,iGRND2,iGDAW2,iGDSC2,iGRND3,iGDAW3,
     . iGDSC3
C
C       Nonarrayed maximum field lengths (l for length).
C
      integer*4 lGSTI,lGCSN,lGNGS,lGDCL1,lGDCU1,lGRND1,lGRNT1,lGDAW1,
     . lGDSC1,lGRSI1,lGITA1,lGRRC1,lGASI1,lGNGFI,lGDCL2,lGDCU2,lGRND2,
     . lGRNT2,lGDAW2,lGDSC2,lGRSI2,lGITA2,lGRRC2,lGASI2,lGNGO,lGDCL3,
     . lGDCU3,lGRND3,lGRNT3,lGDAW3,lGDSC3,lGRSI3,lGITA3,lGRRC3,lGASI3
C
```

```
C       Nonarrayed null indicators for data transfers (n for null).
C
      integer*2 nGSTI,nGCSN,nGNGS,nGDCL1,nGDCU1,nGRND1,nGRNT1,nGDAW1,
     . nGDSC1,nGRSI1,nGITA1,nGRRC1,nGASI1,nGNGFI,nGDCL2,nGDCU2,nGRND2,
     . nGRNT2,nGDAW2,nGDSC2,nGRSI2,nGITA2,nGRRC2,nGASI2,nGNGO,nGDCL3,
     . nGDCU3,nGRND3,nGRNT3,nGDAW3,nGDSC3,nGRSI3,nGITA3,nGRRC3,nGASI3
C
C       Nonarrayed if-modified indicators (m for modified).
C
      integer*2 mGSTI,mGCSN,mGNGS,mGDCL1,mGDCU1,mGRND1,mGRNT1,mGDAW1,
     . mGDSC1,mGRSI1,mGITA1,mGRRC1,mGASI1,mGNGFI,mGDCL2,mGDCU2,mGRND2,
     . mGRNT2,mGDAW2,mGDSC2,mGRSI2,mGITA2,mGRRC2,mGASI2,mGNGO,mGDCL3,
     . mGDCU3,mGRND3,mGRNT3,mGDAW3,mGDSC3,mGRSI3,mGITA3,mGRRC3,mGASI3
C
C       Equivalence statements for data transfers.
C
      equivalence (fidatG(1)(1:12),fGSTI),(ifnulG(1),nGSTI),
     . (fidatG(2)(1:60),fGCSN),(ifnulG(2),nGCSN),
     . (fidatG(3)(1:1),fGNGS),(ifnulG(3),nGNGS),
     . (fidatG(4)(1:4),fGDCL1),(ifnulG(4),nGDCL1),
     . (fidatG(5)(1:1),fGDCU1),(ifnulG(5),nGDCU1),
     . (fidatG(6)(1:6),fGRND1),(ifnulG(6),nGRND1),
     . (fidatG(7)(1:5),fGRNT1),(ifnulG(7),nGRNT1),
     . (fidatG(8)(1:6),fGDAW1),(ifnulG(8),nGDAW1),
     . (fidatG(9)(1:6),fGDSC1),(ifnulG(9),nGDSC1),
     . (fidatG(10)(1:14),fGRSI1),(ifnulG(10),nGRSI1),
     . (fidatG(11)(1:1),fGITA1),(ifnulG(11),nGITA1),
     . (fidatG(12)(1:20),fGRRC1),(ifnulG(12),nGRRC1),
     . (fidatG(13)(1:1),fGASI1),(ifnulG(13),nGASI1),
     . (fidatG(14)(1:1),fGNGFI),(ifnulG(14),nGNGFI),
     . (fidatG(15)(1:4),fGDCL2),(ifnulG(15),nGDCL2),
     . (fidatG(16)(1:1),fGDCU2),(ifnulG(16),nGDCU2),
     . (fidatG(17)(1:6),fGRND2),(ifnulG(17),nGRND2),
     . (fidatG(18)(1:5),fGRNT2),(ifnulG(18),nGRNT2),
     . (fidatG(19)(1:6),fGDAW2),(ifnulG(19),nGDAW2),
     . (fidatG(20)(1:6),fGDSC2),(ifnulG(20),nGDSC2),
     . (fidatG(21)(1:14),fGRSI2),(ifnulG(21),nGRSI2),
     . (fidatG(22)(1:1),fGITA2),(ifnulG(22),nGITA2),
     . (fidatG(23)(1:20),fGRRC2),(ifnulG(23),nGRRC2),
     . (fidatG(24)(1:1),fGASI2),(ifnulG(24),nGASI2),
     . (fidatG(25)(1:1),fGNGO),(ifnulG(25),nGNGO),
     . (fidatG(26)(1:4),fGDCL3),(ifnulG(26),nGDCL3),
     . (fidatG(27)(1:1),fGDCU3),(ifnulG(27),nGDCU3),
     . (fidatG(28)(1:6),fGRND3),(ifnulG(28),nGRND3),
     . (fidatG(29)(1:5),fGRNT3),(ifnulG(29),nGRNT3),
     . (fidatG(30)(1:6),fGDAW3),(ifnulG(30),nGDAW3),
     . (fidatG(31)(1:6),fGDSC3),(ifnulG(31),nGDSC3),
     . (fidatG(32)(1:14),fGRSI3),(ifnulG(32),nGRSI3),
     . (fidatG(33)(1:1),fGITA3),(ifnulG(33),nGITA3),
     . (fidatG(34)(1:20),fGRRC3),(ifnulG(34),nGRRC3),
     . (fidatG(35)(1:1),fGASI3),(ifnulG(35),nGASI3),
     . (mxnocG(1),lGSTI),(ifmodG(1),mGSTI),
     . (mxnocG(2),lGCSN),(ifmodG(2),mGCSN),
     . (mxnocG(3),lGNGS),(ifmodG(3),mGNGS),
     . (mxnocG(4),lGDCL1),(ifmodG(4),mGDCL1),
     . (mxnocG(5),lGDCU1),(ifmodG(5),mGDCU1),
     . (mxnocG(6),lGRND1),(ifmodG(6),mGRND1),
     . (mxnocG(7),lGRNT1),(ifmodG(7),mGRNT1),
     . (mxnocG(8),lGDAW1),(ifmodG(8),mGDAW1),
     . (mxnocG(9),lGDSC1),(ifmodG(9),mGDSC1),
```

```
     . (mxnocG(10),lGRSI1),(ifmodG(10),mGRSI1),
     . (mxnocG(11),lGITA1),(ifmodG(11),mGITA1),
     . (mxnocG(12),lGRRC1),(ifmodG(12),mGRRC1),
     . (mxnocG(13),lGASI1),(ifmodG(13),mGASI1),
     . (mxnocG(14),lGNGFI),(ifmodG(14),mGNGFI),
     . (mxnocG(15),lGDCL2),(ifmodG(15),mGDCL2),
     . (mxnocG(16),lGDCU2),(ifmodG(16),mGDCU2),
     . (mxnocG(17),lGRND2),(ifmodG(17),mGRND2),
     . (mxnocG(18),lGRNT2),(ifmodG(18),mGRNT2),
     . (mxnocG(19),lGDAW2),(ifmodG(19),mGDAW2),
     . (mxnocG(20),lGDSC2),(ifmodG(20),mGDSC2),
     . (mxnocG(21),lGRSI2),(ifmodG(21),mGRSI2),
     . (mxnocG(22),lGITA2),(ifmodG(22),mGITA2),
     . (mxnocG(23),lGRRC2),(ifmodG(23),mGRRC2),
     . (mxnocG(24),lGASI2),(ifmodG(24),mGASI2),
     . (mxnocG(25),lGNGO),(ifmodG(25),mGNGO),
     . (mxnocG(26),lGDCL3),(ifmodG(26),mGDCL3),
     . (mxnocG(27),lGDCU3),(ifmodG(27),mGDCU3),
     . (mxnocG(28),lGRND3),(ifmodG(28),mGRND3),
     . (mxnocG(29),lGRNT3),(ifmodG(29),mGRNT3),
     . (mxnocG(30),lGDAW3),(ifmodG(30),mGDAW3),
     . (mxnocG(31),lGDSC3),(ifmodG(31),mGDSC3),
     . (mxnocG(32),lGRSI3),(ifmodG(32),mGRSI3),
     . (mxnocG(33),lGITA3),(ifmodG(33),mGITA3),
     . (mxnocG(34),lGRRC3),(ifmodG(34),mGRRC3),
     . (mxnocG(35),lGASI3),(ifmodG(35),mGASI3)
C
C    Database table and attribute names for screen fields.
C
      data atrbtG/
     . 'GAMMA_SCREEN','SAMPLE_TRACKING_ID',
     . 'SAMPLE_INFORMATION','CUSTOMER_SAMPLE_NAME',
     . 'SAMPLE_INFORMATION','NEED_GAMMA_SCREEN',
     . 'GAMMA_SCREEN','DETECTOR_COUNT_LENGTH',
     . 'GAMMA_SCREEN','DETECTOR_COUNT_LENGTH_UNITS',
     . 'GAMMA_SCREEN','RESULTS_NEEDED_BY_DATE',
     . 'GAMMA_SCREEN','RESULTS_NEEDED_BY_TIME',
     . 'GAMMA_SCREEN','DATE_ALL_WORK_COMPLETED',
     . 'GAMMA_SCREEN','DATE_SAMPLE_COUNTED',
     . 'GAMMA_SCREEN','RML_SPECTRAL_ID',
     . 'GAMMA_SCREEN','IS_THIS_A_SPECTRUM_RECOUNT',
     . 'GAMMA_SCREEN','RESULTS_REPORT_CITATION',
     . 'GAMMA_SCREEN','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_GAMMA_FULL_ISOTOPIC',
     . 'GAMMA_FULL_ISOTOPIC','DETECTOR_COUNT_LENGTH',
     . 'GAMMA_FULL_ISOTOPIC','DETECTOR_COUNT_LENGTH_UNITS',
     . 'GAMMA_FULL_ISOTOPIC','RESULTS_NEEDED_BY_DATE',
     . 'GAMMA_FULL_ISOTOPIC','RESULTS_NEEDED_BY_TIME',
     . 'GAMMA_FULL_ISOTOPIC','DATE_ALL_WORK_COMPLETED',
     . 'GAMMA_FULL_ISOTOPIC','DATE_SAMPLE_COUNTED',
     . 'GAMMA_FULL_ISOTOPIC','RML_SPECTRAL_ID',
     . 'GAMMA_FULL_ISOTOPIC','IS_THIS_A_SPECTRUM_RECOUNT',
     . 'GAMMA_FULL_ISOTOPIC','RESULTS_REPORT_CITATION',
     . 'GAMMA_FULL_ISOTOPIC','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_GAMMA_OTHER',
     . 'GAMMA_OTHER','DETECTOR_COUNT_LENGTH',
     . 'GAMMA_OTHER','DETECTOR_COUNT_LENGTH_UNITS',
     . 'GAMMA_OTHER','RESULTS_NEEDED_BY_DATE',
     . 'GAMMA_OTHER','RESULTS_NEEDED_BY_TIME',
     . 'GAMMA_OTHER','DATE_ALL_WORK_COMPLETED',
```

```
     . 'GAMMA_OTHER','DATE_SAMPLE_COUNTED',
     . 'GAMMA_OTHER','RML_SPECTRAL_ID',
     . 'GAMMA_OTHER','IS_THIS_A_SPECTRUM_RECOUNT',
     . 'GAMMA_OTHER','RESULTS_REPORT_CITATION',
     . 'GAMMA_OTHER','ANY_SPECIAL_INSTRUCTIONS'/
C
C    Program variable names (value and null indicator) for screen fields.
C
      data varblG/':fGSTI:nGSTI',':fGCSN:nGCSN',':fGNGS:nGNGS',
     . ':rGDCL1:nGDCL1',':fGDCU1:nGDCU1',':iGRND1:nGRND1',
     . ':fGRNT1:nGRNT1',':iGDAW1:nGDAW1',':iGDSC1:nGDSC1',
     . ':fGRSI1:nGRSI1',':fGITA1:nGITA1',':fGRRC1:nGRRC1',
     . ':fGASI1:nGASI1',':fGNGFI:nGNGFI',':rGDCL2:nGDCL2',
     . ':fGDCU2:nGDCU2',':iGRND2:nGRND2',':fGRNT2:nGRNT2',
     . ':iGDAW2:nGDAW2',':iGDSC2:nGDSC2',':fGRSI2:nGRSI2',
     . ':fGITA2:nGITA2',':fGRRC2:nGRRC2',':fGASI2:nGASI2',
     . ':fGNGO:nGNGO',':rGDCL3:nGDCL3',':fGDCU3:nGDCU3',
     . ':iGRND3:nGRND3',':fGRNT3:nGRNT3',':iGDAW3:nGDAW3',
     . ':iGDSC3:nGDSC3',':fGRSI3:nGRSI3',':fGITA3:nGITA3',
     . ':fGRRC3:nGRRC3',':fGASI3:nGASI3'/
C
C    Gross alpha-beta screen's field descriptors and field sizes.
C
      character*80 fieldC(18),fidatC(18)
      character*30 atrbtC(2,18)
      character*14 varblC(18)
      character*1 fitypC(18)
      integer*4 fdlenC(18),fdrowC(18),fdcolC(18),ficolC(18),mxnocC(18),
     . fdrndC(3,18),firndC(3,18), !Field renditions (add/update/search).
     . numfdC/18/   !The number of fields.
      integer*2 ifnulC(18),   !Null indicator (-1 if null, 0 otherwise).
     . ifrqdC(18)/18*0/,   !Points to if-required dependent field.
     . iditoC(18)/18*0/   !Points to allowed field for dittoing.
      logical ifmodC(18)   !Has a field been modified in an Update?
      common /dbdatC/ fieldC,fidatC,fdlenC,fdrowC,fdcolC,ficolC,mxnocC,
     . fdrndC,firndC,numfdC,fitypC,ifnulC,atrbtC,varblC,ifrqdC,ifmodC,
     . iditoC
C
C     Length-specified character fields for equivalences (1 for length).
C
      character fCSTI*12,fCCSN*60,fCNGAF*1,fCDCL1*4,fCDCU1*1,fCRND1*6,
     . fCRNT1*5,fCDAW1*6,fCRRC1*20,fCASI1*1,fCNGAO*1,fCDCL2*4,fCDCU2*1,
     . fCRND2*6,fCRNT2*5,fCDAW2*6,fCRRC2*20,fCASI2*1
C
C       Real and integer field storage (r for real, i for integer).
C
      real*4 rCDCL1,rCDCL2
      integer*4 iCRND1,iCDAW1,iCRND2,iCDAW2
C
C       Nonarrayed maximum field lengths (1 for length).
C
      integer*4 lCSTI,lCCSN,lCNGAF,lCDCL1,lCDCU1,lCRND1,lCRNT1,lCDAW1,
     . lCRRC1,lCASI1,lCNGAO,lCDCL2,lCDCU2,lCRND2,lCRNT2,lCDAW2,lCRRC2,
     . lCASI2
C
C       Nonarrayed null indicators for data transfers (n for null).
C
      integer*2 nCSTI,nCCSN,nCNGAF,nCDCL1,nCDCU1,nCRND1,nCRNT1,nCDAW1,
     . nCRRC1,nCASI1,nCNGAO,nCDCL2,nCDCU2,nCRND2,nCRNT2,nCDAW2,nCRRC2,
     . nCASI2
```

```
C
C       Nonarrayed if-modified indicators (m for modified).
C
      integer*2 mCSTI,mCCSN,mCNGAF,mCDCL1,mCDCU1,mCRND1,mCRNT1,mCDAW1,
     . mCRRC1,mCASI1,mCNGAO,mCDCL2,mCDCU2,mCRND2,mCRNT2,mCDAW2,mCRRC2,
     . mCASI2
C
C       Equivalence statements for data transfers.
C
      equivalence (fidatC(1)(1:12),fCSTI),(ifnulC(1),nCSTI),
     . (fidatC(2)(1:60),fCCSN),(ifnulC(2),nCCSN),
     . (fidatC(3)(1:1),fCNGAF),(ifnulC(3),nCNGAF),
     . (fidatC(4)(1:4),fCDCL1),(ifnulC(4),nCDCL1),
     . (fidatC(5)(1:1),fCDCU1),(ifnulC(5),nCDCU1),
     . (fidatC(6)(1:6),fCRND1),(ifnulC(6),nCRND1),
     . (fidatC(7)(1:5),fCRNT1),(ifnulC(7),nCRNT1),
     . (fidatC(8)(1:6),fCDAW1),(ifnulC(8),nCDAW1),
     . (fidatC(9)(1:20),fCRRC1),(ifnulC(9),nCRRC1),
     . (fidatC(10)(1:1),fCASI1),(ifnulC(10),nCASI1),
     . (fidatC(11)(1:1),fCNGAO),(ifnulC(11),nCNGAO),
     . (fidatC(12)(1:4),fCDCL2),(ifnulC(12),nCDCL2),
     . (fidatC(13)(1:1),fCDCU2),(ifnulC(13),nCDCU2),
     . (fidatC(14)(1:6),fCRND2),(ifnulC(14),nCRND2),
     . (fidatC(15)(1:5),fCRNT2),(ifnulC(15),nCRNT2),
     . (fidatC(16)(1:6),fCDAW2),(ifnulC(16),nCDAW2),
     . (fidatC(17)(1:20),fCRRC2),(ifnulC(17),nCRRC2),
     . (fidatC(18)(1:1),fCASI2),(ifnulC(18),nCASI2),
     . (mxnocC(1),lCSTI),(ifmodC(1),mCSTI),
     . (mxnocC(2),lCCSN),(ifmodC(2),mCCSN),
     . (mxnocC(3),lCNGAF),(ifmodC(3),mCNGAF),
     . (mxnocC(4),lCDCL1),(ifmodC(4),mCDCL1),
     . (mxnocC(5),lCDCU1),(ifmodC(5),mCDCU1),
     . (mxnocC(6),lCRND1),(ifmodC(6),mCRND1),
     . (mxnocC(7),lCRNT1),(ifmodC(7),mCRNT1),
     . (mxnocC(8),lCDAW1),(ifmodC(8),mCDAW1),
     . (mxnocC(9),lCRRC1),(ifmodC(9),mCRRC1),
     . (mxnocC(10),lCASI1),(ifmodC(10),mCASI1),
     . (mxnocC(11),lCNGAO),(ifmodC(11),mCNGAO),
     . (mxnocC(12),lCDCL2),(ifmodC(12),mCDCL2),
     . (mxnocC(13),lCDCU2),(ifmodC(13),mCDCU2),
     . (mxnocC(14),lCRND2),(ifmodC(14),mCRND2),
     . (mxnocC(15),lCRNT2),(ifmodC(15),mCRNT2),
     . (mxnocC(16),lCDAW2),(ifmodC(16),mCDAW2),
     . (mxnocC(17),lCRRC2),(ifmodC(17),mCRRC2),
     . (mxnocC(18),lCASI2),(ifmodC(18),mCASI2)
C
C     Database table and attribute names for screen fields.
C
      data atrbtC/
     . 'GROSS_ALPHA_BETA_AIR_FILTERS','SAMPLE_TRACKING_ID',
     . 'SAMPLE_INFORMATION','CUSTOMER_SAMPLE_NAME',
     . 'SAMPLE_INFORMATION','NEED_GROSS_ALPHA_BETA_FILTERS',
     . 'GROSS_ALPHA_BETA_AIR_FILTERS','DETECTOR_COUNT_LENGTH',
     . 'GROSS_ALPHA_BETA_AIR_FILTERS','DETECTOR_COUNT_LENGTH_UNITS',
     . 'GROSS_ALPHA_BETA_AIR_FILTERS','RESULTS_NEEDED_BY_DATE',
     . 'GROSS_ALPHA_BETA_AIR_FILTERS','RESULTS_NEEDED_BY_TIME',
     . 'GROSS_ALPHA_BETA_AIR_FILTERS','DATE_ALL_WORK_COMPLETED',
     . 'GROSS_ALPHA_BETA_AIR_FILTERS','RESULTS_REPORT_CITATION',
     . 'GROSS_ALPHA_BETA_AIR_FILTERS','ANY_SPECIAL_INSTRUCTIONS',
     . 'SAMPLE_INFORMATION','NEED_GROSS_ALPHA_BETA_OTHER',
```

```
     . 'GROSS_ALPHA_BETA_OTHER','DETECTOR_COUNT_LENGTH',
     . 'GROSS_ALPHA_BETA_OTHER','DETECTOR_COUNT_LENGTH_UNITS',
     . 'GROSS_ALPHA_BETA_OTHER','RESULTS_NEEDED_BY_DATE',
     . 'GROSS_ALPHA_BETA_OTHER','RESULTS_NEEDED_BY_TIME',
     . 'GROSS_ALPHA_BETA_OTHER','DATE_ALL_WORK_COMPLETED',
     . 'GROSS_ALPHA_BETA_OTHER','RESULTS_REPORT_CITATION',
     . 'GROSS_ALPHA_BETA_OTHER','ANY_SPECIAL_INSTRUCTIONS'/
C
C    Program variable names (value and null indicator) for screen fields.
C
      data varblC/':fCSTI:nCSTI',':fCCSN:nCCSN',':fCNGAF:nCNGAF',
     . ':rCDCL1:nCDCL1',':fCDCU1:nCDCU1',':iCRND1:nCRND1',
     . ':fCRNT1:nCRNT1',':iCDAW1:nCDAW1',':fCRRC1:nCRRC1',
     . ':fCASI1:nCASI1',':fCNGAO:nCNGAO',':rCDCL2:nCDCL2',
     . ':fCDCU2:nCDCU2',':iCRND2:nCRND2',':fCRNT2:nCRNT2',
     . ':iCDAW2:nCDAW2',':fCRRC2:nCRRC2',':fCASI2:nCASI2'/
C
C    Special instruction screen's field descriptors and field sizes.
C
      character*80 fieldI(32),fidatI(32)
      character*30 atrbtI(2,5)
      character*14 varblI(5)
      character*1 fitypI(32)
      integer*4 fdlenI(32),fdrowI(32),fdcolI(32),ficolI(32),mxnocI(32),
     . fdrndI(3,32),firndI(3,32), !Field renditions (add/update/search).
     . numfdI/32/   !The number of fields.
      integer*2 ifnulI(32),   !Null indicator (-1 if null, 0 otherwise).
     . ifrqdI(32)/32*0/,   !Points to if-required dependent field.
     . iditoI(32)/32*0/   !Points to allowed field for dittoing.
      logical ifmodI(32)   !Has a field been modified in an Update?
      common /dbdatI/ fieldI,fidatI,fdlenI,fdrowI,fdcolI,ficolI,mxnocI,
     . fdrndI,firndI,numfdI,fitypI,ifnulI,atrbtI,varblI,ifrqdI,ifmodI,
     . iditoI
C
C      Length-specified character fields for equivalences (f for field).
C
      character fISTI*12,fICSN*60,fIOS1*16,fISI1*50,fIOS2*16,fISI2*50,
     . fIOS3*16,fISI3*50,fIOS4*16,fISI4*50,fIOS5*16,fISI5*50,fIOS6*16,
     . fISI6*50,fIOS7*16,fISI7*50,fIOS8*16,fISI8*50,fIOS9*16,fISI9*50,
     . fIOS10*16,fISI10*50,fIOS11*16,fISI11*50,fIOS12*16,fISI12*50,
     . fIOS13*16,fISI13*50,fIOS14*16,fISI14*50,fIOS15*16,fISI15*50,
     . fIOS1t*16,fISI1t*50   !And spares for inserts and fetchs.
C
C       Integer field storage (i for integer).
C
C      integer*4 siline   !Declared in the main routine.
C
C       Nonarrayed maximum field lengths (l for length).
C
      integer*4 lISTI,lICSN,lIOS1,lISI1,lIOS2,lISI2,lIOS3,lISI3,lIOS4,
     . lISI4,lIOS5,lISI5,lIOS6,lISI6,lIOS7,lISI7,lIOS8,lISI8,lIOS9,
     . lISI9,lIOS10,lISI10,lIOS11,lISI11,lIOS12,lISI12,lIOS13,lISI13,
     . lIOS14,lISI14,lIOS15,lISI15,
     . lIOS1t,lISI1t   !And spares for inserts and fetchs.
C
C       Nonarrayed null    icators for data transfers (n for null).
C
      integer*2 nISTI,nICSN,nIOS1,nISI1,nIOS2,nISI2,nIOS3,nISI3,nIOS4,
     . nISI4,nIOS5,nISI5,nIOS6,nISI6,nIOS7,nISI7,nIOS8,nISI8,nIOS9,
     . nISI9,nIOS10,nISI10,nIOS11,nISI11,nIOS12,nISI12,nIOS13,nISI13,
```

```
     . nIOS14,nISI14,nIOS15;nISI15,
     . nIOS1t,nISI1t   !And spares for inserts and fetchs.
C
C       Nonarrayed if-modified indicators (m for modified).
C
      integer*2 mISTI,mICSN,mIOS1,mISI1,mIOS2,mISI2,mIOS3,mISI3,mIOS4,
     . mISI4,mIOS5,mISI5,mIOS6,mISI6,mIOS7,mISI7,mIOS8,mISI8,mIOS9,
     . mISI9,mIOS10,mISI10,mIOS11,mISI11,mIOS12,mISI12,mIOS13,mISI13,
     . mIOS14,mISI14,mIOS15,mISI15,
     . mIOS1t,mISI1t   !And spares for inserts and fetchs.
C
C       Equivalence statements for data transfers.
C
      equivalence (fidatI(1)(1:12),fISTI),(ifnulI(1),nISTI),
     . (fidatI(2)(1:60),fICSN),(ifnulI(2),nICSN),
     . (fidatI(3)(1:16),fIOS1),(ifnulI(3),nIOS1),
     . (fidatI(4)(1:50),fISI1),(ifnulI(4),nISI1),
     . (fidatI(5)(1:16),fIOS2),(ifnulI(5),nIOS2),
     . (fidatI(6)(1:50),fISI2),(ifnulI(6),nISI2),
     . (fidatI(7)(1:16),fIOS3),(ifnulI(7),nIOS3),
     . (fidatI(8)(1:50),fISI3),(ifnulI(8),nISI3),
     . (fidatI(9)(1:16),fIOS4),(ifnulI(9),nIOS4),
     . (fidatI(10)(1:50),fISI4),(ifnulI(10),nISI4),
     . (fidatI(11)(1:16),fIOS5),(ifnulI(11),nIOS5),
     . (fidatI(12)(1:50),fISI5),(ifnulI(12),nISI5),
     . (fidatI(13)(1:16),fIOS6),(ifnulI(13),nIOS6),
     . (fidatI(14)(1:50),fISI6),(ifnulI(14),nISI6),
     . (fidatI(15)(1:16),fIOS7),(ifnulI(15),nIOS7),
     . (fidatI(16)(1:50),fISI7),(ifnulI(16),nISI7),
     . (fidatI(17)(1:16),fIOS8),(ifnulI(17),nIOS8),
     . (fidatI(18)(1:50),fISI8),(ifnulI(18),nISI8),
     . (fidatI(19)(1:16),fIOS9),(ifnulI(19),nIOS9),
     . (fidatI(20)(1:50),fISI9),(ifnulI(20),nISI9),
     . (fidatI(21)(1:16),fIOS10),(ifnulI(21),nIOS10),
     . (fidatI(22)(1:50),fISI10),(ifnulI(22),nISI10),
     . (fidatI(23)(1:16),fIOS11),(ifnulI(23),nIOS11),
     . (fidatI(24)(1:50),fISI11),(ifnulI(24),nISI11),
     . (fidatI(25)(1:16),fIOS12),(ifnulI(25),nIOS12),
     . (fidatI(26)(1:50),fISI12),(ifnulI(26),nISI12),
     . (fidatI(27)(1:16),fIOS13),(ifnulI(27),nIOS13),
     . (fidatI(28)(1:50),fISI13),(ifnulI(28),nISI13),
     . (fidatI(29)(1:16),fIOS14),(ifnulI(29),nIOS14),
     . (fidatI(30)(1:50),fISI14),(ifnulI(30),nISI14),
     . (fidatI(31)(1:16),fIOS15),(ifnulI(31),nIOS15),
     . (fidatI(32)(1:50),fISI15),(ifnulI(32),nISI15),
     . (mxnocI(1),lISTI),(ifmodI(1),mISTI),
     . (mxnocI(2),lICSN),(ifmodI(2),mICSN),
     . (mxnocI(3),lIOS1),(ifmodI(3),mIOS1),
     . (mxnocI(4),lISI1),(ifmodI(4),mISI1),
     . (mxnocI(5),lIOS2),(ifmodI(5),mIOS2),
     . (mxnocI(6),lISI2),(ifmodI(6),mISI2),
     . (mxnocI(7),lIOS3),(ifmodI(7),mIOS3),
     . (mxnocI(8),lISI3),(ifmodI(8),mISI3),
     . (mxnocI(9),lIOS4),(ifmodI(9),mIOS4),
     . (mxnocI(10),lISI4),(ifmodI(10),mISI4),
     . (mxnocI(11),lIOS5),(ifmodI(11),mIOS5),
     . (mxnocI(12),lISI5),(ifmodI(12),mISI5),
     . (mxnocI(13),lIOS6),(ifmodI(13),mIOS6),
     . (mxnocI(14),lISI6),(ifmodI(14),mISI6),
     . (mxnocI(15),lIOS7),(ifmodI(15),mIOS7),
```

```
     . (mxnocI(16),lISI7),(ifmodI(16),mISI7),
     . (mxnocI(17),lIOS8),(ifmodI(17),mIOS8),
     . (mxnocI(18),lISI8),(ifmodI(18),mISI8),
     . (mxnocI(19),lIOS9),(ifmodI(19),mIOS9),
     . (mxnocI(20),lISI9),(ifmodI(20),mISI9),
     . (mxnocI(21),lIOS10),(ifmodI(21),mIOS10),
     . (mxnocI(22),lISI10),(ifmodI(22),mISI10),
     . (mxnocI(23),lIOS11),(ifmodI(23),mIOS11),
     . (mxnocI(24),lISI11),(ifmodI(24),mISI11),
     . (mxnocI(25),lIOS12),(ifmodI(25),mIOS12),
     . (mxnocI(26),lISI12),(ifmodI(26),mISI12),
     . (mxnocI(27),lIOS13),(ifmodI(27),mIOS13),
     . (mxnocI(28),lISI13),(ifmodI(28),mISI13),
     . (mxnocI(29),lIOS14),(ifmodI(29),mIOS14),
     . (mxnocI(30),lISI14),(ifmodI(30),mISI14),
     . (mxnocI(31),lIOS15),(ifmodI(31),mIOS15),
     . (mxnocI(32),lISI15),(ifmodI(32),mISI15)
C
C    Database table and attribute names for screen fields.
C
      data atrbtI/
     . 'SPECIAL_INSTRUCTIONS','SAMPLE_TRACKING_ID',
     . 'SAMPLE_INFORMATION','CUSTOMER_SAMPLE_NAME',
     . 'SPECIAL_INSTRUCTIONS','ORIGIN_OF_SPECIAL_INSTRUCTION',
     . 'SPECIAL_INSTRUCTIONS','SPECIAL_INSTRUCTION',
     . 'SPECIAL_INSTRUCTIONS','SPECIAL_INSTRUCTION_LINE'/
C
C    Program variable names (value and null indicator) for screen fields.
C
      data varblI/':fISTI:nISTI',':fICSN:nICSN',':fIOS1t:nIOS1',
     . ':fISI1t:nISI1',':siline:ifnsil'/
C
C    Possessor screen's field descriptors and field sizes.
C
      character*80 fieldT(23),fidatT(23)
      character*30 atrbtT(2,9)
      character*14 varblT(9)
      character*1 fitypT(23)
      integer*4 fdlenT(23),fdrowT(23),fdcolT(23),ficolT(23),mxnocT(23),
     . fdrndT(3,23),firndT(3,23), !Field renditions (add/update/search).
     . numfdT/23/   !The number of fields.
      integer*2 ifnulT(23),   !Null indicator (-1 if null, 0 otherwise).
     . ifrqdT(23)/23*0/,   !Points to if-required dependent field.
     . iditoT(23)/23*0/   !Points to allowed field for dittoing.
      logical ifmodT(23)   !Has a field been modified in an Update?
      common /dbdatT/ fieldT,fidatT,fdlenT,fdrowT,fdcolT,ficolT,mxnocT,
     . fdrndT,firndT,numfdT,fitypT,ifnulT,atrbtT,varblT,ifrqdT,ifmodT,
     . iditoT
C
C      Length-specified character fields for equivalences (f for field).
C
      character fTSTI*12,fTCSN*60,fTPSN1*3,fTPN1*25,fTPPN1*12,fTPA1*60,
     . fTPPR1*50,fTDAB1*6,fTDRB1*6,fTPSN2*3,fTPN2*25,fTPPN2*12,fTPA2*60,
     . fTPPR2*50,fTDAB2*6,fTDRB2*6,fTPSN3*3,fTPN3*25,fTPPN3*12,fTPA3*60,
     . fTPPR3*50,fTDAB3*6,fTDRB3*6,
     . fTPN1t*25,fTPPN1t*12,fTPA1t*60,fTPPR1t*50   !Temporary fields.
C
C       Integer field storage (i for integer).
C
      integer*4 iTPSN1,iTDAB1,iTDRB1
```

```
C
C       Nonarrayed maximum field lenghts (1 for length).
C
      integer*4 lTSTI,lTCSN,lTPSN1,lTPN1,lTPPN1,lTPA1,lTPPR1,lTDAB1,
     . lTDRB1,lTPSN2,lTPN2,lTPPN2,lTPA2,lTPPR2,lTDAB2,lTDRB2,lTPSN3,
     . lTPN3,lTPPN3,lTPA3,lTPPR3,lTDAB3,lTDRB3
C
C       Nonarrayed null indicators for data transfers (n for null).
C
      integer*2 nTSTI,nTCSN,nTPSN1,nTPN1,nTPPN1,nTPA1,nTPPR1,nTDAB1,
     . nTDRB1,nTPSN2,nTPN2,nTPPN2,nTPA2,nTPPR2,nTDAB2,nTDRB2,nTPSN3,
     . nTPN3,nTPPN3,nTPA3,nTPPR3,nTDAB3,nTDRB3
C
C       Nonarrayed if-modified indicators (m for modified).
C
      integer*2 mTSTI,mTCSN,mTPSN1,mTPN1,mTPPN1,mTPA1,mTPPR1,mTDAB1,
     . mTDRB1,mTPSN2,mTPN2,mTPPN2,mTPA2,mTPPR2,mTDAB2,mTDRB2,mTPSN3,
     . mTPN3,mTPPN3,mTPA3,mTPPR3,mTDAB3,mTDRB3
C
C       Equivalence statements for data transfers.
C
      equivalence (fidatT(1)(1:12),fTSTI),(ifnulT(1),nTSTI),
     . (fidatT(2)(1:60),fTCSN),(ifnulT(2),nTCSN),
     . (fidatT(3)(1:3),fTPSN1),(ifnulT(3),nTPSN1),
     . (fidatT(4)(1:25),fTPN1),(ifnulT(4),nTPN1),
     . (fidatT(5)(1:12),fTPPN1),(ifnulT(5),nTPPN1),
     . (fidatT(6)(1:60),fTPA1),(ifnulT(6),nTPA1),
     . (fidatT(7)(1:50),fTPPR1),(ifnulT(7),nTPPR1),
     . (fidatT(8)(1:6),fTDAB1),(ifnulT(8),nTDAB1),
     . (fidatT(9)(1:6),fTDRB1),(ifnulT(9),nTDRB1),
     . (fidatT(10)(1:3),fTPSN2),(ifnulT(10),nTPSN2),
     . (fidatT(11)(1:25),fTPN2),(ifnulT(11),nTPN2),
     . (fidatT(12)(1:12),fTPPN2),(ifnulT(12),nTPPN2),
     . (fidatT(13)(1:60),fTPA2),(ifnulT(13),nTPA2),
     . (fidatT(14)(1:50),fTPPR2),(ifnulT(14),nTPPR2),
     . (fidatT(15)(1:6),fTDAB2),(ifnulT(15),nTDAB2),
     . (fidatT(16)(1:6),fTDRB2),(ifnulT(16),nTDRB2),
     . (fidatT(17)(1:3),fTPSN3),(ifnulT(17),nTPSN3),
     . (fidatT(18)(1:25),fTPN3),(ifnulT(18),nTPN3),
     . (fidatT(19)(1:12),fTPPN3),(ifnulT(19),nTPPN3),
     . (fidatT(20)(1:60),fTPA3),(ifnulT(20),nTPA3),
     . (fidatT(21)(1:50),fTPPR3),(ifnulT(21),nTPPR3),
     . (fidatT(22)(1:6),fTDAB3),(ifnulT(22),nTDAB3),
     . (fidatT(23)(1:6),fTDRB3),(ifnulT(23),nTDRB3),
     . (mxnocT(1),lTSTI),(ifmodT(1),mTSTI),
     . (mxnocT(2),lTCSN),(ifmodT(2),mTCSN),
     . (mxnocT(3),lTPSN1),(ifmodT(3),mTPSN1),
     . (mxnocT(4),lTPN1),(ifmodT(4),mTPN1),
     . (mxnocT(5),lTPPN1),(ifmodT(5),mTPPN1),
     . (mxnocT(6),lTPA1),(ifmodT(6),mTPA1),
     . (mxnocT(7),lTPPR1),(ifmodT(7),mTPPR1),
     . (mxnocT(8),lTDAB1),(ifmodT(8),mTDAB1),
     . (mxnocT(9),lTDRB1),(ifmodT(9),mTDRB1),
     . (mxnocT(10),lTPSN2),(ifmodT(10),mTPSN2),
     . (mxnocT(11),lTPN2),(ifmodT(11),mTPN2),
     . (mxnocT(12),lTPPN2),(ifmodT(12),mTPPN2),
     . (mxnocT(13),lTPA2),(ifmodT(13),mTPA2),
     . (mxnocT(14),lTPPR2),(ifmodT(14),mTPPR2),
     . (mxnocT(15),lTDAB2),(ifmodT(15),mTDAB2),
     . (mxnocT(16),lTDRB2),(ifmodT(16),mTDRB2),
```

```
     . (mxnocT(17),lTPSN3),(ifmodT(17),mTPSN3),
     . (mxnocT(18),lTPN3),(ifmodT(18),mTPN3),
     . (mxnocT(19),lTPPN3),(ifmodT(19),mTPPN3),
     . (mxnocT(20),lTPA3),(ifmodT(20),mTPA3),
     . (mxnocT(21),lTPPR3),(ifmodT(21),mTPPR3),
     . (mxnocT(22),lTDAB3),(ifmodT(22),mTDAB3),
     . (mxnocT(23),lTDRB3),(ifmodT(23),mTDRB3)
C
C    Database table and attribute names for screen fields.
C
      data atrbtT/
     . 'TRACKING','SAMPLE_TRACKING_ID',
     . 'SAMPLE_INFORMATION','CUSTOMER_SAMPLE_NAME',
     . 'TRACKING','POSSESSOR_SEQUENCE_NUMBER',
     . 'TRACKING','POSSESSOR_NAME',
     . 'TRACKING','POSSESSOR_PHONE_NUMBER',
     . 'TRACKING','POSSESSOR_ADDRESS',
     . 'TRACKING','POSSESSOR_POSSESSION_REASON',
     . 'TRACKING','DATE_ACCEPTED_BY_POSSESSOR',
     . 'TRACKING','DATE_RELINQUISHED_BY_POSSESSOR'/
C
C    Program variable names (value and null indicator) for screen fields.
C
      data varblT/':fTSTI:nTSTI',':fTCSN:nTCSN',':iTPSN1:nTPSN1',
     . ':fTPN1:nTPN1',':fTPPN1:nTPPN1',':fTPA1:nTPA1',':fTPPR1:nTPPR1',
     . ':iTDAB1:nTDAB1',':iTDRB1:nTDRB1'/
C
C    Possessor name list field descriptors and sizes.
C
      character*80 fieldN(4,15)
      integer*4 fdlenN(4,15),fdrowN(15),fdcolN(15),ficolN(15),
     . numfdN    !The number of fields.
      common /lsdatN/ fieldN,fdlenN,fdrowN,fdcolN,ficolN,numfdN
C
C For the Main screen, . . .
C
C   Set defaults for display field renditions now so they can be modified
C    with each field as necessary.  Set all input fields to blanks.
C
      do i=1,numfdM
         fidatM(i)=blanks
         do j=1,3    !1 for adding, 2 for updating, and 3 for searching.
            fdrndM(j,i)=foptnl
            enddo
         enddo
C
      fieldM(1)='Tracking ID:'
      fdlenM(1)=12
      fdrowM(1)=2
      fdcolM(1)=1
      ficolM(1)=15
      mxnocM(1)=12
      fitypM(1)='l'
C
      fieldM(2)='Date Entered:'
      fdlenM(2)=13
      fdrowM(2)=2
      fdcolM(2)=31
      ficolM(2)=46
      mxnocM(2)=6
```

```
      fitypM(2)='d'
C
      do i=1,2
         do j=1,2    !Fixed when adding and updating.
            fdrndM(j,i)=ffixed
         enddo
      enddo
C
      fieldM(3)='Completed:'
      fdlenM(3)=10
      fdrowM(3)=2
      fdcolM(3)=56
      ficolM(3)=68
      mxnocM(3)=6
      fitypM(3)='d'
C
      fieldM(4)='Customer''s Sample ID:'
      fdlenM(4)=22
      fdrowM(4)=3
      fdcolM(4)=4
      ficolM(4)=27
      mxnocM(4)=12
      fitypM(4)='a'
C
      fieldM(5)='Desired Completion Date:'
      fdlenM(5)=24
      fdrowM(5)=3
      fdcolM(5)=45
      ficolM(5)=71
      mxnocM(5)=6
      fitypM(5)='d'
      do j=1,2    !Required when adding and updating.
         fdrndM(j,5)=freqd
         enddo
C
      fieldM(6)='Project:'
      fdlenM(6)=8
      fdrowM(6)=4
      fdcolM(6)=4
      ficolM(6)=14
      mxnocM(6)=40
      fitypM(6)='a'
C
      fieldM(7)='Charge #:'
      fdlenM(7)=9
      fdrowM(7)=4
      fdcolM(7)=58
      ficolM(7)=69
      mxnocM(7)=9
      fitypM(7)='l'
C
      fieldM(8)='Sample Name:'
      fdlenM(8)=12
      fdrowM(8)=5
      fdcolM(8)=4
      ficolM(8)=18
      mxnocM(8)=60
      fitypM(8)='a'
      iditoM(8)=4
C
```

```
      fieldM(9)='Sample Type:'
      fdlenM(9)=12
      fdrowM(9)=6
      fdcolM(9)=4
      ficolM(9)=18
      mxnocM(9)=10
      fitypM(9)='a'
C
      fieldM(10)='Sample Size:'
      fdlenM(10)=12
      fdrowM(10)=6
      fdcolM(10)=32
      ficolM(10)=46
      mxnocM(10)=5
      fitypM(10)='n'
C
      fieldM(11)='Size Units:'
      fdlenM(11)=11
      fdrowM(11)=6
      fdcolM(11)=55
      ficolM(11)=68
      mxnocM(11)=5
      fitypM(11)='a'
C
      fieldM(12)='Collection Date:'
      fdlenM(12)=16
      fdrowM(12)=7
      fdcolM(12)=4
      ficolM(12)=22
      mxnocM(12)=6
      fitypM(12)='d'
C
      do i=7,12
         do j=1,2   !Required when adding and updating.
            fdrndM(j,i)=freqd
            enddo
         enddo
C
      fieldM(13)='Time:'
      fdlenM(13)=5
      fdrowM(13)=7
      fdcolM(13)=32
      ficolM(13)=39
      mxnocM(13)=5
      fitypM(13)='t'
C
      fieldM(14)='Special Instructions?'
      fdlenM(14)=21
      fdrowM(14)=7
      fdcolM(14)=48
      ficolM(14)=71
      mxnocM(14)=1
      fitypM(14)='?'
C
      fieldM(15)='HP Surveyed?'
      fdlenM(15)=12
      fdrowM(15)=8
      fdcolM(15)=4
      ficolM(15)=18
      mxnocM(15)=1
```

```
      fitypM(15)='?'
      ifrqdM(15)=16
C
      fieldM(16)='If Yes, Sample Activity (mrem/h):'
      fdlenM(16)=33
      fdrowM(16)=8
      fdcolM(16)=23
      ficolM(16)=58
      mxnocM(16)=6
      fitypM(16)='n'
      ifrqdM(16)=15
C
      fieldM(17)='Submitter:'
      fdlenM(17)=10
      fdrowM(17)=9
      fdcolM(17)=1
      ficolM(17)=21
      mxnocM(17)=25
      fitypM(17)='a'
C
      lincnt=7
      do i=18,30,3
      lincnt=lincnt+2
         fieldM(i)='Phone Number:'
         fdlenM(i)=13
         fdrowM(i)=lincnt
         fdcolM(i)=50
         ficolM(i)=65
         mxnocM(i)=12
         fitypM(i)='a'
         ifrqdM(i)=i-1
         if(i.ne.18)iditoM(i)=i-3
C
         fieldM(i+1)='Address:'
         fdlenM(i+1)=8
         fdrowM(i+1)=lincnt+1
         fdcolM(i+1)=4
         ficolM(i+1)=14
         mxnocM(i+1)=60
         fitypM(i+1)='a'
         ifrqdM(i+1)=i-1
         if(i.ne.18)iditoM(i+1)=i-2
         enddo
C
      do i=17,19
         do j=1,2    !Required when adding and updating.
            fdrndM(j,i)=freqd
            enddo
         enddo
C
      fieldM(20)='Technical Contact:'
      fdlenM(20)=18
      fdrowM(20)=11
      fdcolM(20)=1
      ficolM(20)=21
      mxnocM(20)=25
      fitypM(20)='a'
      iditoM(20)=17
C
      fieldM(23)='Send Results To:'
```

```
      fdlenM(23)=16
      fdrowM(23)=13
      fdcolM(23)=1
      ficolM(23)=21
      mxnocM(23)=25
      fitypM(23)='a'
      iditoM(23)=20
C
      fieldM(26)='Sample Pickup By:'
      fdlenM(26)=17
      fdrowM(26)=15
      fdcolM(26)=1
      ficolM(26)=21
      mxnocM(26)=25
      fitypM(26)='a'
      iditoM(26)=23
C
      do i=23,28
         do j=1,2   !Required when adding and updating.
            fdrndM(j,i)=freqd
            enddo
         enddo
C
      fieldM(29)='Current Possessor:'
      fdlenM(29)=18
      fdrowM(29)=17
      fdcolM(29)=1
      ficolM(29)=21
      mxnocM(29)=25
      fitypM(29)='a'
C
      do i=29,31
         fdrndM(1,i)=ffixed   !Fixed when adding and updating.
         fdrndM(2,i)=ffixed
         enddo
C
      fieldM(32)='Chain of Custody Number:'
      fdlenM(32)=24
      fdrowM(32)=19
      fdcolM(32)=1
      ficolM(32)=27
      mxnocM(32)=10
      fitypM(32)='a'
C
      fieldM(33)='Analyses Needed:  Gross Alpha-Beta?'
      fdlenM(33)=35
      fdrowM(33)=20
      fdcolM(33)=1
      ficolM(33)=38
      mxnocM(33)=1
      fitypM(33)='?'
C
      fieldM(34)='Alpha?'
      fdlenM(34)=6
      fdrowM(34)=20
      fdcolM(34)=43
      ficolM(34)=51
      mxnocM(34)=1
      fitypM(34)='?'
C
```

```
      fieldM(35)='Beta?'
      fdlenM(35)=5
      fdrowM(35)=20
      fdcolM(35)=56
      ficolM(35)=63
      mxnocM(35)=1
      fitypM(35)='?'
C
      fieldM(36)='Gamma?'
      fdlenM(36)=6
      fdrowM(36)=20
      fdcolM(36)=68
      ficolM(36)=76
      mxnocM(36)=1
      fitypM(36)='?'
C
C     Set all input field renditions to the bit-wise .or. of the display
C      field rendition with finput.  Note that all flags are required
C      when adding and updating.
C
      do i=1,numfdM
         do j=1,3
            if((j.ne.3).and.(fitypM(i).eq.'?'))fdrndM(j,i)=freqd
            firndM(j,i)=finput.or.fdrndM(j,i)
            enddo
         enddo
C
C For the Alpha screen, . . .
C
C   Set defaults for display field renditions now so they can be modified
C    with each field as necessary.  Set all input fields to blanks.
C
      do i=1,numfdA
         fidatA(i)=blanks
         do j=1,3    !1 for adding, 2 for updating, and 3 for searching.
            fdrndA(j,i)=foptnl
            enddo
         enddo
C
      fieldA(1)='Tracking ID:'
      fdlenA(1)=12
      fdrowA(1)=3
      fdcolA(1)=1
      ficolA(1)=15
      mxnocA(1)=12
      fitypA(1)='1'
C
      fieldA(2)='Sample Name:'
      fdlenA(2)=12
      fdrowA(2)=4
      fdcolA(2)=4
      ficolA(2)=18
      mxnocA(2)=60
      fitypA(2)='a'
C
      do i=1,2
         do j=1,2    !Fixed when adding and updating.
            fdrndA(j,i)=ffixed
            enddo
         enddo
```

```
C
      fieldA(3)='Uranium isotopes?'
      fdlenA(3)=17
      fdrowA(3)=6
      fdcolA(3)=1
      ficolA(3)=20
      mxnocA(3)=1
      fitypA(3)='?'
C
      lincnt=4
      do i=4,34,5
         lincnt=lincnt+2
         fieldA(i)='Needed by:'
         fdlenA(i)=10
         fdrowA(i)=lincnt
         fdcolA(i)=36
         ficolA(i)=48
         mxnocA(i)=6
         fitypA(i)='d'
         ifrqdA(i)=i-1
         if(i.ne.4)iditoA(i)=i-5
C
         fieldA(i+1)='Completed:'
         fdlenA(i+1)=10
         fdrowA(i+1)=lincnt
         fdcolA(i+1)=58
         ficolA(i+1)=70
         mxnocA(i+1)=6
         fitypA(i+1)='d'
         ifrqdA(i+1)=i+2
         if(i.ne.4)iditoA(i+1)=i-4
C
         fieldA(i+2)='Results report citation:'
         fdlenA(i+2)=24
         fdrowA(i+2)=lincnt+1
         fdcolA(i+2)=4
         ficolA(i+2)=30
         mxnocA(i+2)=20
         fitypA(i+2)='a'
         ifrqdA(i+2)=i+1
         if(i.ne.4)iditoA(i+2)=i-3
C
         fieldA(i+3)='Special Instructions?'
         fdlenA(i+3)=21
         fdrowA(i+3)=lincnt+1
         fdcolA(i+3)=54
         ficolA(i+3)=77
         mxnocA(i+3)=1
         fitypA(i+3)='?'
         ifrqdA(i+3)=i-1
         if(i.ne.4)iditoA(i+3)=i-2
         enddo
C
      fieldA(8)='Thorium isotopes?'
      fdlenA(8)=17
      fdrowA(8)=8
      fdcolA(8)=1
      ficolA(8)=20
      mxnocA(8)=1
      fitypA(8)='?'
```

```
C
      fieldA(13)='Plutonium isotopes?'
      fdlenA(13)=19
      fdrowA(13)=10
      fdcolA(13)=1
      ficolA(13)=22
      mxnocA(13)=1
      fitypA(13)='?'
C
      fieldA(18)='Am-241 separate from Pu-238?'
      fdlenA(18)=28
      fdrowA(18)=12
      fdcolA(18)=1
      ficolA(18)=31
      mxnocA(18)=1
      fitypA(18)='?'
C
      fieldA(23)='Am-241 combined with Pu-238?'
      fdlenA(23)=28
      fdrowA(23)=14
      fdcolA(23)=1
      ficolA(23)=31
      mxnocA(23)=1
      fitypA(23)='?'
C
      fieldA(28)='Total Spectrometric Alpha?'
      fdlenA(28)=26
      fdrowA(28)=16
      fdcolA(28)=1
      ficolA(28)=29
      mxnocA(28)=1
      fitypA(28)='?'
C
      fieldA(33)='Other?'
      fdlenA(33)=6
      fdrowA(33)=18
      fdcolA(33)=1
      ficolA(33)=9
      mxnocA(33)=1
      fitypA(33)='?'
C
C     Set all input field renditions to the bit-wise .or. of the display
C      field rendition with finput.  Note that some flags are required
C      when adding and updating.
C
      do i=1,numfdA
         do j=1,3
            if((j.ne.3).and.(fitypA(i).eq.'?').and.
     .        (fieldA(i).ne.'Special Instructions?'))fdrndA(j,i)=freqd
            firndA(j,i)=finput.or.fdrndA(j,i)
            enddo
         enddo
C
C For the Beta screen, . . .
C
C   Set defaults for display field renditions now so they can be modified
C    with each field as necessary.  Set all input fields to blanks.
C
      do i=1,numfdB
         fidatB(i)=blanks
```

```
         do j=1,3    !1 for adding, 2 for updating, and 3 for searching.
            fdrndB(j,i)=foptnl
            enddo
         enddo
C
      fieldB(1)='Tracking ID:'
      fdlenB(1)=12
      fdrowB(1)=2
      fdcolB(1)=1
      ficolB(1)=15
      mxnocB(1)=12
      fitypB(1)='1'
C
      fieldB(2)='Sample Name:'
      fdlenB(2)=12
      fdrowB(2)=3
      fdcolB(2)=4
      ficolB(2)=18
      mxnocB(2)=60
      fitypB(2)='a'
C
      do i=1,2
         do j=1,2    !Fixed when adding and updating.
            fdrndB(j,i)=ffixed
            enddo
         enddo
C
      fieldB(3)='Strontium-90?'
      fdlenB(3)=13
      fdrowB(3)=5
      fdcolB(3)=1
      ficolB(3)=16
      mxnocB(3)=1
      fitypB(3)='?'
C
      lincnt=2
      do i=4,28,8
         lincnt=lincnt+3
         fieldB(i)='Length of count:'
         fdlenB(i)=16
         fdrowB(i)=lincnt
         fdcolB(i)=29
         ficolB(i)=47
         mxnocB(i)=4
         fitypB(i)='n'
         ifrqdB(i)=i+1
         if(i.ne.4)iditoB(i)=i-8
C
         fieldB(i+1)='min or hr?'
         fdlenB(i+1)=10
         fdrowB(i+1)=lincnt
         fdcolB(i+1)=55
         ficolB(i+1)=67
         mxnocB(i+1)=1
         fitypB(i+1)='u'
         ifrqdB(i+1)=i
         if(i.ne.4)iditoB(i+1)=i-7
C
         fieldB(i+2)='Results needed by:  date:'
         fdlenB(i+2)=25
```

```
          fdrowB(i+2)=lincnt+1
          fdcolB(i+2)=4
          ficolB(i+2)=31
          mxnocB(i+2)=6
          fitypB(i+2)='d'
          ifrqdB(i+2)=i-1
          if(i.ne.4)iditoB(i+2)=i-6
C
          fieldB(i+3)='time:'
          fdlenB(i+3)=5
          fdrowB(i+3)=lincnt+1
          fdcolB(i+3)=41
          ficolB(i+3)=48
          mxnocB(i+3)=5
          fitypB(i+3)='t'
          if(i.ne.4)iditoB(i+3)=i-5
C
          fieldB(i+4)='Completed:'
          fdlenB(i+4)=10
          fdrowB(i+4)=lincnt+1
          fdcolB(i+4)=57
          ficolB(i+4)=69
          mxnocB(i+4)=6
          fitypB(i+4)='d'
          ifrqdB(i+4)=i+5
          if(i.ne.4)iditoB(i+4)=i-4
C
          fieldB(i+5)='Results report citation:'
          fdlenB(i+5)=24
          fdrowB(i+5)=lincnt+2
          fdcolB(i+5)=4
          ficolB(i+5)=30
          mxnocB(i+5)=20
          fitypB(i+5)='a'
          ifrqdB(i+5)=i+4
          if(i.ne.4)iditoB(i+5)=i-3
C
          fieldB(i+6)='Special Instructions?'
          fdlenB(i+6)=21
          fdrowB(i+6)=lincnt+2
          fdcolB(i+6)=54
          ficolB(i+6)=77
          mxnocB(i+6)=1
          fitypB(i+6)='?'
          ifrqdB(i+6)=i-1
          if(i.ne.4)iditoB(i+6)=i-2
          enddo
C
        fieldB(11)='Strontium-89 and -90?'
        fdlenB(11)=21
        fdrowB(11)=8
        fdcolB(11)=1
        ficolB(11)=24
        mxnocB(11)=1
        fitypB(11)='?'
C
        fieldB(19)='Total Strontium?'
        fdlenB(19)=16
        fdrowB(19)=11
        fdcolB(19)=1
```

```
      ficolB(19)=19
      mxnocB(19)=1
      fitypB(19)='?'
C
      fieldB(27)='Tritium?'
      fdlenB(27)=8
      fdrowB(27)=14
      fdcolB(27)=1
      ficolB(27)=11
      mxnocB(27)=1
      fitypB(27)='?'
C
      fieldB(35)='Other?'
      fdlenB(35)=6
      fdrowB(35)=17
      fdcolB(35)=1
      ficolB(35)=9
      mxnocB(35)=1
      fitypB(35)='?'
C
      fieldB(36)='Length of count:'
      fdlenB(36)=16
      fdrowB(36)=17
      fdcolB(36)=29
      ficolB(36)=47
      mxnocB(36)=4
      fitypB(36)='n'
      iditoB(36)=28
C
      fieldB(37)='min or hr?'
      fdlenB(37)=10
      fdrowB(37)=17
      fdcolB(37)=55
      ficolB(37)=67
      mxnocB(37)=1
      fitypB(37)='u'
      ifrqdB(37)=36
      iditoB(37)=29
C
      fieldB(38)='Nickel-63?'
      fdlenB(38)=10
      fdrowB(38)=18
      fdcolB(38)=4
      ficolB(38)=16
      mxnocB(38)=1
      fitypB(38)='?'
      ifrqdB(38)=35
C
      fieldB(39)='Iron-55?'
      fdlenB(39)=8
      fdrowB(39)=18
      fdcolB(39)=21
      ficolB(39)=31
      mxnocB(39)=1
      fitypB(39)='?'
      ifrqdB(39)=35
C
      fieldB(40)='Sulfur-35?'
      fdlenB(40)=10
      fdrowB(40)=18
```

```
      fdcolB(40)=36
      ficolB(40)=48
      mxnocB(40)=1
      fitypB(40)='?'
      ifrqdB(40)=35
C
      fieldB(41)='Plutonium-241?'
      fdlenB(41)=14
      fdrowB(41)=18
      fdcolB(41)=53
      ficolB(41)=69
      mxnocB(41)=1
      fitypB(41)='?'
      ifrqdB(41)=35
C
      fieldB(42)='Results needed by:  date:'
      fdlenB(42)=25
      fdrowB(42)=19
      fdcolB(42)=4
      ficolB(42)=31
      mxnocB(42)=6
      fitypB(42)='d'
      ifrqdB(42)=35
      iditoB(42)=30
C
      fieldB(43)='time:'
      fdlenB(43)=5
      fdrowB(43)=19
      fdcolB(43)=41
      ficolB(43)=48
      mxnocB(43)=5
      fitypB(43)='t'
      iditoB(43)=31
C
      fieldB(44)='Completed:'
      fdlenB(44)=10
      fdrowB(44)=19
      fdcolB(44)=57
      ficolB(44)=69
      mxnocB(44)=6
      fitypB(44)='d'
      ifrqdB(44)=45
      iditoB(44)=32
C
      fieldB(45)='Results report citation:'
      fdlenB(45)=24
      fdrowB(45)=20
      fdcolB(45)=4
      ficolB(45)=30
      mxnocB(45)=20
      fitypB(45)='a'
      ifrqdB(45)=44
      iditoB(45)=33
C
      fieldB(46)='Special Instructions?'
      fdlenB(46)=21
      fdrowB(46)=20
      fdcolB(46)=54
      ficolB(46)=77
      mxnocB(46)=1
```

```
      fitypB(46)='?'
      ifrqdB(46)=35
      iditoB(46)=34
C
C     Set all input field renditions to the bit-wise .or. of the display
C      field rendition with finput.  Note that some flags are required
C      when adding and updating.
C
      do i=1,numfdB
         do j=1,3
            if((j.ne.3).and.(fitypB(i).eq.'?').and.(i.le.35).and.
     .        (fieldB(i).ne.'Special Instructions?'))fdrndB(j,i)=freqd
            firndB(j,i)=finput.or.fdrndB(j,i)
            enddo
         enddo
C
C For the Gamma screen, . . .
C
C   Set defaults for display field renditions now so they can be modified
C    with each field as necessary.  Set all input fields to blanks.
C
      do i=1,numfdG
         fidatG(i)=blanks
         do j=1,3    !1 for adding, 2 for updating, and 3 for searching.
            fdrndG(j,i)=foptnl
            enddo
         enddo
C
      fieldG(1)='Tracking ID:'
      fdlenG(1)=12
      fdrowG(1)=2
      fdcolG(1)=1
      ficolG(1)=15
      mxnocG(1)=12
      fitypG(1)='1'
C
      fieldG(2)='Sample Name:'
      fdlenG(2)=12
      fdrowG(2)=3
      fdcolG(2)=4
      ficolG(2)=18
      mxnocG(2)=60
      fitypG(2)='a'
C
      do i=1,2
         do j=1,2    !Fixed when adding and updating.
            fdrndG(j,i)=ffixed
            enddo
         enddo
C
      fieldG(3)='Screening/Shipping Count?'
      fdlenG(3)=25
      fdrowG(3)=6
      fdcolG(3)=1
      ficolG(3)=28
      mxnocG(3)=1
      fitypG(3)='?'
C
      lincnt=1
      do i=4,26,11
```

```
      lincnt=lincnt+5
      fieldG(i)='Length of count:'
      fdlenG(i)=16
      fdrowG(i)=lincnt
      fdcolG(i)=37
      ficolG(i)=55
      mxnocG(i)=4
      fitypG(i)='n'
      if(i.ne.4)iditoG(i)=i-11
C
      fieldG(i+1)='min or hr?'
      fdlenG(i+1)=10
      fdrowG(i+1)=lincnt
      fdcolG(i+1)=63
      ficolG(i+1)=75
      mxnocG(i+1)=1
      fitypG(i+1)='u'
      ifrqdG(i+1)=i
      if(i.ne.4)iditoG(i+1)=i-10
C
      fieldG(i+2)='Results needed by:  date:'
      fdlenG(i+2)=25
      fdrowG(i+2)=lincnt+1
      fdcolG(i+2)=4
      ficolG(i+2)=31
      mxnocG(i+2)=6
      fitypG(i+2)='d'
      ifrqdG(i+2)=i-1
      if(i.ne.4)iditoG(i+2)=i-9
C
      fieldG(i+3)='time:'
      fdlenG(i+3)=5
      fdrowG(i+3)=lincnt+1
      fdcolG(i+3)=41
      ficolG(i+3)=48
      mxnocG(i+3)=5
      fitypG(i+3)='t'
      if(i.ne.4)iditoG(i+3)=i-8
C
      fieldG(i+4)='Completed:'
      fdlenG(i+4)=10
      fdrowG(i+4)=lincnt+1
      fdcolG(i+4)=57
      ficolG(i+4)=69
      mxnocG(i+4)=6
      fitypG(i+4)='d'
      ifrqdG(i+4)=i+8
      if(i.ne.4)iditoG(i+4)=i-7
C
      fieldG(i+5)='Date counted:'
      fdlenG(i+5)=13
      fdrowG(i+5)=lincnt+2
      fdcolG(i+5)=4
      ficolG(i+5)=19
      mxnocG(i+5)=6
      fitypG(i+5)='d'
      ifrqdG(i+5)=i+4
C
      fieldG(i+6)='RML Spectral ID:'
      fdlenG(i+6)=16
```

```
            fdrowG(i+6)=lincnt+2
            fdcolG(i+6)=29
            ficolG(i+6)=47
            mxnocG(i+6)=14
            fitypG(i+6)='1'
            ifrqdG(i+6)=i+5
C
            fieldG(i+7)='Recount?'
            fdlenG(i+7)=8
            fdrowG(i+7)=lincnt+2
            fdcolG(i+7)=65
            ficolG(i+7)=75
            mxnocG(i+7)=1
            fitypG(i+7)='?'
            ifrqdG(i+7)=i+5
C
            fieldG(i+8)='Results report citation:'
            fdlenG(i+8)=24
            fdrowG(i+8)=lincnt+3
            fdcolG(i+8)=4
            ficolG(i+8)=30
            mxnocG(i+8)=20.
            fitypG(i+8)='a'
            ifrqdG(i+8)=i+4
            if(i.ne.4)iditoG(i+8)=i-3
C
            fieldG(i+9)='Special Instructions?'
            fdlenG(i+9)=21
            fdrowG(i+9)=lincnt+3
            fdcolG(i+9)=54
            ficolG(i+9)=77
            mxnocG(i+9)=1
            fitypG(i+9)='?'
            ifrqdG(i+9)=i-1
            if(i.ne.4)iditoG(i+9)=i-2
            enddo
C
         fieldG(14)='Full Isotopic Gamma Analysis?'
         fdlenG(14)=29
         fdrowG(14)=11
         fdcolG(14)=1
         ficolG(14)=32
         mxnocG(14)=1
         fitypG(14)='?'
C
         fieldG(25)='Other Gamma Analysis?'
         fdlenG(25)=21
         fdrowG(25)=16
         fdcolG(25)=1
         ficolG(25)=24
         mxnocG(25)=1
         fitypG(25)='?'
C
C     Set all input field renditions to the bit-wise .or. of the display
C      field rendition with finput.  Note that some flags are required
C      when adding and updating.
C
         do i=1,numfdG
            do j=1,3
               if((j.ne.3).and.(fitypG(i).eq.'?').and.
```

```
     .        (fieldG(i).ne.'Recount?').and.
     .        (fieldG(i).ne.'Special Instructions?'))fdrndG(j,i)=freqd
              firndG(j,i)=finput.or.fdrndG(j,i)
              enddo
           enddo
C
C For the Gross Alpha-Beta screen, . . .
C
C   Set defaults for display field renditions now so they can be modified
C    with each field as necessary.  Set all input fields to blanks.
C
      do i=1,numfdC
         fidatC(i)=blanks
         do j=1,3   !1 for adding, 2 for updating, and 3 for searching.
            fdrndC(j,i)=foptnl
            enddo
         enddo
C
      fieldC(1)='Tracking ID:'
      fdlenC(1)=12
      fdrowC(1)=5
      fdcolC(1)=1
      ficolC(1)=15
      mxnocC(1)=12
      fitypC(1)='1'
C
      fieldC(2)='Sample Name:'
      fdlenC(2)=12
      fdrowC(2)=6
      fdcolC(2)=4
      ficolC(2)=18
      mxnocC(2)=60
      fitypC(2)='a'
C
      do i=1,2
         do j=1,2   !Fixed when adding and updating.
            fdrndC(j,i)=ffixed
            enddo
         enddo
C
      fieldC(3)='Gross Alpha-Beta for Air Filters?'
      fdlenC(3)=33
      fdrowC(3)=8
      fdcolC(3)=1
      ficolC(3)=36
      mxnocC(3)=1
      fitypC(3)='?'
C
      lincnt=4
      do i=4,12,8
         lincnt=lincnt+5
         fieldC(i)='Length of count:'
         fdlenC(i)=16
         fdrowC(i)=lincnt
         fdcolC(i)=4
         ficolC(i)=22
         mxnocC(i)=4
         fitypC(i)='n'
         if(i.ne.4)iditoC(i)=i-8
C
```

```
      fieldC(i+1)='min or hr?'
      fdlenC(i+1)=10
      fdrowC(i+1)=lincnt
      fdcolC(i+1)=30
      ficolC(i+1)=42
      mxnocC(i+1)=1
      fitypC(i+1)='u'
      ifrqdC(i+1)=i
      if(i.ne.4)iditoC(i+1)=i-7
C
      fieldC(i+2)='Results needed by:  date:'
      fdlenC(i+2)=25
      fdrowC(i+2)=lincnt+1
      fdcolC(i+2)=4
      ficolC(i+2)=31
      mxnocC(i+2)=6
      fitypC(i+2)='d'
      ifrqdC(i+2)=i-1
      if(i.ne.4)iditoC(i+2)=i-6
C
      fieldC(i+3)='time:'
      fdlenC(i+3)=5
      fdrowC(i+3)=lincnt+1
      fdcolC(i+3)=41
      ficolC(i+3)=48
      mxnocC(i+3)=5
      fitypC(i+3)='t'
      if(i.ne.4)iditoC(i+3)=i-5
C
      fieldC(i+4)='Completed:'
      fdlenC(i+4)=10
      fdrowC(i+4)=lincnt+1
      fdcolC(i+4)=57
      ficolC(i+4)=69
      mxnocC(i+4)=6
      fitypC(i+4)='d'
      ifrqdC(i+4)=i+5
      if(i.ne.4)iditoC(i+4)=i-4
C
      fieldC(i+5)='Results report citation:'
      fdlenC(i+5)=24
      fdrowC(i+5)=lincnt+2
      fdcolC(i+5)=4
      ficolC(i+5)=30
      mxnocC(i+5)=20
      fitypC(i+5)='a'
      ifrqdC(i+5)=i+4
      if(i.ne.4)iditoC(i+5)=i-3
C
      fieldC(i+6)='Special Instructions?'
      fdlenC(i+6)=21
      fdrowC(i+6)=lincnt+2
      fdcolC(i+6)=54
      ficolC(i+6)=77
      mxnocC(i+6)=1
      fitypC(i+6)='?'
      ifrqdC(i+6)=i-1
      if(i.ne.4)iditoC(i+6)=i-2
      enddo
C
```

```
      fieldC(11)='Gross Alpha-Beta for Other Samples?'
      fdlenC(11)=35
      fdrowC(11)=13
      fdcolC(11)=1
      ficolC(11)=38
      mxnocC(11)=1
      fitypC(11)='?'
C
C    Set all input field renditions to the bit-wise .or. of the display
C     field rendition with finput.  Note that some flags are required
C     when adding and updating.
C
      do i=1,numfdC
         do j=1,3
            if((j.ne.3).and.(fitypC(i).eq.'?').and.
     .          (fieldC(i).ne.'Special Instructions?'))fdrndC(j,i)=freqd
            firndC(j,i)=finput.or.fdrndC(j,i)
            enddo
         enddo
C
C For the Special Instruction screen, . . .
C
C   Set defaults for display field renditions now so they can be modified
C    with each field as necessary.  Set all input fields to blanks.
C
      do i=1,numfdI
         fidatI(i)=blanks
         do j=1,3    !1 for adding, 2 for updating, and 3 for searching.
            fdrndI(j,i)=foptnl
            enddo
         enddo
C
      fieldI(1)='Tracking ID:'
      fdlenI(1)=12
      fdrowI(1)=2
      fdcolI(1)=1
      ficolI(1)=15
      mxnocI(1)=12
      fitypI(1)='l'
C
      fieldI(2)='Sample Name:'
      fdlenI(2)=12
      fdrowI(2)=3
      fdcolI(2)=4
      ficolI(2)=18
      mxnocI(2)=60
      fitypI(2)='a'
C
      do i=1,2
         do j=1,2    !Fixed when adding and updating.
            fdrndI(j,i)=ffixed
            enddo
         enddo
C
      do i=3,31,2
         fieldI(i)='From'
         fdlenI(i)=4
         fdrowI(i)=3+(i+1)/2
         fdcolI(i)=4
         ficolI(i)=9
```

```
            mxnocI(i)=16
            fitypI(i)='a'
            fdrndI(1,i)=ffixed    !Fixed when adding.
            fdrndI(2,i)=ffixed    !Fixed when updating.
C
            fieldI(i+1)=':'
            fdlenI(i+1)=1
            fdrowI(i+1)=3+(i+1)/2
            fdcolI(i+1)=25
            ficolI(i+1)=28
            mxnocI(i+1)=50
            fitypI(i+1)='a'
            enddo
C
C     Set all input field renditions to the bit-wise .or. of the display
C      field rendition with finput.
C
      do i=1,numfdI
         do j=1,3
            firndI(j,i)=finput.or.fdrndI(j,i)
            enddo
         enddo
C
C For the Possessor screen, . . .
C
C   Set defaults for display field renditions now so they can be modified
C    with each field as necessary.  Set all input fields to blanks.
C
      do i=1,numfdT
         fidatT(i)=blanks
         do j=1,3    !1 for adding, 2 for updating, and 3 for searching.
            fdrndT(j,i)=foptn1
            enddo
         enddo
C
      fieldT(1)='Tracking ID:'
      fdlenT(1)=12
      fdrowT(1)=3
      fdcolT(1)=1
      ficolT(1)=15
      mxnocT(1)=12
      fitypT(1)='1'
C
      fieldT(2)='Sample Name:'
      fdlenT(2)=12
      fdrowT(2)=4
      fdcolT(2)=4
      ficolT(2)=18
      mxnocT(2)=60
      fitypT(2)='a'
C
      do i=1,2
         do j=1,2    !Fixed when adding and updating.
            fdrndT(j,i)=ffixed
            enddo
         enddo
C
      lincnt=1
      do i=3,17,7
         lincnt=lincnt+5
```

```
      fieldT(i)='Possessor #'
      fdlenT(i)=11
      fdrowT(i)=lincnt
      fdcolT(i)=1
      ficolT(i)=13
      mxnocT(i)=3
      fitypT(i)='n'
      fdrndT(1,i)=ffixed   !Fixed when adding and
      fdrndT(2,i)=ffixed   !updating.
C
      fieldT(i+1)=':'
      fdlenT(i+1)=1
      fdrowT(i+1)=lincnt
      fdcolT(i+1)=16
      ficolT(i+1)=22
      mxnocT(i+1)=25
      fitypT(i+1)='a'
      if(i.eq.3)then
            fdrndT(1,(i+1))=ffixed   !Fixed when adding and
            fdrndT(2,(i+1))=ffixed   !updating.
         else
            ifrqdT(i+1)=i-1
         endif
C
      fieldT(i+2)='Phone Number:'
      fdlenT(i+2)=13
      fdrowT(i+2)=lincnt
      fdcolT(i+2)=51
      ficolT(i+2)=66
      mxnocT(i+2)=12
      fitypT(i+2)='a'
      if(i.eq.3)then
            fdrndT(1,(i+2))=ffixed   !Fixed when adding and
            fdrndT(2,(i+2))=ffixed   !updating.
         else
            ifrqdT(i+2)=i+1
         endif
C
      fieldT(i+3)='Address:'
      fdlenT(i+3)=8
      fdrowT(i+3)=lincnt+1
      fdcolT(i+3)=4
      ficolT(i+3)=14
      mxnocT(i+3)=60
      fitypT(i+3)='a'
      if(i.eq.3)then
            fdrndT(1,(i+3))=ffixed   !Fixed when adding and
            fdrndT(2,(i+3))=ffixed   !updating.
         else
            ifrqdT(i+3)=i+1
         endif
C
      fieldT(i+4)='Reason for possession:'
      fdlenT(i+4)=22
      fdrowT(i+4)=lincnt+2
      fdcolT(i+4)=4
      ficolT(i+4)=28
      mxnocT(i+4)=50
      fitypT(i+4)='a'
      if(i.eq.3)then
```

```
            fdrndT(1,(i+4))=ffixed    !Fixed when adding and
            fdrndT(2,(i+4))=ffixed    !updating.
         else
            ifrqdT(i+4)=i+1
         endif
C
      fieldT(i+5)='Date sample accepted:'
      fdlenT(i+5)=21
      fdrowT(i+5)=lincnt+3
      fdcolT(i+5)=4
      ficolT(i+5)=27
      mxnocT(i+5)=6
      fitypT(i+5)='d'
      if(i.eq.3)then
            fdrndT(1,(i+5))=ffixed    !Fixed when adding and
            fdrndT(2,(i+5))=ffixed    !updating.
         else
            ifrqdT(i+5)=i+1
         endif
      if(i.ne.3)iditoT(i+5)=i-1    !To last date sample relinquished.
C
      fieldT(i+6)='Date sample relinquished:'
      fdlenT(i+6)=25
      fdrowT(i+6)=lincnt+3
      fdcolT(i+6)=37
      ficolT(i+6)=64
      mxnocT(i+6)=6
      fitypT(i+6)='d'
      enddo
C
C     Set all input field renditions to the bit-wise .or. of the display
C      field rendition with finput.
C
      do i=1,numfdT
         do j=1,3
            firndT(j,i)=finput.or.fdrndT(j,i)
            enddo
         enddo
C
C     Possessor name list data (set up to ease adding and removing
C      names - note that the current limit is fifteen names).
C
      numfdN=1
      fieldN(1,numfdN)='J. L. DOHERTY (JODIE)'
      fdlenN(1,numfdN)=21
      fieldN(2,numfdN)='6-6573'
      fdlenN(2,numfdN)=6
      fieldN(3,numfdN)='RML, MS 7111; TRA-604, ROOM 123'
      fdlenN(3,numfdN)=31
      fieldN(4,numfdN)='GAMMA SPECTRUM ACQUISITION'
      fdlenN(4,numfdN)=26
C
      numfdN=numfdN+1
      fieldN(1,numfdN)='T. J. HANEY (TOM)'
      fdlenN(1,numfdN)=17
      fieldN(2,numfdN)='6-4158'
      fdlenN(2,numfdN)=6
      fieldN(3,numfdN)='RML, MS 7111; TRA-604, ROOM 126'
      fdlenN(3,numfdN)=31
      fieldN(4,numfdN)='SAMPLE ROUTING COORDINATION'
```

```
      fdlenN(4,numfdN)=27
C
      numfdN=numfdN+1
      fieldN(1,numfdN)='R. K. MURRAY (RON)'
      fdlenN(1,numfdN)=18
      fieldN(2,numfdN)='6-4182'
      fdlenN(2,numfdN)=6
      fieldN(3,numfdN)='RML, MS 7111; TRA-604, ROOM 123'
      fdlenN(3,numfdN)=31
      fieldN(4,numfdN)='GAMMA SPECTRUM ACQUISITION'
      fdlenN(4,numfdN)=26
C
      numfdN=numfdN+1
      fieldN(1,numfdN)='C. L. ROWSELL (CAL)'
      fdlenN(1,numfdN)=19
      fieldN(2,numfdN)='6-4182'
      fdlenN(2,numfdN)=6
      fieldN(3,numfdN)='RML, MS 7111; TRA-604, ROOM 123'
      fdlenN(3,numfdN)=31
      fieldN(4,numfdN)='GROSS ALPHA-BETA COUNTING'
      fdlenN(4,numfdN)=25
C
      numfdN=numfdN+1
      fieldN(1,numfdN)='C. W. SILL (CLAUDE)'
      fdlenN(1,numfdN)=19
      fieldN(2,numfdN)='6-0605'
      fdlenN(2,numfdN)=6
      fieldN(3,numfdN)='RADIOCHEMISTRY, MS 7111; TRA-604, ROOM 111'
      fdlenN(3,numfdN)=42
      fieldN(4,numfdN)='ALPHA ANALYSES'
      fdlenN(4,numfdN)=14
C
      numfdN=numfdN+1
      fieldN(1,numfdN)='D. S. SILL (DAVE)'
      fdlenN(1,numfdN)=17
      fieldN(2,numfdN)='6-8031'
      fdlenN(2,numfdN)=6
      fieldN(3,numfdN)='RADIOCHEMISTRY, MS 7111; TRA-604, ROOM 110'
      fdlenN(3,numfdN)=42
      fieldN(4,numfdN)='ALPHA ANALYSES'
      fdlenN(4,numfdN)=14
C
      numfdN=numfdN+1
      fieldN(1,numfdN)='L. D. SMITH (LARRY)'
      fdlenN(1,numfdN)=19
      fieldN(2,numfdN)='6-4405'
      fdlenN(2,numfdN)=6
      fieldN(3,numfdN)='RML, MS 7111; TRA-604, ROOM 123'
      fdlenN(3,numfdN)=31
      fieldN(4,numfdN)='GAMMA SPECTRUM ACQUISITION'
      fdlenN(4,numfdN)=26
C
      numfdN=numfdN+1
      fieldN(1,numfdN)='G. K. TAYLOR (GENE)'
      fdlenN(1,numfdN)=19
      fieldN(2,numfdN)='6-4041'
      fdlenN(2,numfdN)=6
      fieldN(3,numfdN)='RML, MS 7111; TRA-604, ROOM 123'
      fdlenN(3,numfdN)=31
      fieldN(4,numfdN)='GAMMA SPECTRUM ACQUISITION'
```

```
      fdlenN(4,numfdN)=26
C
      numfdN=numfdN+1
      fieldN(1,numfdN)='L. A. WEINRICH (LOU)'
      fdlenN(1,numfdN)=20
      fieldN(2,numfdN)='6-4404'
      fdlenN(2,numfdN)=6
      fieldN(3,numfdN)='RADIOCHEMISTRY, MS 7111; TRA-661, ROOM 129/130'
      fdlenN(3,numfdN)=46
      fieldN(4,numfdN)='BETA ANALYSES'
      fdlenN(4,numfdN)=13
C
      numfdN=numfdN+1
      fieldN(1,numfdN)='R. P. WELLS (RICH)'
      fdlenN(1,numfdN)=18
      fieldN(2,numfdN)='6-7870'
      fdlenN(2,numfdN)=6
      fieldN(3,numfdN)='RADIOCHEMISTRY, MS 7111; TRA-661, ROOM 129/130'
      fdlenN(3,numfdN)=46
      fieldN(4,numfdN)='BETA ANALYSES'
      fdlenN(4,numfdN)=13
C
      do i=1,numfdN
         fdrowN(i)=i+1
         fdcolN(i)=2
         ficolN(i)=25
         enddo
```

F. SCREEN TEMPLATES SUBMITTED TO BUILD_SCREEN

F.1 Screen Template for the Main (M) Screen

```
 12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678
 1
 2Tracking ID: _____ Date Entered: _____ Completed: _____
 3   Customer's Sample ID: _____ Desired Completion Date: _____
 4   Project: _____ Charge #: _____
 5   Sample Name: _____
 6   Sample Type: _____ Sample Size: _____ Size Units: _____
 7   Collection Date: _____ Time: _____ Special Instructions? _
 8   HP Surveyed? _ If Yes, Sample Activity (mrem/h): _____
 9Submitter: _____ Phone Number: _____
10   Address: _____
11Technical Contact: _____ Phone Number: _____
12   Address: _____
13Send Results To: _____ Phone Number: _____
14   Address: _____
15Sample Pickup By: _____ Phone Number: _____
16   Address: _____
17Current Possessor: _____ Phone Number: _____
18   Address: _____
19Chain of Custody Number: _____
20Analyses Needed: Gross Alpha-Beta? _ Alpha? _ Beta? _ Gamma? _
```

F.2 Screen Template for the Possessors (T for tracking) Screen

```
 12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678
 1
 2
 3Tracking ID: _____
 4   Sample Name: _____
 5
 6Possessor # ___: _____ Phone Number: _____
 7   Address: _____
 8   Reason for possession: _____
 9   Date sample accepted: _____ Date sample relinquished: _____
10
11Possessor # ___: _____ Phone Number: _____
12   Address: _____
13   Reason for possession: _____
14   Date sample accepted: _____ Date sample relinquished: _____
15
16Possessor # ___: _____ Phone Number: _____
17   Address: _____
18   Reason for possession: _____
19   Date sample accepted: _____ Date sample relinquished: _____
20
```

F.3 Screen Template for the Alpha Analyses (A) Screen

```
 12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678
 1
 2
 3Tracking ID: _____
 4   Sample Name: _____________________________________________________________
 5
 6Uranium isotopes? _              Needed by: _____    Completed: _____
 7   Results report citation: _____________________    Special Instructions? _
 8Thorium isotopes? _              Needed by: _____    Completed: _____
 9   Results report citation: _____________________    Special Instructions? _
10Plutonium isotopes? _            Needed by: _____    Completed: _____
11   Results report citation: _____________________    Special Instructions? _
12Am-241 separate from Pu-238? _   Needed by: _____    Completed: _____
13   Results report citation: _____________________    Special Instructions? _
14Am-241 combined with Pu-238? _   Needed by: _____    Completed: _____
15   Results report citation: _____________________    Special Instructions? _
16Total Spectrometric Alpha? _     Needed by: _____    Completed: _____
17   Results report citation: _____________________    Special Instructions? _
18Other? _                         Needed by: _____    Completed: _____
19   Results report citation: _____________________    Special Instructions? _
20
```

F.4 Screen Template for the Beta Analyses (B) Screen

```
 12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678
 1
 2Tracking ID: _____
 3   Sample Name: _____________________________________________________________
 4
 5Strontium-90? _              Length of count: ____   min or hr? _
 6   Results needed by: date: _____   time: _____   Completed: _____
 7   Results report citation: _____________________   Special Instructions? _
 8Strontium-89 and -90? _      Length of count: ____   min or hr? _
 9   Results needed by: date: _____   time: _____   Completed: _____
10   Results report citation: _____________________   Special Instructions? _
11Total Strontium? _          Length of count: ____   min or hr? _
12   Results needed by: date: _____   time: _____   Completed: _____
13   Results report citation: _____________________   Special Instructions? _
14Tritium? _                  Length of count: ____   min or hr? _
15   Results needed by: date: _____   time: _____   Completed: _____
16   Results report citation: _____________________   Special Instructions? _
17Other? _                    Length of count: ____   min or hr? _
18   Nickel-63? _    Iron-55? _    Sulfur-35? _    Plutonium-241? _
19   Results needed by: date: _____   time: _____   Completed: _____
20   Results report citation: _____________________   Special Instructions? _
```

F.5 Screen Template for the Gross Alpha-Beta Analyses (C for combined) Screen

```
 12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678
 1
 2
 3
 4
 5Tracking ID: _____ .
 6   Sample Name: _____
 7
 8Gross Alpha-Beta for Air Filters? _
 9    Length of count: ____    min or hr? _
10    Results needed by:  date: _____    time: _____    Completed: _____
11    Results report citation: _____    Special Instructions? _
12
13Gross Alpha-Beta for Other Samples? _
14    Length of count: ____    min or hr? _
15    Results needed by:  date: _____    time: _____    Completed: _____
16    Results report citation: _____    Special Instructions? _
17
18
19
20
```

F.6 Screen Template for Gamma Analyses (G) Screen

```
 12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678
 1
 2
 3Tracking ID: _____
 4   Sample Name: _____
 5
 6Screening/Shipping Count? _        Length of count: ____    min or hr? _
 7    Date counted: _____    RML Spectral ID: _____    Recount? _
 8    Results needed by:  date: _____    time: _____    Completed: _____
 9    Results report citation: _____    Special Instructions? _
10
11Full Isotopic Gamma Analysis? _    Length of count: ____    min or hr? _
12    Date counted: _____    RML Spectral ID: _____    Recount? _
13    Results needed by:  date: _____    time: _____    Completed: _____
14    Results report citation: _____    Special Instructions? _
15
16Other Gamma Analysis? _        Length of count: ____    min or hr? _
17    Date counted: _____    RML Spectral ID: _____    Recount? _
18    Results needed by:  date: _____    time: _____    Completed: _____
19    Results report citation: _____    Special Instructions? _
20
```

F.7 Screen Template for Special Instructions (I) Screen

```
 12345678901234567890123456789012345678901234567890123456789012345678901234567 8
1
2Tracking ID:  ____________
3   Sample Name:  __________________________________________________
4
5   :____________________:  __________________________________________________
6   :____________________:  __________________________________________________
7   :____________________:  __________________________________________________
8   :____________________:  __________________________________________________
9   :____________________:  __________________________________________________
10  :____________________:  __________________________________________________
11  :____________________:  __________________________________________________
12  :____________________:  __________________________________________________
13  :____________________:  __________________________________________________
14  :____________________:  __________________________________________________
15  :____________________:  __________________________________________________
16  :____________________:  __________________________________________________
17  :____________________:  __________________________________________________
18  :____________________:  __________________________________________________
19  :____________________:  __________________________________________________
20
```

F.8 Screen Template for RML/Analytical Radiochemistry Analysts (N for names) Screen

```
 12345678901234567890123456789012345678901234567890123456789012345678901234567 8
1
2 J. L. Doherty (Jodie)    _
3 T. J. Haney (Tom)        _
4 R. K. Murray (Ron)       _
5 C. L. Rowsell (Cal)      _
6 C. W. Sill (Claude)      _
7 D. S. Sill (Dave)        _
8 L. D. Smith (Larry)      _
9 G. K. Taylor (Gene)      _
10 L. A. Weinrich (Lou)    _
11 R. P. Wells (Rich)      _
12 Other                   _
13
14
15
16
17
18
19
20
```